Spektrum Industri
Vol. 21, No. 1, 2023, pp. 1-8
ISSN 1693-6590
http://journal3.uad.ac.id/index.php/spektrum

Design and Integration of a Robotic Welding Parametrized Procedure for Industrial Applications

Sangeeth Puthussery[a,1], Emanuele Lindo Secco[a,2,*]

[a] Robotics Lab, School of Mathematics, Computer Science & Engineering, Liverpool Hope Uni-versity)
[b] Second affiliation, Address, City and Postcode, Country (9pt)
[1] 21012690@hope.ac.uk; [2] seccoe@hope.ac.uk
* Corresponding Author

ARTICLE INFO

Article history
Received Month xx, 20xx
Revised Month xx, 20xx
Accepted Month xx, 20xx

Keywords
Sensor;
Controller;
Actuators;
Robot

ABSTRACT

This paper explores the development of an effective motion planning strategy for robotics welding in tube to tubesheet joints, a critical process in heat exchanger manufacturing. The research methodology follows an experimental paradigm, investigating two distinct approaches to tackle the intricacies of this task. The initial approach, involving a welding torch affixed to the robotic arm's flange, proved ineffective due to the complexity of continuous 360° orbital welding. This led to the adoption of a custom end effector in the second approach, designed to enhance adaptability and precision.

Key tools and materials employed in this research include the *Robot Operating System* (ROS), *Rviz* for 3D visualization, *MoveIt* for motion planning, *SolidWorks* for CAD modelling, and the *xArm7 Robotic Arm*. These tools facilitated the creation of a comprehensive planning environment. The motion planning process relies on three essential parameters: tube diameter, type of tube to tubesheet joint (flush or protruding), and the 3D coordinates of tube centers.

A Python scripts control the robot's movements, with specific joint state and pose goals for precise positioning. Finally, this study presents a program that orchestrates the robotic arm's motion, simulating the welding process for tube to tubesheet joints. This comprehensive research endeavor contributes to the optimization of motion planning strategies in the context of tube to tubesheet welding, with practical applications in the manufacturing industry.

1. Introduction

Tube to tubesheet *welding* in *heat exchangers* is a critical process in various industries, including petrochemical, power generation, and manufacturing (Ge et al., 2019). Achieving efficient and high-quality welds in this context is paramount for the performance and safety of heat exchangers. In recent years, the field of robotics and automation has offered innovative solutions to enhance the precision and repeatability of tube to tubesheet welding (Dongming et al., 1995).

Traditionally, *tube to tubesheet* welding was a manual process, relying on the skills and expertise of welders. While this approach yielded satisfactory results, it had limitations, including

inconsistency in weld quality and the inherent risks associated with manual labour in hazardous environments (Thekkuden et al, 2021). The advent of automation, driven by advancements in robotics and control systems, opened new possibilities for improving the welding process (Eren et al, 2023).

According to the research (Er and Das, 2023) early automation efforts in tube to tubesheet welding primarily focused on mechanized systems that used predefined trajectories for welding torches. These systems were an improvement over manual welding but lacked adaptability and the ability to handle variations in tubesheet geometry and tube layouts. Furthermore, they often required extensive setup time and manual adjustments, limiting their efficiency gains.

In this context, *motion planning* emerged as a pivotal component in automating tube to tubesheet welding. This process involves determining the optimal path and control commands for the welding torch or end effector to traverse the tubesheet and perform welds accurately. Effective motion planning is crucial for achieving consistent and high-quality welds, as it addresses the challenges posed by the complex geometry of tubesheets and the need for precise torch positioning (Gao et al., 2020).

The integration of simulation and visualization tools has been a significant advancement in motion planning for automated welding applications. Software platforms like *Gazebo*, *MoveIt* and *Robot Operating System* (ROS) have enabled researchers to create virtual environments for testing and validating motion plans before implementing them on physical robots (Hernandez-Mendez et al., 2017b). Simulation provides a risk-free means of refining motion plans and assessing their performance under various conditions.

Motion planning also necessitates careful consideration of *end effector design*. Haldankar et al. (2022) have explored custom end effectors tailored to the specific requirements of automated robotic welding. These end effectors are designed to provide the necessary torch orientation and alignment for different types of weld joints, including flush and protruding tubes. The flexibility and adaptability of end effector designs play a crucial role in achieving precise and efficient welding.

In the scope of this paper, the primary objective is to formulate a comprehensive motion plan for the welding process involved in joining *tubes to a tubesheet* within the context of a *Heat Exchanger*. A Heat Exchanger can be defined as a vital component in various industrial applications that facilitates the efficient transfer of heat between two distinct fluids.

Specifically, this research focuses on a particular category of heat exchanger known as a shell and tube heat exchanger. In this type of heat exchanger, one fluid traverses through a network of tubes, while another fluid of different temperature circulates within the shell surrounding these tubes, facilitating the heat exchange process (Linquip, 2021). During the fabrication process of shell and tube heat exchangers, a critical step involves the insertion of tubes into the tubesheet, followed by welding the end of the tubes to the tubesheet at both ends. The tube-to-tubesheet weld is undeniably pivotal within the manufacturing of these heat exchangers. Unfortunately, it frequently emerges as the weakest link, often resulting in premature failures. Welding the tube to the tubesheet demands a high degree of precision and expertise, and only highly skilled and extensively trained personnel are entrusted with this intricate task due to its critical nature. The significance of this task is amplified by the scarcity of skilled labor available, thereby necessitating strategies to address labor shortages within this specialized domain. Examples provide a detailed depiction of the orbital weld connecting a *tube to tube* sheets, offering a closer examination of this welding process. Moreover, welding the tube-to-tubesheet joints is often perceived as monotonous, as it requires workers to remain in a stationary position for prolonged periods. Considering that a single heat exchanger may comprise a multitude of tubes, the repetitive and tedious nature of this work can lead to worker fatigue and a potential loss of concentration. Such lapses in concentration can adversely affect the safety of the work environment and the quality of the welds. Consequently, there is a compelling case for the automation of this particular manufacturing process. The journey undertaken in this research aims to address these multifaceted challenges associated with the tube-to-tubesheet welding process in shell and tube heat exchangers. By creating an efficient and optimized motion plan, the objective is to

enhance the quality, reliability, and safety of the welding procedure while mitigating the dependence on highly specialized labor. The incorporation of automation into this critical manufacturing step not only promises to alleviate labor shortages but also reduces the risk of human error and fatigue-induced lapses in concentration.

In the following sections, we will delve deeper into the results and implications of these methodologies in the pursuit of optimized motion planning for tube-to-tubesheet welding. The outcomes of this research represent a significant advancement in the field of heat exchanger manufacturing, with far-reaching implications for industries reliant on these critical components for efficient heat transfer processes. The potential benefits of the motion planning strategy developed in this project have multiple use scenarios. First and foremost, it promises to enhance the overall quality and reliability of tube-to-tubesheet welds. The precision afforded by automated motion planning reduces the risk of defects, ensuring consistent weld quality across a range of tube configurations. The adoption of automation in the welding process addresses the persistent challenge of skilled labor shortages in the manufacturing sector. By minimizing the dependence on highly specialized personnel, industries can mitigate the risks associated with workforce availability and skill gaps, ultimately contributing to more stable and cost-effective production processes. Additionally, the integration of advanced software tools, such as the Robot Operating System (*ROS*), *Rviz*, *MoveIt* and *Gazebo*, showcases the potential of cutting-edge technology to reform traditional manufacturing practices. This interdisciplinary approach, bridging the realms of robotics, simulation, and engineering design, exemplifies the power of cross-disciplinary collaboration in solving complex industrial challenges.

