REINFORCEMENT LEARNING FOR SMALL BODY SCIENCE OPERATIONS

Adam Herrmann[*] **and Hanspeter Schaub**[†]

On-board planning and scheduling will become a requirement for future missions to small bodies due to the uncertainty in the environment and round-trip light-time delay. Reinforcement learning is well-suited for on-board planning and scheduling because of the observation-action-observation feedback loop. This work formulates a Markov decision process for a small body science operations problem. The objective of the MDP is to maximize the sum of targets imaged and downlinked and the amount of spectroscopy map collected and downlinked while avoiding resource constraint failures. In contrast to past work, this formulation of the problem considers attitude dynamics, fuel consumption, and available power. Deep Q-Learning is applied to train a policy that is compared to a representative reference mission scenario. Deep Q-Learning manages to compute a policy that avoids some resource constraint violations and complete a portion of the science objectives. However, it does not reach the expected reward achieved in the reference mission scenario.

INTRODUCTION

Missions to small bodies such as asteroids and comets present several challenges for planning and scheduling. First and foremost, apiori uncertainty regarding the environment about small bodies necessitates the development of tools that can quickly adjust to the discovered environmental parameters upon arrival to the body. Secondly, large navigational uncertainties can lead to challenges in resource modeling and science operations. Either the uncertainty in task execution times and resource consumption must be handled explicitly in the scheduling algorithm or a buffer must be added to the end of every task to account for variations in execution time and resource consumption. Finally, the round-trip light-time delay can present challenges, especially during critical maneuvers such as Touch-and-Go (TAG). While these challenges are often addressed by work in autonomous guidance, navigation, and controls (GNC), planning and scheduling must be able to support rapid changes in the trajectory due to autonomous GNC.

On-board planning and scheduling has been implemented on several missions in recent decades. The ASPEN and CASPER systems developed by the Jet Propulsion Laboratory have been used in various forms for the Earth-Observing 1 mission,[1,2] IPEX mission,[3] and even the Perseverance Rover.[4,5] Recently, Markov decision processes (MDPs) and reinforcement learning (RL) have been posed as candidates for on-board planning and scheduling, particularly in the Earth-orbiting domain.[6,7] Reinforcement learning is attractive for on-board planning and scheduling because of

[*] PhD Student, Ann and H.J. Smead Department of Aerospace Engineering Sciences, University of Colorado, Boulder, Boulder, CO, 80309. AIAA Member.

[†] Glenn L. Murphy Chair of Engineering, Ann and H.J. Smead Department of Aerospace Engineering Sciences, University of Colorado, Boulder, 431 UCB, Colorado Center for Astrodynamics Research, Boulder, CO, 80309. AAS Fellow, AIAA Fellow.

the observation-action-observation feedback loop. The agent selects the next action based on the current state of the environment. Furthermore, reinforcement learning algorithms are capable of finding optimal policies for a given Markov decision process. In practice, optimality is difficult to achieve because of the assumptions made when casting a real-world problem as an MDP, but high-performing policies, often outperforming humans, have been demonstrated repeatedly in the literature.[8,9] In the small body domain, deep RL has been applied to global mapping for shape modeling and target imaging. Chan and Agha-Mohammadi formulate a small body mapping problem as a partially-observable Markov decision process (POMDP) where the objective is to improve the quality of a map assembled using stereophotoclinometry (SPC).[10] The authors apply the REINFORCE algorithm to generate policies over the belief-space, showing that the trained policies perform better than heuristic policies. Piccinin et al. formulate a global mapping problem for SPC as an MDP.[11] In this problem, the spacecraft enters an orbit about the body, and the decision-making agent determines whether or not to take an image. The authors compare Deep Q-Learning (DQN) and Neural Fitted Q (NFQ) learning, showing that these two algorithms outperform random and heuristic policies. Takahashi and Scheeres formulate a surface imaging problem about a small body as an MDP where the output of the policy is a change in elevation and a transfer time, which is fed into a two-point boundary value solver that generates a fuel-optimal control solution.[12] An extended Kalman filter is implemented to provide a state estimate to the two-point boundary value problem solver and decision-making agent. The authors apply Proximal Policy Optimization to train decision-making agents, showing how autonomous GNC technologies may be combined with reinforcement learning for surface imaging.

Past work has demonstrated how various proximity operations problems about small bodies may be formulated as (PO)MDPs and solved with reinforcement learning algorithms. However, these problem formulations typically fail to account for resource constraints such as on-board storage and power. Because on-board storage is not modeled, communication with the ground is typically left out of the problem formulations as well. Attitude guidance and control and its relation to the aforementioned resource constraints, particularly power, is also not considered. The addition of these aspects of the problem are important because they have serious implications for the learned policies. Furthermore, while many of these problem formulations add partial observability, the impact of partial observability on performance, particularly the quality of science observations, is not explored. It should also not be assumed that the navigation architecture supports continuous measurement updates. Instead, one should assume that the measurement update either requires communication with the ground or dedicated imaging for optical navigation, which means that the estimation error covariance should grow between navigation updates. This work formulates a small body science proximity operations problem with on-board storage, power consumption and generation, data downlink, and attitude guidance and control. This work does not add partial observability or navigation updates to the problem formulation, which future work will address.

This paper first provides an overview of the small body proximity operations mission phases, from approach to landing, and defines them using past missions as examples. Then, the small body science operations problem of interest is defined. A Markov decision process formulation of the problem is presented, and the Basilisk simulation used to model the problem is described in detail. An overview of the methods used to solve the Markov decision process is presented, namely Deep Q-Learning. Finally, preliminary results are presented and discussed. A human-designed reference mission scenario is presented and compared to the Deep Q-Learning results.

SMALL BODY PROXIMITY OPERATIONS PHASES

Small body proximity operations may be decomposed into several different phases, each with its own objectives and data products. Each of these phases may be thought of as separate operations problems where the science and data products from one phase are utilized in the next. Past work in spacecraft autonomy for small body exploration has defined these mission phases in various ways.[13,14] This work will provide its own summary for clarity. Because these phases are defined using concepts of operations from several different missions, the boundaries between them are fluid. Ashman et al. provide a detailed summary of the Rosetta operations phases,[15] and Lauretta et al. provide a summary of the OSIRIS-REx operations phases.[16] The phases this work defines are a.) Approach, B.) Characterization, C.) Science Operations, D.) Landing. The characteristics of each are summarized in Table 1.

Table 1: Small Body Mission Phases

	Approach	Characterization	Science Operations	Landing
Data Products	Body Ephemeris, Spin State, Preliminary Shape Model	Preliminary Science, Gravity Estimate, Improved Shape Model	Science Maps, Landing Site Images, Detailed Shape Model	Surface Science
Optical Navigation	Centroid-Based	Centroid-Based	Feature-Tracking	Feature-Tracking
Dynamics	Approach Trajectory	Hyperbolic Fly-bys	Orbital Motion, Inertial Waypoints, Low-Altitude Fly-bys	Descent & Ascent Trajectory
Analogous Rosetta Phases	Far Approach Trajectory	Close Approach Trajectory and Characterization	Global Mapping, Close Observation	Philae
Analogous OSIRIS-REx Phases	Approach	Preliminary Survey	Detailed Survey, Orbital B, Reconnaissance	Touch-And-Go

The first phase is the approach phase. During the approach phase, the spacecraft performs trajectory correction maneuvers to rendezvous with the asteroid. During this phase, a low fidelity shape model is constructed, a refined estimate of the spin state is gathered, and the ephemeris of the body is improved.[14] This phase is analogous to Rosetta's Far Approach Trajectory (FAT) Phase and OSIRIS-REx's Approach Phase. The second phase is typically a characterization phase. During this phase, the spacecraft enters the body's sphere of influence, performing hyperbolic flybys about the body. The shape model is improved, preliminary science data is gathered, and an estimate of the body's gravitational parameter is generated. This phase is analogous to Rosetta's Close Approach Trajectory (CAT) and Characterization Phase and OSIRIS-REx's Preliminary Survey Phase. Finally, the spacecraft enters the science operations phase, which may be decomposed further into more specific operations phases depending on the mission. This is when the detailed science campaign about the body begins, which is highly dependent on the mission. During this phase, the spacecraft either enters into a stable orbit about the body, transfers between or holds a position at an inertial waypoint(s), or performs low-altitude fly-bys about the body. This also marks the transition from centroiding-based optical navigation to feature-tracking optical navigation. This phase typically includes some sort of mapping to build temperature maps, reflectance maps, and identify candidate landing sites. In the case of Rosetta, the Global Mapping and Close Observation Phases fall into this category. In the case of OSIRIS-REx, the Detailed Survey, Orbital B, and Reconnaissance Phases fall into this category. The final phase of proximity operations is often some sort of landing phase. In the case of Rosetta this includes the landing of the Philae lander, and in the case of OSIRIS-REx this includes the Touch-and-Go phase.

