JOURNAL OF SPACECRAFT AND ROCKETS
Vol. 61, No. 1, January–February 2024

Single-Agent Reinforcement Learning for Scalable Earth-Observing Satellite Constellation Operations

Adam Herrmann,* Mark A. Stephenson,† and Hanspeter Schaub‡
University of Colorado, Boulder, Colorado 80303

https://doi.org/10.2514/1.A35736

This work explores single-agent reinforcement learning for the multi-satellite agile Earth-observing scheduling problem. The objective of the problem is to maximize the weighted sum of imaging targets collected and downlinked while avoiding resource constraint violations on board the spacecraft. To avoid the computational complexity associated with multi-agent deep reinforcement learning while creating a robust and scalable solution, a policy is trained in a single satellite environment. This policy is then deployed on board each satellite in a Walker-delta constellation. A global set of targets is distributed to each satellite based on target access. The satellites communicate with one another to determine whether an imaging target is imaged or downlinked. Free communication, line-of-sight communication, and no communication are explored to determine how the communication assumptions and constellation design impact performance. Free communication is shown to produce the best performance, and no communication is shown to produce the worst performance. Line-of-sight communication performance is shown to depend heavily on the design of the constellation and how frequently the satellites can communicate with one another. To explore how higher-level coordination can impact performance, a centralized mixed-integer programming optimization approach to global target distribution is explored and compared to a decentralized approach. A genetic algorithm is also implemented for comparison purposes, and the proposed method is shown to achieve higher reward on average at a fraction of the computational cost.

Nomenclature

$\mathcal{A}$	=	action space
a_i	=	action at interval i
$\mathcal{B}$	=	spacecraft body coordinate frame
b	=	stored data in buffer
D_k	=	imaged and passed ground target set for spacecraft k
$\mathcal{E}$	=	Earth-centered, Earth-fixed coordinate frame
$G(s_i, a_i)$	=	generative transition function
g	=	ground station access indicator
$\mathcal{H}$	=	Hill coordinate frame
h	=	data transmitted
i	=	decision-making interval
J	=	number of targets in U_k
k	=	spacecraft indicator
M	=	global ground target set
m	=	global target indicator
$\mathcal{N}$	=	inertial coordinate frame
$o_{i,m,k}$	=	binary target access variable
p	=	ground target priority
$Q(s_i, a_i)$	=	state-action value function
$R(s_i, a_i, s_{i+1})$	=	reward function
r_i	=	reward at interval i
${}^{\mathcal{E}}\boldsymbol{r}$	=	spacecraft position expressed in Earth-centered, Earth-fixed coordinate frame
${}^{\mathcal{H}}\boldsymbol{r}_j$	=	ground target position expressed in Hill
$\mathcal{S}$	=	state space
s_i	=	state at interval i
$T(s_{i+1}\|s_i, a_i)$	=	discrete transition function
T_k	=	local ground target set for spacecraft k
U_k	=	upcoming ground target set for spacecraft k
${}^{\mathcal{E}}\boldsymbol{v}$	=	spacecraft velocity expressed in Earth-centered, Earth-fixed coordinate frame
$x_{i,m,k}$	=	binary target distribution decision variable
z	=	battery charge
$\pi(s_i)$	=	policy
$\boldsymbol{\sigma}_{\mathcal{B}/\mathcal{R}}$	=	modified Rodrigues parameter attitude error
$\boldsymbol{\Omega}$	=	reaction wheel speeds
${}^{\mathcal{B}}\boldsymbol{\omega}_{\mathcal{B}/\mathcal{N}}$	=	spacecraft angular velocity expressed in body frame components

Subscript

θ	=	neural network approximation

Received 13 March 2023; revision received 13 July 2023; accepted for publication 15 September 2023; published online 30 October 2023. Copyright © 2023 by the authors. Published by the American Institute of Aeronautics and Astronautics, Inc., with permission. All requests for copying and permission to reprint should be submitted to CCC at www.copyright.com; employ the eISSN 1533-6794 to initiate your request. See also AIAA Rights and Permissions www.aiaa.org/randp.

