Debris in Space Autonomous Removal Mechanism (DISARM)

Design Document

Team Members

Project Manager: Kyle Watkins

Project Systems Engineer: Luca Rizza

Electronics System Lead: Michael Leard

Electronics System Supporting Engineers: Nouraldean El-Chariti, Ali Lebbar

Grappling System Lead: Daniel Soto

Grappling System Supporting Engineers: Laura Guziczek, Ali Lebbar, Davey Renoid

Control System Lead: Nouraldean El-Chariti

Control System Supporting Engineers: Laura Guziczek, Matthew Intriago, Michael Leard

Structure System Lead: Vincent Panichelli

Structure System Supporting Engineers: Davey Renoid, Ali Lebbar, Daniel Soto

Client: Dr. Markus Wilde

Faculty Advisor: Dr. Silaghi

The trade studies conducted started by breaking down the initiative to capture space debris into that of either a contact or non-contact mechanism. This led to a unanimous group decision of opting for the contact method as this method is not limited to testing at Florida Tech facilities and labs. The next approach taken was to research welding to achieve contact with the debris. Three welding mechanisms were chosen: stud welding, capacitor discharge welding and cold welding. Currently, a trade study between capacitor discharge welding and cold welding are being conducted and will be carried through onto next semester. The control subsystem shall administer the autonomous welding process of either method chosen. This process shall be coded into the mainframe of the welding mechanism, which will enforce autonomous welding upon contact with debris. The position of the debris shall be established and shall provide and maintain location feed in real time by sensors. Current sensor being researched is the retro-reflective sensor.

