# Conceptual Design Document

# DRAGON

**D**eployed **R**F **A**ntennas for **G**PS-denied **O**ptimization and Environmental **N**avigation

Monday 1st October, 2018

| | |
|---|---|
| Name: Professor Nisar Ahmed<br>Email: nisar.ahmed@colorado.edu<br>Phone: 303-492-0286 | Name: Steve McGuire<br>Email: stephen.mcguire@colorado.edu |

## Team Members

| | |
|---|---|
| Name: Dawson Beatty<br>Email: Dawson.Beatty@colorado.edu<br>Phone: 303-906-6702 | Name: Kyle Nieukirk<br>Email: Kyle.Nieukirk@colorado.edu<br>Phone: 303-717-1345 |
| Name: Christian Carmack<br>Email: Christian.Carmack@colorado.edu<br>Phone: 719-551-9118 | Name: Virginia Nystrom<br>Email: Virginia.Nystrom@colorado.edu<br>Phone: 303-829-8794 |
| Name: Jeremy Fie<br>Email: jefi2765@colorado.edu<br>Phone: 303-808-8152 | Name: Amanda Siirola<br>Email: Amanda.Siirola@colorado.edu<br>Phone: 303-913-7923 |
| Name: Chris Greer<br>Email: James.Greer@colorado.edu<br>Phone: 303-877-9798 | Name: Ryan Stewart<br>Email: Ryan.J.Stewart@colorado.edu<br>Phone: 720-757-4003 |
| Name: Jack Maydan<br>Email: Jack.Maydan@colorado.edu<br>Phone: 203-617-5128 | Name: Luke Tafur<br>Email: Joshua.Tafur@colorado.edu<br>Phone: 214-476-1368 |
| Name: Ross Kloetzel<br>Email: Ross.Kloetzel@colorado.edu<br>Phone: 808-348-7405 | Name: Ivan Yurkin<br>Email: Ivan.Yurkin@colorado.edu<br>Phone: 303-881-0922 |

# Table of Contents

# Acronyms

**DW1000** Decawave 1000 RF module

**I2C** Inter-Integrated Circuit

**SPI** Serial Peripheral Interface

**SW** Software

**RN** Rover Navigation

**PD** Pod Deployment

**GNSS** Global Navigation Satellite System (GPS/GLONASS/BeiDou)

**MIP** Mission Information Package: Customer provided data which defines the mission including maps/waypoints/obstacles/science values/timing

**MOX** Metal Oxide

**VOC** Volatile Organic Compound

**TVOC** Total Volatile Organic Compound

**MEMS** Micro-Electro-Mechanical System

# 1. Project Description

## 1.1. Purpose

The field of autonomous navigation is prevalent in a variety of environments, from Martian deserts to battlefield urban canyons; as such, the availability of GPS is never guaranteed. Presently, autonomous navigation without GPS is limited to small scale, high error, inertial/odometry dead-reckoning measurements, or to highly complicated visual-inertial systems using stereoscopy. The DRAGON team will design a system that enables an unmanned rover to accurately and autonomously navigate complex terrain, which will be done by replacing GPS with dispersed RF-Localization beacon pods, henceforth called Pods.

Based on a customer-supplied 'Mission Information Package' (MIP), the rover will attempt to follow a software generated path relative to its last known position using inertial measurements; these position estimates are subject to unacceptable error and drift. The system will use said pods to provide a second, more accurate, position estimate as feedback for correction. Ranging is done by measuring transmission time between the pods and the rover, a functionality provided by customer-dictated hardware. In order for the pods to triangulate the rover they must be deployed at some distance away from the rover, and because everything is relative to the last known position, their deployment accuracy is imperative. The primary mission is to demonstrate a rover can navigate autonomously without GPS by deploying pods along its path which assist in position estimates.

In the mindset of a resource-limited mission, it is natural to add a secondary functionality to the deployed pods because they will remain in the environment permanently, and because they have the benefit of being able to reach rover-inaccessible locations. This adds two major aspects to the overall project. First, the pods must be able to collect and transmit environmental data; the type of data is not important, but demonstrating the functionality is. Second, software must now consider whether it is more advantageous to deploy a pod to a location where it helps the rover navigate, or to deploy somewhere that is of scientifically high interest; software will make this decision based on MIP provided data.

The DRAGON team will provide a fully autonomous method to improve unmanned navigation in complex GPS-denied environments by designing the pods, deployment mechanism, all software for placement, and interface to the rover, with the secondary benefit being that the pods will remain in the environment for future navigation use while collecting intermittent scientific data.

## 1.2. Project Objectives

The levels of success (LOS) have been broken into categories for each major element of project DRAGON. Each category has a minimum "Level 1," which are defined as the most basic criteria of success for the project. Level 2 is a more intermediate set of accomplishments which describe the customer's excepted outcome, while Level 3 defines the total completion of customer specified objectives. Level 4 consists of difficult goals defined by the customer to be completed if time and budget permit. Each level of success acts as a funnel towards the next objective in each project element category. Lower levels represent simplifications to the project if a requirement goal becomes unachievable with the given resources. Table 1 describes the project elements in each row and the level of success to be achieved in the columns. A project element is referenced in the table by its abbreviation, while a previous level is compressed to L#. For example, referring to the first level of success for rover navigation would be abbreviated as RN-L1. In addition, square-bracketed numbers, such as [1], reference additional information at the bottom of the table.

In addition, the DRAGON team has intent to develop a Ground Segment (GS) that is not directly demonstrated in the LOS. The GS will have two development stages. The first will have only remote Emergency Power Off (EPO) and rover state indication. All data will be stored on the rover and analyzed in post. This is implied in the 1st and 2nd LOS, in the way that no test can be performed safely without EPO capability, and post-test data analysis is sufficient for proof of concept and well understood by the team. The second stage of development for the GS, implied in SW-L4, will include online (real-time) telemetering of various rover data such as location and operational monitoring parameters. This data will be received and displayed in a GUI such that data can be presented live during a test for monitoring.

**Table 1. Project DRAGON Success Criteria**

| **Project Elements** | **Level 1** | **Level 2** | **Level 3** | **Level 4** |
|---|---|---|---|---|
| **Rover Navigation (RN)** | The rover can locate itself within a deployed pod array to an accuracy of 1m using RF localization, after a short (15m TBR) pilot controlled traversal. [1] | The rover autonomously completes RN-L1 using computer closed loop location error correction. Correction uses relative position data from pods to correct dead reckoning to 1m. [1] | Rover autonomously maneuvers along the SW-L1 path. Rover uses RN-L2 to reach up to 10 waypoints within 1m tolerance. | RN-L3 and reaches waypoints within user specified times and within 30 seconds (TBR) of user specified times. |
| **Pod Deployment (PD)** | Pods are hand placed, in analytically [2] predetermined locations. | Pod deployment system on the rover has a range capability of beyond 10m. Pods are deployed within a 1m radius of intended absolute location. User inputs trajectory. | Software provides desired pod location. Deployment system uses rover-based software to calculate trajectories and commands to deploy. | |
| **Scientific Data (SD)** | | Pods functionally collect environmental data. | SD-L2 and the rover is able to selectively request/receive data from pods. [3] Pods collect environmental data at 5 minute frequency. | The pods are able to communicate amongst themselves using a mesh network, such that only communicating to one pod enables access to all deployed pods data. |
| **Pod (PO)** | Pods will continuously broadcast RF-localization signals using required presupplied beacons. Broadcast will be received and recorded by the rover (via RN-L1). | Pods can transmit environmental science data, in addition to RF-localization data, to rover. Pod internal components can function after deployment stresses. | Pods demonstrate ability to toggle between low-power data recording mode (20mA TBR) and high-powered transmission mode (200mA TBR). | Pods records environmental observations for a 2 hour duration in low-power mode after navigation completion. |
| **Software (SW)** | Software can determine a path for the rover to get to up to 10 waypoints around obstacles and hazards, using terrain map and user specified waypoints. | SW-L1 and software can determine pod placement which will provide online correction assistance to rover along SW-L1 defined path. | SW-L2 and software can determine pod placement which will balance pods dual function of assisting rover, and collecting science data. [4] | Ground segment provides online [5] communication with rover to display rover position and system-health-monitoring parameters with a 10Hz (TBR) update rate |

[1] GPS data on rover and pods will be captured at all times for validation testing only; it will never be used by pod or rover, except for rover starting position known. It will be used as truth to determine accuracy.

[2] Analytical determination of pod location is an indicator that the team has developed a by hand calculation/algorithm for determining pod location to assist rover on SW-L1 path.

[3] Selectively is with respect to information such as which pods host the most valuable environmental data (based on heat map) at a specified time.

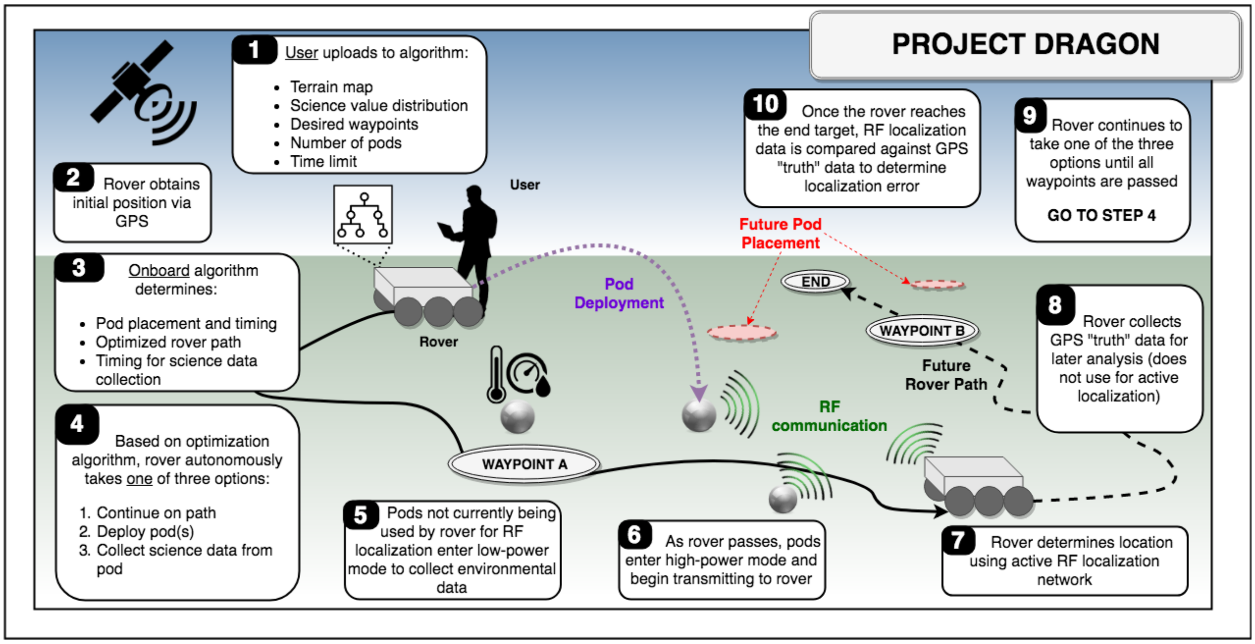
[4] Balance will be based on weighting algorithm (MIP defined weights) which weights ability to assist rover and value of science (value of science defined by pre-supplied heat map) to determine pod location.

[5] Online is defined as near real-time data relay between rover and ground segment.

## 1.3. Concept of Operations

The CONOPS in Figure 1 shows how the system will operate in its test environment, one similar in size and terrain to CU Boulder South or the Business field. Using the specified initial conditions from the MIP, the onboard path-finding algorithm will decide the desired path and its next steps without human intervention (see Step 4). The onboard algorithm determines if the rover should continue to drive on the path, deploy pod(s), or collect science data from previously deployed pods. This process is a loop; the rover will continue to do this step until the end of the test. As the rover traverses along the path, it uses the active RF localization network to determine its location. In addition, the rover collects GPS truth data which is used to compare against the RF positioning in post; GPS data is not used by the onboard navigation algorithm.

After deployment, the pods alternate between a high-power and low-power mode on a cyclical schedule. When the pods wake-up from low-power mode (i.e. hibernation) and enter high-power mode, they either transmit data science and/or positioning data to the passing rover, or collect environmental data and then return to low-power mode.



1 User uploads to algorithm:
- Terrain map
- Science value distribution
- Desired waypoints
- Number of pods
- Time limit

2 Rover obtains initial position via GPS

3 Onboard algorithm determines:
- Pod placement and timing
- Optimized rover path
- Timing for science data collection

4 Based on optimization algorithm, rover autonomously takes one of three options:
1. Continue on path
2. Deploy pod(s)
3. Collect science data from pod

5 Pods not currently being used by rover for RF localization enter low-power mode to collect environmental data

6 As rover passes, pods enter high-power mode and begin transmitting to rover

7 Rover determines location using active RF localization network

8 Rover collects GPS "truth" data for later analysis (does not use for active localization)

9 Rover continues to take one of the three options until all waypoints are passed

GO TO STEP 4

10 Once the rover reaches the end target, RF localization data is compared against GPS "truth" data to determine localization error

**Figure 1. Concept of operations of mission testing**

## 1.4. Functional Block Diagram

The Functional Block Diagram (FBD) in Figure 2 illustrates the connections between the rover, pods, and user. The customer-provided rover from Clearpath Robotics includes two on-board computers, a battery, an Inertial Measurement Unit (IMU), odometer, and GPS antenna. The low-level computer will control the actuators for the rover's wheels, while the high-level computer will host the software for navigation and deployment targeting. The rover will host the team-designed pod deployment mechanism and a customer-provided RF beacon. The only physical interaction between rover and user occurs only when the user inputs the initial conditions. A ground segment will provide online (real-time) information about the state of the system and allow for emergency termination of the test if necessary. GPS data will be collected and used for post-test verification.

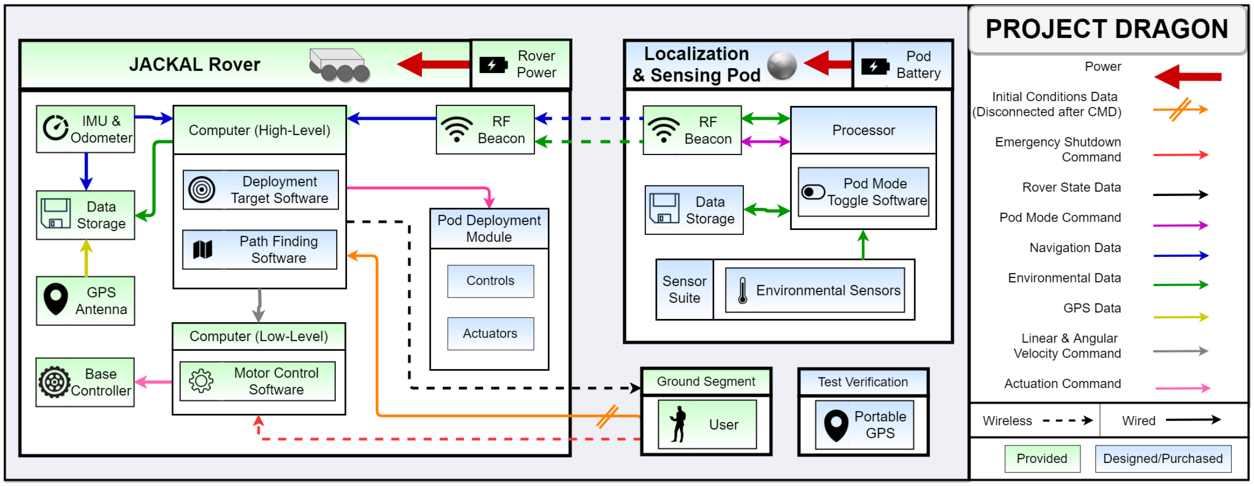


**Figure 2. Functional Block Diagram showing major system components**

### 1.5. Mission Requirements (Functional Requirements)

**Table 2. Mission requirements**

| | |
|---|---|
| M1 | Rover shall autonomously navigate along software generated path within 1m accuracy using RF-Localization Beacon correction to inertial navigation |
| M2 | The rover shall estimate its absolute position |
| M3 | The deployment mechanism shall have capability to deploy pods to software defined locations |
| M4 | The rover and ground inputs shall prevent damage to all hardware systems |
| M5 | The pods shall function as RF navigation beacons and as environmental data monitors, to the rover |
| M6 | The pods shall be able to function as a long-term deployable environmental data monitor |
| M7 | The team shall verify absolute navigation ability |
| M8 | The team shall use the customer-provided hardware |

## 2. Design Requirements

Below are the functional and derived design requirements from level 1-3. The team does have lower level requirements which were excluded for brevity. Phrasing such as “inspection”, ‘verify by observation’, or ‘verify by existence’ are simple descriptors for easily verified requirements such as confirming that a part number matches the requirement, or a physical observation that certain hardware is the correct desired hardware, or test conductors that observe a test and can easily report results by visual observation of the test.