PROBLEM STATEMENT

Small Body Science Operations Problem

This works formulates a small body science operations problem where a spacecraft maneuvers between waypoints defined in the sun-asteroid Hill frame, performing science activities while managing on-board resources such as power and data storage. The objective is to maximize the number of targets imaged and downlinked and the amount of mapping performed and downlinked. There are two simultaneous science objectives - spectroscopy mapping and high-resolution target imaging. For the spectroscopy mapping, there are $j = 3$ separate maps that must be collected, one at each of the following solar longitudes: $\boldsymbol{\lambda} = \{90°, 30°, -30°\}$. Each map is represented by a set of $k = 500$ points, $\mathbf{M}_j$, evenly distributed on the surface of the body, where j is the map number. These points are generated using a Fibonacci lattice. The high resolution imagery is represented by a set of surface targets that are referred to as $\mathbf{T}$. The spacecraft has pre-planned access with the deep space network (DSN) once every 24 hours. Furthermore, the spacecraft must avoid collision with the body as it maneuvers between waypoints. This small body science operations problem is most closely related to the OSIRIS-REx Detailed Survey Phase. However, this work adds additional waypoints (i.e. maneuvers are not only performed in the northern and southern solar latitudes), moves the spacecraft closer to the body, and increases the half field-of-view of the mapping instrument. This is primarily done to reduce the amount of time the spacecraft is coasting in regions where there is no science value, decreasing the simulation time required to complete the mapping campaign. To complete the mission, the spacecraft enters different modes of operation. These spacecraft modes abstract the continuous, low-level behaviors of the spacecraft (i.e. attitude guidance and control, instrument status, etc.) into discrete modes of operations. These modes are shown in Figure 1.

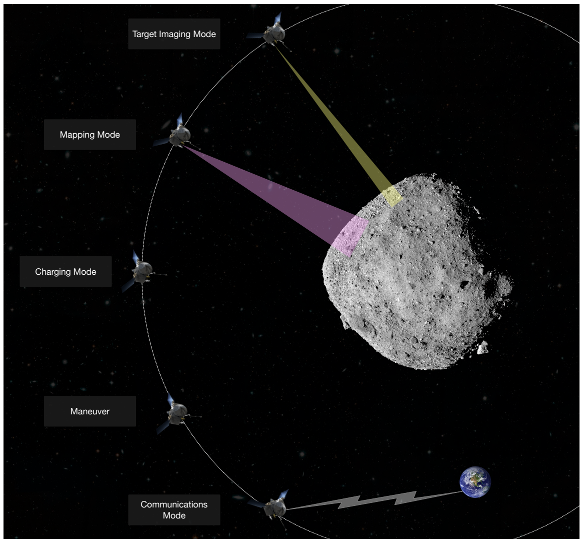
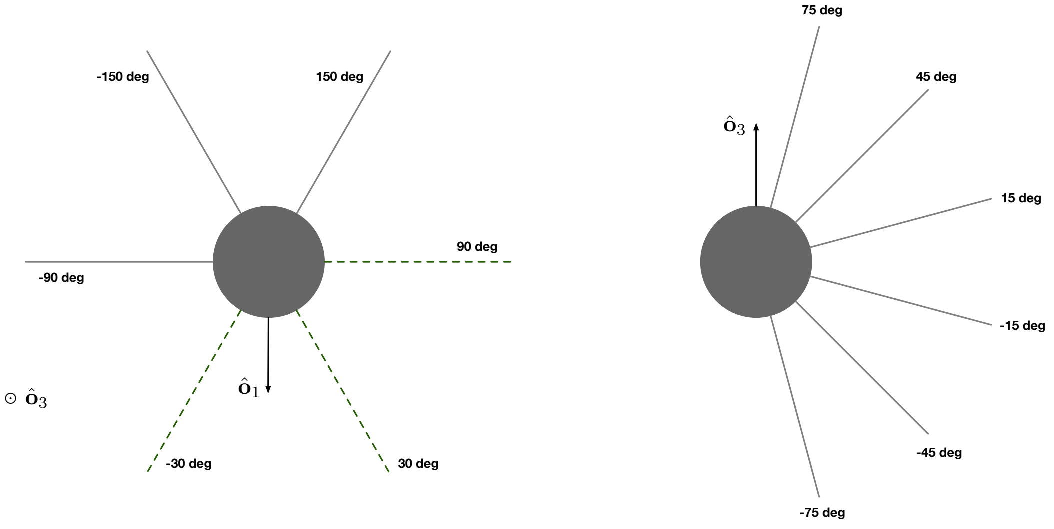


Figure 1: Flight Modes

The waypoints the spacecraft maneuvers between are defined in the sun-asteroid Hill frame. The spacecraft maneuvers between or holds its position at specific waypoints, performing the tasks in

Figure 1 as the asteroid rotates beneath it. The waypoints are evenly distributed across six solar longitudes and latitudes, as shown in Figure 2, numbering 36 in total. In Figure 2a, the dotted lines represent the solar longitudes where spectroscopy mapping may take place. The $\hat{\mathbf{o}}_1$ vector denotes the direction of the sun. There are three maps in total that must be collected. In Figure 2b, the various latitudes are displayed. Mapping may occur at any of these latitudes if the spacecraft is at the correct solar longitude. During the Detailed Survey Phase, OSIRIS-REx had seven total maps to collect, each at a specific solar longitude. Furthermore, the mapping had to take place at a relatively narrow band of solar latitudes. This work selects three maps at specific solar longitudes and removes the narrow solar latitude requirement to maintain minimal simulation time.



(a) Solar longitudes (λ).

(b) Solar latitudes (ϕ).

Figure 2: Solar longitudes and latitudes. Dotted lines represent the solar longitudes of the three maps, $\boldsymbol{\lambda} = \{90^\circ, 30^\circ, -30^\circ\}$.

Markov Decision Process

The small body science operations problem is formulated as a Markov decision process. A Markov decision process a sequential decision-making problem in which an agent selects an action a_i in some state s_i following a policy $\pi : \mathcal{S} \times \mathcal{A}$, which maps states to actions. The agent observes a new state s_{i+1} and receives a reward r_i. The reward is a function of the state and action taken, $R : \mathcal{S} \times \mathcal{A} \to \mathcal{R}$. Markov decision processes follow the Markov assumption. The next state is conditionally dependent only on the current state and action. This may be stated as $T(s_{i+1}|s_i, a_i) = T(s_{i+1}|s_i, a_i, s_{i-1}, a_{i-1}, ..., s_0, a_0)$, where $T(s_{i+1}|s_i, a_i)$ is the probability of transitioning to state s_{i+1} given s_i and a_i. All relevant state information for the purposes of maintaining this assumption must be included in the state space.