The proposed system demonstrates a capacity for accommodating various categories of tube-to-tubesheet joints, thereby offering a versatile and inclusive solution applicable to a wide array of heat exchanger designs. Finally, the exploration of methodologies, the integration of state-of-the-art tools, and the development of an efficient motion planning strategy collectively lay the foundation for safer, more reliable, and automated welding processes in the production of shell and tube heat exchangers. The evolution of automation in tube-to-tubesheet welding has transformed this critical industrial process. From manual welding to sophisticated robotic systems, motion planning has become central to achieving consistent and high-quality welds. This project explores the comprehensive approach adopted to develop an effective motion planning strategy for tube-to-tubesheet welding. It begins by investigating various methodologies, leading to a shift from using a weld torch affixed to the flange to designing and implementing a custom end effector. The paper further delves into the used tools and materials and then deeper into the results and implications of these methodologies, shedding light on the journey to optimizing motion planning for tube-to-tubesheet welding.

2. Method

The research methodology employed in this study is grounded in an experimental paradigm aimed at developing an effective motion planning strategy for the robotics welding of tube to tubesheet joints. To accomplish this, a series of distinct approaches were explored, each tailored to address the intricacies of the task. Two primary methodologies were investigated sequentially, with the first involving the utilization of a model mimicking weld torch affixed to the flange, and the subsequent approach incorporating a custom end effector. It is imperative to underline that the initial method, involving the weld torch and flange, did not yield the desired outcomes, prompting the transition to the second approach.

2.1. Tools and Materials

The following section details the main tools this project is making use of.

Robot Operating System - The foundation of this research was built upon the Robot Operating System (ROS), a versatile middleware that seamlessly integrates various software components for motion planning and simulation. Specifically, ROS Noetic Ninjemys, in conjunction with Ubuntu 20.04, served as the platform for experimentation. ROS's open-source nature and exceptional capacity for software integration make it an ideal choice for orchestrating complex robotic tasks (wiki.ros.org, 2021). Primarily, ROS libraries are developed with a preference for Unix-like operating systems, in contrast to the more widely adopted Windows systems. Consequently, for the execution of this project, the Unix-based operating system Ubuntu 20.04 was selected as the platform of choice. Although ROS does offer an experimental version compatible with Windows 10, the decision to employ Ubuntu was made in order to streamline the development process, ensure the seamless functionality of libraries and packages, and mitigate potential compatibility concerns, thereby rendering Ubuntu a more advantageous selection for this endeavor.

Rviz - Rviz, an abbreviation for "ROS visualization," represents a noteworthy software tool within the field of robotics, offering three-dimensional (3D) visualization capabilities tailored for robots, sensors, and algorithms. Its principal function is to provide users with a means to observe and comprehend a robot's perception of its surroundings, whether in a real-world or simulated context (Sear-Collins ,2021). The fundamental objective of Rviz is to facilitate the visualization of a robot's state. This objective is realized through the utilization of sensor data, which is harnessed to construct a coherent and faithful representation of the robot's environment (Hershberger, 2019). In the context of the research endeavor under consideration, Rviz, as a sophisticated 3D visualization software solution, assumed a pivotal role. Its capabilities were instrumental in visualizing the movements of the robot and the incorporation of supplementary objects designed for the analysis of collision avoidance strategies. This, in turn, significantly contributed to the enhancement of accuracy and safety within the motion planning process.

MoveIt! - MoveIt, denoted as the Motion Planning Framework for the Robot Operating System (ROS), assumes a pivotal role within the ROS ecosystem by facilitating the configuration of the robot and the establishment of crucial pose configurations essential for diverse actions, notably encompassing operations such as pick and place tasks. The employment of the MoveIt Python interface conferred the capacity to command the robot's motion, expediting the programming process. According to Tellez (2018) at its core, the fundamental objective of the MoveIt system resides in its competence to furnish requisite trajectory planning for the robotic arm, thereby orchestrating the precise positioning of the end effector at specified target locations. This capability serves as a cornerstone for the effective execution of intricate robotic tasks.

Gazebo - Gazebo, a 3D robot simulator, serves the primary purpose of replicating the behavior of a robot within a simulated environment, providing a valuable approximation of its real-world counterparts. In this simulation framework, Gazebo possesses the capability to calculate and model the influence of various forces, including gravitational effects (open robotics, 2022). In the process of selecting suitable simulation software for the research endeavor, a comprehensive evaluation was conducted, with Gazebo and V-REP emerging as the primary contenders. Both options exhibited comparable functionalities and strengths in their respective capacities as simulation platforms (Lenka Pitonakova et al., 2018). Ultimately, the decision to adopt Gazebo was predicated upon its distinct advantage: its seamless integration into the Robot Operating System (ROS) environment, thereby rendering it the more practical and advantageous choice for the intended research objectives.

SolidWorks - SolidWorks stands as a renowned Computer-Aided Design (CAD) and computer-aided engineering (CAE) software solution of significant prominence made by *Dassault Systèmes*. Its pervasive utilization extends across diverse industrial sectors, encompassing mechanical and electrical engineering, product design, and manufacturing disciplines. This software's primary utility

resides in its capacity to facilitate the conception and evaluation of both 2-dimensional (2D) and 3-dimensional (3D) models, undertake intricate simulations, and produce comprehensive technical drawings.

xArm7 Robotic Arm - In this project, a commercial xArm7 device, namely a 7-axis Robotic Arm produced by UFACTORY, has been designated as the principal platform for the advancement of motion planning. The rationale behind this selection is predicated upon the multifaceted attributes possessed by the xArm7, which include its exceptional versatility, precision, and harmonious integration with the software tools employed within the research context, notably the *Robot Operating System* (ROS) and *MoveIt*, as detailed by Daniel (2021). Additionally, the availability of the xArm7 within the laboratory environment facilitated thorough assessments and further comprehensive investigations, rendering it an optimal choice for the research undertaking.

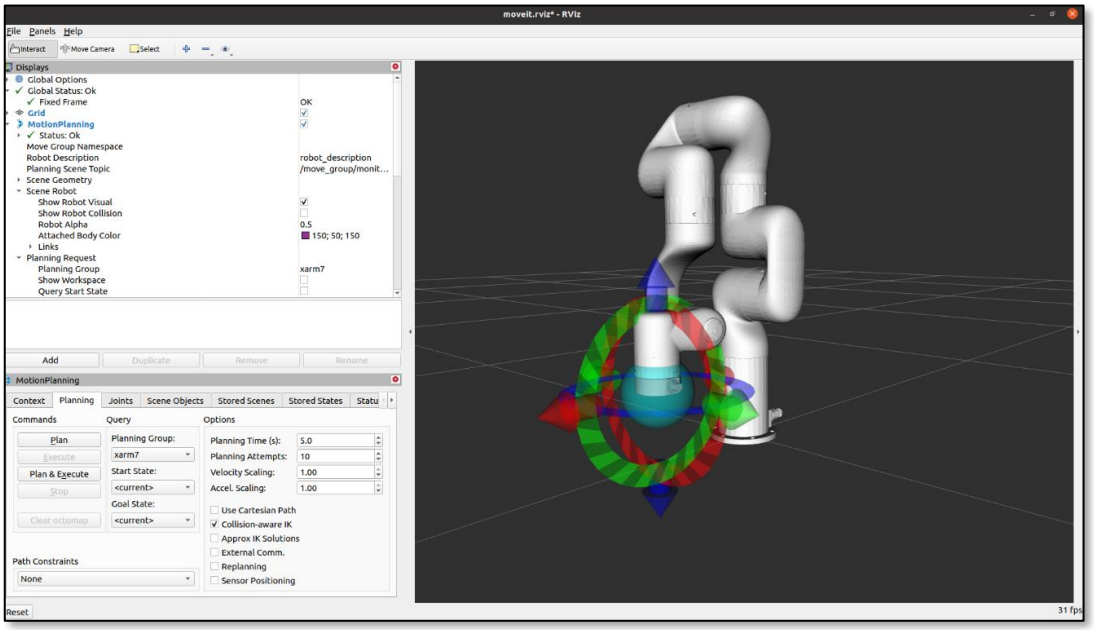

Fig. 1. The xArm7 Robotic Arm visualized within the *Rviz* scene with the *MoveIt* motion planning.

