Space Jockey

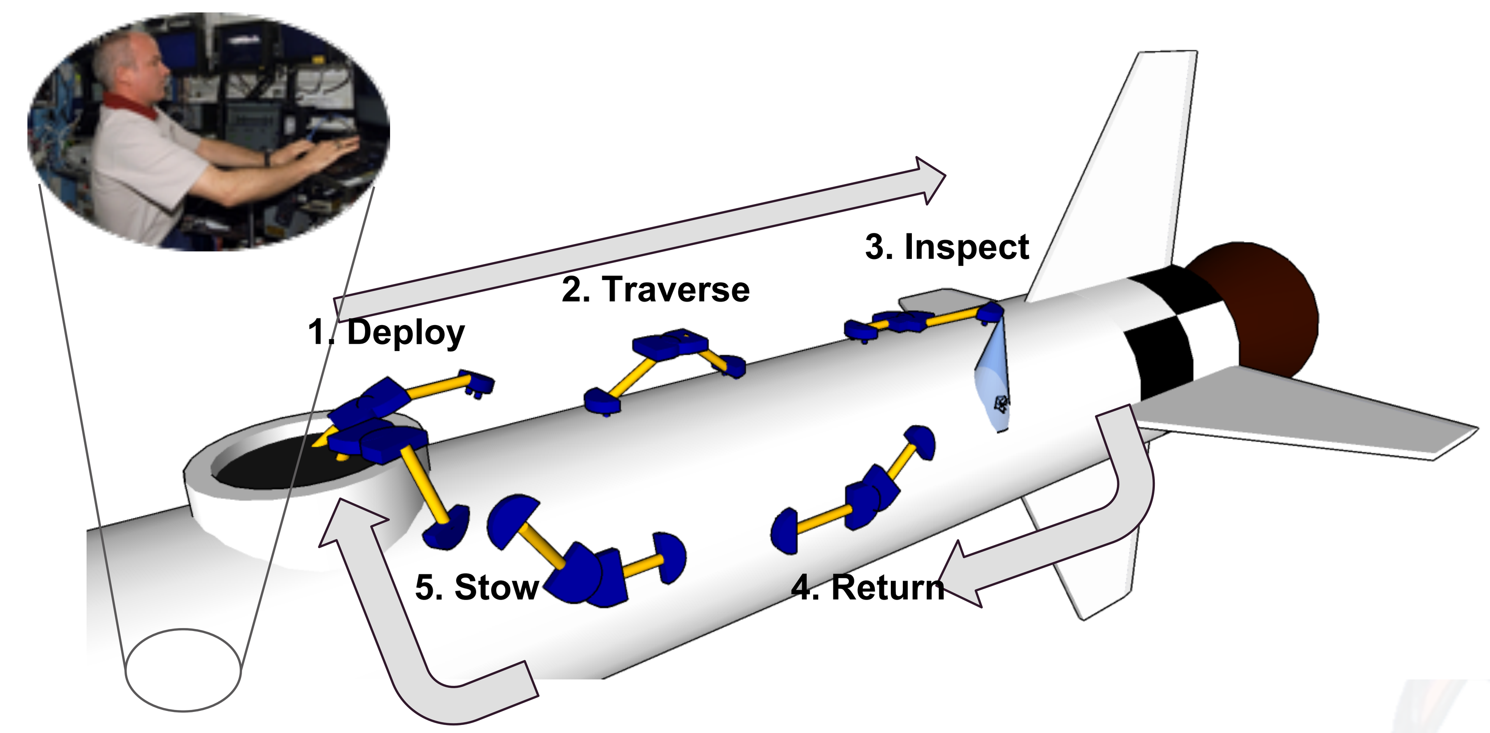
A Mobile Robot Platform for Spacecraft Exterior Inspection & Maintenance

Project Description

Even under typical operations, ambient hazards such as radiation and micro-meteorites slowly degrade the surfaces and components of spacecraft over time. In some cases, even reasonable wear and tear can lead to loss of life if not detected and addressed, as evidenced by the Space Shuttle Columbia Disaster in 2003. Manual inspections during astronaut EVAs are one viable solution to the problem, but they increase mission costs and astronaut hazards.

We believe this problem can be solved using a lightweight mobile robot which is deployed onto the surface of a spacecraft, and traverses the hull, performing an autonomous inspection of the surface. Mission Control is then notified of any damage detected during the inspection, and astronauts may be dispatched on a more limited mission to enact repairs.

