Ruben Konings (student);Thom Peters (student)

Project S3 – Rev up your robot coding skills

Plan of approach

# Version revision

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Date | Autor | Note |
| 0.1 | 28-8-2023 | Ruben and Thom | Lay-out and info |
| 0.2 | 6-9-2023 | Ruben and Thom | Full version |
| 0.3 | 12-9-2023 | Ruben and Thom | Implemented feedback and functional design |
| 1.0 | 14-19-2023 | Ruben and Thom | Implemented feedback for final version |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Table of contents

[1. Version revision 1](#_Toc145582703)

[3. Problem 3](#_Toc145582704)

[4. To do plan 4](#_Toc145582705)

[5. Deliverables 7](#_Toc145582706)

[6. Planning 8](#_Toc145582707)

[7. Functional design 9](#_Toc145582708)

# Problem

Improving a Delta Gripper Robot: Fixing Bugs and Creating a User-Friendly Interface

In this project, we have received a Delta Gripper Robot without a working operating system. Our goal is to identify and resolve any issues in the existing system, as well as develop a user-friendly interface for controlling the robot. This involves debugging the legacy system and designing a simple interface to make it easier to operate the Delta Gripper Robot. The project aims to enhance the robot's functionality and user experience.

# To do plan

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| What | How | When | Deadline | Who |
| Research | In-depth research on the functioning and applications of the delta picker based on prior student projects and industry case studies. | Week 35: Gather initial research materials.  Week 36: Analyse gathered data and identify key insights. | Week 37: Complete the research phase. | Ruben & Thom |
| * Documentation | Gather academic papers, technical documentation, and user manuals related to delta grippers. |
| * Contact project teams | Contact previous project teams and experts in the field for insights. |
| * Experiments | Conduct experiments if necessary to understand the delta gripper's capabilities.  And see how other robot’s work. |
| * Qube operating system | Meet with the one of the developers of both the cube and the delta robot to get more insite and a understanding for both of them. |  |  |  |
| * Other operating systems | Search for programms and docmunentation of other ROS. |  |  |  |
| Jogging system | Develop a robust control system that allows the delta gripper to move in any direction with an accuracy of 5 mm if the hardware is capable. | Week 37: Initial code development and testing.  Week 38: Iterative improvements and fine-tuning. | Week 39: Achieve seamless multidirectional movement. | Ruben & Thom |
| * Test code | Write and test code for delta gripper movement, focusing on directional accuracy and control responsiveness. |
| * Position | Implement feedback mechanisms to ensure the tool position accuracy. |
| Coordinate system | Create a comprehensive coordinate system that accurately determines the robot's position and destination within its workspace. And move to these position smoothly. | Week 40: Begin coordinate system design.  Week 41-42: Algorithm development and testing. | Week 43: Full implementation of the coordinate system and testing improvements. |  |
| * Framework | Design a coordinate framework that integrates with the robot's hardware and software. |
| * Algorithms | Develop algorithms for real-time position tracking and mapping. |
| GUI (Graphical User Interface) | Develop an intuitive and user-friendly GUI to control the robot's actions and monitor its status. | Week 44: GUI design and initial coding.  Week 45-46: Integration and testing. | Week 47: Fully functional GUI. |  |
| * Design | Design a user interface with clear visuals, controls, and feedback elements. |
| * Integrate | Integrate the GUI with the robot's control software. |
| Programs | Create a suite of programs that enable the robot to execute complex tasks, including moving to multiple positions without interruption. | Week 48: Program architecture design.  Week 49-50: Coding and initial testing. | Week 51: Reliable program suite. |  |
| * Routines | Develop software routines that coordinate movements and task sequences. |
| * Error handling | Implement error handling and recovery mechanisms. |
| Tool Integration | Develop a modular system that allows the robot to adapt to various tools, considering attributes like length and input requirements. | Week 52: Tool interface design.  Week 1-2 (Next Year): Software integration and tool testing. | Week 3 (Next Year): Complete tool integration. |  |
| * Calibrate | Implement software features to recognize and calibrate with different tools. |  |  |  |
| Pendant | if we have time left to make a tablet into a “pendent” by remote controlling the delta robots | To be determined |  |  |

# Deliverables

1. Blog, deadline: 14 weeks of the start.
2. Demonstration + presentation: 2 weeks before ending.
3. Documentation and presentation:
   * User manual – a step by step guide on how to use the system
   * Commented and structured code on GitHub
   * Project report
   * Architecture document –software architecture (it can be an appendix in the report)
   * Everything uploaded on Teams in a final version folder
4. Learning material for partners, other students:
   * Blog
   * Worked out demonstration.
   * Video of the demonstration
   * Poster

# Planning

The planning can be found in the folder: \Project S3\Documentation\Planning S3.