2.2. Planning environment setup Process

The 3D models of the xArm7 Robotic Arm, along with its associated ROS packages, were obtained from the official xArm GitHub repository (UFACTORY, 2023). Subsequently, these resources were integrated into the development environment comprising ROS Noetic Ninjemys and Ubuntu 20.04.

Figure 1, presented herein, provides a visual representation of the xArm7 Robotic Arm as it is depicted within the *Rviz* environment. Furthermore, it underscores the seamless integration of MoveIt, a critical component for motion planning. This integration, as depicted in the figure, signifies the arm's readiness for the forthcoming motion planning endeavors, a pivotal phase in the research process. Figure 2, displays a simplified 3D model, incorporating both the tubesheet and tubes, which was created using the SolidWorks CAD software. It is noteworthy that in practical scenarios, the tubesheet typically accommodates a substantial number of tubes. However, for the purpose of this research study, a deliberate reduction in the number of tubes was implemented, ensuring a more comprehensible visualization of the system. This comprehensive model was subsequently introduced into the robot's planning environment within the Rviz platform, serving as an element in the collision avoidance strategies employed during the motion planning process.

Spektrum Industri
Vol. 21, No. 1, 2023, pp. 1-8
ISSN 1693-6590

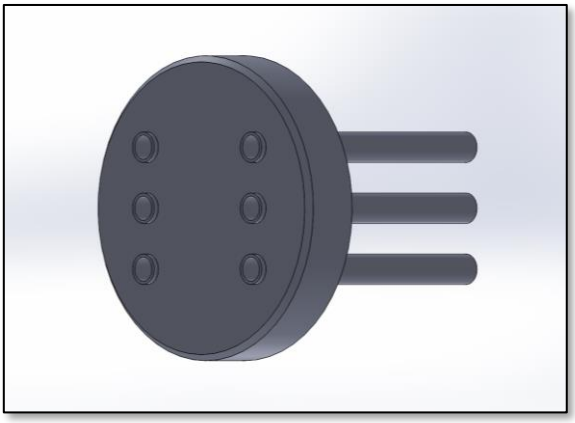

Fig. 2. Simplified Three-Dimensional Model of a Tubesheet accompanied by Tubes.

The three-dimensional model was seamlessly incorporated into the planning scene interface of MoveIt, as an STL object. The integration of this 3D model within the planning environment contributes significantly to the overall safety and effectiveness of the motion planning phase. This integration facilitates the comprehensive visualization of the tubesheet and tubes, enabling the robot to make informed decisions and adjustments to avoid potential collisions and ensure the successful execution of motion plan. The research methodology employed in this study is grounded in an experimental paradigm aimed at developing an effective motion planning strategy for the welding process. To accomplish this, a series of distinct approaches were explored, each tailored to address the intricacies of the task.

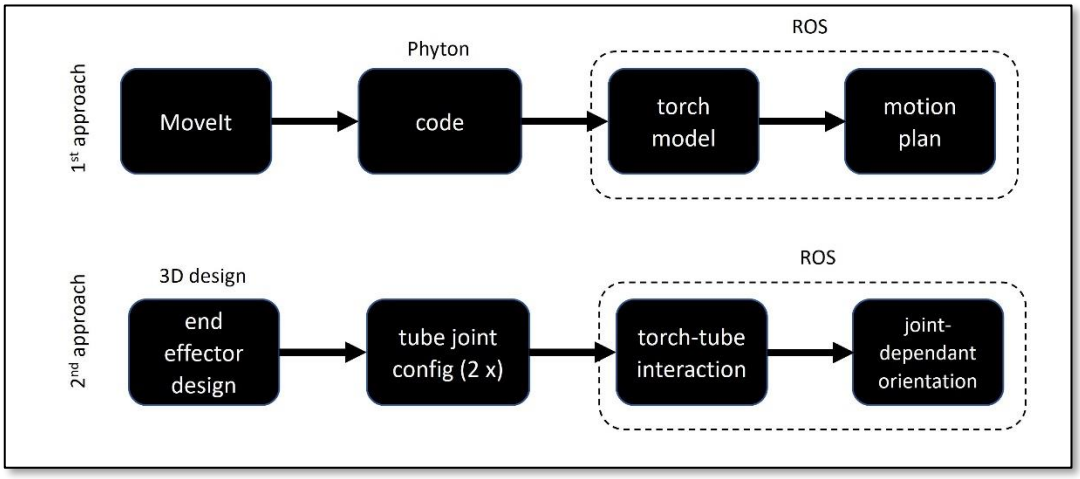

Fig. 3. Set-up of the two methodological approaches (details in Section 2.3 and 2.4)

2.3. First Approach

Initially, the integration process commenced by affixing a three-dimensional model of a welding torch to the flange of the xArm7 Robotic Arm, followed by the formulation of a motion plan tailored for the orbital welding of tube-to-tubesheet joints. Within the repository of ROS packages hosted on the xArm GitHub platform, provisions were available to secure an end-effector onto the robotic arm's flange. Subsequently, the endeavor sought to generate a motion plan utilizing the MoveIt framework. Nevertheless, this initial approach encountered notable inefficacy, primarily attributed to the inherent complexity of the motion plan's requirements (Figure 3, top panel).

Specifically, a key challenge emerged as the robotic arm exhibited limitations in its ability to execute continuous 360° rotations around the tube, a fundamental requisite for conducting orbital welding operations. Repeated attempts at motion planning yielded suboptimal outcomes, characterized by either plan failures or the robot's erratic and discontinuous movements during its circumferential traversal of the tube. These outcomes failed to align with the precise motion plan specifications demanded by the task at hand, namely *the coordinates of center point of tubes*, *the diameter of the tubes*, and the *overall number of tubes*. Consequently, certain robotic arm positions necessitated a revaluation of the chosen methodology, prompting a quest for alternative strategies.

2.4. Second Approach

Subsequently, the research progressed to the second approach, which encompassed the conceptualization and realization of a bespoke end effector (Figure 3, bottom panel). This pivotal component, integral to the robotic system, was crafted using SolidWorks, as illustrated in Figure 4. Comprising three distinct links, this end effector was engineered to fulfil indispensable functions. The initial link is affixed to the xArm7 Robotic platform's seventh link, also referred to as the flange. The second link establishes a connection with the first link through the implementation of a prismatic joint, enabling linear motion capabilities. The third link, designed to emulate a welding torch, interfaces with the second link via a revolute joint, facilitating dynamic orientation adjustments to accommodate the specific configurations of various tubes effectively. This purpose-driven end effector design significantly enhances adaptability, rendering it particularly well-suited for the precision execution of welding tasks across a spectrum of diverse tube layouts (Figure 3, bottom panel).

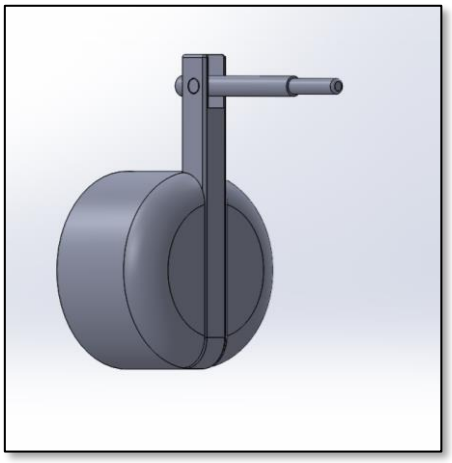

Fig. 4. Custom end effector, designed using SolidWorks, consists of three links.