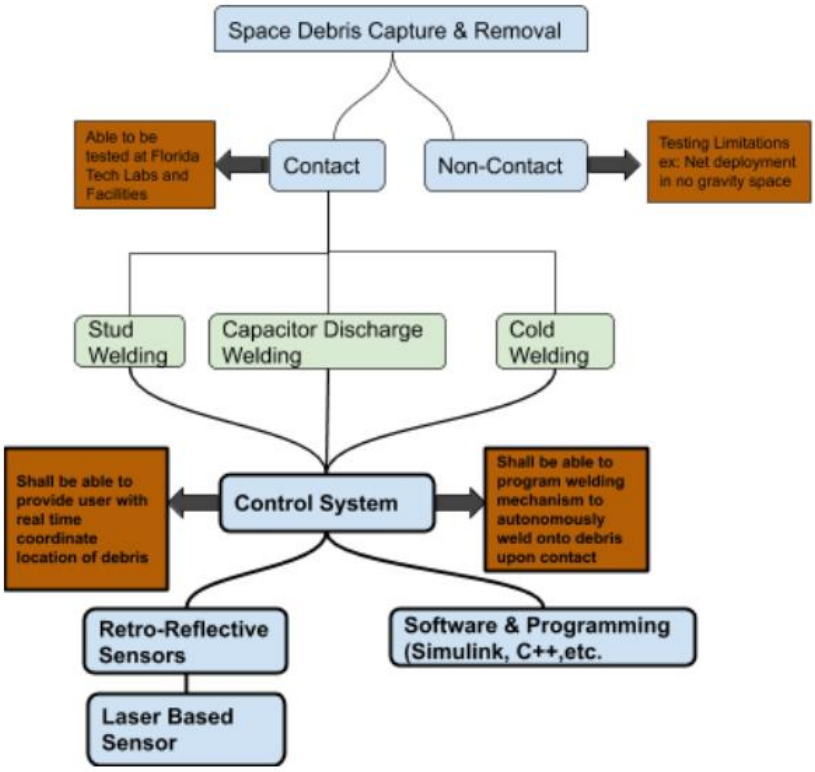

Figure 1: Diagram of project trade studies

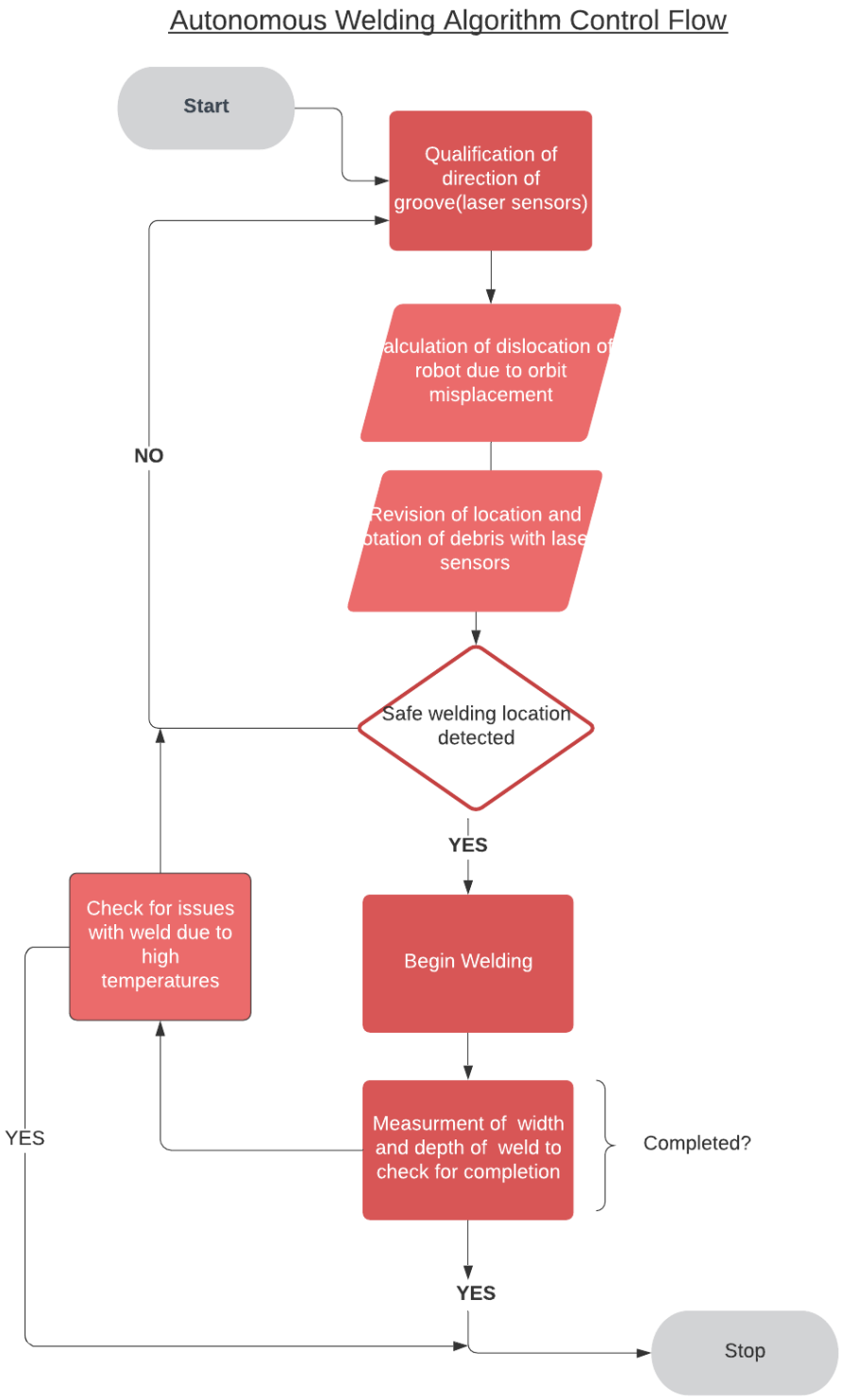

Figure 2: UML Diagram of Algorithm Control Flow

Algorithm Control Flow Breakdown

1) **Qualification of direction of groove**

   The xyz rotation axis of the DISARM device is calculated, to align with welding points located on target debris through the use of sensors.

2) **Calculation of dislocation of robot due to orbit displacement**

   Once the potential groove locations are found, check for any displacement of the actual system and if found allocate the calculated welding points to take into account this displacement, otherwise continue with the process.

3) **Revision of location and rotation of debris**

   Once again verify if the variables of momentum, velocity, and location of debris have changed, and make any adjustments with groove locations taking into account the changes.

4) **Safe welding location detecte**d
5) **Begin Welding**

   During welding, system will be actively relaying information back to the GUI with completion percentage of weld, live location of target debris, and related variables to be displayed for the user.

6) **Measurement of width and depth of weld to check for completion**

   After welding has been executed, revise depth and width of weld to make sure that the system has successfully captured the debris without the potential of debris breaking loose and creating more debris.

7) **Checking for issues caused by high temperature**s

   If there is any overheating that is putting stress on the system during or after the weld, stop the process and now the user can manually fix the weld or cancel the weld.

8) **End**

Mock-Up Interface

Figure 3: Mock-Up User-Interface

Even though the welding process will autonomously be executed, the system will display welding information for the user to see the progress of the capture, as well as any necessary variables to make decisions on continuing the weld or cancelling. The variables will include the live location of the target debris, the velocity and momentum, the temperatures on the system, which will then be used to calculate the strain on the system. In the case that the system doesn't automatically shut-off when encountering an error, the user will be able to cancel the weld using the button displayed above. The user-interface will be coded using Visual Basic combined with HTML, CSS, and Javascript using Visual Studio tools.