Use Case

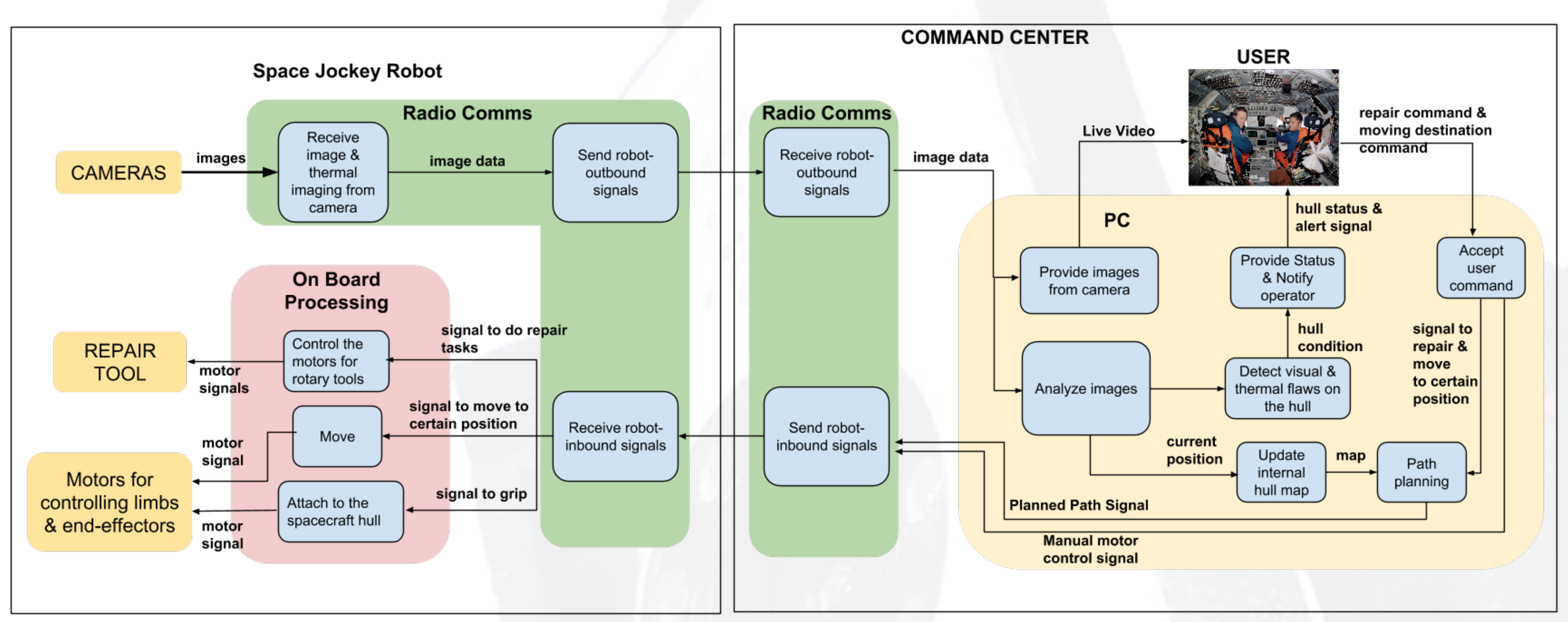


System Requirements

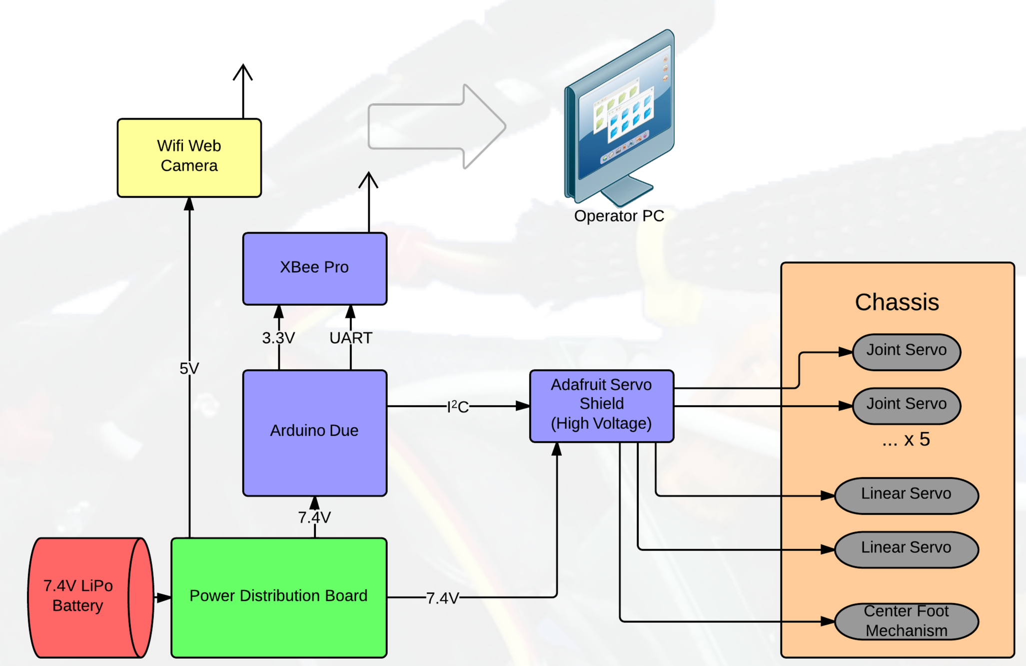
Functional
Maintain attachment to the hull TPM: Supports own weight on a 1G vertical surface
Inspect the surface at a reasonable speed TPM: 6 square meters per hour
Provide visual feedback of the robot state and inspection data
Autonomous identification of surface flaws TPM: 90% Detection rate, with less than 2 false positives

Non-Functional
System must be battery powered and Wirelessly controlled
System can be validated in terrestrial conditions
As light-weight as possible to reduce launch requirements TPM: <10kg
Minimal volume for efficient storage TPM: 8 Cubesats (10cm cube)