| Identifier | Identifier | Identifier | Title | Motivation | Validation |
|---|---|---|---|---|---|
| M1 | | | **Rover shall autonomously navigate along software generated path within 1m accuracy using RF-Localization Beacon correction to inertial navigation** | | *Full System Demonstration* |
| | G1.1 | | G - Rover location accuracy shall be determined with respect to GNSS truth. GNSS data not used by rover | *Serves to provide the highest location accuracy, GNSS data will be used as the location truth for validating for DRAGON's navigation methods* | *Vendor provided data that indicates which constellations are supported, user collected 'connection log' can also ensure full usage of GLONASS, GPS, and BEIDOU.* |
| | | G1.1.1 | GNSS accuracy shall be 1m or less | *Truth data must have greater accuracy than the data it's going to be used to validate* | *Vendor certification, supplement with landmark test* |
| | S1.2 | | S - Rover shall have feedback control to physically correct for path deviance | *Must be able to correct rover's motion to maintain path* | *Full System Demonstration,*<br>*Sub-testing via:*<br>*Rover autonomous motion with intentional disturbance, demonstrate correction to disturbance, can be verified visually or in post.*<br>*Rover autonomous motion with large beacon correction input, demonstrate correction to error, can be verified visually because of large devation in rover motion.* |
| | | S1.2.1 | S - Software shall be capable of controlling rover's physical motion through wheel actuators/ROS nodes | *Control over rover motion is essential to correcting its navigation, using the built in ROS nodes is the most concise solution.* | *Software can command a planned route (closing the square) and test conductors can physically measure error in path deviance or final position.* |
| | | S1.2.2 | S - Software shall have closed loop correction of inertial/odometry location measurement error by using pod RF-Localization | *There must be a feedback loop for correcting error in rover's motion in order to ensure it's accurate traversal.* | *Full system demonstration* |
| | S1.3 | | S - Software shall generate a path through the terrain to reach up to 10 waypoints | *The rover needs to follow some path to hit MIP desired waypoints, and must stay safe during traversal* | *Software demonstration of path, path hits waypoints and avoids obstacles, verified by insepction of environment or of terrain/obstcacle map.* |
| | | S1.3.1 | S - Software shall record initial GNSS position of rover | *An initial point of reference is required for inertial navigation, and to indicate starting position on MIP maps* | *GNSS subsystem demonstration, listed as consistent prelim step in every test procedure.* |
| | | S1.3.2 | S - SW shall be capable of ingesting MIP, data types must be compatible | *MIP defines the mission, and so the software must be compatible in order to use it.* | *Software demonstration of interaction with MIP* |
| | | S1.3.3 | S - Software path shall meet MIP time requirements, 'on-time, on-target' for each waypoint | *Per customer, mission success is not clear if the rover arrives late or early to destinations* | *Test Conductor can observe test and record timing, or post test location data can be compared against desired results* |
| M2 | | | **The rover shall estimate its absolute position** | *The rover's ability to estimate its position in the environment without GNSS is the crux of this project. Without known position FR 1.0 is impossible* | *System demonstration* |
| | S2.1 | | S - The software shall combine RF-Localization and inertial/odometry position estimates in order to enhance position estimation accuracy | *Using these two data types in combination is the only way to correct inertial/odometry position estimates* | *Partial System Demonstration* |
| | | S2.1.1 | S - Software shall be able to determine position relative to any pod in range via RF-Localization | *The ranging measurement between pod and rover is what's used for error feedback, and is an essential functionality.* | *Partial System Demonstration* |
| | | S2.1.2 | S - The rover shall estimate position using onboard odometer and IMU | *This measurement is considered high error, but serves as a baseline to correct.* | *Rover only demonstration, this functionality is provided by vendor and well demonstrated.* |
| | | S2.1.3 | S - The ingest of position data shall be continuous, and the combination of position estimates shall be continuous | *This functionality should not be jerky, inconsistent, or piecewise, otherwise this is indicative of a poorly functioning control system.* | *System demonstration subject to customer approval of functionality.* |
| M3 | | | **The deployment mech shall have capability to deploy pods to software defined locations** | *The pod's ability to provide RF-Localization data to the rover is entirely dependent upon its location in the test environment* | *Full System Demonstration* |
| | S3.1 | | S - Software shall determine pod location for deployment, weighted between science value and effective RF-Localization, per MIP weight specification | *This is the crux of the project, knowing where to place the pods based on terrain and env data value is necessary to enable accurate navigation and also maximize env data collection.* | *Full software demonstration of ingesting MIP, and returning pod locations. MIP can be selected to derive obvious results and confirm that software will generate expected results. A scale system test is likely required for total verification that pod locations are correct.* |
| | | S3.1.1 | S - Software can determine pod location for placement for most effective ranging | *Software must understand the environment and determine where to place pods for most effective localization and ranging data.* | *Software demonstration* |
| | | S3.1.2 | S - Software can determine pod location for placement for for most valuable environmental data. | *Software must understand the environment and determine where to place pods for most valuable env. science data.* | *Software demonstration* |
| | | S3.1.3 | S - Software shall incorporate terrain obstacles into deployment location selection to avoid collisions with obstacles (tree, wall) | *Must avoid deploying pods into obstacles which will cause damage or inaccuracy.* | *Software demonstration* |

| Identifier | Identifier | Identifier | Title | Motivation | Validation |
|---|---|---|---|---|---|
| | | S3.1.4 | S - Software has a weighting algorithm, and method for combining 3.3.1, 3.3.2, and 3.3.3 into a valid placement location | *In order to meet highest level of success, the system will need to be capable of selecting locations which serve both purposes for pod deployment.* | *Software demonstration* |
| | D3.2 | | D - DM shall have the following range: No less than 5m, at least 10m, no more than 20m | *This range was determined by customer as necessary for system function* | *Independent deployment mechanism demonstration at each required range.* |
| | D3.3 | | D - DM shall have deployed pods land within 1m radius accuracy of the SW commanded location | *The accuracy of the pod's location in space directly transfers to the accuracy of the rover's position in space, error must be minimized* | *Multiple deployment tests with the same target, standard deviation demonstrated to be within bound* |
| | | P3.3.1 | **Sub to change** P - Deployed pod accuracy shall be determined with respect to GNSS truth. GNSS data not used by pods | *Subject to change, GNSS truth is an excellent method of verifying a pod's location after deployment, other methods are being considered.* | |
| | D3.4 | | D - DM shall have interface with rover SW for commanding deployment mechanism to deploy pods | *The DM will 'piggyback' on the rover, and must be commandable by the rover for truly autonomous operation.* | *System demonstration of rover ability to command a pod deployment* |
| | D3.5 | | D - DM shall be capable of deploying 10 pods within a 20 minute (TBR) duration | *Many pods will be deployed during any given mission and the 'reload time' cannot obstruct the overall mission* | *Demonstration of DM's total deploy-reload-deploy process* |
| | | D3.5.1 | D - DM shall be capable of reloading and deploying a new pod every 2 minutes | *There is an expected 20 minute mission duration, and the more pods deployed the more accurate the data, so a fast reload rate is desirable.* | *DM Subsystem demonstration.* |
| | D3.6 | | D - DM shall withstand stresses incurred during rover motion | *The entire DM 'piggybacks' on the rover, and therefore must not break during rover traversal.* | *Verified before production by analysis and inclusion of robust FOS. Potential 'vibe testing' can be performed, but likely not needed.* |
| | D3.7 | | D - DM shall not interfere with rover's GNSS or pod communication | *The DM cannot obstruct rover functionality by blocking communications* | *Demonstration of GNSS recording and pod communication of fully integrated system* |
| M4 | | | **The rover and ground inputs shall prevent damage to all hardware systems** | *Hardware is expensive and keeping it safe is essential to mission success.* | *Pre-test analysis, include robust FOS on any hazardous action/system.* |
| | D4.1 | | D - Rover shall be uninhibited by deployment module | *The rover must maintain its manufacturer designed ability to traverse complex terrain* | *This is validated by not exceeded the rover's mass specification, not altering is CG significantly.*<br>*Deployment mechanism's transfer of momentum to the rover will be calculated in pre-analysis, and verified post construction before installation.* |
| | | D4.1.1 | D - DM and mounting interface shall not modify Jackal's center of gravity to a point where it cannot sustain 30 degree slope in any orientation. | *The rover has ability to traverse large hills or 'rough terrain' and that functionality should be maintained.* | *Pre-test analysis, robust FOS. Can also measure CG of installed system before functional testing.* |
| | | D4.1.2 | D - Module and pods shall weigh less than 20kg together | *Maximum payload of rover stated by vendor.* | *Pre-test weigh in, weigh in also before installation.* |
| | G4.2 | | G - Rover shall have a two-level safety system to enable emergency takeover and emergency power off remotely | *To prevent runaway scenarios, and to prevent system damage* | *Demonstrate controller operation, and controller takeover ability* |
| | | G4.2.1 | G - Vehicle shall have remote 'kill switch', as in a hard shutdown | *As a backup to an emergency takeover, a more draconian kill switch shall exist.* | *Demonstrate takeover and kill capability in safe environment, such as interupting the software closing the square test.* |
| | | G4.2.2 | G - Vehicle shall have remote 'torque off' which keeps rover power, as in a soft shutdown | *As a less draconian response, this will prevent system damage or runaway, but enable data to be recovered and communication lines to stay open.* | *Demonstrate takeover and all stop capability in safe environment, such as interupting the software closing the square test.* |
| | D4.3 | | D - DM shall have a mechanical safety inhibit | *DM may have stored potential energy, it is desirable to prevent its release for safety* | *Analysis of expected stresses of a full strength deployment, inhibit must sustain stress and prevent any projectile or debris motion. Can test by demonstration.* |
| | D4.4 | | D - DM shall have a remote safety inhibit, such as an arm/disarm system, to enable safe approach to rover | *DM's deployment may be hazardous, remote disable is desirable for safety* | *Demonstration of remote shutdown during autonomous operation.* |
| | D4.5 | | D - DM shall have indicated keep out range/FOV | *Making a prominent indication of dangerous zones is paramount for safety.* | *Physical indicators on rover, analysis of FOV performed before system test.* |
| M5 | | | **The pods shall function as RF navigation beacons and as environmental data monitors, to the rover** | *Pod functionality is the only way for the rover to correct inertial traversals.* | *Full System Demonstration* |
| | P5.1 | | P - Pods are deployable and compatible with DM | *The pods cannot obstruct the DM functionality for needed range/accuracy* | *DM demonstration with shell only and mass simulator* |
| | | P5.1.1 | P - Pods shall have under .75kg mass TBR | *Must not exceed DM capability* | *Weigh in periodically during development, and at final build pre-deploy.* |
| | | P5.1.2 | P - Pods shall have diametric dimensions less than 3.5in, and total length that shall not impair DM reload capability. | *Must not exceed DM capability* | *Measurement periodically during development, and of final build pre-deploy.* |

| Identifier | Identifier | Identifier | Title | Motivation | Validation |
|---|---|---|---|---|---|
|  | P5.2 |  | P - The pods shall communicate data to the rover and amongst themselves | *The pods must share RF-Localization data and environmental data for mission success* | *Subsystem demonstration of communication between pod-rover and pod-pod. Requries test to confirm data receiption, processing/extraction of relevant info (aka turn RF time into a ranging distance).* |
|  |  | P5.2.1 | P - Pods shall have localization data transmit capability over DW1000 module (DW1001 accepted) | *Customer requirement* | *Verify purchased part.* |
|  |  | P5.2.2 | P - Pods shall have env-data transmit capability over DW1000 module to rover (DW1001 accepted) | *Customer requirement* | *Verify purchased part.* |
|  |  | P5.2.3 | P - Pods shall have env-data transmit capability over DW1000 module to other pods (DW1001 accepted) | *Customer requirement* | *Verify purchased part.* |
|  | P5.3 |  | P - Pods shall be powered by a rechargeable power source | *In order to readily conduct multiple tests, the pods must have easily renewable power sources* | *Vendor confirmation, can demonstrate rechargeability.* |
|  |  | P5.3.1 | P - Battery shall be readily replaceable or rechargable in-situ | *To enable ease of testing, customer requirement* | *Inspection and demonstration.* |
|  |  | P5.3.2 | P - Batteries shall have sufficient capacity to meet 5% duty cycle between low and high power mode for 2 hour duration test at 200mA high, 20mA low. Usable capacity shall be 60mAh*FOS, TBR. | *Desireable to have beyond sufficient battery power for the mission duration.* | *Power budget shall be completed, battery purchased with capacity beyond expected need, inspection of capacity on component.* |
|  |  | P5.3.3 | P - On board battery shall be protected against damage due to DOD exceedance | *A battery that self regulates, or by using a power management system, battery health should be maintained.* | *Inspection and demonstration, confirm via vendor data.* |
|  | P5.4 |  | P - The pods shall be ready to function immediately upon landing post deployment | *The pods must perform their function once deployed, they cannot be damaged or otherwise inhibited from this without impacting mission success.* | *Pod and Deployment subsystem deployment demonstration followed by pod functionality test, pods can be powered on pre-deployment or after landing.* |
|  |  | P5.4.1 | P - Pod shall have water resistance to IP 54 | *To enable 'light rain' testing, and to prevent damage of wet grass or otherwise.* | *Inspection, analysis using an empty shell and water damage indicators during light water splash test.* |
|  |  | P5.4.2 | P - Pods shall be durable enough to withstand at least 30 deployments | *Repeatability in testing without having to go back and remanufature components is desireable, though some intential crumple zones are allowed.* | *Analysis of structural wear over time, proven after multiple tests, can be tested without fully integrated system by way of drop test.* |
|  |  | P5.4.3 | P - Pod shall survive 4.5m/s TBR impact to dirt/terf surface | *Desire for pods to function after deployment, deployment will induce high accelerations.* | *Analysis of structural accleration, can be tested without fully integrated system by way of drop test.* |
|  |  | P5.4.4 | P - Pod shall be designed to maintain position upon impact at final landing position, eg prevent bouncing outside of accuracy tolerance. | *The 1m accuracy target is difficult to reach, but hitting it and bouncing away is not useful either, so the pod must stay near its first impact.* | *Inspection, proof of concept testing to be performed by drop test or preliminary deployment tests.* |
|  |  | P5.4.5 | P - Pods shall be powered once they have landed post deployment, and can turn on before this time | *The pods must be powered on to function.* | *Electrical design, inspection of functionality to power up on impact.* |
| M6 |  |  | **The pods shall be able to function as a long-term deployable environmental data monitor** | *Provides dual functionality to an otherwise disposable system, customer requirement* | *Pod Subsystem Demonstration* |
|  | P6.1 |  | P - Pod shall have an environmental sensor package | *Adds dual functionality to the pods during their permanent existance in the environemt.* | *Parts List of Pod, inclusion in electrical subsystem within pod.* |
|  |  | P6.1.1 | P - Temperature Sensor | *Chosen to emulate some basic 'weather/environmental' data* | *Inspection, verified by existence.* |
|  |  | P6.1.3 | P - Altimeter | *Can also be used for deployment accuracy check against terrain map, chosen to emulate some basic 'weather/environmental' data* | *Inspection, verified by existence.* |
|  |  | P6.1.4 | P - Accelerometer | *Customer Requirement, can be used for position estimation and to see if the pod moved unexpectedly during a test.* | *Inspection, verified by existence.* |
|  |  | P6.1.5 | **Subj to Change** P - GPS Sensor | *Subject to change, GPS truth may be needed to verify deployment accuracy, but other methods are being explored.* | *Inspection, verified by existence.* |
|  | P6.2 |  | P - Pod shall operate above 3.3V and under 20mA TBR current in ‘low power mode’, refer to DR 5.3 | *The pods have to operate in the environment post rover traversal, and should do so in a way that consumes battery life within budget* | *Verified by analysis of power budget, testable in subsystem level demonstration.* |
|  | P6.3 |  | P - Pod shall operate aboe 3.3V and under 200mA TBR current in ‘high power mode’, refer to DR 5.3 | *Pods should provide sufficient power for ranged transmission without exceeding battery limitations.* | *Verified by analysis of power budget, testable in subsystem level demonstration.* |
|  | P6.4 |  | P - Pod shall have ability to toggle between low power and high power modes | *In order to collect valuable data in the environment post primary mission, a low power consumption is essential.* | *Verified by design, and testable in subsystem level. Autonomous toggle requires rover subsystem for a full demonstration.* |
|  |  | P6.4.1 | P - In high power mode the pod shall transmit data | *No low power transmission needed, full power resources avail to transmitter.* | *Subystem Demonstration, Full System Demonstration* |

| Identifier | Identifier | Identifier | Title | Motivation | Validation |
|---|---|---|---|---|---|
| | | P6.4.2 | P - In low power mode the pod will wake up on a set interval, check if rover is nearby and take environmental data then go back to sleep | *Staying nominally in low power mode is to preserve battery life, wake up, sample, check for rover, is the second level of functionality so that the pods can be reconnected to and the extended period data can be recovered.* | *Subsystem Demonstration.* |
| M7 | | | **The team shall verify absolute navigation ability** | *The verification of data is essential to validating any system tests* | *Post Data Processing Demonstration* |
| | S7.1 | | S - Software shall be capable of recording GPS data of the rover for the duration of the mission | *The duration of mission GPS data will be used as truth, required for ultimate validation most easily.* | *Inspecting rover generated GPS after any rover subsystem demonstration, verified by existence of data. Data can be validated further via landmark test.* |
| | | P7.1.1 | ** Subj to change** P - Pod records GPS data for duration of its individual on time | *Subject to change, GPS truth may be needed to verify deployment accuracy, but other methods are being explored.* | |
| | G7.2 | | G - Team shall build data analysis tools to generate graphs, plots, etc. in order to demonstrate quantitatively and qualitatively the mission success. | *A suite of software that ingests test data and outputs user-digestable content to easily verify the success or issues in a test is essential to debugging and customer satisfaction* | *Presentation of post data-analysis tools, acceptance by customer/PAB of sufficiency.* |
| | G7.3 | | G - Rover shall capture first person video during test | *Customer requirement* | *Install a COTS action camera on exterior of rover, human interaction to power on camera and visually confirm it is recording.* |
| | | G7.3.1 | G - Video time shall be synchronized with rover time | *Customer Requirement* | *Verification in post, inspect footage with known event times and compare* |
| M8 | | | **The team shall use the customer-provided hardware** | *Customer requirement* | *Inspection, verified by existence.* |
| | S8.1 | | S - The software shall run on the rover | *Customer requirement* | *Inspection, verified by existence.* |
| | | S8.1.1 | S - Programming language used shall be ROS compatible/portable | *Customer requirement* | *Inspection, verified by existence.* |
| | P8.2 | | P - Pods shall contain DW1000 and STM 320 for RF-localization and communication | *Customer requirement* | *Inspection, verified by existence.* |
| | G8.3 | | Rover shall be a Clearpath Jackal | *Customer requirement* | *Inspection, verified by existence.* |
| | | D8.3.1 | D - Deployment module shall be compatible with provided onboard power (5v @ 5A, 12V @ 10A, 24V @ 20A) | *Vendor requirement* | *Inspection, design by DM team.* |

# 3. Key Design Options Considered

## 3.1. Rover Navigation

At the start of the mission, the rover will receive a terrain contour map, a bitmap of keep-out zones, and a set of waypoints to traverse, as well as the heat map to show high science areas. Before moving, the rover will process these inputs and plan a path for traversing the waypoints while avoiding keep-out zones and steep gradients. It is important that this algorithm properly consider high science areas while also determining a short, safe path between all of the rover waypoints.