State Space The state space, $\mathcal{S}$, is designed to retain enough information to satisfy the Markov assumption. A complete list of the state space is provided in the bulleted list below:

- Hill frame spacecraft position, ${}^{\mathcal{O}}\mathbf{r}_{\text{s/c}}$

- Hill frame spacecraft velocity, ${}^{\mathcal{O}}\mathbf{v}_{\text{s/c}}$
- Hill frame position of nearest imaging target, ${}^{\mathcal{O}}\mathbf{r}_{t_{\text{nearest}}}$
- Hill frame position of current waypoint target, ${}^{\mathcal{O}}\mathbf{r}_{w_{\text{ref}}}$
- Hill frame position of previous waypoint target, ${}^{\mathcal{O}}\mathbf{r}_{w_{\text{prev}}}$
- Number of imaged targets
- Number of downlinked targets
- For map $\mathbf{M}_j$, j =1:3:
  - Amount of region 1 mapped
  - Amount of region 2 mapped
  - Amount of region 3 mapped
- Battery charge
- Eclipse indicator
- Data buffer storage
- $\Delta\mathbf{v}$ consumed
- Ground station indicator

Geometric information is included in the state space to capture the spatial relationship between the science objectives. It can also provide information on resource management states and the risk of collision. These states include the spacecraft position, spacecraft velocity, position of the nearest imaging target, position of the current waypoint, and position of the previous waypoint. These states are all expressed in the Hill frame, $\mathcal{O} : \{\hat{\mathbf{o}}_1, \hat{\mathbf{o}}_2, \hat{\mathbf{o}}_3\}$, which is computed using the asteroid's orbit about the sun.

Several states are also included to provide a measure of science objective completion. The number of imaged and downlinked targets in $\mathbf{T}$ are included in the state space. For each map $\mathbf{M}_j$, the mapping points are partitioned into three equally sized groups based on the value of the z-component of the body-fixed position of the mapping points. The body frame of the asteroid is defined as $\mathcal{A} : \{\hat{\mathbf{a}}_1, \hat{\mathbf{a}}_2, \hat{\mathbf{a}}_3\}$. The three regions are displayed in Figure 3. This state provides the agent information on which regions still need to be mapped.

Finally, several states are included to retain information on resource constraints and safety. The data stored in the buffer and ground station indicator provide state information for the on-board data system. The battery charge and eclipse indicator provide information for the purposes of power management. The available $\Delta\mathbf{v}$ state indicates how much fuel the spacecraft has available to use.

Each state is normalized to a range of approximately [-1, 1]. The spacecraft position, position of the nearest imaging target, and position of the current and previous waypoint are all normalized by the radius of the body. The number of imaged and downlinked targets, the mapped regions, and resource states are all normalized by their respective max values such that they are within a range of [0, 1].

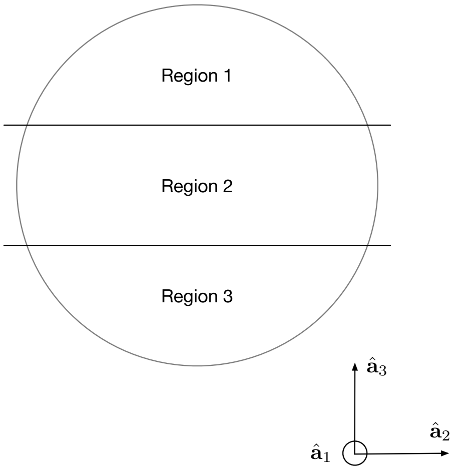


Figure 3: Map regions.

Action Space A mode-based planning approach is taken in the action space. A spacecraft mode turns certain models on or off and sets the attitude reference for a prescribed amount of time, abstracting continuous low-level behavior into higher-level abstractions of spacecraft behavior. Each mode lasts for 2,000 seconds, with the exception of the mapping mode. The mapping mode lasts for 4,000 seconds, which is approximately one full rotation of the body. The action space, $\mathcal{A}$, is provided by the bulleted list below:

- Charge
- Waypoint Reference Actions
  - $\phi_{\text{ref}} = \phi_{\text{ref}} + 30^o$, $\lambda_{\text{ref}} = \lambda_{\text{ref}}$
  - $\phi_{\text{ref}} = \phi_{\text{ref}} + 30^o$, $\lambda_{\text{ref}} = \lambda_{\text{ref}} + 60^o$
  - $\phi_{\text{ref}} = \phi_{\text{ref}}$, $\lambda_{\text{ref}} = \lambda_{\text{ref}} + 60^o$
  - $\phi_{\text{ref}} = \phi_{\text{ref}} - 30^o$, $\lambda_{\text{ref}} = \lambda_{\text{ref}} + 60^o$
  - $\phi_{\text{ref}} = \phi_{\text{ref}} - 30^o$, $\lambda_{\text{ref}} = \lambda_{\text{ref}}$
  - $\phi_{\text{ref}} = \phi_{\text{ref}} - 30^o$, $\lambda_{\text{ref}} = \lambda_{\text{ref}} - 60^o$
  - $\phi_{\text{ref}} = \phi_{\text{ref}}$, $\lambda_{\text{ref}} = \lambda_{\text{ref}} - 60^o$
  - $\phi_{\text{ref}} = \phi_{\text{ref}} + 30^o$, $\lambda_{\text{ref}} = \lambda_{\text{ref}} - 60^o$
- Map
- Image
- Downlink

In the charging mode, the spacecraft turns off all instruments and the transmitter and points the solar panels at the sun to charge the battery. The action space also includes eight separate waypoint reference change actions. When a waypoint reference change action is taken, the current waypoint reference $w_{ref} = \{\phi_{ref}, \lambda_{ref}\}$ changes to the selected adjacent waypoint reference. If one of these modes is selected, the last time the waypoint was changed is checked to see if a new waypoint can be selected. The current waypoint does not change unless 8,000 seconds have passed since the last switch to ensure convergence to the current waypoint. After each change, the new waypoint latitude and longitude is checked to ensure it is wrapped to the appropriate latitude and longitude boundaries. An example of this is provided in Figure 4. The nominal transitions are shown in the dotted green line. Wrapped transitions are shown in the solid red line.

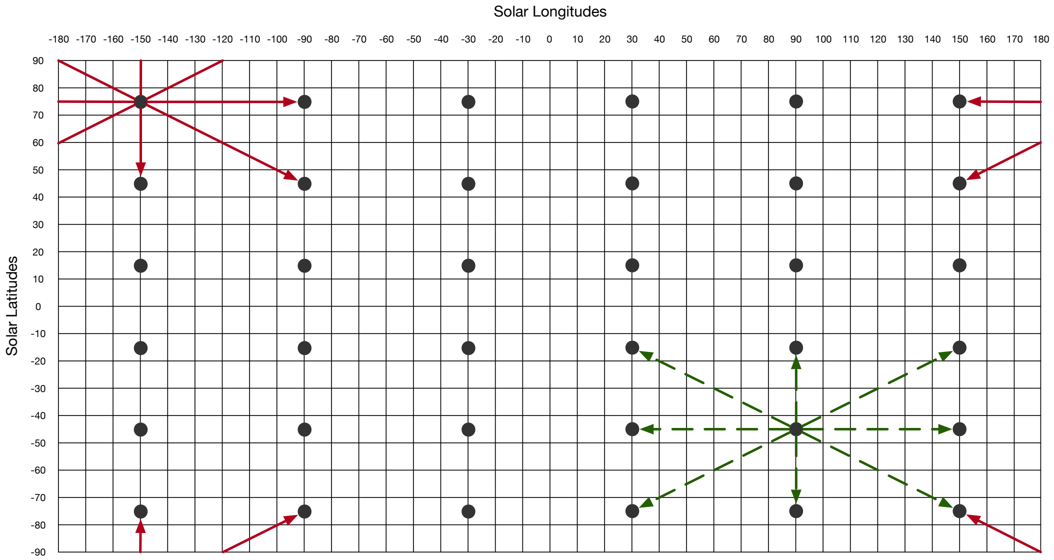


Figure 4: Waypoint reference transitions.