This shift in methodology was driven by the shortcomings encountered in the initial approach. The custom end effector was envisioned to offer increased flexibility and precision in addressing the challenges associated with tube-to-tubesheet welding. In conjunction with this shift, a *Universal Robot Description Format* (URDF) model was generated from the SolidWorks environment using the SolidWorks URDF Exporter (Brawner, 2021). The URDF, an XML-based language, is used to describe and seamlessly integrate the robot into the ROS environment. The URDF representation of the xArm7 Robotic system was flawlessly integrated with the custom end effector. Additionally, crucial modifications were introduced to the URDF model to ensure compatibility with the visualization and simulation aspects of the research conducted in Rviz and Gazebo, respectively. This iterative approach, marked by a constant refinement of methodologies and models, underscores

the research's dynamic and adaptive nature, ultimately contributing to the pursuit of an optimal motion planning strategy for tube-to-tubesheet welding in heat exchangers. The URDF representation of the end effector was integrated with the URDF model of the xArm Robotic Arm, thereby establishing a comprehensive robot URDF configuration. Subsequently, this URDF model served as the foundation for the development of a MoveIt package, with the assistance of the MoveIt Setup Assistant—a valuable tool commonly employed in robotics application development.

2.5. Motion Planning

The initial phase in motion planning entails the generation of a MoveIt package derived from the URDF representation of the robotic system, a task facilitated by the MoveIt Setup Assistant. During the MoveIt package setup process, a deliberate decision was made to define two distinct planning groups: one designated for the robot arm and the other tailored specifically for the end effector. The two distinct planning groups were designated as 'arm' and 'tool,' as depicted in Figure 5, with the former tasked with controlling the robot arm and the latter responsible for overseeing the manipulation of the end effector. This configuration was developed within the MoveIt Setup Assistant, which provides a user-friendly graphical interface for defining robot kinematics, dynamics, and other essential parameters. The sepearation of planning groups was undertaken to enable independent control of both the arm and the end effector, thus affording enhanced flexibility and adaptability in task execution.

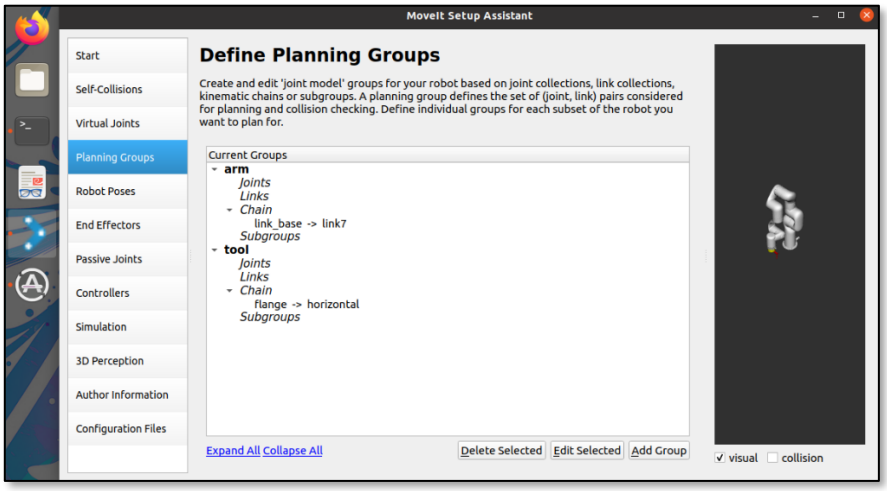

Fig. 5. Defining separate planning groups for arm and end effector.

To facilitate the generation of control code for the robot arm, the MoveIt Python interface was harnessed. Python was chosen as the preferred programming language for this purpose due to its inherent advantages, including ease of debugging and its capacity to expedite the prototyping phase of development. This selection was grounded in the practicality and efficiency it affords in the context of robotics control software development, which is greatly facilitated by the integrated capabilities of the MoveIt framework. The visual representation depicted in the Figure 6, illustrates the robotic system in conjunction with its associated end effector, as simulated within the Gazebo software environment. To achieve motion planning and solve for the kinematics of the robotic motion, an advanced algorithm called the *Open Motion Planning Library* (OMPL) is employed. OMPL, defined as a highly regarded motion planning framework, serves as the computational backbone for devising efficient and feasible motion trajectories for the robot (Ucan, Moll and Kavraki, 2012).

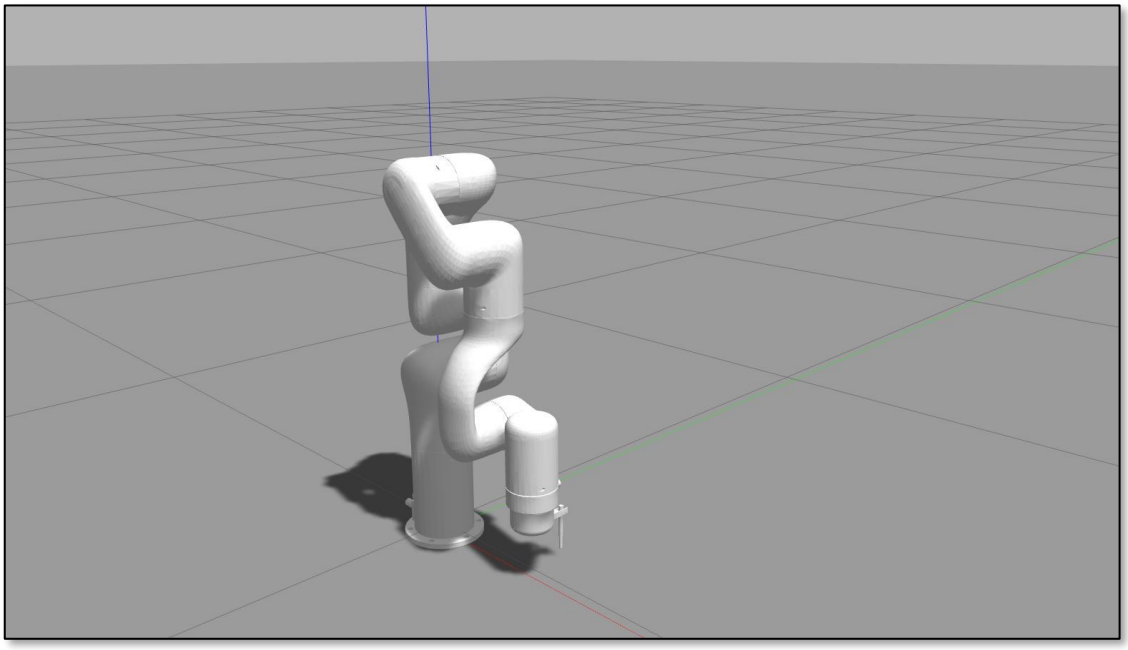

Fig. 6. The xArm Robotic Arm with the customized end effector within the *Gazebo* simulation environment.