### *3.1.1. Dijkstra's Algorithm*

Dijkstra's algorithm is one of the most ubiquitous classes of path-finding algorithm, and is provably the fastest shortest-path algorithm for arbitrary directed graphs with unbounded weights. The solution is deterministic– inputting the same set of initial conditions will always return the same result. Because of its simplicity and efficiency, Dijkstra's algorithm has been thoroughly studied and has many existing implementations, including ROS nodes and MATLAB functions.

The algorithm relies on a graph consisting of vertices $V$ connected by edges $E$ weighted according to distance or some other measure of traversability (e.g. terrain gradient). Dijkstra's begins with a single starting node and maps out the shortest path to all other nodes in the graph. In the case of continuous regions– such as maps– the region must be discretized to the desired level of precision. By weighting obstacles as arbitrarily high or excluding vertices in that region altogether, the shortest path will avoid any keep-out zones. The edge weight will primarily depend on distance, but steepness of ascent may also be taken into account depending on the final test location.

The computation grows as a function of the number of vertices $V$ and $E$ as $O(|E| + |V| \log |V|)$. The resolution and discretization method will require extensive testing in order to confirm the viability of Dijkstra's algorithm. One major downfall is that the algorithm makes no attempt to move in a desired direction; the map expands only according to next-available shortest distance, and may cover paths away from the desired target first.

**Table 3. Dijkstra's Algorithm Pros and Cons**

| Pros | Cons |
|---|---|
| Deterministic, reliably shortest path (unbounded weights) | Computationally intensive for high-resolution discretization $O(|E| + |V| \log |V|)$ |
| Handles obstacles | Doesn't consider desired direction |
| Simple implementation | |
| Ubiquitous, lots of resources and sample implementations | |
| Existing ROS implementations | |

### *3.1.2. A Algorithm*

The A class of algorithms are essentially "parents" to Dijkstra's algorithm, but use a heuristic function to give the algorithm a sense of direction towards the shortest path. For instance, the heuristic function may bias the algorithm in favor of exploring nodes in the desired direction of travel. The A algorithm provably (under certain restrictions) considers the fewest nodes, considerably speeding up computation, depending on the topology and nature of the heuristic function. This algorithm finds common use in robotics, and as such there are plenty of preexisting base implementations, including nodes already included in ROS.

There is no universal growth order estimate for the A algorithm, as the complexity depends on the selected heuristic function. The chosen function determines both the speed and the accuracy of the solution: a complex heuristic will increase computation time, while a simplistic one may result in a non-optimal solution. An added benefit of this is that depending on the scope of the implementation, the heuristic can be arbitrarily relaxed or constricted in order to speed up computation or promote accuracy in the path generation.

Table 4. A Algorithm Pros and Cons

| Pros | Cons |
|---|---|
| Generally computationally faster than Dijkstra's | Slightly more complex than Dijkstra's |
| Heuristic function allows for guided solutions | Implementation may be challenging with multiple targets |
| Ubiquitous, lots of resources and sample implementations | Complex heuristic may slow computation time |
| Existing ROS implementations | Simplistic heuristic may provide non-optimal solution |
| Allows for relaxation of heuristic criteria to speed computation | |

### 3.1.3. D Algorithm

The D class of algorithm is also derived from A*, with the addition of dynamic replanning upon discovery of obstacles[8]. This algorithm determines the path “backwards” starting at the goal and expanding towards the current node. This is beneficial for the dynamic replanning portion of the algorithm, since the paths to the end are well-mapped. Some variants of the algorithm include a heuristic function similar to A. At its core, this algorithm behaves almost exactly like A* with the exception of the obstacle weighting.

In the case of the DRAGON mission, the obstacles will be known in advance, so implementing dynamic replanning may be unnecessary and cumbersome in terms of prototyping and testing.

Table 5. D Algorithm Pros and Cons

| Pros | Cons |
|---|---|
| Dynamic updating desirable autonomous missions | Possibly too complex for application |
| No heuristic function to deal with | Slower computation time– A* plus dynamic changes |
| Same optimality as A* | |
| Existing ROS implementation | |

### 3.1.4. Rapidly-Exploring Random Tree

The Rapidly-Exploring Random Tree method (RRT) works differently from the previous algorithms in that points for inspection are chosen randomly from the starting point, and the tree structure extends incrementally towards that newly chosen point. Should the path encounter an obstacle or keep-out zone the branch does not extend any further. In the event that no obstacle is encountered, the tree will continue branching out until it has reached the new node. This is done recursively until the goal node has been picked up by a branch of the tree. The rover path can then be drawn from the final node back to the starting node. This method works very well in unmapped environments with lots of irregularities, but since a bitmap of obstacles is provided for Project DRAGON, this method may be unnecessary.

The main drawback of this method is that it does not provide necessarily the optimal path. By nature, the paths are randomly generated, and the branch that eventually reaches the final node may not be the shortest path for the vehicle. Given a time constraint, this may not be viable in terms of the mission. The random nature of this method also does not bode well in terms of processing speed, as points are chosen randomly to be explored without any real sense of direction towards the goal.

Table 6. RRT Algorithm Pros and Cons

| Pros | Cons |
|---|---|
| Random exploration covers more paths than other methods | Path not guaranteed to be optimal |
| Dynamic obstacle detection | Unknown computational speed – depends on region |
| ROS Package exists | |

## 3.2. Feedback Control Law

Feedback control will be implemented on the rover to maintain the rover's position and movement along the path to satisfy the path accuracy requirement G1.1.1. PID control uses feedback based on state measurements to correct the desired variable (i.e., trajectory, velocity) using proportional, derivative, and integral control. Proportional control applies a direct gain based on direct deviation in the feedback variable, while derivative and integral apply gains based on the rate of change of the feedback variable and the steady-state error in the signal, respectively.[14] These gains can be selected and tuned based on response timing requirements for the mission. This can not only be done in ROS, but also in MATLAB/Simulink, where extensive tuning tools exist and would allow for rapid prototyping for different control laws the team wishes to test on the Jackal. A sample image of a typical PID control block diagram is shown in Figure 3 for clarity.

Figure 3. PID Diagram

In terms of feedback control, both feedback and feedforward PID control law implementations were studied, but it was decided that feedback PID will be used for the rover navigation due to the fact that not only are PID controllers used extensively in industry, but ROS already has multiple packages nodes for implementing PID control. As a consequence, no trade study for the control law was conducted for Project DRAGON.

### 3.3. Software Languages

The Jackal rover is capable of running x86_64 Linux programs, with official support for ROS and Mathworks software. The code must be compatibel with ROS per customer requirement. This limits the available languages to the following: C++, Python, Lisp, C, Go, Java, Haskell, Node.JS, Julia, Lua, Smalltalk, R, Ruby, and Simulink. The two languages that the most team members are familiar with are MATLAB and Python. Simulink is a software that is very similar to MATLAB, which is a program every member of the team is proficient in; moreover, some members have directly worked with Simulink. While Simulink is essentially an extension of MATLAB, Python is another language entirely. It is similar enough to MATLAB to not pose a significant learning curve for unfamilar team members. The major benefits of both Simulink and Python over other languages/programs include the teams experience with these languages and the vast availability of documentation and resources. The final software package will likely be a combination of both, based on what team members are comfortable coding in. Furthermore, it is necessary to use a language appropriate for the intended application. For example, when dealing with wireless communication, a Javascript or Python library may be most effective. For these reasons, a combination of several coding languages will be the best option. Compiling these scripts to be run with a single start command on the rover will be beneficial as well.

### 3.4. State Estimation

The GPS-denied rover must depend on relative position measurements alone such as odometer and inertial measurements. Position estimation is based on earlier measurements, so error will increase over time. Odometers are especially error-prone in actuators with high slippage (such as the ClearPath Jackal) or in regions with significant terrain irregularities.

In contrast to relative navigation, absolute position measurements do not experience unbounded error growth. Active landmarks such as beacons are a form of absolute positioning. DRAGON will use trilateration– position determination using measured distance from at least 3 points– from a set of beacons as its absolute measurement to correct errors inherent in relative navigation.

The localization task falls into the category of Bayesian estimation problems– the rover has some initial “belief” about its location, which will be revised based on future measurements. With access to absolute and relative navigation information, the prior belief $Bel\ (x_k)$ is the belief the rover has after incorporating all information up to step $k$, including the most recent relative measurement, but not including the latest absolute measurement. The posterior belief $Bel^{+}(x_k)$ also includes the latest absolute measurement. State estimation methods fall into one of two distinct categories depending on the assumptions made in the formulation: Gaussian filters and nonparametric filters.

### 3.4.1. Gaussian Filters

All Gaussian filtering techniques rest on the assumption that all beliefs are multivariate normally distributed. To be considered a linear Gaussian system, the initial probability distribution function belief must be normally distributed, the state transition must be linear with Gaussian noise, and the measurement estimation must be linear with Gaussian noise.[23]

In many cases the linear assumption doesn't hold– for instance the updated odometer reading is not a linear function. This means that the initial belief of the position will be Gaussian (determined by GPS), but subsequent estimates from the odometer will be non-normally distributed.

KALMAN FILTERS The most basic form of the Kalman filter considers the linear system

$$x_t = A_t x_{t-1} + B_t u_t + \varepsilon_t \tag{1}$$

Where $x_t$ is the state vector, $A_t$ is the dynamics matrix representing the transition from state $x_{t-1}$ to $x_t$, $B_t u_t$ considers the controls acting on the system, and $\varepsilon_t$ is Gaussian white noise. The Kalman "filter" attempts to filter Gaussian white noise from the system estimate.

Measurements of the state of the system are denoted as

$$z_t = C_t x_t + \delta_t \tag{2}$$

Where $z_t$ is an observation, $C_t$ is the observability matrix (expressing which elements of state are able to be observed), and $\delta_t$, which is Gaussian white noise.

The noise can be characterized as the difference between the actual measurement $z_t$ and the estimated measurement based on the previous state and the system dynamics, $\hat{z}_t$. This error term also depends on the location of the observer, L. The estimated change in state can be written as

$$\hat{x} = A_t \hat{x} + B_t u_t + L(z_t - \hat{z}_t) \tag{3}$$

The Filter determines the location of the observer which minimizes the error term.

EXTENDED KALMAN FILTERS The "Extended Kalman Filter" (EKF) relaxes the assumption of linearity; state $x_t$ and measurement $z_t$ are instead written as:

$$x_t = g(u_t; x_{t-1}) + \varepsilon_t \qquad z_t = h(x_t) + \delta_t \tag{4}$$

This relaxation means that the beliefs are no longer necessarily Gaussian– the initial state will be Gaussian, but any non-linear mapping will make the resulting probability distribution non-normal. The EKF calculates the Gaussian approximation to the output of $g$, but it will not be exact. The EKF relies on linearization of the transformation by taking the Taylor series expansion of $g$ at the point to be evaluated. The quality of the approximation depends on the degree of non-linearity and the width of the posterior belief.

UNSCENTED KALMAN FILTERS The Extended Kalman Filter linearizes the transformation function $g$ with a Taylor series; the Unscented Kalman Filter (UKF) performs stochastic linearization through use of a linear regression process. The sigma points of the distribution (mean and 1 points for each dimension) are computed and fed through the function $g$, which characterizes how the shape of the Gaussian changes.

The asymptotic complexity of UKF is the same as EKF (lower bound $O(\dim z_t^{2.4} + \dim x_t^2)$), but in practice UKF is often slightly slower. However, for non-linear systems in general, UKF performs more accurately than UKF, and is always more accurate than the vanilla Kalman filter except in the linear case.

**INFORMATION FILTERS** Information Filters (IF) are often called the “dual” of Kalman filters in that problems that are simple in one become challenging in the other. In particular, information filters incorporate measurements easily but controls are more complicated– vice versa for Kalman filters. This difference arises from how the information filter represents the Gaussians: Kalman filters use the mean and covariances, while the information filter uses the “canonical representation” of an information matrix and vector.

Information filters perform especially well in scenarios with high uncertainty, many state variables (dim $x_t$ (> 100)), or multi-robot deployments.

Analogously to the EKF, there is an Extended Information Filter (EIF) which extends the filter to allow non-linear state transformations.

### 3.4.2. Nonparametric Filters

State estimation may also be accomplished using nonparametric methods that do not rely on Gaussians, but instead approximate posterior beliefs using a finite set of possible values in the state space.[23] Nonparametric filters do not make strong parametric assumptions about the posterior, and therefore can handle more complex situations such as non-unimodal probability distributions.

Nonparametric filters discretize the state space– naturally the quality of approximation increases as the resolution increases. Such filters are often implemented as “resource-adaptive”, where the discretization scales with available computing resources. For instance, if the rover needed to devote computing resources to deploy a pod, the state estimation algorithm could be scaled down to release memory and processor time. The two relevant types of parametric filter are called histogram filters and particle filters.

**HISTOGRAM FILTER** The Histogram Filter (HS) discretizes the state space with a single number representing the probability of each quantum being the true state. This type of filter is significantly less intensive than a Kalman filters since it relies only on summations– no matrix inversions.

The algorithm can be adapted to remain within computing constraints by setting the granularity of the approximation, which also affects the computation time and resolution. Discretization can be accomplished either statically (uniformly partitioning the state space) or dynamically (partitioning to maintain higher resolution at higher probabilities).

**PARTICLE FILTERS** In a Particle Filter (PF) the porterior is represented by a set of random state samples, allowing the determination of the probability distribution function numerically before and after the transformation. This allows for modelling of non-linear transformations given a sufficiently large number of particles. Typically the number of particles will be on the order of thousands– less than a hundred is not recommended.

During each cycle of the algorithm, each particle will be weighted according to the probability of the most recent measurement given the particle position. The lower weighted particles are culled and redistributed near the high-probability density, allowing for higher resolution in that region of state space.

One downsides of nonparametric methods is that states between time steps are difficult to estimate since the transformation is discrete.

Table 7. State Estimation Summary

| Descriptions | KF | EKF | UKF | IF | EIF | HF | PF |
|---|---|---|---|---|---|---|---|
| State/measurement transition | Linear | Non-linear (Taylor Series) | Non-linear (Stochastic) | Linear | Non-linear (Taylor Series) | Discretized | Discretized |
| All beliefs Gaussian | X | X | X | X | X | | |
| Resource-adaptive | | | | | | X | X |
| Computation time | [1] | > KF | > EKF | [1] | > IF | [2] | [2] |
| Accuracy | – | > KF | > EKF | – | > IF | [2] | [2] |
| Algorithm Complexity | Moderate | High | High | Moderate | High | Low | Low |

[1] Computation of KF vs IF depends on the complexity of measurement/control dynamics
[2] Depends on discretization resolution

Since the information filters are effectively equivalent to the Kalman filter in terms of complexity and accuracy, we will continue to the trade study with the vanilla Kalman, Extended Kalman Filter, Unscented Kalman Filter, Histogram Filter, and Particle Filter.

