## 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

This section builds, and expands, on material previously included in the project Initial Report (see Appendix B)

## Background and Context

Space systems have rapidly developed in recent years, with a global drive to increase commercial availability. Current commercial systems designed under the limitation of mass and launch costs, are traditionally highly customized whole systems which consequently have very limited or no maintenance and repair capabilities. The number of ageing satellites is rapidly increasing and upon reaching end-of-life, are discarded through atmospheric deconstruction methods if possible, or left in orbit contributing to space debris build-up.

Technology to circumvent these conditions is not currently available and as such, the HORIZON 2020 EU-funded MOdular Spacecraft Assembly and Reconfiguration (MOSAR) project was launched to develop novel technologies that would allow standardising satellites and components [%1]. The modularisation and standardisation of space systems will benefit the European space industry by facilitating mass production of standard components and therefore decreasing assembly costs, reducing time between customer orders and commissioning in space, and allowing repair and upgrading of components directly in-orbit.

MOSAR primarily aims to produce on-orbit modular and reconfigurable satellites. At present the project has developed a demonstrator for re-configuring cubic modules to simulate the movement of modules through the use of a mobile robotic manipulator. Currently fixed instructions facilitating module mobility are sent to the manipulator from a software simulation on earth [%]; This research project aims to further develop the capabilities of the overall system by developing an algorithm to automate the module reconfiguration process, facilitating self-repair and self-assembly. Following development, this technology has the potential to facilitate the automated assembly of space systems and platforms directly in space, expanding the limitations currently imposed on the space industry.

## Project Objectives and Specification

This project intends to introduce the capability of autonomous assembly and reconfiguration of a modular space system by implementing a reconfiguration planning program made up of simple algorithms. This program, given the initial state and final state of a modular craft as parameters, will produce a list of commands to send to the mobile manipulator. These commands can then be used to autonomously rearrange modules on a spacecraft or space platform in operation. This program must consider the physical constraints placed on the system by the mobile manipulator present on the modular system, therefore this project will also strive to explore methods of introducing physical constraints into the planning program.

To accomplish the research goal, a functional planning program must be implemented and be demonstratable through software simulation. Though if time permits, there is the additional goal of demonstrating the planning program by physically re-configuring real modules in the lab through integrating the system with the available manipulator arm.

The above goals of producing, testing and demonstrating the system has been broken down into the following sub-objectives:

1. Development of a reconfiguration planning program to produce module movement instructions to send to a mobile manipulator. The program must take as arguments the initial and final state configuration and return an instruction set.
2. Further expansion of the reconfiguration planning program to take into consideration the physical constraints placed onto the modular system by the mobile manipulator present.
3. Development of a display function to produce reconfiguration slideshows/videos, allowing users to view a simulation of the modular system reconfiguration from start to finish.
4. Analysis of the system through testing a variety of inputs and recording the time taken to find a solution. This is to benchmark the system against current and future alternative systems.
5. Demonstration of the system through integration with the robot arm available in the laboratory.

## 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

## Report Structure

This document aims to provide a detailed report of the research and development completed during the Autonomous Re-Configuration of Modular Spacecraft with Manipulator Arm project. The report includes:

* Research on the state-of-the-art of modular reconfiguration and review of existing literature.
* Development of a detailed design to guide the implementation of the reconfiguration planning program.
* Description of the final implemented design.
* Analysis of the final implemented design and records of analysis results.
* Discussion into the results and their relevance to the field of study.
* Explanation of the project management approach and the evolution of the project plan throughout the duration of the project.
* Suggestions for further work to expand on completed work detailed in this report.

# Literature Review - around 3000 words

This section builds, and expands, on material previously included in the project Initial Report (see Appendix B)

# System Design and Development

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

The reconfiguration task and motion planner (TAMP) program will be designed around a python implementation as there is currently an existing python implementation for controlling the robot arm present in the lab [%][%].

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

* Critical analysis how well my product would work in certain applications given the obtained results

## Discussion

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

### Comparison to existing work

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

* Demonstration day feedback – usage in warehouses

## Planning and Time Management

### Project Management Procedures

### Project Management Reflection

### Risk Assessment

### Evolution of Project Plan

## Conclusion

* Refer back to objectives/specification in introduction

## 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. This would also allow the system to recover from temporary power failure pointed out in the literature review challenges section
* 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 (need to cite libraries used)