Within the OMPL framework, the *Rapidly-Exploring Random Trees* (RRT) algorithm is strategically utilised. RRT, an integral component of OMPL, plays a pivotal role in the context of motion planning. Specifically, it is employed to explore and navigate the configuration space of the robot, facilitating the discovery of feasible and collision-free paths for the robot's end effector. This use of the RRT algorithm within OMPL is essential for optimising the robot's motion planning process, enhancing its precision, and ensuring the safe execution of tasks. The process of seamlessly integrating the Gazebo simulation environment with the MoveIt framework necessitates the creation of specific configuration files. They are ROS controllers tailored for Gazebo, as well as ROS controllers designed to facilitate the integration of MoveIt and Gazebo. The initial procedural step entails the creation of a configuration file intended for trajectory controllers in Gazebo. Subsequently, following the generation of Gazebo configuration files, the ensuing actions involve the development of both a MoveIt controller configuration file and a corresponding launch file (Joseph and Cacace, 2021). In Figure 7, a schematic representation illustrates the sequential steps essential for simulating the robot, encompassing the entire workflow from the inception of the Unified Robot Description Format (URDF) to the realisation of a comprehensive simulation. As depicted in the figure, the initial phase involves the construction of the complete URDF representation of the robot, including all requisite details crucial for both robot control and simulation purposes.

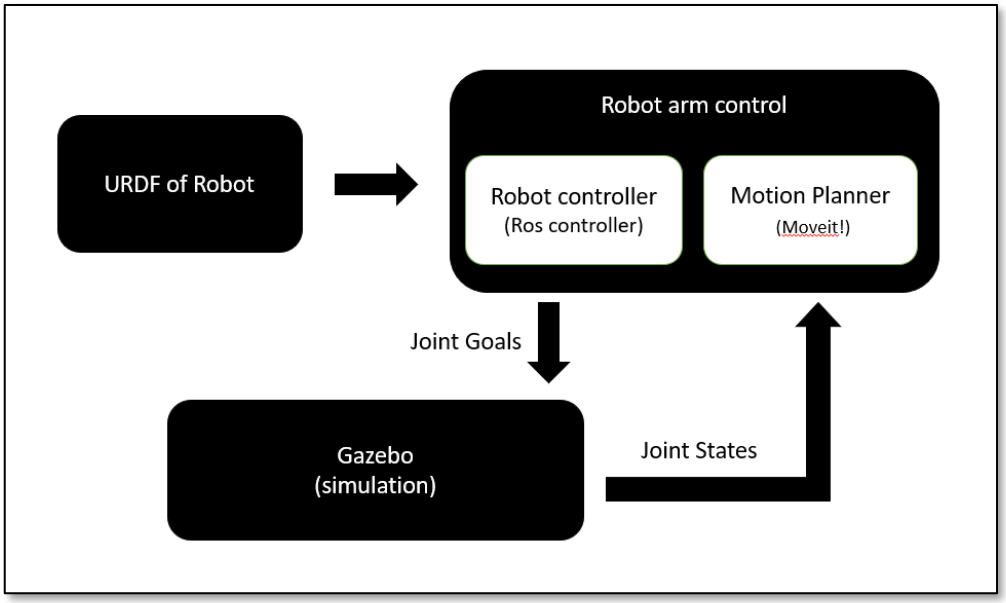

Fig. 7. The schematic diagram of the system combining three elements, namely the (1) *URDF*, (2) the *arm controller* and (3) the *Gazebo* simulation tool.

Afterwards, a ROS MoveIt package is generated, leveraging the URDF model coupled with the MoveIt Setup Assistant. This integration equips the robot with advanced motion planning and control capabilities. To execute specific joint movements, the joint goals are transmitted to the simulation software, while the Gazebo controllers are responsible for overseeing the simulation and ensuring accurate joint states are transmitted back to MoveIt. This bi-directional communication loop effectively synchronizes the robot's virtual representation in Gazebo with the motion planning and control capabilities facilitated by MoveIt, enabling a comprehensive and accurate simulation environment.

Subsequently, the Python code for robot control was carefully developed. During the program's creation, a deliberate effort was made to minimize the quantity of required input parameters, with the aim of diminishing complexity and enhancing practicality. The outcomes of this program shall be expounded upon in the ensuing section. In conclusion, this methodology section has detailed the comprehensive approach undertaken in this project to develop an effective motion planning strategy for the welding process, specifically in the context of tube-to-tubesheet joints in heat exchangers. The research journey began with an exploration of different methodologies, leading to a shift in approach from utilizing a weld torch affixed to the flange to designing and implementing a custom end effector. Throughout this process, a range of advanced tools and materials were leveraged, including the Robot Operating System (ROS), Rviz, MoveIt, Gazebo, SolidWorks, and the xArm7 Robotic Arm. The ensuing sections of this project will delve deeper into the results and implications of these methodologies in the pursuit of optimized motion planning for tube-to-tubesheet welding.

3. Results and Discussion

3.1. Configuration Setting

In the formulation of the motion plan, a deliberate effort was made to minimize the number of required inputs. In total, the motion plan relies on just three essential parameters. The first parameter pertains to the tube diameter, while the second parameter specifies the type of tube-to-tubesheet joint. Notably, the motion plan accommodates two distinct types of tube-to-tubesheet welds: one where

the tube does not protrude from the tubesheet, and the other where the tube protrudes from the tubesheet. These two types of welds are the most commonly seen in the industry. These two parameters enable the end effector to orient itself accordingly. The end effector orients itself according to the type to tube to tubesheet weld. In Figure 8 (left panel), the end effector's configuration is examined under conditions where no tube protrusion is present from the tubesheet. In this particular configuration, the third link assumes a perpendicular orientation in relation to the second link. Additionally, the second link exhibits linear motion, aligning the end of the third link precisely with the outer diameter of the tube. Conversely, Figure 8 (right panel) illustrates the end effector's orientation when the tube protrudes from it.

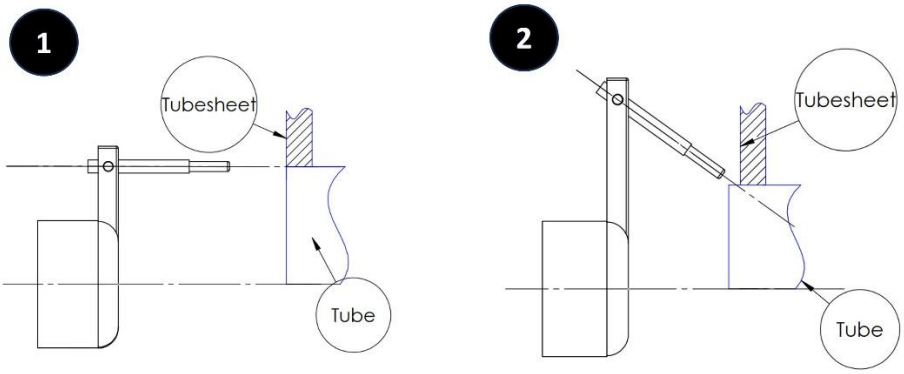

Fig. 8. Alignment of the end effector in instances where the tube remains flush with the tubesheet and when the tube protrudes from the tubesheet on the left and right panel, respectively.

According to this design, incorporating recessed tubes will require only minor modifications to the end effector. The design of the end effector, in fact, is specifically tailored to accommodate medium to large scale heat exchangers, which are predominantly used within the industry. These heat exchangers feature tubes with a diameter between 15 mm and 25 mm. Consequently, the end effector is engineered to facilitate the accommodation of the holes corresponding to these dimensions. During the welding process of the tube to the tubesheet, the heat exchangers will be isolated, so that the robot will have room to move around without impacting into the other tubesheets.