### 3.5. Pod Deployment Method

In order to place the pods in desirable locations, it is critical that there is an accurate deployment method. The deployment method will not only dictate the shape of the pods, but will also weigh heavily on the stopping method and reloading method. The trade study metrics have been designed to reflect the dependence between the two systems. The concepts evaluated include drone deployment, mechanical arm, pneumatic cannon, mechanically propelled projectile deployments, and a catapult. For all the considered designs, excluding the drone, the deployment devices shall be mounted to a variable pitch platform, which shall then be attached to a base that can swivel on the rover. Actuators will then be attached to these platforms, allowing for the deployment method to move independently of the rover. The software can dictate a deployment azimuth and elevation angle and maneuver the deployment method accordingly.

It should also be noted that for all of the methods discussed below, there will not be active control devices on the pods. Though active control devices could allow for more accurate flight and better dead-band orientation upon landing, they add unnecessary complexity to the pods. Keeping the primary purpose of the pods in mind, active control devices would require additional software and hardware within the pods that will serve no purpose once the pod has arrived at its location. This additional level of complexity is beyond the scope of this project and will not be pursued in future designs.

#### 3.5.1. Drone

One potential design concept to achieve pod deployment is through the use of a fully autonomous drone. For the purpose of this design, the drone needs to hover and conduct vertical takeoff which makes a quad-copter the clear choice over a fixed wing drone. As Fig. 4 indicates, the drone will begin the mission by waiting idly on the rover. Once the algorithm on board the rover has determined where a pod is needed, the drone will then launch and attach to a pod. The drone could attach to the pod through a variety of mechanisms with the most desirable being electromagnetism due to a wider connection range and easy detachment. The drone will fly to the designated coordinates dictated by the algorithm and place the pod within a 1m tolerance. Once the pod has been placed, the drone will fly back to the rover and land. This process will be repeated as often as the pod placement algorithm dictates until the completion of the mission.

Figure 4. CONOPS for drone deployment

While this method would allow for low impact pod deployment and nearly guarantee-able orientation, integrating the drone with the rover would be a complex task. COTS drones such as the DJI Phantom 4 series have flight modes that return the drone to a specified destination without user input, however these functions are entirely dependent on GNSS navigation which will not be available for this project. Research areas that may help solve this problem include the vision-based cooperative localization developed by RAVEN or the use of optical flow navigation. However, integrating RAVEN's design as a pod deployment mechanism could be a whole other project of its own as the the drone would naturally drift over time with-out GPS correction.

Table 8. Drone Pros and Cons

| Pros | Cons |
|---|---|
| Predictable pod orientation | Complex integration |
| Low impact pod placement | Limited battery life |
| Adaptable for several pod designs | Limited payload capacity |
| - | Slow deployment rate |
| - | Complex controls |

### 3.5.2. Mechanical Arm

A mechanical arm deployment uses a arm or boom structure to drop the pod at the desired location. An angle and range would be provided, and the mechanical arm would then extend to that position. The structure would then retract to the rover, reload a pod, and repeat this process as commanded. A CONOPS of a possible mechanical arm deployment can be seen below:

Figure 5. CONOPS for mechanical arm deployment

There are three main types of mechanical arms: telescoping, hinged, and rolling. A telescoping arm would compact the arm into sections, either tubular that are slid into each other, or stacked onto each other. A full segment of known length would be extended before the next section is extended. A hinged boom would act like a scissor lift to extend the pods to their locations. A rolling mechanical arm would consist of a tape-measure shaped material rolled into a spool. To deploy, an actuator would rotate the spool and a tape-measure like structure. Each of these methods would require similar components: a controller (microcontroller or microprocessor), multiple actuators, and the arm structure itself. This deployment method would need to be built from the ground up; the mechanical arms required would need to be manufactured with very little off-the-shelf solutions.

Table 9. Mechanical arm Deployment Pros and Cons

| Pros | Cons |
|---|---|
| Precise placement | Slow deployment time |
| Easy control algorithm | High stress on structure |
| Low impact force for pods | Difficult to fabricate |
| Easy to control range and angle | Significantly changes rover's CG |
|  | Low TRL |

### 3.5.3. Spring Cannon

Several design options to deploy the pods lie under the category of mechanically propelled projectiles. These choices all launch the pods through the air using electrical energy from the rover that is then converted into mechanical energy to provide the variable force necessary for the pods to land between the 5-20m range. All concepts considered in this section came from designs already found in other mechanical projectile systems: a football pitching machine, a clay pigeon trap thrower, and an air-soft style spring cannon.

An air-soft gun is essentially just a spring cannon mixed with a hydraulic cannon, but the design concept could still be used. When the trigger is pulled, a motor is powered that turns a gear which pushes a gear rack which, in turn, compresses a spring. The turning gear is designed in such a way that the last tooth releases from the gear rack when the spring is fully compressed. When released the spring decompresses and pushes air through a pump, hence air-soft. This design could be easily modified to release at any point in the springs compression, varying the range of the cannon between the desired 5-20m range. This design could use either a gravity feed or a spring assisted clip to push the next pod into place when the spring is fully compressed before decompressing the spring as necessary to obtain the calculated force needed for launch. This method could be easily modelled using Hooke's Law to determine the needed compression for the spring along with some modelled friction and drag during launch. The downside of this concept is the fact that it would be much more mechanically complex than the other mechanical cannons, and the orientation of the pod cannot necessarily be predicted upon landing.

Figure 6. CONOPS for a Spring Cannon

Table 10. Spring Canon Deployment Pros and Cons

| Pros | Cons |
|---|---|
| Simpler Modelling | More mechanically complex |
| Dependable range variability | More safety mechanisms required |
|  | Unpredictable pod orientation without pod modifications |

### 3.5.4. Pneumatic Cannon

A pneumatic cannon was considered in addition to the spring-cannon and features a similar CONOPS, with a pressurized air as the source of energy as opposed to a spring. While a pneumatic cannon can be modeled simply and is more tunable than a spring system, it also has many major drawbacks. The largest drawback of this design is that it requires consumables, which leads to a need for a fuel storage and a means of re-filling the consumables. In the context of the project – a remote, autonomous system – the customer and team seek to reduce physical user interaction with the system during demonstration, making the need for consumables a significant drawback. Another disadvantage of the pneumatic system is that it requires pressurized gas, which adds a level of safety risk to the project – not only to personnel in the testing vicinity, but also the rover itself. Additional preventative measures would be necessary to ensure the safety of this system.

Table 11. Spring Canon Deployment Pros and Cons

| Pros | Cons |
|---|---|
| More tunable than a spring system | Requires consumables |
| Consistent and predictable projectile motion | More safety mechanisms required |
| | Unpredictable pod orientation without pod modifications |
| | Complex mechanisms and storage required |

### 3.5.5. Disk Thrower

The next mechanically propelled design comes in the form of a disk thrower. A clay pigeon disk throwing machine is a quite simple design. It uses a spring in tension that, when the “trigger” is pulled, compresses and rotates the swinging arm. The arm pushes and spins the clay pigeon off the slide. COTS products can launch a 100g clay disk approximately 41 meters[5]. This acceleration all occurs within about a quarter turn, before the arm and spring are cocked by a motor the rest of the way around. The next pod is then loaded from a gravity fed chute. When spinning, the disk can keep its angular momentum vector perpendicular to the ground, ensuring that the pod stays in the correct orientation during a gliding flight. Even if a disk-shaped pod flipped, the dead-band orientation would still remain vertical. This method, however, requires that the pod is in a flat disk shape, greatly limiting the size and weight of the actual pods.

Figure 7. CONOPS for a Disk Thrower

Table 12. Disk Thrower Deployment Pros and Cons

| Pros | Cons |
|---|---|
| Limited pod orientations upon landing | Projectile path more dependent on wind |
| Gliding reduces impact force on the ground | Poorer accuracy during non-ideal conditions |
| Simple, single motor and spring design for swinging arm | Limited pod shape, size, and weight |
|  | Only able to vary angle, not power |

### 3.5.6. Pitching Machine

A football pitching machine essentially is a set of two spinning wheels with off-set angular momentum vectors that push and spin the ball in a tight spiral when the football is fed into the wheels on a slider. The practical benefits of a pitching machine are ease of creation and varying the force supplied to the pods during launch by simply changing the angular velocities of the wheels. The reloading mechanism could also be a simple gravity fed chute or spring-assisted clip, similar to the other design options. The pods are also spin-stabilized, allowing them to fly on a more predictable path. However, the spin-stabilization greatly reduces the ability to ensure that the pods are in a desirable orientation–if at all–without some sort of additional stopping or orientation mechanism on the pod. If desired, the wheels could be set level with the slide and each other, which would not spin stabilize the pods.

Figure 8. CONOPS for a Pitching Machine

Table 13. Pitching Machine Deployment Pros and Cons

| Pros | Cons |
|---|---|
| Simple, COTS design | Spin stabilization highly reduces orientation predictability |
| Easily re-loadable | Must model friction between wheels and pod to determine launching force |
| Desirable controls modelling | Would likely need to prototype before CDR |
| Straight-forward mechanical and fabrication complexity |  |

### 3.5.7. Catapult

The last mechanically deployed design option comes in the form of a catapult. Much like the disk thrower–which is essentially a catapult on its side–the main force for deployment comes from a spring in tension which compresses when a clutch mechanism is released. The arm would swing and the pod would be released at the top of the arc before the arm is cocked back using a motor. The difficulty in this design comes from ensuring that the pod is released in the correct part of the swing and ensuring the reloading mechanism does not interfere with the catapult's swing.

Table 14. Catapult Deployment Pros and Cons

| Pros | Cons |
|---|---|
| Few moving parts | Releasing pod at correct location in the arm's swing could be difficult |
| Can easily change force applied to pod on launch | Ranging could involve more inaccuracy |
|  | Reloading mechanism would need to be separate from the catapult's swinging arm |

### 3.6. Pod Stopping Method

Upon deployment, the pods will need a significant amount of kinetic energy to satisfy the 5 - 20 meter range. Due the 1-meter precision required for placement, the pod should not bounce or roll after landing. Furthermore, the RF chip inside the pod will need to be oriented in such a way the antenna dead-band points vertically (exact keep-out zones are being determined). The following conceptual designs are possible options for stopping the pods. The orientation method depends on the pod design, but one over-all consideration is a two-axis gimbal that uses gravity to correctly align itself, regardless of how it lands. However, this design may be difficult to implement since the gimbal would have to be quite small and able to fit inside the pod. When the pod lands, the gimbal may break upon impact. No trade study will be conducted until a pod deployment method is finalized since the stopping method is heavily dependent on deployment method.

#### 3.6.1. Attachment

Attachments to the pod structure was the first pod design option considered. These could be in the form of metal rods or fins which aim to prevent the pod from rolling upon impact. Ideally, it would only allow a few rotations after impact and then keep the pod oriented with the deadbands vertical. Due to the high forces at impact, the attachments would need to be manufactured out of a strong material and could also prevent a challenge when the pods are being reloaded. A possible design for attachments to the pod can be seen below.

Figure 9. Leg attachments on pod for stopping and proper orientation

When this design lands, the pod may roll over a few times, but would likely stay within the 1m target area. The vertical attachments on the ends of the arms would also orient the pods correctly without the need for a gimbal device.

Table 15. Attachments Pros and Cons

| Pros | Cons |
|---|---|
| Correctly orients pod | No shock absorption for pod |
| Easy to manufacture | Large forces on attachment may cause them to break |
| Does not significantly affect trajectory model | Difficult to reload |

### 3.6.2. Crumple Mechanism

One potential method for slowing the pod upon arrival to the target location is through the use of a crumpling shell. For this method the pod shape is not a design driver because the crumple material can be used to encase the pod. Through the use of 3-D printed arms or honey-combed cardboard for example, a majority of the initial impact can be absorbed. This method of shock absorption and pod-stopping allows for a large design space while not drastically increasing the mass of the pod. Because the crumpling material must withstand the force of deployment, the implantation and strength of the material will be an area of concern. Additionally, the crumple material will be subject to a wide variety of forces associated with short and far deployment which may be problematic. However, an algorithm that decides the trajectory of the pods could be made to deploy the pods in such a way so that each pod will experience the same impact force, regardless of range.

Table 16. Crumple Mechanism Pros and Cons

| Pros | Cons |
|---|---|
| Effective shock absorption | Needs to be replaced every launch |
| Pod is not subjected to initial impact | Deployment mechanism forces may cause premature crumpling |
| Lightweight | Not adjustable for different impact forces |

### 3.6.3. Sandbag Mechanism

Another potential method for maximizing the accuracy of landing the pod in a specific location would be to use an energy damping system similar to how packing peanuts protect contents of packages. The pod would have a bag similar in design to a hacky sack or sandbag strapped to the leading edge. Upon impact with the ground, the energy would be dispersed through the contents of the sandbag leaving the pod and its internal components in tact. Unfortunately, there are some issues with this system that must be dealt with. Manufacturing a pod completely surrounded by this sandbag system would be very difficult and would make replacing internal components or charging the batteries a hassle. If the sandbag was only covering the leading edge, flight stability becomes much more important to ensure a safe landing.

Table 17. Sandbag Mechanism Pros and Cons

| Pros | Cons |
|---|---|
| Effective shock absorption | Difficult to manufacture |
| Reusable | May cause deployment difficulty |
|  | Heavy |

## 3.7. Pod Structures

Pod survivability is one critical project element as it is necessary for the internal pod electronics to survive deployment. While there are many factors that will influence the survivability, none are as influential as the structural material. Therefore, three very different types of materials were selected to be studied. These materials are polycarbonate, polyurethane, and ABS plastic filament.

### 3.7.1. Polycarbonate

Polycarbonate is a type of plastic polymer with a spectrum of different engineering uses. Polycarbonate is known for its robust durability and its ability to be manipulated and molded, however, the material is not flexible and internal electronics secured directly to the shell might be at risk for damage.

Table 18. Polycarbonate Pros and Cons

| Pros | Cons |
|---|---|
| Little fatigue | Heavy |
| Inexpensive | Little energy dampening |
| Machinable | |

### 3.7.2. Polyurethane

Polyurethane is a polymer just like polycarbonate but is much less dense and is often used in the production of anything from foam sponges to surface sealants. The density and hardness can vary depending on the production method meaning it can suit many different applications. In general, polyurethane is easily pliable but will return to its previous form once the force is removed which makes it a great option to seat the pod electronics. This material is not nearly as impact resistant or durable as polycarbonate and has a higher chance of failing after each successive launch.

Table 19. Polyurethane Pros and Cons

| Pros | Cons |
|---|---|
| Lightweight | Expensive |
| Exceptional energy absorption | Prone to rips/tears |
| Packaging Legacy | Difficult to tether |

### 3.7.3. ABS

The main advantage of using and ABS (Acrylonitrile Butadiene Styrene) material is the ease of manufacturability. It can be easily molded but can also be used in additive manufacturing techniques. A key drawback to using additive manufacturing is the large reduction in strength that is introduced from impurities in the material. These impurities should not affect the hardness or ability to survive impacts but may cause greater fatigue and a lower overall durability.

Table 20. ABS Pros and Cons

| Pros | Cons |
|---|---|
| Very manufacturable | Brittle |
| Cost efficient | Many impurities from printing |
| Relatively durable | Little energy absorption |

## 3.8. Localization/Pod Processing

In order to broadcast localization information the DW1000[9] RF module will be used. This module has the ability to transmit ranging information as well as transmit data packages for environmental data. To implement the DW1000 it needs to be interfaced with a microprocessor to command it to transmit and receive data. To accomplish this task four options are considered. The four options are using a Localino[12] development board, using a STM32F103 processor with a custom PCB, using a DWM1001[10] development board, and using an Adafruit Feather nRF52 Bluefruit[2].

### 3.8.1. Using a Localino for interfacing with DW1000 and sensor interfacing

The Localino uses a STM32F103 processor and is programmed with the Arduino IDE and has a built in interface for the DW1000 module. In the summer of 2018 the customer developed software to use the ranging and data transfer components of DW1000. This software was made available from the customer, is well tested, and well suited for the application of this project. The Localino board has limited number of I/O pins that could be used to interface additional sensors including I2C and SPI interfaces. However the board only offers 128 KB of flash memory which means an external memory unit would be required. The Localino costs roughly $100 and is 29mm by 52.7mm in size.