In the mapping mode, the spacecraft points the mapping instrument at the asteroid. Data is collected in the on-board storage unit, and only the portion of the map collected within requirements is considered mapped. Mapping requirements are provided in Table 2. In the imaging mode, the spacecraft points the imager at the nearest target and attempts to take an image of the target. The image is collected if the spacecraft is within the elevation and range requirements of the target image. In the downlink mode, the spacecraft points the transmitter in the direction of the Earth. Data is downlinked once the spacecraft is within elevation and range requirements of the DSN and the prescribed downlink time occurs.

Table 2: Science Requirements

Imaging	
Elevation	60^o
Attitude Error Norm	0.1 rad
Mapping	
Elevation	45^o
Instrument Half-FOV	22.5^o
Solar Longitude Tolerance	1^o

Reward Function The reward function $R(s_i, a_i, s_{i+1})$ is a piecewise function of the current state, action, and next state. The return at state i is given by:

$$r_i = \begin{cases} -10 & \text{if failure} \\ \dfrac{1}{|T|} H(c_j) & \text{if } \neg\text{failure} \wedge a_i \text{ is image} \\ \dfrac{1}{3|M|} \sum_j^3 \sum_k^{|\mathbf{M}_j|} H(m_{j,k}) & \text{if } \neg\text{failure} \wedge a_i \text{ is map} \\ \dfrac{10}{|T|} H(c_j) \sum_j^{|\mathbf{T}|} H(d_j) + \dfrac{10}{3|M|} \sum_j^3 \sum_k^{|\mathbf{M}_j|} H(f_{j,k}) & \text{if } \neg\text{failure} \wedge a_i \text{ is downlink} \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

If the agent fails, a failure penalty of -10 is returned and the episode terminates. The failure condition is true if the spacecraft expends all charge in the battery, overfills the data buffer, or collides with the body. Mathematically, this is represented as with Equation (2), where z is the normalized charge of the battery and b is the normalized data buffer level.

$$\text{failure} = (z = 0 \ \vee \ b \geq 1 \ \vee \ \text{any}(||^{\mathcal{H}}\mathbf{r}_{\text{s/c}}|| \leq \mathbf{r}_{\text{ast}}) \tag{2}$$

A function $H(x_j)$ is formulated to check if the state variable x is false at step i and true at step $i+1$, returning 1 if these conditions are met.

$$H(x_j) = 1 \text{ if } \neg x_{j_i} \ \wedge \ x_{j_{i+1}} \tag{3}$$

The variable c_j represents whether or not target j has been imaged. If the imaging mode is initiated and a failure does not occur, target j is checked to determine if it was imaged for the first time. This reward component is normalized by the total number of targets.

The variable $m_{j,k}$ represents whether or not mapping point k for map number j has been mapped. If the mapping mode is initiated and a failure does not occur, all map points are checked to determine if they were collected for the first time or not. The summation of this reward is normalized by $3|M|$ such the total possible reward for this component totals to 1.

The variable d_j represents whether or not target j has been downlinked, and the variable $f_{j,k}$ represents whether or not mapping point k for map number j has been downlinked. Both the set of targets and all map points are looped through to determine if they have been downlinked for the first time or not. Both the imaging and mapping components are multiplied by 10 and divided by the total number of targets or mapping points such that the maximum possible reward for each component is 10.

Transition Function Due to the continuous dynamics of the small body proximity operations science problem, it is difficult to construct a transition function with conditional probabilities that accurately captures state transitions. The transition function is instead represented by a generative model $G(s_i, a_i)$ given in Equation (4). The generative model returns a new state s_{i+1} and reward r_i by integrating equations of motion forwards in time.

$$s_{i+1}, \ r_i = G(s_i, a_i) \tag{4}$$

The Basilisk astrodynamics software architecture[17] is used to construct the simulation, which models the complex behavior of the spacecraft and environment. The Basilisk simulation is wrapped within a Gym environment. The Gym environment provides a standard interface for the agent to interact with the Basilisk simulation.

Simulation Architecture

Basilisk Simulation Overview A Basilisk simulation is implemented to serve as the generative transition function for the MDP. In Figure 5, the task groupings and modules in the Basilisk simulation are provided. Several flight software tasks are implemented. These include a Sun-pointing task, Earth-pointing task, target-pointing task, map-pointing task, MRP control task, and a waypoint feedback control task. Depending on the flight mode, these tasks are turned on or off, primarily to determine which attitude reference should be used. A summary of each task's status in each flight mode is provided in Table 3. The sun-pointing, earth-pointing, target-pointing, and map-pointing tasks all use Basilisk's `locationPointing()` module and output an attitude guidance message which includes the MRP attitude error $\boldsymbol{\sigma}_{B/R}$. The attitude guidance message is ingested by the `mrpFeedback()` module, which outputs a commanded torque. This commanded torque is utilized by the `rwMotorTorque()` module to compute reaction wheel motor torques and send a motor command message to the three reaction wheel state effectors in the dynamics task.

Table 3: Basilisk Model and Task Status in Different Modes

	Modes				
Basilisk Tasks & Models	Charge	Waypoint Change	Map	Image	Downlink
Sun-Pointing Task	Enabled	Enabled	Disabled	Disabled	Disabled
Earth-Pointing Task	Disabled	Disabled	Disable	Disabled	Enabled
Location-Pointing Task	Disabled	Disabled	Disabled	Enabled	Disabled
Map-Pointing Task	Disabled	Disabled	Enabled	Disabled	Disabled
MRP Control Task	Enabled	Enabled	Enabled	Enabled	Enabled
Waypoint Control Task	Enabled	Enabled	Enabled	Enabled	Enabled
Mapping Task	Disabled	Disabled	Enabled	Disabled	Disabled
Imager Power Model	Off	Off	Off	On	Off
Imager Data Model	Off	Off	On	On	Off
Mapping Power Model	Off	On	Off	Off	Off
Mapping Data Model	Off	On	Off	Off	Off
Transmitter Power Model	Off	Off	Off	Off	On
Transmitter Data Model	Off	Off	Off	Off	On

The waypoint feedback control task utilizes a feedback control law to regulate the state of the spacecraft to the desired Hill frame waypoint. The feedback control law outputs a force command, which the `externalForceTorque()` dynamics module utilizes to pass the commanded force to the spacecraft. The thrust is computed with the following equation:

$$\mathbf{u} = -(f(\mathbf{x}) - f(\mathbf{x}_{ref})) - [K_1]\Delta\mathbf{x}_1 - [K_2]\Delta\mathbf{x}_2 \tag{5}$$

The derivation of the relative dynamics are not discussed here for brevity, but may be found in work from Scheeres[18] and Takahashi.[19] In this problem, $f(\mathbf{x})$ is computed using a cannonball SRP model, third body perturbations from the sun, and point-mass gravity from the asteroid. The feedback control law provides no guarantees on fuel optimality. Future work should consider the use of a Lambert solver to compute a fuel-optimal two-burn solution. However, the feedback control law fulfills the function of a control solution from one waypoint to another. Furthermore, the total