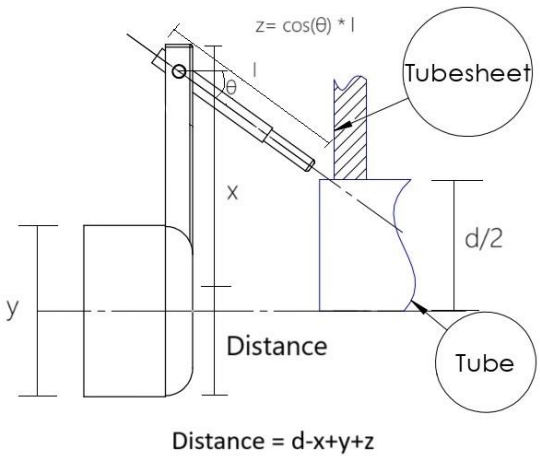

Fig. 9. Angular displacement of the end effector according to the *American Society of Mechanical Engineers* standards.

Spektrum Industri
Vol. 21, No. 1, 2023, pp. 1-8
ISSN 1693-6590

To achieve an optimal welding torch angle under these circumstances, the third link rotates such that it is positioned at an angle 45° to the second link. The angle was selected according to the *American Society of Mechanical Engineers* (ASME) Standards, which regulate the manufacturing process of heat exchangers (ASME, 2015). According to Figure 9, the second link's displacement can be determined using the following equation:

$$distance = d - x + y + z \quad (1)$$

where:

- *distance* represents the required displacement of the second link from its zero position.
- *d* signifies the outer diameter of the tube.
- *x* corresponds to the length of the second link.
- *y* represents the radius of the flange of the x-arm.
- *z* is equal to the cosine of the angle of rotation of the third link, multiplied by the length of the third link.

The third input crucial to the process is the location of the tubes. Specifically, the three-dimensional coordinates of the center points of the tubes are provided to the robot. With this positional information in hand, the robot systematically moves to the center point of each tube and faithfully replicates the motion plan necessary to execute a tube-to-tubesheet weld. The control flow diagram depicting the code utilized to govern the robot arm in Python is presented in Figure 10.

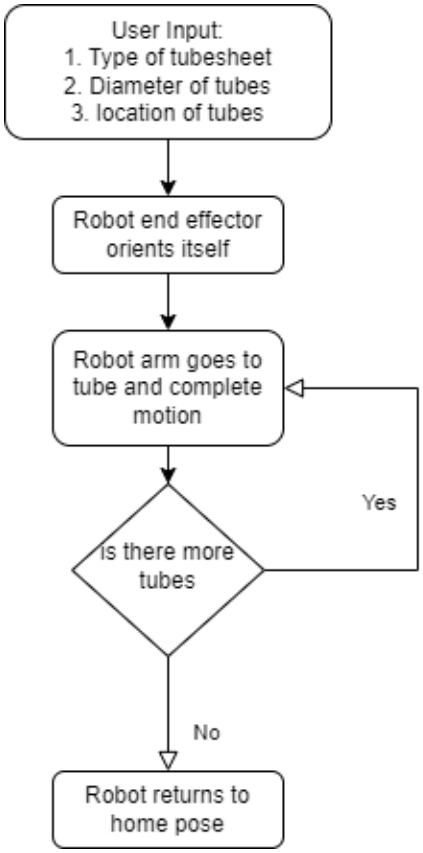

Fig. 10. A schematic representation of the control flow of the code governing the generation of the motion plan.

The flow of control in the program is shown in the Figure 10. The whole process involves a sequence of steps wherein the user provides three key inputs: firstly, the specification of the tube-to-tubesheet joint, secondly the diameter of the tubes, and thirdly the precise three-dimensional coordinates of the center points of the tubes. Subsequently, the robotic end effector undergoes an orientation adjustment according to the characteristics of the tube-to-tubesheet joint type and the tube diameter. Following this, the robotic arm proceeds to reposition itself to align with the coordinates of the first tube and performs a controlled rotational motion of the end effector around the tube, thereby emulating the welding process typically associated with tube-to-tubesheet joints. Upon the completion of the welding motion for the first tube, the robotic system conducts an assessment to determine the presence of any remaining tubes. In the event that additional tubes are identified, the system transitions to the next tube for a subsequent welding operation. Afterwards, if no further tubes remain to be processed, the system returns to its designated home position.

3.2. Program Setting

The script starts by importing imports requisite libraries and modules for ROS integration, motion planning, geometry, mathematics, and message handling.

1. *MotionPlanXarm Class*

This class encapsulates the core functionalities of the robotic arm control. It carries out the following tasks:

- Initialisation of ROS and MoveIt! configurations.
- Establishment of the robot's planning scene.
- Definition of arm and end-effector groups.
- Implementation of methods for joint state control, end-effector control, and pose goal setting.
- Retrieval of pertinent robot information.

2. *Joint State Control Methods*

The *go_to_joint_state* and *go_to_joint_state_tool* methods facilitate the manipulation of joint states for the arm and the end-effector, respectively. They facilitate precise positioning of the robotic arm.

3. *Pose Goal Control Method*

The *go_to_pose_goal* method enables the arm to achieve specific pose goals in Cartesian space, with specified position and orientation.

4. *Main Function*

The script's entry point is the main function. It orchestrates the execution of the robotic arm for a sequence of predefined tasks. The main steps involve initialisation, configuration of tube parameters, and arm movements to manipulate objects, potentially for assembly tasks.

5. *Conditional Statements*

The script incorporates conditional statements to adapt arm movements based on the type of tube and its radius. These conditions ensure that the arm's positioning and movements are appropriate for the task at hand.

6. *Loop and Joint Movements*

Within a loop structure, the script iterates through a series of tube positions, moving the arm to each location. Notably, it alternates the direction of flange rotation for every other tube.

7. *all_close Function:*

This function serves as a fundamental utility for assessing the proximity of the actual position to the desired goal position, with an allowance for tolerance. It can handle various input types, including joint states and poses.

The snippet of the programming code containing the main function is illustrated in Figure 11.

```
def main():
    try:
        input("Press Enter to start the program")
        planXarm = MotionPlanXarm()

        radiusTube = 0.025 # Diameter of tubes
        Tube_type = 1 # 1 = extended tubes, 0 flushed tubes

        # to move the according to the tube type and its radius
        input("press enter to move the tool")
        if Tube_type == 1:
            planXarm.go_to_joint_state_tool(1,-0.78)
            planXarm.go_to_joint_state_tool(0,(radiusTube-0.0075))
        else:
            planXarm.go_to_joint_state_tool(0,(radiusTube-0.0575))

        # the center points of tubes in tubesheet
        centersOfTubes = [[0.425,0.1,0.375],[0.425,0.0,0.375],
                          [0.425,-0.1,0.375],[0.425,-0.1,0.125],
                          [0.42,0.0,0.125],[0.425,0.1,0.125]]


        #iterating through each tube
        for x in range(0,len(centersOfTubes)):
            planXarm.go_to_pose_goal(centersOfTubes[x][0],centersOfTubes[x][1],
                                     centersOfTubes[x][2],0,pi/2,0)
            joint_position = planXarm.get_current_position(6)

            #to change direction  of roatation of flange every other tube
            if (joint_position < 0):
                g=-3.11
                while(g<=3.11):
                    planXarm.go_to_joint_state(6,g)
                    g += 0.1555
            else:
                g=3.11
                while(g>=-3.11):
                    planXarm.go_to_joint_state(6,g)
                    g -= 0.1555
```

Fig. 11. Code's snippet.

The depiction of the robotic arm's execution, as observed within the *Rviz* visualization tool, is elucidated through a sequential series of images, as presented in Figure 11. In the initial frame,

denoted as "Image 1," the robotic arm assumes its home position. Transitioning to "Image 2," the 3D representation of the tubesheet is integrated into the robot's planning environment. Subsequently, within the frames labelled "Images 3 to 4," the robotic arm methodically navigates itself to the precise coordinates associated with the initial tube. Continuing in "Images 5 to 6," the robotic arm undergoes iterates through all tubes, mirroring the welding procedure employed in affixing the tubes to the tubesheet. Finally, the robotic system concludes its operation by returning to the designated home position, an action that is perceptible in "Image 8."