Table 21. Using Localino for all Pod computing

| Pros | Cons |
|---|---|
| RF functionality has been tested | Will require external data storage |
| Arduino interface is easy to implement | Limited number of I/O pins |
| Low cost | |

### 3.8.2. Designing a PCB integrating a DW1000 and a STM32F103 Processor

Instead of using a COTS board, the same processor(STM32F103) can be integrated with the DW1000 RF module on a custom board. This would allow the use of all the necessary pin outs for integrating the RF module and all required sensors. The custom board has the potential of lowering the size of processing board. However this does come at the cost of increased complexity of the design. Designing a board is a difficult process that would require a substantial amount of time and manpower. Additionally any errors in the board design could lead to processors being damaged or destroyed. There is also potential that this board can be expensive to produce considered the 10+ required for the total number of pods.

Table 22. Designing a PCB to integrate a STM32F103 with a DW1000

| Pros | Cons |
|---|---|
| More design freedom | Increased complexity |
| | Reduced reliability |
| | Increased development time |

### 3.8.3. Using a DWM1001 Dev Board

The DWM1001 Dev board is a new release from Decawave that incorporates the DW1000 RF module in a prebuilt system. This dev board is packaged completely by Decawave with a C debugger. The DWM1001 Dev board contains $I^2C$, SPI, and multiple GPIO pins. This board will still require expandable memory but is significantly cheaper at $39 for the whole board. The dev board is 62 mm by 43 mm in size which will help keep the pods within the size requirements.

Table 23. Using Localino for DW1000 interfacing and an additional processor for data collection and storage

| Pros | Cons |
|---|---|
| Provided debugger | Will require expanded memory |
| Multiple I/O pins | New system(might contain bugs or lack examples) |
| Low Cost | |
| Size | |

### 3.8.4. Using an Adafruit Feather nRF52 Bluefruit

The Bluefruit uses a nRF52832 Bluetooth chip for its processing and is programmed with the Arduino IDE. The Bluefruit does not use the DW1000 RF module and instead uses an integrated Bluetooth module. This board supports 19 GPIO pins that include $I^2C$, SPI and UART interfaces. It also contains 8, 12 bit, ADCs. It comes in a 51mm by 23 mm size and costs $25. As with the other COTS boards it will require expanded memory as it only has 512 KB of flash memory. Like the DWM1001, this board will require new software to be written for ranging and environmental data transfer.

Table 24. Using Localino for all Pod computing

| Pros | Cons |
|---|---|
| Size | Will require external data storage |
| Large number of I/O pins | Not tested by campus personal |
| Arduino interface is easy to implement | Does not use the DW1000 |
| Low cost | |

## 3.9. Pod Electronics

### 3.9.1. Environmental Sensors and Altimeter Introduction

Each pod will contain an environmental sensor suite that will measure various elements of the surrounds. The environmental sensors that will be included in the pod are to the team's discretion; however, the sensors chosen must obey the customer's requirement which has been established through the direct requirement P6.1. Three environmental sensor suites will be studies in the following trade study.

All three sensor suites are COTS components. The choices of strictly COTS components was purposeful. The collection of weather data is strictly a requirement set out by the customer for the purpose of having some form of science data to transmit from the pods to the rover over the course of the mission from the supposed high science areas that are input by the user. The environmental sensors data collection is a secondary mission compared to the primary mission of RF localization. Benefits of a pre-packaged board are that that COTS boards offer a reduced cost, high reliability, customer friendly method of sensor integration. COTS parts are easily adaptable and usually offer clear and well-supported documentation of how the boards and integrated components operate. Hook up guides and documentation make set-up, integration, calibration, de-bugging, and overall system analysis much simpler. COTS boards allow the team to focus more on the communication aspect of the project while still maintaining quality and ability to achieve P6.1. Even through designing a PCB for individual environmental sensors would allow the team to have more control over the size/shape of the board as well as its layout, it would be more effective and efficient to use COTS breakout boards in this project.

The amount of data the the pod will need to save will be larger than what each processor can save; therefore, a micro SD card is necessary. All data collected from the accelerometer and environmental breakout boards will require external data storage. Data from the environmental sensor suite will be saved to a Micro SD card on the pod. No trade study was conducted on this component, but the team will be using the Sparkfun OpenLog as the SD card reader.[21] This reader can be integrated with any of the environmentally and accelerometer breakout boards in the below trade studies.

### 3.9.2. SparkFun Altitude/Pressure/Temperature Sensor Breakout - MPL3115A2

The MPL3115A2 sensor suite embodies a MEMS (Micro-Electro-Mechanical System) sensor that provides accurate measurements for pressure, altitude, and temperature in a small and compact breakout board.[22] The board, standard to Sparkfun specifications, has designated spaces for easy integration of headers and breadboard testing as well as mounting holes. The sensors outputs are digitized by a high resolution 24-bit ADC (Analog to Digital Converter) and are transmitted over an I2C interface, meaning that this sensors can be integrated with most controllers. The MEMS sensor is specifically a pressure sensor that outputs pressure in fractions of Pascals, temperature in Celsius, and altitude in meters. This sensors can be used in a variety of applications including high accuracy altimeter, GPS emergency services, and weather station equipment which is ideal for our mission. The team also has heritage with the component, for a team member has utilized this specific component before on a sounding rocket payload.[22]

Table 25. Pros and Cons of the MPL3115A2 Altitude/Pressure/Temperature Sensor Breakout

| Pros | Cons |
|---|---|
| Altitude resolution of 30cm | Can have +/- 10 meters in altitude variation |
| Can easily be integrated onto an 3.3V or 5V micro-controller | Cannot be integrated onto Raspberry Pi |
| 20-bit measurements for pressure and temperature | |
| Multiple sources of documentation and hook-up guide | |
| Company nearby | |
| Component Heritage / Team members have used before | |

### 3.9.3. Adafruit - BME680 Environmental Sensor Breakout Board

Adafruit's BME680 environmental sensor breakout board offers temperature, humidity, barometric pressure, altitude sensing capabilities.[1] The BME680 also contains a small Metal Oxide (MOX) sensor. Changes of resistance determined by the sensor and determines various volatile organic compound (VOC) gases such carbon monoxide, ethanol, and alcohol in the surrounding atmosphere. The board has designated spaces for easy integration of headers and breadboard testing as well as designated mounting holes. The sensor is a compact board that has the ability to communicate using either SPI or I2C communication. This allows for easy integration on numerous micro-controllers. The small dimensions and low power consumption of this sensor makes it ideal for simple environmental observations with high accuracy. This sensor can be used for a variety for applications including making a personalized weather stations, making outdoor and indoor air quality measurements, moisture detection, environmental change detection, and even mobile phones. With the wide variety of applications of this sensor, it proves to be versatile and user friendly.[1]

Table 26. Pros and Cons of Adafruit's BME680 Environmental Sensor Breakout Board

| Pros | Cons |
|---|---|
| Voltage Regulator on breakout board | 48 hour burn in time |
| Independently enable/disable individual sensors | 30 minute heat up time for accurate measurements |
| Multiple sources of documentation | Sensor cannot differentiate gases and alcohols |
| Can easily be integrated onto 3.3V or 5V micro-controller | |
| SPI and I2C compatible | |

### 3.9.4. Sparkfun - Environmental Combo Breakout - CCS811/BME280

The Sparkfun CCS811/BME280 environmental breakout board provides a variety of environmental data including pressure, humidity, temperature, altitude, total volatile organic compound gases (TVOCs), and equivalent CO2 levels.[19] [18] The board, standard to Sparkfun specifications, has designated spaces for easy integration of headers and breadboard testing as well as mounting holes. This board contains two separate sensors. The CCS811 sensor is an ultra-low power digital sensor solution which integrates a MOX gas sensor to detect a wide range of VOCs in parts per million (PPM) or parts per billion (PPB) by measuring resistance of the surrounding atmosphere.[19] The BME280 is a combined humidity, pressure, altitude, and temperature sensor that has high accuracy and response times. Using a simple interface with I2C communication, this breakout board can be used in a wide variety of applications including use in smart phones, wearable technology, indoor and outdoor air quality monitoring, and weather forecasting which makes it ideal for this project.[18]

Table 27. Pros and Cons of Sparkfun's Environmental Combo Breakout with CCS811 and BME280 Sensors

| Pros | Cons |
|---|---|
| Designed for high volume and reliability (5 years) | Backordered |
| Simplified hardware and software integration | Relatively large (2.5 cm x 2.5 cm) |
| Proven technology platform | Expensive (34.95 US dollars per unit) |
| Multiple sources of documentation | Cannot independently enable/disable individual sensors |
| Company nearby | |

### 3.9.5. Accelerometer

Each pod will contain an accelerometer. The sensor chosen will assist in fulfilling requirement DR 5.4 and providing important regarding the impact conditions of the pod during deployment as well as requirement DR 6.1 as being apart of the environmental sensors suite. It is important that the internal and external components of the pod can remain functional after being exposed to multiple g-forces, and this sensor will assist the understanding of the conditions of the pod post-deployment. After deployment, and the sensor will measure the initial impact of the pod. As a secondary possible purpose, the accelerometer will be to determine possible undesired movement or rolling of the pod during the mission duration. It is important to note that including this sensor is a requirement of the customer and the possible data we get from this sensor can possibly be used for post-mission analysis, but it not necessary to use the data to fulfill a mission requirement. Three COTS accelerometer breakout boards will be studies in the following trade study. These boards are designed to withstand a extensive range of g-forces well beyond there g-force range for data collection. Designing a board to withstand similar (if not a higher g-force limit) woulds pose as a major and unnecessary challenge to the team.

### 3.9.6. Sparkfun - Triple Axis Accelerometer Breakout - ADXL345

Sparkfun's Triple Axis Accelerometer Breakout board includes the ADXL345 sensor that will measure accelerations in the X, Y, and Z directions. This sensor is a small, thin, low-powered 3-axis MEMS accelerometer with high resolution (13 bit) measurements up to +/- 16 g.[16] This sensor measures not only static acceleration for tilting-sense applications but also dynamic acceleration due to shocks or sudden movements. This sensor also includes a feature for free-fall detection. Embodied in a compact breakout board with designated locations for header integration and mounting holes, this sensor utilizes SPI or I2C interfaces that allow for simple, quick, and user-friendly integration. Extensive documentation is also available for this specific sensor. Applications of this sensor range from gaming devices to medical instruments due to its high measurement accuracy and g-force range and its low power requirements. Its high precision and reliability make it a good candidate for this project.[16]

Table 28. Pros and Cons of Sparkfun's Triple Axis Accelerometer Breakout - ADXL345

| Pros | Cons |
|---|---|
| 10,000 g shock survival | Accuracy degrades at higher g-forces |
| Free fall detection | Range of only +/- 16 g's |
| Can code sensor for +/- 2, 4, 8, or 16 g outputs | Backordered |

### 3.9.7. Sparkfun - Triple Axis Accelerometer Breakout - ADXL377

Sparkfun's Triple Axis Accelerometer Breakout board includes the ADXL377 sensor that has the ability to measure accelerations in the X, Y, and Z directions. This sensor is a small, low-powered 3-axis MEMS accelerometer that is capable of outputting a full-scale range of +/- 200 g's.[17] Being strictly analog, this sensor can be easily integrated on to multiple micro-controllers. Similar to most breakout boards of this type, this accelerometer breakout board has designated locations for header integration and mounting holes. This board also has signal conditioner for analog voltage outputs. This wide range of outputs allows for extreme measurements in motion, shocks, and vibration. This board is used commonly in applications involving high force events such as concussion investigation and head trauma detection making its high g-force scale ideal for this mission. The team also has heritage using this component, for a team member has utilized this specific breakout board on a previous sounding rocket mission.[17]

Table 29. Pros and Cons of Sparkfun's Triple Axis Accelerometer Breakout - ADXL377

| Pros | Cons |
|---|---|
| 10,000 g shock survival | Accuracy degrades at low g-forces |
| Range of +/- 200 g's | No I2C |
| Easy to code (analog pins) | Expensive (24.95 US dollars per unit) |
| Multiple resources and code documentation | |

### 3.9.8. Sparkfun - Triple Axis Accelerometer Breakout - MMA8452Q

Sparkfun's Triple Axis Accelerometer Breakout board includes the MMA8452Q sensor that can measure accelerations in the X, Y, and Z directions. This sensor is a thin, low powered 3-axis MEMS accelerometer with 12 bit resolution that is capable of outputting a full-scale range of +/- 2, +/- 4, or +/-8 g's of force.[20] Embedded interrupt functions allow for overall power saving relieving the host processor from continuously polling data. The device can be configured by the used to remain in low power mode during long periods of inactivity. This compact breakout board has designated locations for header integration and mounting holes, but this specific board already has the pre-soldered headers. The I2C interface allows for easy integration on to multiple micro-controllers. Applications using this sensor range from shock and vibration motoring to static orientation detection, making it a good candidate for this project.[20]

Table 30. Pros and Cons of Sparkfun's Triple Axis Accelerometer Breakout - MMA8452Q

| Pros | Cons |
|---|---|
| Comes with pre-soldered header | Only a maximum range of +/- 8 g's |
| Three channels of motion detection | Accuracy degrades at higher g-forces |
| Multiple resources and code documentation | Max 1.8m drop height unprotected |
| Can code sensor for +/- 2,4, or 8 outputs | |

### 3.10. Pod Power

The pods require a power source to function for the duration of the mission. The power source options evaluated are lithium-polymer, lithium-ion, nickel-metal hydride, and nickel-cadmium. All of the power sources listed are rechargeable and require a charger, which is not discussed since each choice needs it. The ability to recharge the power source is a customer requirement.

### 3.10.1. Lithium-Polymer (LiPo)

Lithium-polymer batteries are lightweight and capable of very high capacities packing into a small space. This option can be very useful for cases where space is an issue, but they are fragile and require maintenance. This option would require a battery protection circuit to ensure that the battery does not over-volt or under-volt as that would result in the battery dying for good. They are also sensitive to pressure on them on impacts, which can cause the batteries to be pierced leading to expansion with possible ignitions and explosions. The batteries would need a safe storage container to keep them in. Although the prices are high, only one battery would need to be used as the 1s (1-cell) nominally outputs 3.7 volts. The lithium-polymer pros and cons are located in table 31.[15]

Table 31. Lithium-Polymer Battery Pros and Cons

| Pros | Cons |
|---|---|
| High power density | Fragile to pressure and sharp edges |
| Lightweight | Require additional protection circuit for safety |
| Would only need to have 1 to get enough voltage | Risk of combustion |
| | Require more care and maintenance |
| | High cost |

### 3.10.2. Lithium-Ion (Li-ion)

Lithium-ion batteries are similar to lithium-polymer batteries with the high power density, but have a larger form factor and are usually heavier. The size of lithium-ion batteries is reasonable and since only one is required to get to 3.3 volts, then the volume taken up is relatively small. There is a combustion risk associated along with the requirement of a protection circuit. They do still require some maintenance to ensure stability, but a lot less than the lithium-polymer. They tend to deplete with age, which can be bad for longevity.[4]

Table 32. Lithium-Ion Battery Pros and Cons

| Pros | Cons |
|---|---|
| High power density | Require additional protection circuit for safety |
| Low cost | Depletes with age |
| Low maintenance | Risk of combustion |
| Would only need to have 1 to get enough voltage | |

### 3.10.3. Nickel-Metal Hydride (NiMH)

Nickel-metal hydride batteries are the most popular and widely available rechargeable battery for everyday use. They are often used in place of AA batteries as they have the same voltage and form factor. They are capable of very high capacities and contain no toxic materials; meaning that they could just be thrown away. Another positive is that unlike the lithium batteries they can be run all way down to 0 charge without worry of the battery being dead forever. A downside to this is that instead of having a voltage drop off as the battery dies there is a sudden cut of power. Another downside is that if left alone they will discharge quickly; losing their entire capacity in months if not used. These also have a long charging time and would require 3 to achieve the desired voltage, which would either mean waiting around or purchasing more of them. They have little to no maintenance and are non-combustible. The nickel-metal hydride battery pros and cons are located in Table[3] 33.

Table 33. Nickel-Metal Hydride Battery Pros and Cons

| Pros | Cons |
|---|---|
| No toxic materials | Sudden power cutoff |
| High capacity | Discharge if left unused |
| Can be completely discharged | Requires 3 to get required voltage |
| Non-combustible | Long charge time |

### 3.10.4. Nickel-Cadmium (NiCd)

Nickel-cadmium batteries are similar to nickel-metal hydride in that they are available in common sizes along with having the same sudden power cutoff feature. They are resistant to high voltage, which allows them to be charged very quickly. They are inexpensive but have a low power density and are not available in very large capacities. They have a memory effect, which means that if they are not fully discharged every use that they actually lose some of their maximum capacity. The nickel-cadmium battery pros and cons are located in Table[3] 34.

Table 34. Nickel-Cadmium Battery Pros and Cons

| Pros | Cons |
|---|---|
| Durable | Sudden power cutoff |
| Quick charging | Toxic |
| Inexpensive | Memory effect |
| | Low power density |

## 3.11. Pod GPS/GNSS

Absolute location based on GPS/GNSS for the pods is required to ensure that the pods actually landed within the desired deployment area. Since the accuracy of the pod deployment is 1 [m], normal GPS/GNSS would not be accurate enough to find absolute location. Due to this real time kinematics (RTK) in combination had to be used to get sub-meter accuracy. This option is very expensive and requires a large form factor. This brought the options down to having a GPS/GNSS unit with RTK on each pod or going around after the test is completed with a GPS/GNSS unit with RTK and recording the position in post. Since the cost of these units are so expensive and such a high percentage of the total budget, there was not an official trade study completed.