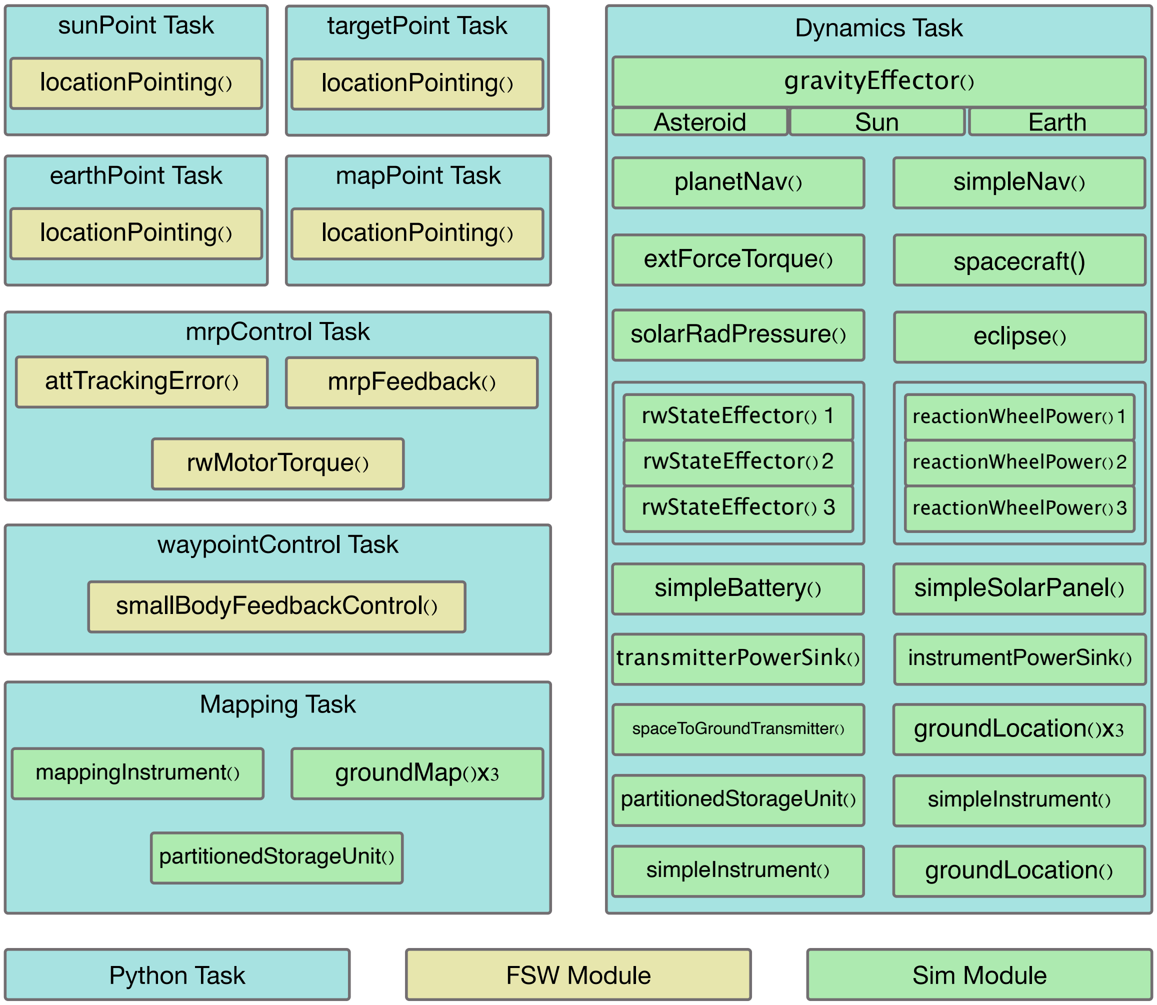


Figure 5: Basilisk Simulation Diagram

$\Delta\mathbf{v}$ can be computed and compared to the $\Delta\mathbf{v}$ budget. For the purposes of this planning problem, this simple solution is sufficient.

In addition to several flight software tasks, a dynamics tasks is also implemented which holds the majority of the modules in the simulation. Gravity effectors for the asteroid, sun, and the Earth are implemented. A `planetNav()` module is also implemented for the asteroid, which creates an ephemeris message utilized by the relevant flight software modules. Likewise, a `simpleNav()` module performs the same function, but for the spacecraft state. The `planetNav()` and the `simpleNav()` modules can optionally add noise to the states to imitate a navigation system. This work does not add noise to these states.

Several dynamics modules are connected to the spacecraft. As previously stated, the commanded force is passed to the spacecraft with the `extForceTorque()` module. Additionally, a `solarRadiationPressure()` module is implemented. A cannonball SRP module is utilized. Finally, each reaction wheel state effector is connected to the spacecraft for the purposes of attitude control. Lastly, the `eclipse()` module utilizes the state of the asteroid and the spacecraft to indicate whether or not the spacecraft is in eclipse.

A representative power system is modeled on-board the spacecraft. At the center of the power system is a `simpleBattery()` module. The battery receives power generation and consumption messages from each other power module to compute the storage level at each time step. Solar panels are modeled using the `simpleSolarPanel()` module, which computes power generation based on the area of the panels, the efficiency of the panels, and the solar incidence angle. Instrument and transmitter power models are also implemented with the `simplePowerSink()` module.

An on-board data system is also modeled. This system is modeled using two tasks - the dynamics task and the mapping task. The dynamics tasks is always on, but the mapping pass is disabled for all modes except for the mapping mode. This is done to minimize required computation. In the mapping task, three `groundMap()` modules are connected to a `mappingInstrument()`. The `groundMap()` module loops through each mapping point and checks for three things: a.) the spacecraft is within the elevation requirements of the point, b.) the point is within the instrument's field-of-view, and c.) the spacecraft is within the required solar longitude band. A vector of access messages are then passed to the `mappingInstrument()`, which passes the data on to a `partitionedStorageUnit()`. This `partitionedStorageUnit()` in the maping task keeps track of the points that have been imaged and those that have not. This serves a different function than the `partitionedStorageUnit()` in the dynamics task. In the dynamics task, two `simpleInstrument()` modules are implemented. One `simpleInstrument()` module is used in conjunction with the `simpleInstrumentController()` to image the ground targets if the imaging mode is entered. The other `simpleInstrument()` module is used keep track of the amount of data generated by mapping. This module provides a scalar value for data generated and does not keep track of the specific points. Both of these instruments pass the data to the `partitionedStorageUnit()` in the dynamics task.

Initial Conditions The parameters of the spacecraft may be found in Table 4. The modeled spacecraft is a small satellite used frequently in past work. These parameters were balanced to create a scenario in which the spacecraft must make tradeoffs between resource constraints, science collection, and downlink. The initial conditions for the asteroid orbit, size, and rotation may be found in Table 5. These parameters are based on those of Bennu,[20] with the exception of the radius and mass which were slightly increased.

DEEP Q-LEARNING

This work implements Deep Q-Learning from the Tensorflow Agents library, which is based on work from Mnih et al.[21] The Tensorflow Agents library provides collection of tools for designing, implementing, and testing reinforcement learning algorithms.[22] Deep Q-Learning aims to approximate the optimal state-action value function using a deep neural network. The state-action value function is described by the equation below, which states it is the expected value of all future return given the current state s_i and some action a_i, following a policy π:

$$Q^*(s,a) = \max_\pi \mathbb{E}[r_i + \gamma r_{i+1} + \gamma^2 r_{i+2} + \cdots | s_i = s, a_i = a, \pi] \tag{6}$$

The parameters of the deep neural network, i.e. the weights and biases, are referred to as θ_k, where k is the iteration number. The neural network representation of the state-action value function is referred to as $Q(s, a; \theta_k)$. Furthermore, tuples of the agent's experience (s_i, a_i, r_i, s_{i+1}) are stored in a replay buffer D. The replay buffer is sampled uniformly at each iteration to compute the loss. The state-action value network is updated with the following loss function:

Table 4: Spacecraft Parameters

General Spacecraft Parameters	
Mass	330 kg
Dimensions	1.38 x 1.04 x 1.58 m
$\Delta\mathbf{v}$ Budget	40 m/s
Power System	
Solar Panel Area	1.0 m^2
Solar Panel Efficiency	0.20
Instrument Power Draw	30 W
Transmitter Power Draw	15 W
Battery Capacity	100 Whr
Data & Communications System	
Data Buffer Storage Capacity	125 GB
Transmitter Baud Rate	120 Mbps
Instrument Baud Rate	8 Mbps
Map Instrument Baud Rate	8 Mbps

Table 5: Asteroid Parameters

Orbital Parameters	
Semi-Major Axis, a	1.1259 AU
Eccentricity, e	0.016975
Inclination, i	0.0027666 deg
Long. of Ascend. Node, Ω	177.42 deg
Arg. of Periapsis, ω	284.26 deg
True Anomaly, f	357.30 deg
Size and Rotation	
Shape	Spherical
Rotation Period	4.297461 hr
Radius	800 m
Mass	5.278e12 kg
Gravitational Constant	352.25 m^3/s^2

$$L_k(\theta_k) = \mathbb{E}_{(s,a,r,s')\sim U(D)}\left[\left(r + \gamma \max_{a'} Q(s', a'; \theta_k^-) - Q(s, a; \theta_k)\right)^2\right] \tag{7}$$

After training, the learned state-action value function can be utilized to create a deterministic policy.

$$\pi(s_i) = \arg\max_{a_i} Q(s_i, a_i; \theta) \tag{8}$$

Deep Q-Learning is applied to the small body science environment. The neural network is updated 800 times during training. During each update, 50 episodes are executed to generate data with the new policy. This data is added to the replay buffer, and the network is updated. Five evaluations with the new policy are performed on the environment to measure the average reward. A summary of the training parameters are provided in Table 6.