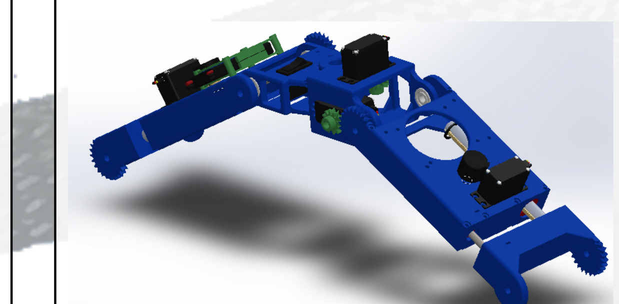
Functional Architecture

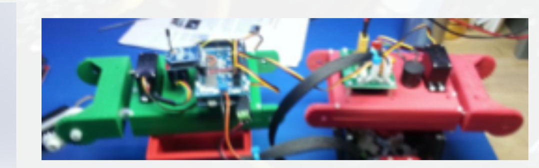


Physical Architecture

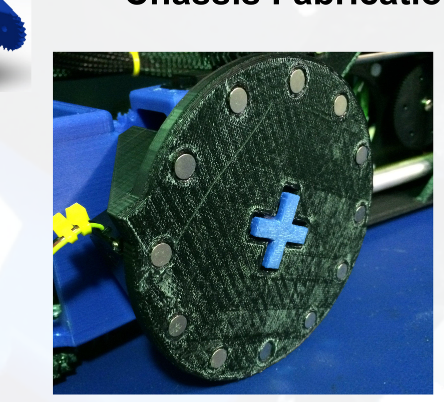


System Design

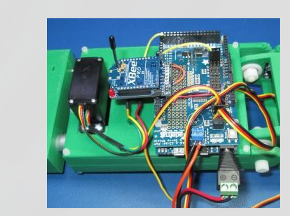

CAD Design

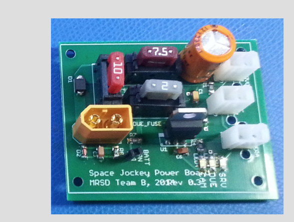

Chassis Fabrication

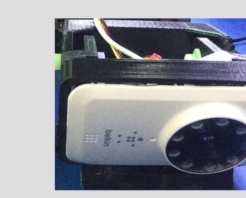

Prototype Magnetic Foot Design

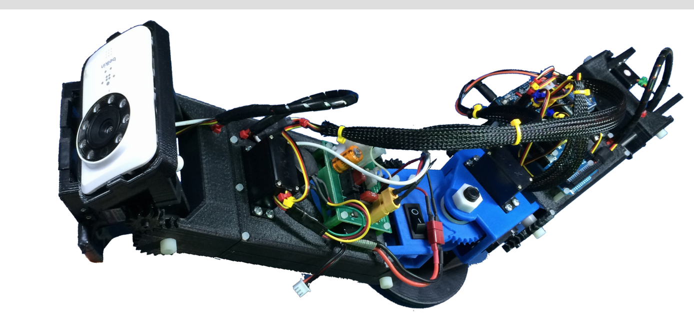

Magnetic foot surface and detachment mechanism

Electrical Subsystems

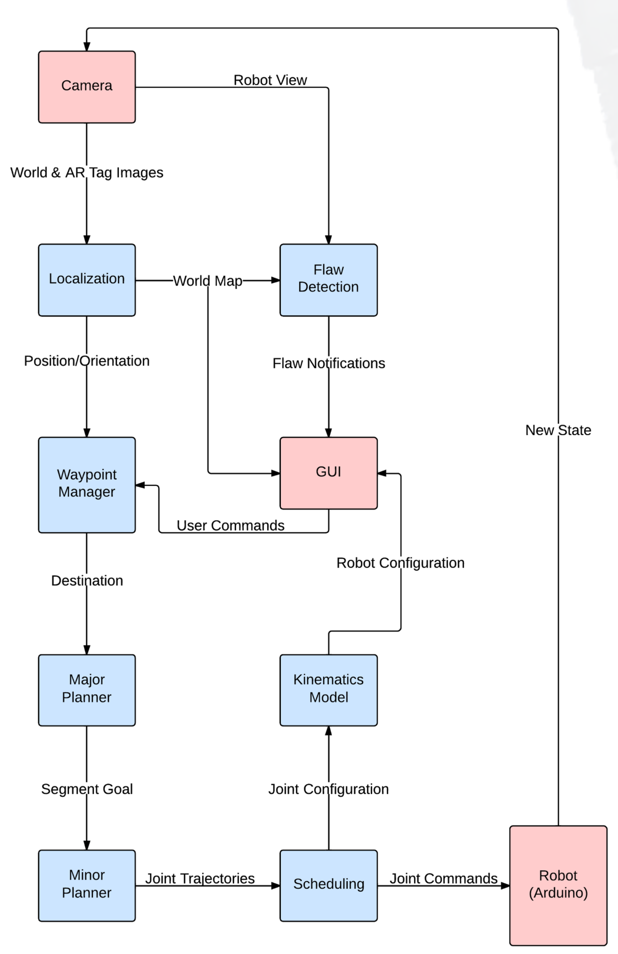

Avionics


Power Board

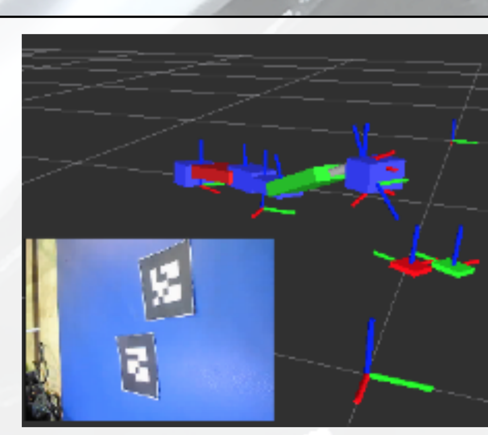

Wifi Camera


Completely Assembled Robot

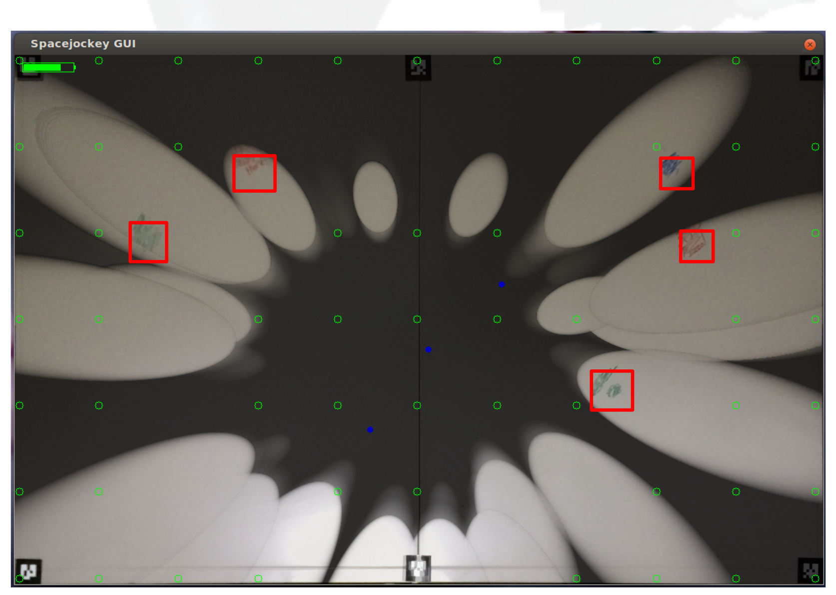
Software Design



Software Architecture



AR Tag Localization markers in RVIZ

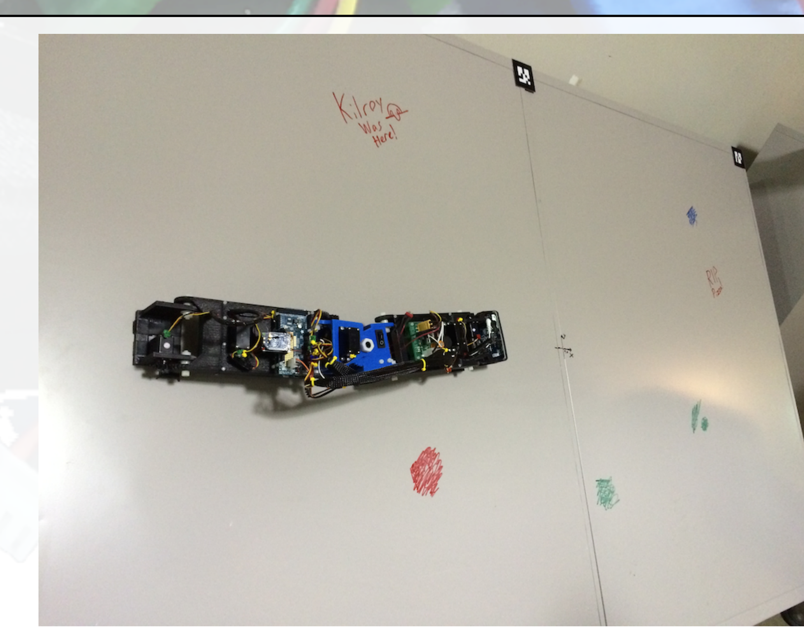


GUI screenshot showing flaw notifications
(simulated data)

Results

Test Procedure

The robot is powered on, and an inspection region is specified using the operator interface. The robot then navigates and inspects a 3 meter^2 vertical test environment, notifying operator of defects encountered. Notifications will be displayed as a bounding box encompassing the entire flaw in the operator GUI.