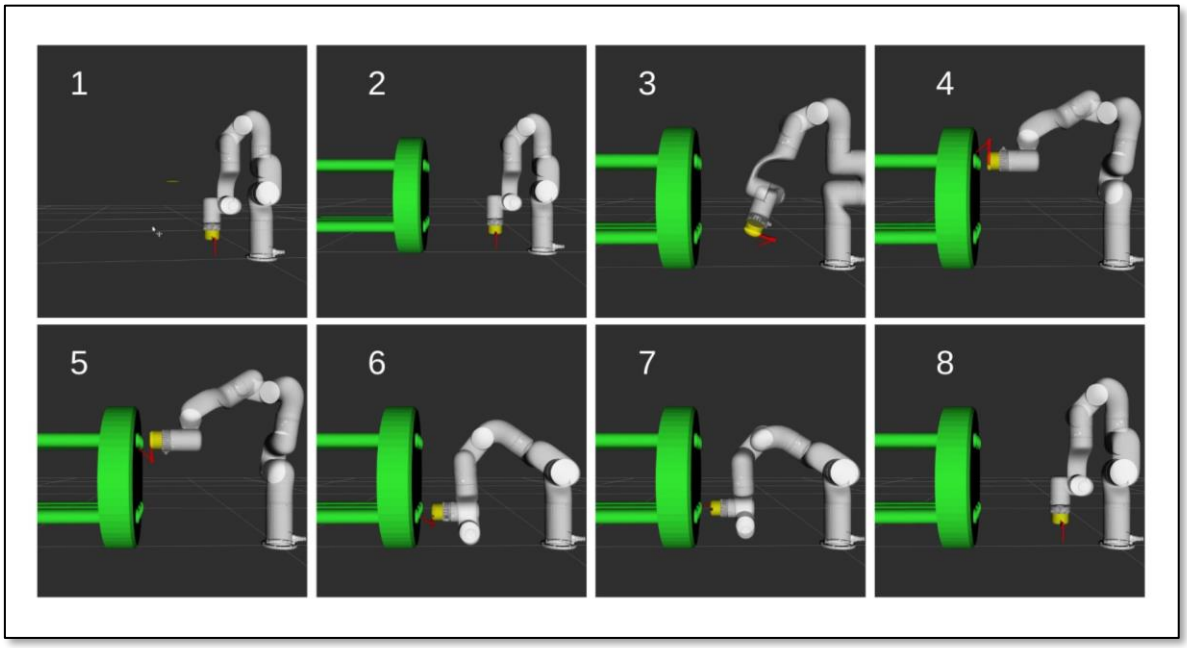

Fig. 11. Sequence of images showing the visualization of the robot arm in *Rviz*.

4. Discussion

The adoption of the *Robot Operating System* (ROS) within the context of this project is considered advantageous, as evidenced by the outcomes achieved and the progression of the project throughout its development. ROS is an open-source software framework that enjoys substantial support from a thriving community. This community support proved invaluable when addressing obstacles encountered during the project, as swift resolutions were readily accessible through collaborative contributions from community members. While there exist alternative platforms for robot simulation, such as MATLAB and ROBO DK, boasting extensive user support, it is worth noting that these alternatives are accompanied by substantial costs. Particularly for small-scale industries, the financial burden associated with maintaining a robot arm under these proprietary solutions may not be cost-effective. In contrast, open-source solutions like ROS offer a more financially viable alternative. However, it is important to acknowledge that ROS presents a steep learning curve, and comprehensive documentation and tutorials for all potential use cases are not yet fully developed. Nevertheless, ROS introduces numerous benefits, such as seamless integration with Gazebo, a simulation environment.

The motion planning interface relies on MoveIt, which interfaces effectively with ROS. Moreover, MoveIt offers a Python interface that simplifies the programming and deployment of programs into the simulation environment, further enhancing project efficiency. While the motion

planning was implemented in Python, it is conceivable to improve efficiency by transitioning to C++, as the Python interface of MoveIt essentially wraps the C++ functions. This transition can potentially lead to code optimization, reduced code size, and enhanced processing speed in future iterations. It is essential to highlight that the time parameterization of the motion was not addressed in the project. While the OMPL motion planner provides basic time parameterization functionalities, future enhancements could explore alternative motion planners, such as the Descartes planner or Pilz Industrial motion planner, to gain more precise control over the robot arm's motion.

Notably, Gazebo emerged as an ideal companion to ROS due to its seamless integration, surpassing V-REP in this regard. While V-REP boasts an extensive library of available robot models and shares the open-source ethos, its integration with ROS proved more complex compared to Gazebo (Arregi, 2023). The selection of SolidWorks as the primary modelling software was propelled by its efficient URDF exporter plugin, which facilitates the generation of URDF models replete with intricate details. Fusion 360 is a potential alternative, favored in some industry settings due to its cost-effectiveness. However, the URDF exporter plugin in SolidWorks provided a valuable asset for this project, simplifying the integration of the end effector and the xArm Robot. MoveIt incorporates an integrated collision avoidance system that identifies discrepancies within the motion plan by leveraging the 3-dimensional models of both the robot and the tubesheet, even if the detection of faults during the welding process was not implemented at this stage. The versatility of the xArm Robot, endowed with 7-axes, rendered it well-suited for orienting itself efficiently during motion planning. The manufacturer-provided documentation for the xArm Robot, while comprehensive, must be underscored as compensating for a lack of robust user support. The successful creation of the motion plan using open-source software implies the feasibility of automating the welding process for tube-to-tubesheet applications in industrial settings. Simulation software played a pivotal role in this project by enabling safe and cost-free testing of robot control code, thereby circumventing safety concerns and the high expenses associated with real-world testing. This underscores the advantages of simulation software for research purposes, as it allows for iterative testing of prototypes in a virtual environment with minimal effort compared to physical testing. In the initial attempt, a rudimentary 3D model of the welding torch was employed to simulate joint welding motions, but it proved ineffective. Subsequently, a custom end effector was designed and incorporated, demonstrating the rapid prototyping and testing capabilities facilitated by simulation software.

In summary, the adoption of ROS within this project offers a cost-effective, community-supported framework, albeit with a demanding learning curve and limited documentation. Consideration should be given to transitioning to C++ for enhanced code efficiency. Motion planning enhancements can be pursued through more sophisticated planners. Gazebo stands out as a ROS-compatible simulation environment, and SolidWorks facilitated URDF model generation for complex robot structures, notably the xArm Robot with its seven-axis capability.

5. Conclusion

In conclusion, this project has undertaken a comprehensive exploration of motion planning strategies for tube-to-tubesheet welding within the context of shell and tube heat exchanger manufacturing. The research journey commenced with a thorough investigation of diverse methodologies, culminating in the development of an efficient and adaptive motion planning approach. The utilization of advanced tools and materials, such as *ROS*, *Rviz*, *MoveIt*, *Gazebo*, *SolidWorks*, and the *xArm7 Robotic Arm*, has paved the way for a transformative shift in the industry's welding practices.

The motion planning strategy presented herein promises to reform the field of heat exchanger manufacturing in several critical ways. Firstly, it elevates the quality and reliability of tube-to-tubesheet welds through precision and consistency, thereby reducing the likelihood of defects and premature failures. Secondly, by incorporating automation into the welding process, this research addresses the persistent challenge of skilled labor shortages, offering industries a more stable and efficient production paradigm. Furthermore, the interdisciplinary integration of cutting-edge software tools underscores the transformative potential of cross-disciplinary collaboration in solving complex industrial challenges (Tharmalingam, 2022; Manolescu, 2022). The adaptability of the developed motion planning system to diverse tube configurations reaffirms its versatility and applicability across a spectrum of heat exchanger designs.