### 3.11.1. GPS/GNSS RTK Unit in Each Pod

Having a GPS/GNSS unit in each pod would be great to have location throughout the entire mission. There are many downsides to this option including it being very expensive, having a large form factor and increasing the power draw. The GPS in each pod pros and cons are located in Table 35.

Table 35. GPS in Each Pod Pros and Cons

| Pros | Cons |
|---|---|
| Accurate absolute positioning data | Very expensive |
| | Large form factor |
| | Increase power draw |

### 3.11.2. GPS/GNSS RTK Unit in Post

Having a GPS/GNSS unit in post would require more work during the actual test and mission as the positions would have to be recorded after the test. This option is also prone to error if the pods moved even a little during the actual test. This option is a lot cheaper though as only two units would need to be purchased instead of one per pod. The GPS in post pros and cons are located in Table 36.

Table 36. GPS in Post Pros and Cons

| Pros | Cons |
|---|---|
| Cheap in Comparison | Require recording positions after mission is completed |
| | Uncertainty of position |

# 4. Trade Study Process and Results

## 4.1. Rover Navigation

Table 37. Metrics for Path Algorithm Trade Study

| Metric | Weight | Driving Requirements | Rationale |
|---|---|---|---|
| Reliability/Accuracy | 0.40 | S1.3 | S - Software shall generate a path through the terrain to reach up to 10 waypoints. |
| Computational Speed | 0.30 | Testing Constraints | The algorithm chosen must run in a reasonable amount of time with the processing power provided in order to not push the limitations of the Jackal processor, as well as to allow for rapid testing. |
| Algorithm Complexity | 0.20 | Scope | The development of the algorithm must not be so complicated as to exceed the scope of the project. |
| Code Heritage | 0.10 | N/A | The ideal algorithm will have plenty of pre-existing implementations across different software packages |

Table 38. Explanation of Path Algorithm metrics

| Metric | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| Computational Speed | >1 min | >45s | >30s | >15s | >5s |
| Code Heritage | No documentation/No ROS nodes | Some documentation, no ROS nodes | Some documentation, ROS nodes exis | Articles and extensive documentation, ROS nodes exis | Articles, wikis, papers, ROS nodes exist, MATLAB implementations, Python algorithms |
| Algorithm Complexity | Out of scope/Too large to process | Difficult | Inconvenient | Desirable | Trivial |
| Reliability/Accuracy | Inherently inaccurate , does not meet path accuracy requirement | Fairly inaccurate but meets path requirement sometimes | Inaccurate but can be made more accurate (heuristic), meets path requirement | Accurate and correctable, meets path requirement | Accurate without correction |

Table 39. Path Algorithm Trade Study

| Metric | Weight | Dijkstra's Algorithm | A* | D* | RRT |
|---|---|---|---|---|---|
| Reliability/Accuracy | 0.40 | 5 | 4 | 5 | 2 |
| Computational Speed | 0.30 | 2 | 5 | 3 | 2 |
| Algorithm Complexity | 0.20 | 5 | 4 | 2 | 2 |
| Code Heritage | 0.10 | 5 | 5 | 3 | 3 |
| Total | 1 | 4.1 | 4.4 | 3.6 | 2.1 |

The reliability/accuracy of the algorithm is the first metric in this trade study and carries the highest weight of 0.40. This metric is assigned such a high weight due to the fact that the rover must be capable of generating a path to all waypoints that is both the shortest possible path as well as the safest. Without an accurate path, the rover would not be able to meet all waypoints within the allotted time, nor would it be able to deploy the pods correctly. The first algorithm in the study is Dijkstra's algorithm which gained a 5 for accuracy. This is due to the fact that given a graph with quantified weights for each possible path, the algorithm will find the shortest path to the endpoint, due to the fact that it is an exhaustive search of the grid, and its accuracy can be proved mathematically.[11] The A* algorithm received a slightly lower score of 4, and this is due to the fact that depending on the “cheapness” of the heuristic function, the algorithm can actually generate paths that are not the shortest, which could violate the timing requirement for the mission. The third algorithm, D*, was give a 5 for this category, because it performs a total search of the solution space for the given map, much like Dijkstra's. The RRT algorithm was given the lowest of the four scores, which was a two, due to the fact that, as the name implies, the search tree branches out randomly until it finds the end node, meaning the generated path is not always the best one for the mission, just the first to finish the search.

The second metric of the trade study was the computational speed, with a weight of 0.30. The values in this table are still estimations, but are representative. This aids not only with the final usability, but also with testing, as many different paths will need to be generated before a final algorithm can be chosen, and a high computational speed would require more time to test iteratively. This explains the high weight of 0.30 for this metric. Dijkstra's algorithm was given a two for this metric, due to the fact that it performs an entire grid search of the solution space in order to find the shortest path, meaning unnecessary nodes are checked and time is wasted searching for solutions that do not exist. A* was given a 5, because given a decent heuristic, the algorithm will be able to “short-circuit” certain nodes that may not be heading in the direction of the finish line, which means paths not providing the optimal solution will not even be considered.[7] D* was given a 3 for this section, because even though it also performs a full search of the solution space, it can be upgraded with a heuristic function (Focussed D*) to allow for some shortening in the computation speed, giving it a slight edge over Dijkstra's. RRT also scored low in this section with a two, due to the fact that it randomly searches the entire map until it reaches the end, so depending on the topology, it could end up being even slower than Dijkstra's.

The third metric studied was the algorithm complexity, which was given a weight of 0.20. The slightly lower weight was attributed to the fact that while still important for time management and speed of development, the complexity was not quite as important as directly meeting the mission requirements. (i.e. if the most complicated algorithm solved the problem the best/most efficient way, it would still likely be a front-runner in the study despite its complexity) For this metric, Dijkstra's algorithm was given a five, because the algorithm involves no complicated math and only a few data structures to monitor open and closed nodes in the grid, so implementation is straightforward. A* was given a four for this category, only due to the fact that the heuristic may or may not be slightly more complicated to model depending on the mission requirements and projected topology. D* was given a three for complexity, since it not only has to account for dynamic replanning (i.e., Dijkstra's with an added system to re-weight certain nodes), but also operates from the final point back to the starting node[18], requiring the use of backpointers to keep track of the current path. The RRT method scored a two for this section, as the stochastic model for the random node choice could become arbitrarily complicated if not modeled correctly. More similar to a Monte Carlo study of different paths, the RRT method is more complicated in the sense that it seeks unnecessary solutions more than any of the other methods (i.e. more “moving parts”).

The final metric studied was the code heritage, which was given a weight of 0.10. This is the smallest weight of any of the metrics, and is due to the fact that while important, the code heritage does not necessarily determine the success or failure of one particular algorithm. Rather it determines the ease of implementation during system integration and testing, as more resources mean easier debugging and development. For this metric, Dijkstra's algorithm was given a 5, as the method has widely been used for over half a century and plenty of documentation exists, as well as MATLAB functions and ROS nodes. A* was also given a five for this metric, as this algorithm has also been documented well for its lifespan and plenty of pre-existing code exists for implementation and testing. The D* algorithm was given a three for this method, as a decent amount of articles and documentation exist, but not as many pre-existing implementations available open-source exist, but there are ROS packages, so it is still viable in terms of heritage. RRT was given a three for the same reasons, as RRT has been used on many autonomous projects, but pre-existing code for the algorithm is not necessarily as easy to find with “out-of-box” compatitibility.

## 4.2. State Estimation

Table 40. Metrics for State Estimation Trade Study

| Metric | Weight | Driving Requirements | Rationale |
|---|---|---|---|
| Complexity | 0.20 | Scoping | The development of the algorithm must not be so complicated as to exceed the scope of the project, and unnecessary additions to the algorithm must be avoided if they are not directly needed for the mission |
| Accuracy | 0.30 | S2.1 | S - The software shall combine RF-Localization and inertial/odometry position estimates in order to enhance position estimation accuracy |
| Computation Speed / Resource Scaling | 0.20 | Testing/Validation | The algorithm chosen must run in a reasonable amount of time with the processing power provided in order to not push the limitations of the Jackal processor, as well as to allow for rapid testing |
| Assumption Validity | 0.30 | N/A | The algorithms rely on several key assumptions and should only be used if the assumptions are valid for the system |

Table 41. Explanation of state estimation metrics

| Metric | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| Complexity | Daunting | | Challenging | | Manageable |
| Accuracy | Vanilla Kalman Filter | | > Kalman Filter | | Kalman Filter |
| Computational Speed / Resource Scaling | > KF | | Kalman Filter | | Resource Scaling |
| Assumption Validity | Invalid assumption | | Partially met | | All assumptions met |

Table 42. State Estimation Trade Study

| Metric | Weight | Kalman Filter | Extended Kalman Filter | Unscented Kalman Filter | Histogram Filter | Particle Filter |
|---|---|---|---|---|---|---|
| Complexity | 0.20 | 3 | 1 | 2 | 5 | 4 |
| Accuracy | 0.30 | 1 | 3 | 5 | 1 | 3 |
| Computational Speed / Resource Scaling | 0.20 | 3 | 2 | 2 | 5 | 5 |
| Assumption/Constraint Validity | 0.30 | 1 | 3 | 5 | 3 | 5 |
| Total | 1 | 1.8 | 2.4 | 3.8 | 3.2 | 4.2 |

Algorithm complexity was taken into consideration when determining which state-estimation algorithm to use, but was weighted somewhat low because all of the algorithms discussed are of a similar class. At the bottom we have the Kalman filter options– while the vanilla filter is complex, the extended and unscented add another layer of complication. The histogram filter is the most simple option of all, consisting of a summation of probabilities over a discretized space. The particle filter is simpler than a Kalman filter, but still more involved than the histogram.

The requirements on accuracy are stringent for the project as a whole, and so this metric was weighted relatively high for the study. The Unscented Kalman performs as well or better than the vanilla filter in all cases, and better than the extended in general. The histogram and particle filter scored lower here because of the need to discretize– which introduces some error– and because of the computational deadends that occasionally arise in the particle filter.

The processing power on the rover will be limited, and the state algorithm must share time with controls and pod deployment tasks. The Kalman filters are all similar in time to eachother, with the more complicated filters taking slightly more time. The nonparametric algorithms have the advantage here with their ability to scale resources in accordance with what is available at the time. In the case of the particle filter, the algorithm can reduce the number of particles if computer resources are limited.

The Kalman filters rely on the assumptions of linearity or the ability to linearize accurately– the algorithms should not be used if the assumptions cannot be validated. The vanilla filter relies on linearity, which immediately fails since the odometer measurement is non-linear. The Extended Kalman Filter does better, but the Unscented Kalman Filter generally performs better in non-linear systems. The histogram filter relies on being able to discretize the whole state space, which may prove difficult to implement with complex topology. The particle filter has no such restrictions on region size or linearity.

### 4.3. Pod Deployment Method

Table 43. Metrics for Pod Deployment Trade Study

| Metric | Weight | Driving Requirements | Rationale |
|---|---|---|---|
| Fabrication Difficulty | 0.10 | M3 | The deployment method must be fabricated within the academic year while remaining within the scope of this project and team abilities. This is weighted at .10 because while the design must be feasible, the team has some resources available to assist with a more complicated design. |
| Impact Force (g's) | 0.20 | P5.4 | The pod must be able to function upon landing. This is rated at .20 because the pods are a CPE and the project cannot reach success if the pods do not survive deployment. In addition, the customer requires that the pod be durable enough to withstand repeated deployments, therefore it is desired to reduce the impact energy from deployment. |
| Deployment Rate | 0.10 | D3.5 | The pods must be deployed at the rate dictated by the algorithm such that navigation is not inhibited by the deployment. This is rated at .10 because the accuracy of deployment is more pertinent to achieving the levels of success than the speed of deployment. |
| Controls Complexity | 0.25 | M3 | The deployment method must be achievable within the academic year without going beyond the scope of this project. This is rated the highest due to the complexity of other controls elements of the project. Reducing the complexity of controls while maintaining an accurate deployment significantly reduces scope of the project. |
| Environment | 0.15 | D3.3 | The pod deployment must be repeatable in any reasonable environment. Factors such as wind and hills should not prevent the system from deploying pods within 1m of the desired location. This is important to reducing the error in the RF network placement and demonstrating reliability of the system, so it is weighted at .15. |
| Pod Stopping Difficulty within 1m | 0.20 | D3.3 | The pods must be stopped within the given tolerance in order to properly provide the most accurate information for navigation and localization. This is rated highly at .20 because it applies directly ties to the levels of success for the deployment system, but also because the accuracy of the localization is directly related to the accuracy of the deployment system. |

Table 44. Explanation of pod deployment metrics

| Metric | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| Fabrication Difficulty | Impossible | Difficult | Inconvenient | Desirable | Trivial |
| Impact Force | Guaranteed pod damage | Likely pod damage | Possible pod damage | Unlikely pod damage | Negligible pod damage |
| Deployment Rate | 1 min | 45 sec | 30 sec | 15 sec | 1 sec |
| Controls Complexity | Outside of scope | Difficult | Inconvenient | Desirable | Trivial |
| Environment | System Failure | Detrimental trajectory deviation | Predictable | Trivial trajectory deviation | No effect |
| Necessary pod stopping mechanism | Impossible | Complicated machining required | COTS mechanism available | Trivial machining required | No stopping mechanism needed |

Table 45. Pod Deployment Trade Study

| Metric | Weight | Drone | Mechanical Arm | Pneumatic Cannon | Spring Cannon | Disk Thrower | Pitching Machine | Catapult |
|---|---|---|---|---|---|---|---|---|
| Fabrication Difficulty | 0.10 | 3 | 1 | 2 | 3 | 3 | 4 | 2 |
| Impact Force | 0.20 | 5 | 4 | 3 | 3 | 4 | 3 | 3 |
| Deployment Rate | 0.10 | 1 | 1 | 4 | 4 | 4 | 4 | 2 |
| Controls Complexity | 0.25 | 1 | 4 | 3 | 3 | 3 | 3 | 3 |
| Environment | 0.15 | 1 | 2 | 4 | 4 | 2 | 4 | 4 |
| Necessary pod stopping mechanism | 0.20 | 5 | 4 | 2 | 2 | 2 | 2 | 2 |
| Total | 1.00 | 2.8 | 3.1 | 2.95 | 3.05 | 2.95 | 3.15 | 2.75 |

Fabrication difficulty is the first metric in the trade study and carries a relatively small weight at 10%. The highest scoring design was the pitching machine at 4. Because pitching machines have such extensive heritage and are a widely available for prototyping, this method was desirable. A score of 3 was given to the drone, the spring cannon, and the disk thrower primarily due to the complexity associated with making these mechanisms repeatable. While basic versions of each can be constructed, fine-tuning these three deployment methods and their interface with the pods would be inconvenient. The catapult and pneumatic cannon received a score of 2 because manufacturing a compact yet accurate design to fit on the rover would be difficult, and modifications to the design would be troublesome. The mechanical arm received the lowest score as creating an arm capable of spanning 20m that fits on the rover would be impossible given the project's time constraints.

Impact force carried 20% of the weight on a scale of negligible pod damage to guaranteed pod damage. In this category drone received the highest score because the drone would be able to hover and slowly decrease its elevation until the pod was placed on the ground. The mechanical arm and disk thrower both received 4s because in both instances, the pod is only being subjected to a relatively small impact velocity. In the case of the arm that would be a drop from a trivial height and in the case of the disk thrower, the pod itself would act like a lifting body and decrease the impact velocity. The remaining options all received 3s as they involve projectile motion of some sort and will not be slowed before impacting the ground.

The deployment rate was taken into account with a weight of 10%. In this category the cannons, disk thrower, and pitching machine all received a score of four. For all four methods, the reloading design will be similar and thus so will the deployment rate. The catapult received a much lower score at 2 because of the excess energy in the system following deployment. Without active slowing mechanisms, the catapult will take time to slow down and return to a launch position. The mechanical arm and drone both received 1s as they are required to physically move to the drop location. For considerable distances such as 15m, this task could take upwards of a minute.

The controls complexity metric carried 25% of the weight, on a scale of trivial to outside of scope. The drone came in at a score of 1 since the controls required for a drone to autonomously fly to a position and then return to a potentially moving rover is a project within itself, especially without GPS. With many difficult elements in this project and a limited time frame, drone controls is outside of the scope of the project. All projectile methods received a score of 3 since calculations would need to be made to determine power, azimuth and elevation angle. Potentially complicated designs would need to be implemented to ensure that these projectile deployments can reach all locations. Finally, the mechanical arm got the highest score since only an azimuth angle and distance is necessary, both of which are easy to monitor and control.