Test Results

✘ 90% success rate in identifying defects, with fewer than 2 false positives
- Current flaw detection implementation is overzealous, and sensitive to lighting conditions, producing false positives

✔ The robot must not detach from surface and fall at any point during testing.
- 60 degree final test environment

✔ The robot must inspect at a rate of 6 meters^2 per hour.
✔ The robot must perform autonomously after given an inspection command
✔ The robot has must weigh less than 10 Kg (final weight: 1.65 Kg)
✔ The robot must operate wirelessly.

Future Work

- Improve the attachment foot by using space-ready sticky material to enable attachment to non-magnetic material
- Improve the flaw detection algorithm using machine learning method so that it can be more accurate
- Add obstacle avoidance and planning support for non-planar surfaces, also try on a ceiling as a test environment
- Improve localization and joint position accuracy
- Add simple manipulation function to tighten a fastener using a rotary tool
- Add plane transitions function with at least 60-degree transitions

Lessons Learned

Project Management

- Have regular meetings at least twice a week
- Clearly define tasks for each team member at every meeting.
- Use spiral model for systems engineering.
- Parallelize tasks in each iteration cycle to maximize team productivity.
- Group work sessions facilitate communication and help people stay on track.

Design/Fabrication

- Develop GUI design early to pin down desired functionality at the onset.
- Keep all source code and design docs under version control, for safety and communication.
- Use rapid prototyping techniques to create early versions of all parts to check fits and tolerances before fabricating/machining a final part.
- Add a 2X cost and time buffer to the project to assure sufficient project time.
- Always maintain a separate "production" code branch in git with the last stable build of all demo-ready code, regardless of non-critical bugs
- Always order spare parts for critical components.

Acknowledgments

We would like to thank John Dolan, Hagen Schempf, and Dimitrios Apostolopoulos for their guidance and feedback.
We would also like to thank Sumitra Gopal for ordering parts and supplies.

MRSD Team B, 2013-2014
Brian Boyle - Nathaniel Chapman - Ardya Dipta - Songjie Zhong

Carnegie Mellon
THE ROBOTICS INSTITUTE