*Graduate Research Assistant, Ann & H.J. Smead Department of Aerospace Engineering Sciences, 3775 Discovery Drive, AERO 446; adam.herrmann@colorado.edu. Member AIAA.

†Graduate Research Assistant, Ann & H.J. Smead Department of Aerospace Engineering Sciences, 3775 Discovery Drive, AERO 446; mark.a.stephenson@colorado.edu. Member AIAA.

‡Professor, Schaden Leadership Chair, Ann & H.J. Smead Department of Aerospace Engineering Sciences, 3775 Discovery Drive, AERO 446; hanspeter.schaub@colorado.edu. Associate Fellow AIAA.

I. Introduction

THE need for constellation management tools and on board satellite autonomy is increasing as the number of Earth observation constellations grows. Planet Labs [1], Spire Global [2], and Capella Space [3] are just a few examples of companies with current or upcoming Earth-orbiting constellations. The problem of scheduling the sequence of observation, downlink, and resource management tasks performed by a constellation of Earth-orbiting satellites with three-axis attitude control capabilities is commonly referred to as the multi-satellite agile Earth-observing (MSAEO) scheduling problem. The satellites are referred to as *agile* because of their three-axis attitude control capabilities that enable attitude maneuvers about any axis, as opposed to only pitch and roll maneuvers. The primary challenge associated with the MSAEO scheduling problem is formulating performant, accurate, and computationally tractable problems that are flexible and fast enough to modify and solve again in the inevitable event that replanning is required. The solutions should also be robust and scalable to support the addition and removal of satellites, which is a normal part of sustained constellation operations as new satellites are launched and older satellites are decommissioned. In the case of performance, problem formulations and solutions must be optimal with respect to observations collected and downlinked while respecting the relevant resource constraints.

Problem formulations must also be accurate, representing the real life problem with sufficient fidelity to minimize the frequency in which replanning is required due to mismodeling. Finally, problem formulations and corresponding algorithms must be computationally tractable enough to solve during nominal operations and also fast enough to solve again when replanning is required, such as when opportunistic science events present themselves.

Traditional approaches to MSAEO scheduling use some heuristic, metaheuristic, or exact solution method [4]. Commonly used optimization-based approaches include mixed-integer linear programming (MILP) and/or metaheuristic optimization. An optimization problem is formulated and solved over some planning horizon. The solution to this problem is sequenced into commands and uplinked to the spacecraft for open-loop execution. Metaheuristic optimization algorithms such as genetic algorithms (GAs) have been applied to Earth-observing scheduling with some success [5]. Genetic algorithms do not place constraints on the fitness function. Theoretically, the fitness function could be a high-fidelity spacecraft simulation that returns the number of collected and downlinked targets. However, metaheuristic algorithms are sensitive to initialization, oftentimes do not guarantee global optimality, and can be slow to converge (especially if a high-fidelity simulator is used). Mixed-integer linear programming approaches are perhaps the most popular in the literature due to optimality guarantees and speed of convergence after the relevant data are preprocessed [6–11]. Spire Global [2] and Planet [1] use mixed-integer linear programming for their Earth-observing constellations. However, MILP formulations require linear dynamics for power and data generation and consumption that are a function of the decision variables [6,7,12]. Mixed-integer nonlinear programming (MINLP) formulations can certainly address these issues head on, but many MINLP formulations are difficult to solve, especially if the problem is nonconvex [13]. Therefore, approximations must be made to adhere to the linear requirements of a MILP formulation, but the degradation in model accuracy may affect performance. Even if these approximations are made, MILP formulations do not scale well to increasing numbers of imaging targets and satellites as the number of decision variables explodes. If replanning is required, the MILP must be solve again. Even simple MILP formulations for Earth-observing (EO) scheduling can take seconds to solve on modern computing hardware, nevermind limited space computing hardware. Valicka et al. [14] address some of the issues associated with MILP scheduling for an Earth-observing satellite (EOS) scheduling problem with cloud coverage uncertainty, formulating a multistage stochastic MILP model for a constellation of satellites to avoid replanning. The Scheduling Planning Routing Inter-Satellite Network Tool (SPRINT) addresses the replanning problem by using a global planner for constellation-level scheduling and local on board planners for unexpected opportunities [15,16]. Chien et al. [17–20] use a process known as iterative repair to modify the current working plan for on board replanning on the Earth Observing One mission.

