## Abstract (200 – 300 words)

* Follow guide on VLE (<https://docs.google.com/document/d/1jvCs3of8zjf-G_jIX9kA-ZF32Rhi3oZX0a_fdN6bHeI/edit?pli=1>)

## Acknowledgements

## Ethics

* No current identified ethical considerations? Need to apply for ethical approval if any identified

## Table of Contents – (Include section on Figures, Tables and abbreviations)

## Glossary?

## Introduction – (Use initial report as reference)

* **Project Background and Context**
* **Project Objectives/Specification –** Modify specification from initial report to include enumerated points to refer back to following Marks feedback
* **Project Scope** – outline boundaries of the project and any restraints. There are little restraints other than the requirement to develop hardware for demonstration which puts development of a physical demonstration out of scope, however it can be completely prepared for from a software perspective
* **Dissertation Structure –** To be finalised

## Literature Review – (Use initial report as reference)

I am unsure how much of the initial report should be modified and pulled to the final report, however I will possibly be limited by word count stopping me from including all of it. So, the below sections not in bold are seen as lower priority and will be the first to be removed if a word count problem is identified.

* Overview of Modular Spacecraft
* **State-of-the-art in Spacecraft Modularity and Autonomous Reconfiguration**
* Challenges and Limitations of Automated Reconfiguration in Space
* **Emerging Advancements in Reconfiguration Technologies**
* **Gaps and Opportunities**

## System Design and Development

### Overview – (including input/output display method)

A High-level overview of the software design focused on what the software is doing, not how

This section will include:

* A software data flow diagram
* Data input/output requirements
* Requirements of the logic layer
* Requirements of the physical layer
* Control loop behaviour
* Usage and purpose of feedback strategies

### Logic Layer

**Overview**

* Requirements and design breakdown of logic layer

**Module Representation**

* How individual modules and their variations are stored and manipulated within the system

**State Representation**

* How a state configuration is stored, analysed and manipulated within the system

**Priority Queue – (including heuristics)**

* How a priority queue of states operates within the system and how states are compared and priority is introduced to the queue
* What heuristics are used to compare states and assign them priorities

**Search Algorithm**

* Pseudo-code of the search algorithm
* Detailed explanation of how the algorithm works
* Include the diagram that shows tree expansion and trimming
* Include how the algorithm is modified to implement feedback strategies

### Physical Layer

**Overview**

* Requirements and design breakdown of physical layer

**Inverse Kinematics Module**

* Brief explanation of what inverse kinematics is
* Breakdown of analytical and iterative Jacobian methods for inverse kinematics and method chosen, including reasoning
* Explanation of position and orientation matrices and how they are used to find solutions
* Usage of the kinematics module for pose verification

**Manipulator Base Location Planning**

* Explain the moving arm used in the MOSAR project and how the software has been designed for use with a walking arm which requires a base location parameter when moving modules, however implemented with the stationary automata EVA arm making this function a currently unnecessary addition

**Motion Planning**

* Detail algorithm used for motion planning

**Failure Feedback**

* Detail the types of physical failures that occur and how that information is fed back to the logic layer (out of reach, collision, no base location available, etc)

### Feedback Strategies

**Failure Memory - (possibly out-of-scope but considered and researched)**

**Disallowing moves**

**Application of physical constraints**

## System Implementation

### Hardware Specifications

* Automata EVA arm specification and restrictions

### Software Specifications

* Implemented Software Structure
* Usage of python Modules (not copy module cause it sucks)

### Implementation Challenges – (efficiency and memory use)

## Testing and Results

### Testing Method – (timing/efficiency, varying inputs)

### Performance Metric – (failure rate, timing)

### Analysis of results

## Discussion

### Interpretation of results – (what results say about current system)

### Comparison to existing work

### Implications – (potential impact of work on the field)

## Planning and Time Management

### Project Management Procedures

### Project Management Reflection

### Risk Assessment

### Evolution of Project Plan

## Conclusion

## Further Work

* Making the program work in real-time through a control loop, allowing the program to continue running and working towards rearranging a satellite towards a solution even when blocks have been removed mid-program or there are other moving objects within the surroundings
* Support for multiple manipulator arms
* Support for clustering (moving multiple modules at once)
* Support for modules of different sizes

## References

## Appendix A - Initial Report

## Appendix B - Code