# Functional design

In consultation with the client, the following list of functional specifications has been drawn up based on the SMART criteria (Unknown, SMART criteria, 2022). The specifications are prioritized in the second column according to the MoSCoW method (Unknown, MoSCoW method, 2022). (Arends, 2022)

|  |  |  |
| --- | --- | --- |
| **SMART functional design** | | |
| **#** | **MoSCoW** | **Description** |
| **F1** | **M** | **The Delta robot must work properly** |
| F1.1 | M | This requirement mandates that the Delta robot should consistently and accurately perform its designated tasks. It holds critical importance, especially in applications like manufacturing, assembly, or precision operations. |
| F1.2 | M | Ensuring the robot maintains constant coordinates is essential for precision over time. |
| F1.2.1 | M | * Accurate mathematical calculations are pivotal to maintaining positional stability. |
| F1.2.2 | M | * Avoiding motor and arm flex ensures precise and reliable operation. |
| F1.2.3 | M | * Unrestricted joint movement is crucial for achieving smooth and precise motion. |
| F |  | * Software improvement of the speed of the program or efficiency |
|  |  | * Improve the communication and the speed |
| **F2** | **M** | **The Delta robot must work with the Qube** |
| F2.1 | M | This requirement stipulates that the Delta robot must seamlessly integrate with the Qube platform, guaranteeing compatibility. |
| F2.2 | M | Compatibility with the Qube's software is paramount to ensure effective control and monitoring within the Qube ecosystem. |
| **F3** | **C** | **The Delta X2 should work properly** |
| F.3.1 | C | Similar to the Delta robot, this requirement demands that the Delta X2 performs its tasks consistently and accurately. |
| F3.2 | C | Compatibility with the Qube's software is indispensable for the Delta X2 to enable effective management and control via the same platform. |
| **F4** | **S** | **The Delta robot and Delta X2 must work with plug and play** |
| F4.1 | S | This requirement necessitates straightforward and user-friendly setup and configuration for both robots, emphasizing ease of use. |
| F4.2 | S | Uniform control methods enhance the user experience, simplifying the management of multiple robots concurrently. |
| F4.2.1 | S | * Sharing a single ethernet cable reduces clutter and streamlines physical setup. |
| **F5** | **M** | **There must be a User Interface on a PC to control both the Delta robot and Delta X2** |
| F5.1 | M | This requirement underscores the importance of a user-friendly PC interface to enable operators to interact with and control the robots effectively. |
| F5.2 | M | Individual operation allows for independent control of each robot when necessary. |
| F5.3 | M | The jogging function facilitates manual control for precise positioning, supporting movement in the Y-axes, X-axes, and Z-axes. |
| F5.4 | M | Automated movement programs using XYZ coordinates streamline repetitive tasks. |
| F5.5 | M | The gripper's capability to open and close is essential for tasks involving object manipulation. |
| **F6** | **W** | **The Delta robot and Delta X2 should work with a pendant** |
| F6.1 | W | This suggests that both robots should offer compatibility with a pendant control device, providing an intuitive control option. |
| F6.2 | W | Consistency with the PC UI controls enhances user familiarity and simplifies transitioning between control methods. |
| **F7** | **S** | **Consideration of advanced safety features, such as collision detection and emergency stop mechanisms** |
| F7.1 | S | Advanced safety features add an extra layer of protection for both the Delta robot and Delta X2. |
| F7.1.1 | S | * Collision Detection: When a collision risk is identified, the robot can stop or slow down, preventing damage to itself, nearby objects, and ensuring the safety of human operators. |
| F7.1.2 | S | * Emergency Stop Mechanisms: These mechanisms provide a quick and efficient way to halt robot operations in emergency situations. |
| **F8** | **W** | **Avoidance of implementing features that significantly increase project complexity without providing clear benefits** |
| F8.1 | W | The focus should be on features that directly contribute to the system's efficiency, safety, or usability. Any proposed features should undergo a cost-benefit analysis to assess their impact on project timelines, budgets, and overall feasibility. |
| F8.2 | W | The project team should maintain a balance between feature-rich functionality and simplicity to deliver a product that meets its intended purpose without unnecessary complexity. |

M = Must, S = Should, C = Could, W = Would.