Reinforcement learning has emerged as a popular method for Earth-observing satellite operations [12,21–25]; rendezvous and proximity operations [26,27]; and small body guidance, navigation, and control [28,29]. Reinforcement learning (RL) has also emerged as a potential solution for the MSAEO scheduling problem. Reinforcement learning agents, or policies, are trained to map states to actions to maximize a numerical reward function [30]. The trained policies can be uplinked to a satellite for closed-loop execution, responding to the real states of the environment, which means that replanning is inherent to an RL planning and scheduling paradigm. Furthermore, the trained policies are often optimal or near optimal with respect to the reward function. Many reinforcement learning algorithms only require a generative model of the environment, which allows for the use of high-fidelity simulations to represent the problem. Finally, execution of trained policies is typically very fast. Neural network approximations of the policy can be executed in milliseconds on modern computational hardware. While RL for single satellite EOS scheduling is gaining traction in the literature, few authors have explored using RL for MSAEO scheduling. Cui et al. [31] apply double Deep Q-Networks for communication scheduling of a constellation of Earth-orbiting satellites and demonstrate that their algorithm is superior to a genetic algorithm in terms of performance and computation time. Dalin et al. [32] formulate a scheduling problem for multi-satellite tasking and apply the multi-agent deep deterministic policy gradient (MADDPG) algorithm to solve the problem. The performance of the MADDPG algorithm is shown to be comparable to other solvers for the problem. While each of these authors makes important contributions to RL for MSAEO scheduling, the handling of resource constraints and their relationship to spacecraft position, velocity, and attitude is quite limited. The authors do not fully leverage the blackbox optimization capabilities of reinforcement learning and rely on simple models of the problem. Furthermore, Refs. [31] and [32] do not demonstrate the scalability of their algorithms to constellation parameters beyond those fixed during training.

While RL poses many benefits for MSAEO scheduling, the primary challenge is the computational complexity of the multi-agent problem, especially if a high-fidelity simulation is used. The most general formulation of a multi-agent RL problem is a decentralized partially observable Markov decision process (Dec-POMDP). However, a Dec-POMDP is nondeterministic exponentially complete [33]. If free communication is assumed, a Dec-POMDP can be reduced to a multi-agent Markov decision process (MDP) [34,35]. However, the size of joint action space in both Dec-POMDPs and Multi-Agent Markov Decision Process (MMDPs) is exponential in the number of decision-making agents. Past work has demonstrated that a single agent can be trained in several hours to several days [36,37]. A multi-agent reinforcement learning problem with comparable simulation fidelity could take much longer to train because of the exponential increase in computational complexity. To avoid the increase in computational complexity, this work trains a decision-making agent in a single-agent environment and deploys that agent on board each spacecraft in a Walker-delta constellation. While this problem formulation is suboptimal in terms of global reward because the decision-making agents are competing for reward, the size of the constellation may be readily changed without requiring retraining. To address this issue, this work adds higher-level coordination for the target distribution so the agents are not competing for reward but are instead working through their individual local target lists while managing satellite resources such as power, on board data storage, and reaction wheel speeds.