Environmental effects were taken into account with 15% of the weight. The drone once again came in with a score of 1 indicating that environmental factors could cause system failure. Wind would affect the drone's ability to land back on the rover to reload as well as affect the precision of the placement. If the drone crashes or is unable to return to the rover, the deployment method is a complete failure. A score of 2 was given to the mechanical arm since elevation changes would prevent the mechanical arm from accurately deploying the pods, or entirely hinder deployment. The disk thrower was also scored a 2 due to wind affecting the disk during flight and drastically altering the location of deployment. All other methods received a 4 since the compatible pod shapes will likely be negligibly affected by wind and other factors.

The final metric was the necessity of a pod stopping mechanism. The lowest score was given to everything but the drone and the mechanical arm. The drone received the highest score as it will be able to place the pod gently on the ground without any additional pod stopping assistance. The projectile methods may require a complicated method of stopping the pods since these are high kinetic energy deployments at long ranges. Since a disk thrower would be deploying frisbee-shaped pods, these would land with relatively low translational kinetic energy, but the rotational energy could be detrimental if they were to land on edge. For this reason a complicated stopping mechanism could be needed to stop the disks from rolling. The mechanical arm was second only to the drone, receiving a 4. The mechanical arm would drop the pod at a low height and level thus only a basic mechanism would be necessary.

### 4.4. Pod Structures

The trade study for pod structure was done to compare polycarbonate, polyurethane, and ABS materials. The weighting, driving requirements, and rationale can be shown in Table 46. More information on the specifications of how the metrics were given score can be shown in Table 47. The actual trade study can be shown in Table 48.

Table 46. Metrics for Pod Structure Trade Study

| Metric | Weight | Driving Requirements | Rationale |
|---|---|---|---|
| Strength | 0.20 | P5.4 | The pods must be capable of surviving the initial impact after deployment as required by 5.4. This is tied for second in the weighted as the pod must be strong enough to survive the landing. |
| Cost | 0.05 | Budget | Cost is not very important to the pod structure as quality materials should be used. If the pod structure does not survive, then the mission is a failure. |
| Manufacturability | 0.20 | N/A | There is no direct requirement discussing that the pod should be able to be manufactured, but it is an overlying requirement. If the pod shape cannot be manufactured out of the material requested, then it is obviously not an option. This is tired for second as the pods must be able to be made. |
| Weight | 0.15 | P5.1.1 | There is a weight requirement based on P5.1.1 and the pods should be designed to be as light as possible. |
| Longevity | 0.30 | P5.4.2 | The most important aspect of the pods is the re-usability, so being able to reuse the pods is the highest priority. This metric is weighted the highest because they must be able to last. |
| Coefficient of Restitution | 0.10 | P5.4.4 | The pods must be able to stay in place upon landing and throughout the mission. This is a requirement for precise deployment precision for localization accuracy. |

Table 47. Explanation of Pod Structure Metrics

| Metric | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| Tensile Strength | <4000 psi | 4000-6000 psi | 6000-8000 psi | 8000-10,000 psi | >10,000 psi |
| Cost | $>\$2.00/\text{in}^3$ | $\$2.00\text{-}\$1.50/\text{in}^3$ | $\$1.50\text{-}\$1.00/\text{in}^3$ | $\$1.00\text{-}\$0.50/\text{in}^3$ | $<\$0.50/\text{in}^3$ |
| Manufacturability | Impossible | Difficult | Inconvenient | Desirable | Trivial |
| Weight | >1.5 g/cc | 1.5-1.1 g/cc | 1.1-0.7 g/cc | 0.7-0.3 g/cc | <0.3 g/cc |
| Longevity | Single Use | 2-10 Uses | 11-20 Uses | 21-30 Uses | >30 Uses |
| Coefficient of Restitution | 1-0.81 | 0.8-0.61 | 0.6-0.41 | 0.4-0.21 | 0.2-0 |

Table 48. Pod Structure Trade Study

| Metric | Weight | Polycarbonate | Polyurethane | ABS |
|---|---|---|---|---|
| Tensile Strength | 0.20 | 4 | 2 | 1 |
| Cost | 0.05 | 4 | 1 | 4 |
| Manufacturability | 0.20 | 4 | 2 | 5 |
| Weight | 0.15 | 2 | 4 | 3 |
| Longevity | 0.30 | 5 | 4 | 4 |
| Coefficient of Restitution | 0.10 | 2 | 4 | 2 |
| Total | 1.0 | 3.8 | 3.05 | 3.25 |

Tensile strength was the first metric compared in the pod material trade study. It has a weight of 20% because it is crucial to the success of the mission that the pod and its internal components survives deployment impact. The polycarbonate has the highest tensile strength and is the most adventitious material to use when designing for survivability. ABS has the lowest tensile strength as impurity is introduced to the system when using additive manufacturing.

The cost of the material is the next metric weighing in at only 5%. This is because all the compared materials are very similar in price and will not make a significant difference in the overall budget. That aside, the polyurethane is much more expensive than the polycarbonate and ABS filament which are both very similar in price.

The ability to modify store bought materials to fit design specifications is a significant factor in choosing a material especially when multiple pods must be produced which is why manufacturability holds a weight of 20%. The ABS filament would be the best material to use as it can be used in additive manufacturing. Once the CAD models are made to spec, ABS is used to 3D print the necessary components that can be assembled. Polycarbonate can be drilled, cut, and bent with ease but would still require more manpower to create.

Weight is an important factor for the pod material because the rover has a limit on how much weight it can hold and therefore is given a ranking of 15%. Polyurethane is the least dense material in the trade study and therefore is given the highest ranking while the polycarbonate is the most dense with ABS in the middle. While weight is an important metric, it may be difficult to use this as a driving factor for choosing a material because different amounts of material will be necessary for different pod shapes, sizes, or required strength, all of which are yet to be determined as of this point. Longevity is the most important factor in this trade study and has an associated weighting of 30%. Surviving multiple deployments is a system requirement of the pods and will also save the team money in replacement parts. Polycarbonate is very shock-proof and will be able to survive multiple deployments before showing signs of fatigue. ABS is more brittle than polycarbonate and may crack under pressure. The polyurethane is a spongy material and would absorb impacts very well but may rip or tear after multiple deployments.

The last metric studied was the coefficient of restitution of each material. This was included to help with the deployment accuracy and is weighted 10%. The weighting of this metric was low because there are many ways this issue can be negated but the overall coefficient of restitution for polyurethane was much lower than the polycarbonate and ABS (as to be expected from its cell structure and density).

### 4.5. Localization/Pod Processing

In this trade study four design options are considered, using the Localino dev board, designing a PCB for the STM32F103 processor,using a DWM1001 dev board, and using an Adafruit Feather nRF52 Bluefruit. Below are the metric that will be used to grade the pod processing.

There are five metrics that will be used to grade the trade study: power consumption, cost, size, complexity, and heritage of the processors. Complexity was chosen to be the most heavily weighted metric, due to the short project cycle is. This idea feeds into one of the two second most heavily weighted metrics, the heritage or the processors. The second most heavily weighted metric is the cost of the processor. The design of the pods will be replicated ten times making the impact on the budget ten time higher. The last two metrics, the power consumption and size of the processor, are the lowest weighted but still have a considerable impact on the trade study. The power draw of the processor impacts how long the pods can operate in the environment in high and lower power modes. The processor will likely be the largest component in the pods and therefore will have the largest impact on the size of the pods. It is important to keep the size low to avoid interfering with the deployment method.

In Table 49 is a summary of each metric, their weights, the requirements they were derived from, and the rationale for the weighting of each metric, followed by Table 49 which scores each processor.

Table 49. Metrics for Pod Processing Trade Study

| Metric | Weight | Driving Requirements | Rationale |
|---|---|---|---|
| Power | 0.15 | P6.2, P6.3, P6.4 | Pods must operate for a long duration while cycling between high and low power modes, therefore it is important to minimize overall power consumption to reduce the size of the power source on the pod |
| Cost | 0.20 | Budget | There will be 10 pods that will be designed and constructed; this will be a large impact on the project budget |
| Size | 0.15 | P5.1 | The pods must not be exceedingly large in order to not interfere with the deployment system. The total volume of the pods will likely be determined by the pods processor due to it being the largest component in the pods |
| Complexity | 0.30 | M5 | The team must be able to implement a solution with the skills available to undergraduate students |
| Heritage | 0.20 | N/A | The goal is build upon someone else's work to meet the requirements of the project, not reinvent the way RF localization works |

Table 50. Explanation of Pod Processing Metrics

| Metric | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| Power (Not including sensors or DW1000) | > 5 W | < 3 W | < 1 W | < 0.6 W | < 0.3 W |
| Cost | > $150 | $150 | $100 | $50 | $25 |
| Size[cm$^2$] | > 30 | > 25 | > 20 | > 15 | > 10 |
| Complexity | Develop hardware and software solution from scratch. | Use COTS hardware and develop software and firmware solutions from scratch. | Use COTS hardware and develop software and firmware solutions with readily available debugging tools | Use COTS hardware and firmware while developing a software solution | Build upon a previously COTS hardware using provided software |
| Heritage | New Product/Untested | Processor has been used for any purpose | Processor used for localization before but has not undergone extensive testing | Processor has been extensively used for RF localization but not by any campus personal | Campus personal available that have developed localization software using the processor |

Table 51. Trade study results for Pod Processing

| Metric | Weight | Localino processor | Making a PCB for a STM32F103 | DWM1001 processor | Adafruit Feather nRF52 Bluefruit |
|---|---|---|---|---|---|
| Power | 0.15 | 4 | 5 | 4 | 5 |
| Cost | 0.20 | 3 | 3 | 4 | 5 |
| Size | 0.15 | 4 | 3 | 2 | 4 |
| Complexity | 0.30 | 5 | 1 | 4 | 4 |
| Heritage | 0.20 | 5 | 1 | 3 | 3 |
| Total | 1 | 4.3 | 2.3 | 3.5 | 4.15 |

### 4.6. Pod Electronics

There are two trades studied completed for the pod electronics. The first study will investigate three environmental sensor breakout boards. Four metrics are examined in this trade study including number of environmental senors, accuracy of the altimeter, power consumption, and cost. The second study involves the investigation of three accelerometer breakout boards. Four metrics, similar to the environmental sensor breakout board trade study, are examined in this trade study include G-force loading, accuracy, power consumption, and cost.

### 4.6.1. Environmental Breakout Board

Table 52. Metrics for Environmental Sensor Breakout Board

| Metric | Weight | Driving Requirements | Rationale |
|---|---|---|---|
| Number of Environmental Sensors | 0.10 | P6.1 | More sensors that are available allows for more environmental data that can be recorded. More types of data will provided more science value in the pod's landing location and fulfill P6.1. |
| Accuracy of Altimeter | 0.30 | P6.1.4 | Having higher altitude accuracy and using the attitude data to assist in the verifying deployment accuracy to fulfill P6.1.3. |
| Power Consumption | 0.20 | P5.3 | Environmental data sensing will continue throughout the mission as well as after the mission using the rechargeable power source to fulfill P5.3. |
| Cost | 0.40 | Budget | Minimized cost is ideal for the selected breakout board. |

Table 53. Explanation Environmental Breakout Board Metrics

| Metric | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| Number of Environmental Sensors | 1 | 2 | 3 | 4 | 5+ |
| Accuracy of Altimeter | > 4.0 m | 3.99 m - 3.0 m | 2.99 m - 2.0 m | 1.99 m - 1.0 m | < 1.0 m |
| Power Consumption | > 8 mW | 7.99 mW - 6.00 mW | 5.99 mW - 4.00 mW | 3.99mW - 2.0 mW | < 1.99 mW |
| Cost (per unit) | > $50 | $49 - $40 | $39 - $30 | $29 - $20 | < $10 |

Table 54. Environmental Breakout Board Trade Study

| Metric | Weight | MPL3115A2 | BME680 | CC5811/BME280 |
|---|---|---|---|---|
| Number of Environmental Sensors | 0.10 | 3 | 5 | 5 |
| Accuracy of Altimeter | 0.30 | 5 | 4 | 3 |
| Power Consumption | 0.20 | 4 | 4 | 1 |
| Cost | 0.40 | 4 | 3 | 2 |
| Total | 1.0 | 4.2 | 3.7 | 2.4 |

The reason the team decided to include a metric involving altimeter accuracy instead of the accuracy of pressure is because altimeter data can possibly be utilized to help verify the accuracy of the deployment method and assist in possible post-mission analysis.

Environmental sensing will continue throughout the mission as well as after the mission making power consumption a valuable metric for consideration. P5.3 requires the pods to not only operate for the duration of the mission but after the completion of the mission for long term sensing. Power draw is relatively low compared to the power draw required for the other on board units but it is important to minimize the power draw if possible.

Cost is the largest consideration in this trade study because of the team's limited budget. Since there will be multiple pods designed and manufactured by the team, it is important that the selected sensor is reasonably priced and can be replaced easily if necessary. Accuracy and power consumption can be compromised as long as the sensor is reasonably priced and simply completes the job of collecting data (no matter its accuracy).

### 4.6.2. Accelerometer Breakout Board

Table 55. Metrics for Accelerometer Breakout Board

| Metric | Weight | Driving Requirements | Rationale |
|---|---|---|---|
| G-force Loading | 0.3 | P5.4,P6.1 | The accelerometer, as stated in P6.1, should ideally measure the g-forces acting on the pod during impact. The data from the accelerometer sensor data could possibly be used in order to study the shock and impact of the pods post-deployment as well as the force imparted on the pod from the deployment mechanism as described in P5.4. |
| Accuracy | 0.2 | P6.1.4 | High accuracy of the accelerometer sensor would assist in any post-mission shock analysis by providing the team the ability to accurately quantify the amount of g-forces the pods experiences as well collect useful data to fulfill P6.1.4. |
| Power Consumption | 0.1 | P5.3 | Accelerometer sensing will continue throughout the mission as well as after the mission. The pods must operate for the duration of the mission as well as after the completion of the mission for long term sensing as described in P5.3. |
| Cost (per unit) | 0.40 | Budget | Cost is the largest consideration in this trade study because of the team's limited budget. Since there will be multiple pods designed and manufactured by the team, it is important that the selected sensor is reasonably priced and can be replaced easily if necessary. |

Table 56. Explanation for Accelerometer Breakout Board Metrics

| Metric | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| G-force Loading | < 5 g's | 5.01 g's - 15.00 g's | 15.01 g's - 25.00 g's | 25.01 g's - 35.00 g's | > 35.01 g's |
| Accuracy | > 200mV/g | 199.99 mV/g - 150.0 mV/g | 149.9 mV/g - 100.0 mV/g | 99.9 mV/g - 50.0 mV/g | < 49mV/g |
| Power Consumption | > 700 W | 699 W - 500 W | 499 W - 400 W | 399 W - 300 W | < 299 W |
| Cost (per unit) | > $25 | $24.99 - $20 | $19.99 - $15 | $14.99 - $10 | < $9.99 |

Table 57. Accelerometer Breakout Board Trade Study

| Metric | Weight | ADXL345 | ADXL377 | MMA8152Q |
|---|---|---|---|---|
| G-Force Loading | 0.30 | 3 | 5 | 2 |
| Accuracy | 0.20 | 5 | 4 | 1 |
| Power Consumption | 0.10 | 5 | 1 | 3 |
| Cost (per unit) | 0.40 | 3 | 2 | 3 |
| Total | 1.0 | 3.6 | 3.2 | 2.3 |

The accelerometer breakout board should ideally measure the g-forces acting on the pod during impact. The data from the accelerometer sensor data could possibly be used in order to study the shock and impact of the pods post-deployment as well as the force imparted on the pod from the deployment mechanism. If the range of g-force range of the sensor is too small, the sensor will over-saturate and the amount of shock experienced by the pod cannot be quantified during possible post-mission analysis. Higher g-force rating mitigate the chances of over-saturation.

Accuracy of the accelerometer sensor is important for studying the impact of the pods, but it is not the most important metric to consider. High accuracy of the accelerometer sensor would assist in any post-mission shock analysis by providing the team the ability to accurately quantify the amount of g-forces the pods experiences. However, even though high accuracy would be ideal, the team does not need an exact g-force reading in order to achieve our mission objectives. The accuracy would help assist in possible post-mission analysis and assist in fulfilling DR 5.3.

Accelerometer sensing will continue throughout the mission as well as after the mission. P5.3 requires the pods to not only operate for the duration of the mission but after the completion of the mission for long term sensing. Even if the power consumption is small for the sensor, it is a necessary metric to examine for the overall power budget.

Cost is the largest consideration in this trade study because of the team's limited budget. Since there will be multiple pods designed and manufactured by the team, it is important that the selected sensor is reasonably priced and can be replaced easily if necessary.

### 4.7. Pod Power

The trade study for pod power was done to compare LiPo, Li-ion, NiMH, and NiCd batteries. The weighting, driving requirements, and rationale can be shown in Table 58. More information on the specifications of how the metrics were given score can be shown in Table 59. The actual trade study can be shown in Figure 60.