Table 6: DQN training hyperparameters.

Parameter	Value
Nodes Per Hidden Layer	200
Hidden Layers	3
Activation Function	tanh
Batch Size	64
Optimizer	Adam
Replay Buffer Size	10,000

RESULTS

Reference Mission

A reference mission scenario is designed to validate the simulator and determine an upper bound on reward. The reference mission scenario is designed such that the spacecraft collects the three spectroscopy maps in succession, traveling up and down the solar latitudes at the solar longitudes corresponding with each map. The trajectory in the Hill frame is provided in Figure 6. The spacecraft trajectory is shown in blue, and the asteroid-sun line is provided using the red line. The orange points represent the location of the mapping points in the Hill frame at $t = 0$. Similarly, the green points represent the imaging targets at $t = 0$.

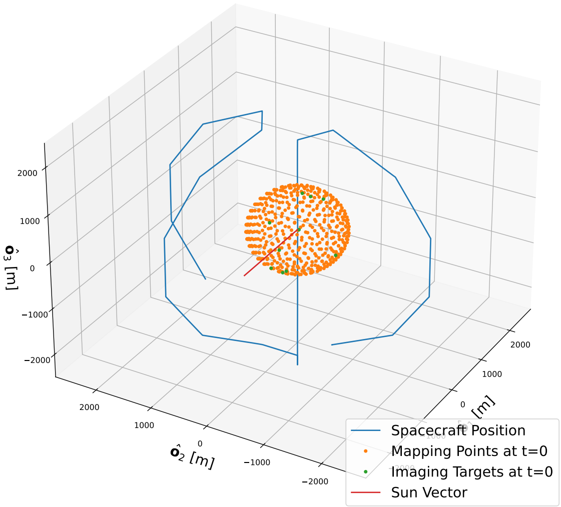
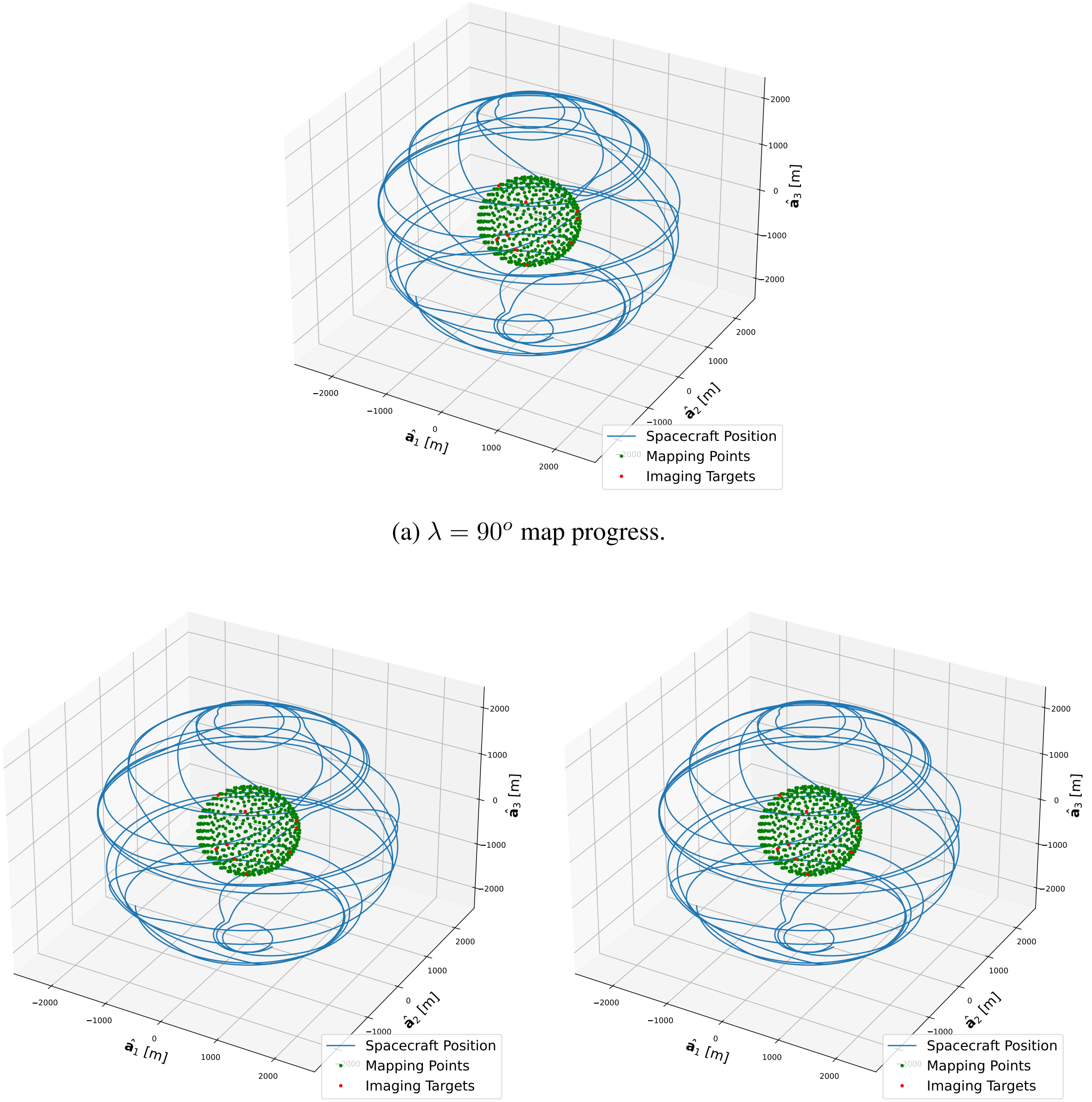


Figure 6: Hill frame trajectory.

The first leg of the trajectory occurs at the $\lambda = 90^o$ solar longitude. The spacecraft begins at the southern-most latitude, which is an ideal case as the location is randomized in training. The spacecraft moves from southern to northern latitudes, periodically switching between mapping and selecting the next waypoint. The spacecraft enters the charge mode at the northern-most latitude before moving to the next solar longitude, $\lambda = 30^o$. The spacecraft moves from the northern latitudes to the southern latitudes, performing a charge-downlink-downlink-charge mode sequence about halfway down to empty the data buffer. The downlink modes are accompanied by charging modes because the transmitter consumes a relatively large amount of power. At the southern latitude of the second leg, the spacecraft moves to the next solar longitude, $\lambda = -30^o$, performing more mapping and waypoint transitions as it moves from southern latitudes to northern latitudes. Halfway through the third leg, the spacecraft performs a charge-charge-downlink-downlink mode sequence to downlink the map data. Once at the northern latitude, the spacecraft transitions to a final solar longitude, moving from south to north while periodically imaging. The spacecraft ends the scenario with two downlink modes, sending the rest of the data on-board the spacecraft to the DSN.



(a) $\lambda = 90^o$ map progress.