This work formulates a multi-satellite agile Earth-observing scheduling problem where a constellation of spacecraft in a Walker-delta formulation attempts to maximize the weighted sum of targets imaged and downlinked while avoiding resource constraint violations concerning power, on-board data storage, and reaction wheel speeds. A Markov decision process formulation of the problem is created for the single satellite training process and multi-satellite deployment. The single- and multi-satellite simulations are also described in detail. Finally, the methods used to train the decision-making agents are presented, and the various communication methods implemented in the multi-satellite scenario are described. The two methods of target distribution (i.e., first come, first served and the mixed-integer programming techniques) are both described. Finally, the performance of the trained agents is benchmarked for each communication method, each target distribution method, and various Walker-delta constellation designs. This performance is compared to that of a genetic algorithm as well.

II. Problem Statement

A. Single Satellite Agile Earth-Observing Scheduling Problem

A single satellite agile Earth-observing (SSAEO) scheduling problem is formulated for the purposes of training the decision-making agents to avoid the computational challenges associated with multi-agent reinforcement learning. The problem statement and corresponding MDP formulation is described in detail in Ref. [37] but will be summarized here for completeness. In the SSAEO scheduling problem, a single satellite in low-Earth orbit attempts to maximize the weighted sum of imaging targets collected and downlinked while avoiding resource constraint violations regarding battery charge, on board data storage, and reaction wheel speeds. The planning horizon, or the total amount of time considered for planning, is equal to approximately three orbits. More precisely, the planning horizon is 90 min long. The planning horizon is divided into 45 discrete planning intervals that last for 6 min each that comprise the set I. The length of the planning interval is driven by the amount of time it takes for the attitude control system to converge to the reference. At each planning interval, the satellite can enter one of the operational modes. The operational modes include battery charging, downlink, reaction wheel desaturation, and imaging. A concept figure of the problem is provided in Fig. 1.

In this formulation of the SSAEO scheduling problem, the satellite has a set of targets available for imaging, ordered by the time the spacecraft has access to the targets. This set of targets is referred to as T. Each target has its own priority p where the maximum priority is 1 and the minimum priority is 3. The priorities are generated by sampling from a uniform probability for each imaging target. A subset of T that includes the next J upcoming targets is also formulated, and this set is referred to as U. Each individual imaging target is referred to as $c_j \in U$. The set D includes all targets that have already been passed or imaged by the satellite,

$$U = \{c_j \in (T - D) | \quad \forall\ j \in [0, J)\} \tag{1}$$

As the satellite orbits the Earth, U is continually updated such that the satellite is only considering the next J targets for imaging, which helps to decrease the size of the action space.

B. Markov Decision Process Formulation

A MDP is a sequential decision-making problem in which a decision-making agent selects an action, $a_i \in \mathcal{A}$, in some state, $s_i \in \mathcal{S}$, following a policy, $\pi: \mathcal{S} \to \mathcal{A}$, which maps states to actions. The agent receives a reward r_i based on some reward function, $R: \mathcal{S} \times \mathcal{A} \to \mathbb{R}$. MDPs follow what is known as the Markov assumption, which states that the probability of transitioning to the next state is conditionally dependent on the current state and action only. All information necessary to maintain this assumption should be included in the state.

1. State

Approximating real-world problems as Markov decision processes can be challenging, largely due to the fact that real-world problems have a mix between discrete and continuous states, and many of the underlying dynamics contain discontinuities and hidden states. The design of the state should closely adhere to the Markov assumption, and in many cases, the process of designing a state that is Markovian enough is nontrivial. In the SSAEO, the state must include information relevant to the collection and downlink of imaging targets, as well as information relevant to the management of on board resources. A description of the state, including information on the normalization of each state for the purposes of function approximation, is provided in Table 1.