Table 58. Metrics for Pod Power Trade Study

| Metric | Weight | Driving Requirements | Rationale |
|---|---|---|---|
| Volume | 0.15 | P5.1.2 | There is a size requirement of the pod based on P5.1.2 and the batteries must fit within the pod. This is tied for lowest as the batteries are all of similar sizes. |
| Weight | 0.25 | P5.1.1 | There is a weight requirement based on P5.1.1 and the pods should be designed to be as light as possible. This is tied as most important because the lighter the pods are, the more the rover is able to hold. |
| Cost | 0.15 | Budget | Cost is very important to the pods as lower cost means more pods. It is tied for lowest because in relation to the whole cost of the pod the batteries do not account for much. |
| Complexity | 0.20 | M5,M6 | The pods must function and making the power require more hardware adds more chances of failure. This metric also relates a little to volume and weight as more hardware leads to a bigger and heavier power source. |
| Durability | 0.25 | P5.3.1, P5.4.2 | The batteries must be capable of having many uses to fulfill P5.3.1 to be able to be recharged. This is useful for the pods being able to be reused and for testing when the pods will have to be consistently used for longer periods of time than the actual mission. |

Table 59. Explanation of Pod Power Metrics

| Metric | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| Volume | > 30,000 mm$^3$ | 25,001 mm$^3$ - 30,000 mm$^3$ | 17,501 mm$^3$ - 25,000 mm$^3$ | 10,000 mm$^3$ - 17,500 mm$^3$ | < 10,000 mm$^3$ |
| Weight | > 200 g | 151 - 200 g | 101 - 150 g | 51 - 100 g | < 50 g |
| Cost | > $ 8.00 | $ 6.01 - $ 8.00 | $ 4.01 - $ 6.00 | $ 2.00 - $ 4.00 | < $ 2.00 |
| Complexity | Requires additional circuits | | Requires additional wiring | | Works out of the box |
| Durability | < 100 uses | 100 - 350 uses | 351 - 700 uses | 701 - 1000 uses | > 1000 uses |

Table 60. Pod Power Trade Study

| Metric | Weight | LiPo | Li-ion | NiMH | NiCd |
|---|---|---|---|---|---|
| Volume | 0.15 | 4 | 3 | 1 | 1 |
| Weight | 0.25 | 5 | 2 | 4 | 5 |
| Cost | 0.15 | 4 | 3 | 2 | 4 |
| Complexity | 0.20 | 1 | 1 | 3 | 3 |
| Durability | 0.25 | 2 | 3 | 5 | 4 |
| Total | 1.0 | 3.15 | 2.35 | 3.3 | 3.6 |

# 5. Selection of Baseline Design

## 5.1. Rover Navigation

From the trade study results for the navigation algorithm, the top two choices the team intends to continue pursing are Dijkstra's algorithm and the A* algorithm. Both choices were only within 0.3 points of each other, so both must be studied more in depth in order to make a more informed choice for the final design. The front-runner, A*, stands out due to its faster processing and directional capability through the use of the heuristic function. Both options will provide the shortest path for the rover, but the development of a robust A* algorithm will allow for much more rapid prototyping, easier testing, and a more efficient product for the final mission.

Both Dijkstra's and A* are usually developed for systems with a single start and endpoint, so the navigation team will be responsible for studying the implementation of multiple waypoints within the algorithm to allow the rover to extend the capability of either algorithm to a multiple ending-node design. Not much documentation exists on such a design, so it will be a challenge the team will have to face regardless of the algorithm chosen. Figure 10 below displays a rudimentary CONOPS based on the functionality of both algorithms.

Figure 10. CONOPS for a Path Algorithm

## 5.2. Feedback Control Law

As mentioned previously, no formal trade study was conducted for the controller used for the rover navigation, as PID was the clear frontfunner based on availability and accuracy. As a consequence, the navigation algorithm will be fitted with a PID controller to monitor and maintain path tracking for the duration of the mission. This PID will ideally be implemented using ROS nodes in tandem with pre-eixisting hardware on the Jackal, but may also be designed by the team depending on the interfacing capability of available nodes, as well as the robustness.

The PID controller will receive data from the on-board state estimation sensors, as well as ranging estimates from the pod, and will determine and command corrections for the rover based upon how deviation is detected in the aforementioned measurements. For tuning, the mission will be simulated (directly in the open-source simulation software or in MATLAB) and the controller gains will be adjusted in order to meet the accuracy requirements for path tracking and waypoint timing. To facilitate this process, the use of ROS nodes and the built in tuning interface will be used extensively in order to avoid having to develop the entire control law and corresponding tuning program from scratch. Extensive resources exist regarding how to accurately model and develop PID controllers, and both Professor Nisar Ahmed and Steve McGuire have experience with their development, so the team will be capable of dealing with any issues that arise with the controller design.

## 5.3. State Estimation

Based on the results of the trade study, a particle filter emerges as the clear choice. The ability to scale computational resources will be key, and the relative simplicity will aid with testing and debugging later in the project lifecycle. As the runner up, the Unscented Kalman Filter will be the next choice if unforeseen issues arise with the implementation of a particle filter. In previous year's senior projects teams have had difficulties with state estimation, but the customers Professor Ahmed and Steve McGuire have offered their assistance in implementing the state estimation algorithm.

## 5.4. Pod Deployment Method

Based on the trade studies above, the top three choices for the deployment mechanisms are the pitching machine, the mechanical arm, and the spring cannon. Because all three of these devices scored within a tenth of a point of each other, a final feasibility discussion was conducted which ultimately resulted in the removal of the mechanical arm as a viable solution method. The actual fabrication of an arm capable of deploying pods at distances upwards of 20 meters would be challenging and impose a large moment on the rover. This method is also very limited by terrain slopes and pod weight. Reliability and fatigue are also a concern for this design due to its considerable length. For these reasons, the spring cannon and pitching machine will both be brought forward for further investigation in the PDR.

As the trade study indicates, both the spring cannon and pitching machine are strong options moving forward. In order to come to a final selection for the deployment method, extensive prototyping and modeling shall be conducted. Both deployment methods can be modeled through trajectory analysis given an initial velocity; however, the pitching machine will add complexity to predicting how the pod will land due to the added angular velocity associated with spinning the pod, if that particular deisgn is chosen. Fundamentally, the spring cannon is easier to analyze because it relies heavily upon Hooke's law. The pitching machine relies on friction to deploy the pods which will be difficult to model since the energy imparted to the pod will be divided between the two rotating wheels. Both methods have extensive heritage that will be taken advantage of in the prototyping and testing phase in order to pick a final design before the Spring semester. Prototyping will take the form of modifying COTS products such as airsoft devices and pitching machines in order to determine the range and fidelity of these deployment methods. Upon completion of modeling, prototyping, and testing, a final decision will be made.

As for ensuring the pod stops within the margin of error and is oriented correctly, crumple zone plastics or other “arm” type attachments seem promising, but are heavily dependant on the actual deployment mechanism, which is yet to be decided. In order to prevent locking this aspect of the project in an unfeasible design option, the pod stopping and orientation mechanisms are yet to be determined for the time being.

### 5.5. Pod Structures

From the trade studies conducted on the pod materials, the best material for the pod shell was polycarbonate. Polycarbonate is by far the strongest of the compared materials and has the greatest reusability, a key factor to the accomplishment of the mission. Polycarbonate is readily available in many different forms and is relatively easy to manipulate for manufacturing. The trade study shows that the polycarbonate is a fairly heavy, however, less material would be necessary because the strength is significantly larger than the ABS or polyurethane.

This material has a high coefficient of restitution which could be a problem for maintaining deployment accuracy. This issue may be mitigated by modifying the pod shape/structure or adding a layer of a different material and, therefore, is not of the utmost concern when choosing the shell material.

### 5.6. Localization/Pod Processing

From this trade study the Localino is the clear best option for the processor. There is software that has been developed by a student working for the customer that is readily available. The biggest problem with this choice is the Localino is no longer in production. There are a limited number that are available and if any break during integration or if stock is depleted, it will not be possible to make additional pods. The second best option would be to use the Adafruit Feather nRF52 Bluefruit. Although the Bluefruit has no readily available localization software, Bluetooth has been used for localization and therefore it would be feasible to use it for this projects application. The next best option is using the DWM1001 processor. This processor however came into production in August of 2018 so there has not been extensive use of the product to date. Making a PCB for a STM32F103 processor is out of scope of this project and would require substantial development time, introducing a major risk to producing the necessary quantity of working pods for testing and final demonstration.

There is a customer requirement to use the DW1000 RF module but the development board that uses it is in short supply. To accomodate this, other options were researched such as the Bluefruit and DWM1001 development board. If the Localino development boards are not available there are other options that can be pursued that will still allow a design that will meet the requirements.

### 5.7. Pod Electronics

The pod internal system can be visualized in 11 below. The FBD shows the components that have been narrowed down using the trade study and how they will integrate with each other. This figure also shows which components are COTS and which components are made by the team.

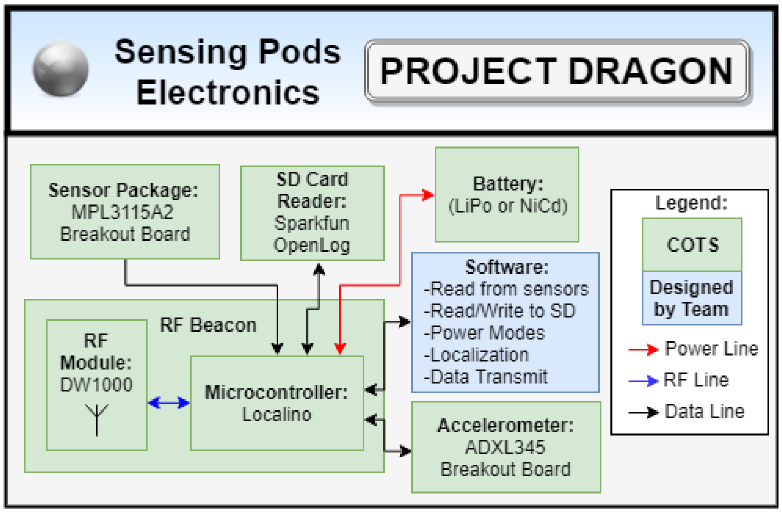


**Figure 11. FBD for Pod Electronics**

### 5.7.1. *Environmental Breakout Board*

From the environmental breakout board trade study, it can be seen that will the given metrics and weights that the Sparkfun MPL3115A2 breakout board is a strong choice for the environmental breakout board for this project. The high accuracy of this sensor as well as its low power consumption and low cost make this an ideal breakout board to be used on our mission.

### 5.7.2. *Accelerometer Breakout Board*

The accelerometer breakout board trade study resulted in two rather close results between the Sparkfun ADXL345 and the Sparkfun ADXL377. The ADXL345 has high accuracy and low power consumption which is ideal for this mission. However, it has limited g-force loading. The ADXL377 has high g-force loading and accuracy, but it has an extremely high power consumption compared to the other options as well as a higher cost. However, the baseline design choice chosen for the accelerometer break out board is the ADXL345 due to its lower cost and overall accuracy, corresponding to the highest metrics in this trade study, making it the logical choice for reaching the project's requirements and goals.

### 5.7.3. *SD Card Reader*

No trade study was conducted on this component, but the team will be using the Sparkfun OpenLog as the SD card reader. This reader can be integrated to both the MPL3115A2 and ADXL345.

## 5.8. Pod Power

Based on the trade study in 4.8, the top choices are NiMH and NiCd. It can be seen that these are both very durable and are lightweight, but NiCd are cheaper. Therefore, if the form factor of AA size was to fit within the pod shape NiCd would be the stronger option. If the pod shape is incapable of fitting that form factor, the LiPo would be a great choice over Li-ion as it is lighter and a little cheaper. The final choice comes down to either NiCd or LiPo based on the form factor required.

## 5.9. Pod GPS/GNSS

Referring back to 3.9, there was no trade study completed for this choice. This is due to the cost of each RTK GPS/GNSS unit being about 5 % of the total budget given. Based on this factor, collecting GPS data in post is the best option available. Furthermore, the main mission objectives is to produce deployable pods that can create a local positioning system in a GPS-denied environment. If this system was actually implemented, the pods would have no use in containing GPS modules. Including the GPS modules inside of the pods now just increase size, weight, complexity, and power draw.

## References

[1] Adafruit, BME680 Datesheet, https://cdn-learn.adafruit.com/downloads/pdf/adafruit-bme680-humidity-temperature-barometic-pressure-voc-gas.pdf

[2] Adafruit, Bluefruit nRF52 Feather Learning Guide, Retrieved September 28, 2018, `https://learn.adafruit.com/bluefruit-nrf52-feather-learning-guide/introduction`

[3] Battery University, BU-203: Nickel-based Batteries, https://batteryuniversity.com/learn/article/nickel_based_batteries

[4] Battery University, Understanding Lithium-ion, https://batteryuniversity.com/index.php/learn/archive/understanding_lithium_ion

[5] Cabela's, DO-All FireFly Auto Trap Thrower with Carry Bag, Retrieved September 25, 2018, from `https://www.cabelas.com/product/DO-ALL-FIREFLY-AUTO-TRAP-W-BAG/`

[6] Chang-an, Liu, et al."Mobile Robot Path Planning Based on an Improved Rapidly-Exploring Random Tree in Unknown Environment - IEEE Conference Publication. An Introduction to Biometric Recognition - IEEE Journals Magazine, 30 Sept. 2008, ieeexplore.ieee.org/document/4636565.

[7] Choset, Howie. "Robotic Motion Planning: A* and D* Search." Carnegie Melon University School of Computer Science, www.cs.cmu.edu/ motionplanning/lecture/AppH-astar-dstar_howie.pdf.

[8] Dakulovic, Marija, et al. "Complete Coverage D* Algorithm for Path Planning of a Floor-Cleaning Mobile Robot." Department of Control and Computer Engineering, Faculty of Electrical Engineering and Computing, University of Zagreb, Croatia, 2 Sept. 2011, ac.els-cdn.com/S147466701644557X/1-s2.0-S147466701644557X-main.pdf?_tid=1db6f2fc-118f-4de8-bbcd-1b749669efd2acdnat=1538188786_4901c483c55c869652bb75f80e9a14ae.

[9] Decawave, DWM100, Retrieved September 17, 2018, `https://www.decawave.com/products/dw1000`

[10] Decawave, DWM1001 Development Board, Retrieved September 17, 2018, `https://www.decawave.com/product/dwm1001-development-board/`

[11] "Dijkstras Algorithm: Correctness by Induction." Oregon State, 29 Jan. 2015, web.engr.oregonstate.edu/ glencora/wiki/uploads/dijkstra-proof.pdf.

[12] Kuek, Localino Indoor Positioning System v2.0 - DIY!, Retrieved September 17, 2018, `https://www.tindie.com/products/kuek/localino-indoor-positioning-system-v20-diy/`

[13] LaValle,Steven. "12.3.2 Stentz's Algorithm (D)." Planning Algorithms, 20 Apr. 2012, planning.cs.uiuc.edu/node616.html.

[14] Popescu, Dumitru, et al. "DESIGN, REFINEMENT AND ACCURACY OF PID CONTROLLERS." 2001, ac.els-cdn.com/S1474667017390705/1-s2.0-S1474667017390705-main.pdf?_tid=48f332be-f7c7-4333-b0e0-99cf70f64bedacdnat=1538189459_f36b27541f71a70482ae0e70879ccb5d.

[15] Roger's Hobby Center, A Guide to Understanding LiPo Batteries, https://rogershobbycenter.com/lipoguide/

[16] Sparkfun, ADXL345 Datasheet, https://www.sparkfun.com/datasheets/Sensors/Accelerometer/ADXL345.pdf

[17] Sparkfun, ADXL377 Datasheet, https://cdn.sparkfun.com/datasheets/Sensors/Accelerometers/ADXL377.pdf

[18] Sparkfun, BME280 Datasheet, https://cdn.sparkfun.com/assets/learn-tutorials/4/1/9/BST-BME280-DS001-10.pdf

[19] Sparkfun, CCS811 Datasheet, https://cdn.sparkfun.com/assets/learn-tutorials/1/4/3/CCS811-Datasheet-DS000459.pdf

[20] Sparkfun, MMA8452Q Datasheet, https://cdn.sparkfun.com/datasheets/Sensors/Accelerometers/MMA8452Q-rev8.1.pdf

[21] Sparkfun, OpenLog Information - Sparkfun, https://www.sparkfun.com/products/13712

[22] Sparkfun, MPL3115A2 Datasheet, https://cdn.sparkfun.com/datasheets/Sensors/Pressure/MPL3115A2.pdf

[23] Thrun, Sebastian, et al. Probabilistic Robotics. MIT Press, 2010.

[24] Wang, X., J. Schwan, H.-W. Hsu, E. Grn, and M. Hornyi (2016). *Dust charging and transport on airless planetary bodies.* Geophys. Res. Lett., 43, 61036110, doi:10.1002/2016GL069491.