In essence, this project provides a significant contribution to the realm of heat exchanger manufacturing, offering a holistic solution to the intricate challenges inherent in tube-to-tubesheet welding. The meticulously explored methodologies, alongside the integration of state-of-the-art tools, have laid the groundwork for safer, more reliable, and automated welding processes within the shell and tube heat exchanger industry. As industries progress towards the adoption of these innovative approaches, the potential for revolutionizing heat exchanger manufacturing becomes palpable, promising enhanced efficiency, consistency, and adaptability in the production of these vital industrial components.

Author Contribution: All authors contributed equally to the main contributor to this paper. All authors read and approved the final paper.

Funding: This research received no external funding.

Acknowledgment: This work was presented in dissertation form in fulfilment of the requirements for the MSc in Robotics Engineering for the student Sangeeth Puthussery from the Robotics Labs, School of Mathematics, Computer Science & Engineering, Liverpool Hope University.

Conflicts of Interest: The authors declare no conflict of interest.

References

American Society of Mechanical Engineers (ASME), 2015. *ASME Boiler and Pressure Vessel Code, Section 8, Division 1: Rules for Construction of Pressure Vessels, p136.*

Brawner, S. (2021). *sw_urdf_exporter - ROS Wiki*. [online] wiki.ros.org. Available at: http://wiki.ros.org/sw_urdf_exporter [5Accessed 20 June. 2023].

Daniel, W. (2021). *The difference between UFACTORY xArm5, UFACTORY xArm6 and UFACTORY xArm7 | UFACTORY Help Center*. [online] help.ufactory.cc. Available at: http://help.ufactory.cc/en/articles/4491842-the-difference-between-ufactory-xarm5-ufactory-xarm6-and-ufactory-xarm7 [Accessed 7 Aug. 2023].

Dongming, W., Zengfu, G., Li, T. and Feng, T. (1995). The technical summary of tube-sheet welding of heat-exchangers in China and out of China. *Press Vessel Technol*, 12, pp.48–53.

Er, B.H. and Das, T. (2023). COMPARISON OF ORBITAL AND MANUAL WELDING FOR MANUFACTURING STAINLESS STEEL HEAT EXCHANGER. *International Journal of Advanced*

Natural Sciences and Engineering Researches, 7(6), pp.358–362. doi:https://doi.org/10.59287/ijanser.1173.

Eren, B., Demir, M.H. and Mistikoglu, S. (2023). Recent developments in computer vision and artificial intelligence aided intelligent robotic welding applications. *The International Journal of Advanced Manufacturing Technology*, [online] 126(11), pp.4763–4809. doi:https://doi.org/10.1007/s00170023114564.

Gao, W., Tang, Q., Yao, J. and Yang, Y. (2020). Automatic motion planning for complex welding problems by considering angular redundancy. *Robotics and Computer-Integrated Manufacturing*, [online] 62, p.101862. doi:https://doi.org/10.1016/j.rcim.2019.101862.

Ge, Y., Xu, Y., Yu, H., Chen, C. and Chen, S. (2019). A method for detecting central coordinates of girth welds based on inverse compositional AAM in tube-tube sheet welding. In: S. Chen, Y. Zhang and Z. Feng, eds. Springer Singapore, pp.65–81.

Haldankar, T., Kedia, S., Panchmatia, R., Parmar, D. and Sawant, D. (2022). Design of robotic manipulator for welding using ROS and gazebo. pp.1–6. doi: https://doi.org/10.1109/DELCON54057.2022.9753305.

Hernandez-Mendez, S., Maldonado-Mendez, C., Marin-Hernandez, A., Rios-Figueroa, Homero Vladimir, Vazquez-Leal, H., and Palacios-Hernandez, E.R. (2017a). Design and implementation of a robotic arm using ROS and MoveIt! pp.1–6. doi: https://doi.org/10.1109/ROPEC.2017.8261666.

Hershberger, D. (2019). *rviz - ROS Wiki*. [online] Ros.org. Available at: http://wiki.ros.org/rviz.

Joseph, L. and Cacace, J. (2021). *Mastering ROS for Robotics Programming*. Packt Publishing Ltd.

Lenka Pitonakova, Giuliani, M., Pipe, A.G. and Alan (2018). Feature and Performance Comparison of the V-REP, Gazebo and ARGoS Robot Simulators. *Lecture Notes in Computer Science*, pp.357–368. doi: https://doi.org/10.1007/978-3-319-96728-8_30.

Linquip, T. (2021). *Heat Exchanger Parts: The Advantages of Each Component*. [online] Linquip. Available at: https://www.linquip.com/blog/heat-exchanger-parts-a-complete-description/.

Lumelsky, V. (1987). Effect of kinematics on motion planning for planar robot arms moving amidst unknown obstacles. *IEEE Journal on Robotics and Automation*, 3, pp.207–223. doi: https://doi.org/10.1109/JRA.1987.1087094.

Open robotics (2022). *Gazebo*. [online] gazebosim.org. Available at: https://gazebosim.org/about.

Ren, K., Lu, Z., Dong, H., Cheng, D., Yu, Y., Cui, R., Yu, Q. and Yan, S. (2018). Target grasping and obstacle avoidance motion planning of humanoid robot. pp.250–255. doi: https://doi.org/10.1109/IISR.2018.8535650.

Sears-Collins, A. (2021). *What is the Difference Between rviz and Gazebo? – Automatic Addison*. [online] Available at: https://automaticaddison.com/what-is-the-difference-between-rviz-and-gazebo/.

Tellez, R. (2018). *What is moveit_ros? All about MoveIt! ROS*. [online] The Construct. Available at: https://www.theconstructsim.com/ros-movelt/.

Thekkuden, Dinu Thomas, Mourad, A.-H.I. and Bouzid, A.-H. (2021). Failures and leak inspection techniques of tube to tubesheet joints: A review. *Engineering Failure Analysis*, 130, p.105798.

Ucan, I., Moll, M. and Kavraki, L. (2012). *The Open Motion Planning Library*. [online] Available at: https://ompl.kavrakilab.org/ieee-ram-2012-ompl.pdf [Accessed 20 June. 2023].

UFACTORY (2023). *Contents:* [online] GitHub. Available at: https://github.com/xArm-Developer/xarm_ros [Accessed 7 June. 2023].

Waldron, S.R. (2020). *Secrets to a Lasting Tube to tubesheet Joint: Properly Pre-Setting Tubes*. [online] hydropro. Available at: https://www.hpro.com/post/secrets-to-a-lasting-tube to tubesheet-joint-properly-pre-setting-tubes [Accessed 9 July. 2023].

Arregi, M.O., Secco, E.L., Operative Guide for 4-wheel Summit XL Mobile Robot set-up, Acta Scientific Computer Sciences, 5(4), 39-47, 2023

Tharmalingam, K., Secco, E.L., A Surveillance Mobile Robot based on Low-Cost Embedded Computers, 3rd International Conference on Artificial Intelligence: Advances and Applications, 25, 323-334, (ICAIAA 2022) DOI : 10.1007/978-981-19-7041-2

Manolescu, V.D., Secco, E.L., Design of an Assistive Low-Cost 6 d.o.f. Robotic Arm with Gripper, 7th International Congress on Information and Communication Technology (ICICT 2022), Lecture Notes in Networks and Systems (ISSN: 2367-3370), 1, 39-56, 2022, DOI: 10.1007/978-981-19-1607-6