Geometric information for the purposes of imaging and downlink is provided with $\mathcal{E}_{\boldsymbol{r}}$, $\mathcal{E}_{\boldsymbol{v}}$, and $\mathcal{H}_{\boldsymbol{r}_j}$. The left superscript denotes to the coordinate frame the vector is expressed in. The selection of the Hill frame, $\mathcal{H}: \{\hat{\boldsymbol{h}}_1, \hat{\boldsymbol{h}}_2, \hat{\boldsymbol{h}}_3\}$, as the coordinate frame for the expression of the target positions is made because the decision-making agent only needs to know the position of the targets relative to itself to make

Table 1 State description

State	Normalization
Spacecraft position (Earth-centered, Earth-fixed frame), ${}^{\mathcal{E}}\boldsymbol{r}$	Radius of Earth
Spacecraft velocity (Earth-centered, Earth-fixed frame), ${}^{\mathcal{E}}\boldsymbol{v}$	Velocity of circular orbit at Earth's surface
Ground target j position (Hill frame), ${}^{\mathcal{H}}\boldsymbol{r}_j$	Radius of Earth
Ground target j priority, $1/p_j$	——
L^2 norm of attitude error, $\Vert\boldsymbol{\sigma}_{\mathcal{B}/\mathcal{R}}\Vert$	——
L^2 norm of angular attitude rate (Spacecraft body frame), $\Vert{}^{\mathcal{B}}\boldsymbol{\omega}_{\mathcal{B}/\mathcal{N}}\Vert$	——
Reaction wheel speeds, $\boldsymbol{\Omega}$	Maximum wheel speeds
Battery charge, z	Maximum battery capacity
Stored data, b	Maximum storage capacity
Data downlinked, h	Maximum storage capacity
Eclipse indicator, q	——

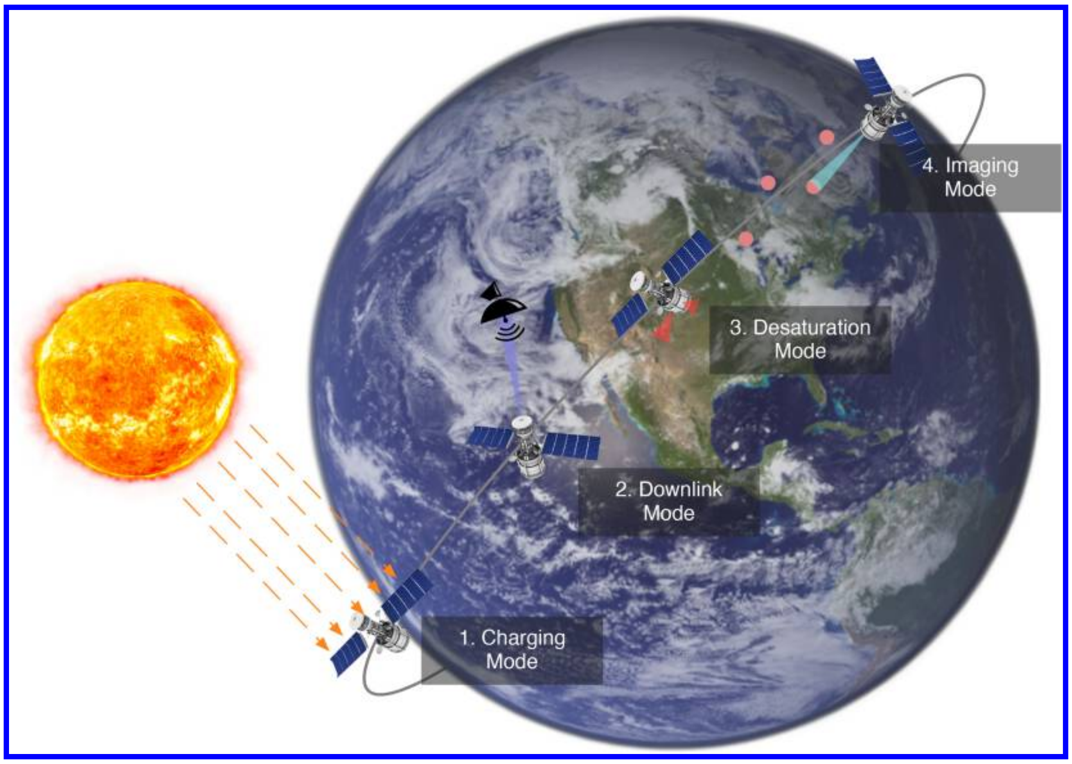


Fig. 1 SSAEO scheduling problem. A satellite in low-Earth orbit attempts to maximize the number of imaging targets collected and downlinked.