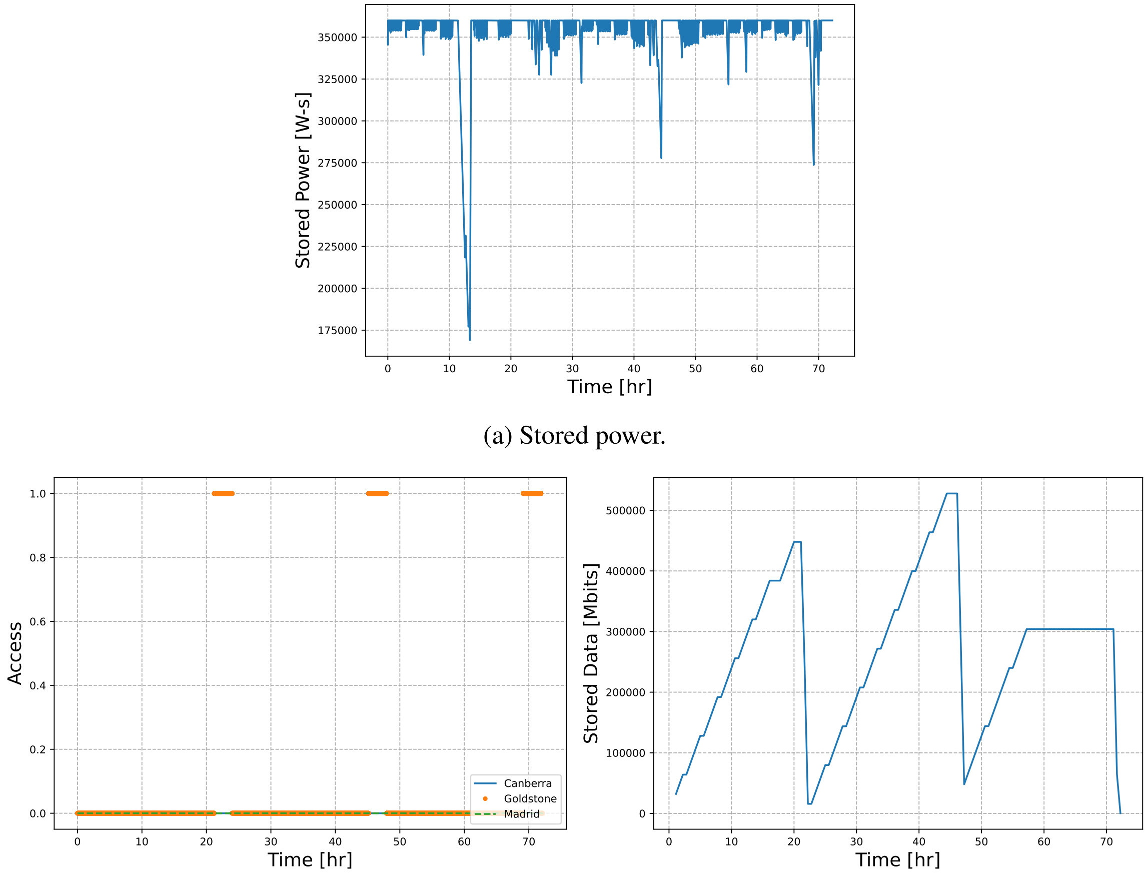
(b) $\lambda = 30^o$ map progress.

(c) $\lambda = -30^o$ map progress.

Figure 7: Spacecraft trajectory in the asteroid body frame.

To qualitatively evaluate the coverage of the body, the trajectory of the spacecraft in the asteroid's body frame is plotted in Figure 7. The green dots represent the mapping points the spacecraft collected within requirements. The red dots represent the body-fixed positions of the imaging targets. The spacecraft is able to collect the majority of each map in the reference mission scenario using a single traversal through all of the latitudes. The total $\Delta \mathbf{v}$ cost is 20.0 m/s.

Spacecraft resources throughout the scenario are provided in Figure 8. The spacecraft maintains a healthy amount of stored power throughout the reference scenario. The minimum amount of stored power is about 72% of the maximum capacity. Furthermore, the spacecraft is able to almost empty the data buffer during each downlink mode. It is important that the spacecraft downlinks at each possible opportunity in this scenario. Skipping an access interval would result in a data buffer overflow. Recall that the DSN is constrained by both physical and temporal access. In addition to range and elevation requirements, downlink may only occur within the specified intervals. In this scenario, all of the specified intervals occur when only the Goldstone station is available.



(a) Stored power.

(b) DSN access.

(c) Stored data.

Figure 8: Spacecraft resources and DSN access.

Finally, the reward achieved by the spacecraft is plotted in Figure 9. The theoretical maximum reward is 20. This assumes all map points and imaging targets have been captured and downlinked without failure. However, the reward achieved in the reference mission scenario provides a realistic upper bound on performance.

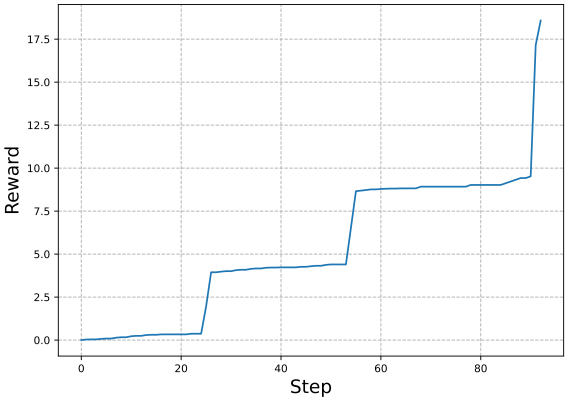


Figure 9: Reference mission reward.

Deep Q-Learning

Preliminary results are generated utilizing Deep Q-Learning. The smoothed average reward achieved by Deep Q-Learning in training is provided in Figure 10. The DQN policy manages to achieve positive reward, but the policy has not learned mapping and does a poor job of managing resources. The policy attempts to image and downlink the ground targets alone. The agent fails periodically as well. The majority of the failures are either exceeding the $\Delta \mathbf{v}$ budget or running out of power. The agent does a decent job of avoiding collision, learning that care must be taken when moving waypoints at the poles when angle wrapping can set the spacecraft on a collision course with the body. Future work will investigate two potential remedies for these issues. First, the total possible mapping reward, relative to the imaging reward, will be doubled or tripled. A larger reward signal for mapping may help balance the two science objectives. Secondly, alternative reinforcement learning algorithms like Proximal Policy Optimization (PPO) will be investigated.

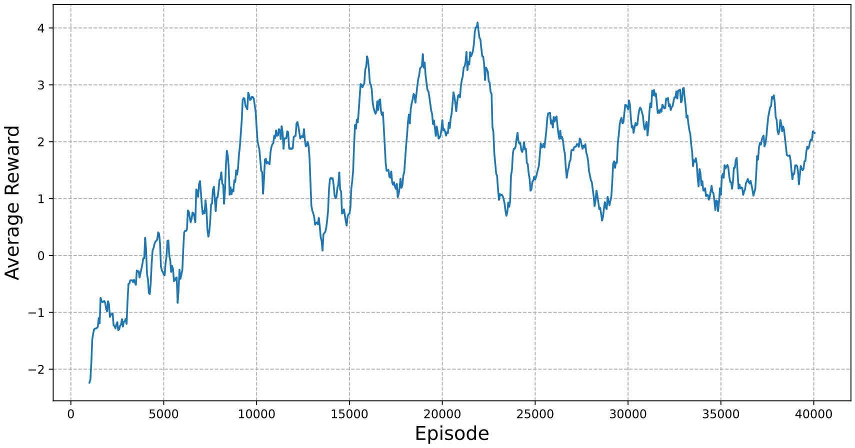


Figure 10: DQN average reward.