imaging decisions. The Hill frame origin is at the spacecraft position. The first direction $\hat{\boldsymbol{h}}_1$ is in the orbit radius direction, and the third direction $\hat{\boldsymbol{h}}_3$ is in the orbit normal direction. The second direction $\hat{\boldsymbol{h}}_2$ completes the right-hand coordinate system. The priority of each target $1/p_j$ is included because it is an important component of the weighting of the reward function. The priority is transformed to $1/p_j$ by dividing it by p_j^2. This is done to ensure the priority represented in the state matches the reward function, which is discussed later. Information on the current state of the attitude control system is provided using $\|\boldsymbol{\sigma}_{\mathcal{B}/\mathcal{R}}\|$, $\|\mathcal{B}_{\boldsymbol{\omega}_{\mathcal{B}/\mathcal{N}}}\|$, and $\boldsymbol{\Omega}$. This information is used to determine when a desaturation mode is performed. Information regarding the power system is provided using z and q, and information regarding the on board data management system is provided using b and h.

2. Action Space

An action space $\mathcal{A}$ is constructed for the SSAEO scheduling problem that allows the decision-making agent to collect and downlink science data as well as manage its resources. A mode-based planning approach is implemented. At each decision-making interval, the satellite turns on or off certain attitude references, instruments, and transmitters for the duration of the 6 min decision-making interval. The continuous behavior of the satellite system is thus abstracted using discrete actions, or modes. The action space, as well as a description for each mode, is provided in the following:

1) **Charge:** The satellite points its solar panels at the sun, turning off all instruments and transmitters to recharge the batteries.

2) **Desaturate:** The satellite points its solar panels at the sun, turning off all instruments and transmitters. Momentum is mapped to thrust commands, which the thrusters execute.

3) **Downlink:** The satellite points the transmitter at the Earth. The transmitter is turned on, and data is downlinked if and when a ground station is available.

4) **Image target** $c_0 \in U$

$$\vdots$$

6) **Image target** $c_j \in U$**:** The satellite points the instrument at the target, taking an image of the target when access requirements are met. The data are stored on board the satellite.

3. Reward Function

A piecewise reward function $R(s_i, a_i, s_{i+1})$ is developed for the SSAEO scheduling problem to ensure that science data are collected and downlinked and that resource constraint violations are avoided. The reward function is provided in Eq. (2). To ensure the problem is numerically well conditioned, the reward function is normalized to a range of approximately [-1, 1]:

$$R(s_i, a_i, s_{i+1}) = \begin{cases} -1 & \text{if failure} \\ \dfrac{1}{|I|}\displaystyle\sum_j^{|T|} H(d_j) & \text{if} \neg \text{failure} \wedge a_i \, \text{is down}n\text{link} \\ \dfrac{0.1}{|I|} H(w_j) & \text{if} \neg \text{failure} \wedge a_i \, \text{is image} \, c_j \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

The failure condition is checked first. Failure occurs if the battery is drained to zero charge, the data buffer is overfilled, or the reaction wheel speeds exceed the maximum speed.

If a failure does not occur and the downlink action is taken, the local target list is looped through to check if a target was downlinked for the first time or not using the downlink variable associated with target j, d_j. The function that performs this check for imaging and downlink is provided in Eq. (3). This function checks to determine if a ground target was imaged or downlinked for the first time or not. If so, then 1 divided by the target priority is returned:

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

A summation over Eq. (3) is performed and normalized by the maximum number of decision-making intervals $|I|$ to ensure the reward contribution from downlinking targets does not exceed 1.