CONCLUSION

This work formulates a small body science operations problem as a Markov decision process (MDP). The objective of the problem is to maximize the amount of map data collected and downlinked as well as the number of surface images collected and downlinked. A generative model of the problem is developed using the Basilisk astrodynamics software. A reference mission scenario is designed to validate the Basilisk model and provide a realistic upper bound on reward. Deep Q-Learning is implemented to train a neural network representation of the state-action value network. Preliminary results show that the DQN agent struggles to match the reward achieved in the reference mission scenario. The learned policy prioritizes imaging, downlink, and collision avoidance, ignoring power and $\Delta\mathbf{v}$ resource constraints. This is not surprising, however, due to the weakness of the DQN algorithm.

Future work will first investigate a.) alternative reinforcement learning algorithms and b.) changes to the action space to produce policies that can meet or exceed the reward achieved in the reference mission scenario for arbitrary initial conditions. Afterwards, future work will take steps to implement a more representative GNC system. A two-point boundary value solver will be utilized to compute fuel-optimal two-burn maneuvers between waypoints. A navigation system will also be implemented, either assuming completely autonomous on-board navigation or a hybrid approach with DSN range and range-rate measurements provided during communication periods. Once the navigation system is implemented, a POMDP formulation of the problem will be created. This problem formulation will include dedicated navigation update modes. Agents will be trained over the belief space generated by the navigation system, and the effect of the state uncertainty on science observations and safety states will be investigated.

REFERENCES

[1] S. Chien, R. Sherwood, D. Tran, B. Cichy, G. Rabideau, R. Castano, A. Davis, D. Mandl, S. Frye, B. Trout, S. Shulman, and D. Boyer, “Using Autonomy Flight Software to Improve Science Return on Earth Observing One,” *Journal of Aerospace Computing, Information, and Communication*, Vol. 2, No. 4, 2005, pp. 196–216, 10.2514/1.12923.

[2] S. A. Chien, D. Tran, G. Rabideau, S. Schaffer, D. Mandl, and S. Frye, “Improving the Operations of the Earth Observing One Mission via Automated Mission Planning,” 2010.

[3] S. Chien, J. Doubleday, D. R. Thompson, K. Wagstaff, J. Bellardo, C. Francis, E. Baumgarten, A. Williams, E. Yee, E. Stanton, and J. Piug-Suari, “Onboard Autonomy on the Intelligent Payload EXperiment (IPEX) CubeSat Mission,” *Journal of Aerospace Information Systems (JAIS)*, April 2016, 10.2514/1.I010386.

[4] A. Yelamanchili, G. Rabideau, J. Agrawal, V. Wong, D. Gaines, S. Chien, E. Fosse, J. Biehl, S. Kuhn, A. Connell, *et al.*, “Ground and Onboard Automated Scheduling for the Mars 2020 Rover Mission,” International Workshop on Planning and Scheduling for Space, 2021.

[5] G. Rabideau, V. Wong, D. Gaines, J. Agrawal, S. Chien, E. Fosse, and J. Biehl, “Onboard Automated Scheduling for the Mars 2020 Rover,” 2020.

[6] A. P. Herrmann and H. Schaub, “Monte Carlo Tree Search Methods for the Earth-Observing Satellite Scheduling Problem,” *Journal of Aerospace Information Systems*, 2021, pp. 1–13, 10.2514/1.I010992.

[7] A. Herrmann and H. Schaub, “Autonomous On-board Planning for Earth-orbiting Spacecraft,” *IEEE Aerospace Conference*, Big Sky, MT, March 5-12 2022.

[8] M. Hessel, J. Modayil, H. Van Hasselt, T. Schaul, G. Ostrovski, W. Dabney, D. Horgan, B. Piot, M. Azar, and D. Silver, “Rainbow: Combining Improvements in Deep Reinforcement Learning,” *arXiv preprint arXiv:1710.02298*, 2017.

[9] D. Silver, J. Schrittwieser, K. Simonyan, I. Antonoglou, A. Huang, A. Guez, T. Hubert, L. Baker, M. Lai, A. Bolton, Y. Chen, T. Lillicrap, F. Hui, L. Sifre, G. Driessche, T. Graepel, and D. Hassabis, “Mastering the Game of Go Without Human Knowledge,” *Nature*, Vol. 550, 10 2017, pp. 354–359, 10.1038/nature24270.

[10] D. M. Chan and A. Agha-mohammadi, "Autonomous Imaging and Mapping of Small Bodies Using Deep Reinforcement Learning," *2019 IEEE Aerospace Conference*, 2019, pp. 1–12, 10.1109/AERO.2019.8742147.

[11] M. Piccinin, P. Lunghi, and M. Lavagna, "Deep Reinforcement Learning-based Policy for Autonomous Imaging Planning of Small Celestial Bodies Mapping," *Aerospace Science and Technology*, Vol. 120, 2022, p. 107224.

[12] S. Takahashi and D. Scheeres, "Autonomous Proximity Operations at Small NEAs," 33rd International Symposium on Space Technology and Science (ISTS), 02 2022.

[13] V. Pesce, A.-a. Agha-mohammadi, and M. Lavagna, "Autonomous Navigation & Mapping of Small Bodies," *IEEE Aerospace Conference*, IEEE, 2018, pp. 1–10.

[14] I. A. Nesnas, B. J. Hockman, S. Bandopadhyay, B. J. Morrell, D. P. Lubey, J. Villa, D. S. Bayard, A. Osmundson, B. Jarvis, M. Bersani, *et al.*, "Autonomous Exploration of Small Bodies Toward Greater Autonomy for Deep Space Missions," *Frontiers in Robotics and AI*, Vol. 8, 2021.

[15] M. Ashman, M. Barthélémy, M. Almeida, N. Altobelli, M. C. Sitjà, J. J. G. Beteta, B. Geiger, B. Grieger, D. Heather, R. Hoofs, *et al.*, "Rosetta Science Operations in Support of the Philae Mission," *Acta Astronautica*, Vol. 125, 2016, pp. 41–64.

[16] D. Lauretta, S. Balram-Knutson, E. Beshore, W. Boynton, C. D. d'Aubigny, D. DellaGiustina, H. Enos, D. Golish, C. Hergenrother, E. Howell, *et al.*, "OSIRIS-REx: Sample Return From Asteroid (101955) Bennu," *Space Science Reviews*, Vol. 212, No. 1, 2017, pp. 925–984.

[17] P. W. Kenneally *et al.*, "Basilisk: A Flexible, Scalable and Modular Astrodynamics Simulation Framework," *7th International Conference on Astrodynamics Tools and Techniques (ICATT)*, DLR Oberpfaffenhofen, Germany, Nov. 6–9 2018.

[18] D. J. Scheeres, "Orbit mechanics about asteroids and comets," *Journal of Guidance, Control, and Dynamics*, Vol. 35, No. 3, 2012, pp. 987–997.

[19] S. Takahashi and D. J. Scheeres, "Autonomous Exploration of a Small Near-Earth Asteroid," *Journal of Guidance, Control, and Dynamics*, Vol. 44, No. 4, 2021, pp. 701–718.

[20] D. Lauretta, A. Bartels, M. Barucci, E. Bierhaus, R. Binzel, W. Bottke, H. Campins, S. Chesley, B. Clark, B. Clark, *et al.*, "The OSIRIS-REx Target Asteroid (101955) Bennu: Constraints on its Physical, Geological, and Dynamical Nature from Astronomical Observations," *Meteoritics & Planetary Science*, Vol. 50, No. 4, 2015, pp. 834–849.

[21] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, *et al.*, "Human-Level Control Through Deep Reinforcement Learning," *Nature*, Vol. 518, No. 7540, 2015, pp. 529–533.

[22] S. Guadarrama, A. Korattikara, O. Ramirez, P. Castro, E. Holly, S. Fishman, K. Wang, E. Gonina, N. Wu, E. Kokiopoulou, L. Sbaiz, J. Smith, G. Bartók, J. Berent, C. Harris, V. Vanhoucke, and E. Brevdo, "TF-Agents: A Library for Reinforcement Learning in TensorFlow," `https://github.com/tensorflow/agents`, 2018.