If no failure occurs and an imaging action is taken, Eq. (3) is applied to that ground target using the imaged variable associated with that image w_j. This component of the reward is sized such that the maximum amount of reward from imaging does not exceed 0.1. Without the small reward bonus for imaging, the sparsity of the reward can impede learning. Furthermore, the decision-making agent needs a reward incentive to image when all downlink windows have been passed but the end of the planning horizon has not yet arrived.

C. Multi-Satellite Agile Earth-Observing Scheduling Problem

The multi-satellite agile Earth-observing scheduling problem extends the SSAEO scheduling problem to multiple satellites in a Walker-delta constellation. Walker-delta constellations contain N satellites are distributed evenly among P orbit planes [38]. The orbital planes are distributed at $360/P$ deg intervals of the longitude of ascending node. A phasing factor between the planes may also be specified, which determines the offset in true anomaly between satellites in adjacent planes. In this work, a phasing factor of 0 is used. The satellites in the constellation have access to a global set of targets M. Furthermore, each satellite k has its own set of targets T_k, but satellites may share targets within M. In this work, the decision-making agents attempt to maximize local reward. That is, each satellite attempts to maximize the weighted sum of targets collected and downlinked within its local target set T_k. The satellites update their local target lists through communication with the other satellites. This concept is demonstrated in Fig. 2.

Each satellite in the Walker-delta constellation maintains an observation over its local state, $s_i^k \in \mathcal{S}^k$. The full state is now $s_i = \{s_i^1, \ldots, s_i^k\}$, and the state space is $S = \mathcal{S}^1 \times \cdots \times \mathcal{S}^k$. The state evolves based on the underlying system dynamics and actions taken by each satellite. The action space is now a joint action space represented as $\mathcal{A} = \mathcal{A}^1 \times \cdots \times \mathcal{A}^k$. Each decision-making agent takes actions following its local copy of the policy, $a_i^k = \pi^k(s_i^k)$. The generative transition function and the reward function are both now functions of the complete state and joint actions. The generative transition function is given by $s_{i+1}, r_i^1, \ldots, r_i^k \sim G(s_i, \boldsymbol{a}_i)$, and the joint reward function is given by $\boldsymbol{R}(s, \boldsymbol{a}) = (R^1(s, \boldsymbol{a}), \ldots, R^k(s, \boldsymbol{a}))$. The local reward function of each agent is the same as it is for the SSAEO scheduling problem, but the environment now enumerates through M instead of T_k to check if a target was imaged or downlinked for the first time or not. If another satellite has already imaged or downlinked a target, no new reward is returned.

D. Simulation Architecture

The generative transition function, $s_{i+1}, r_i^1, \ldots, r_i^k \sim G(s_i, \boldsymbol{a}_i)$, of the MSAEO scheduling problem is implemented using a high-fidelity astrodynamics and mission simulation tool called Basilisk[§] [39]. The components of the simulation architecture are identical for both the MSAEO and SSAEO scheduling problems, and a summary of the simulation is provided in Fig. 3. The Basilisk simulation and flight software (FSW) code is written in C/C++. Users instantiate and interface with the simulation and FSW code using Python scripting. This allows for both the speed of C and C++ and the simplicity of a Python interface. Furthermore, this allows for the use of reinforcement learning packages like Gym, which are implemented in Python. The Basilisk simulation is wrapped with a Gym environment,[¶] which allows for decision-making agents to interact with the simulation in a standardized manner. The decision-making agents pass actions to the Gym environment, which turns dynamics and FSW modules and tasks on or off for each spacecraft and runs the simulation. The Basilisk simulation includes three separate classes

[§]Data available online at https://hanspeterschaub.info/basilisk.

[¶]Data available online at https://www.gymlibrary.dev/.

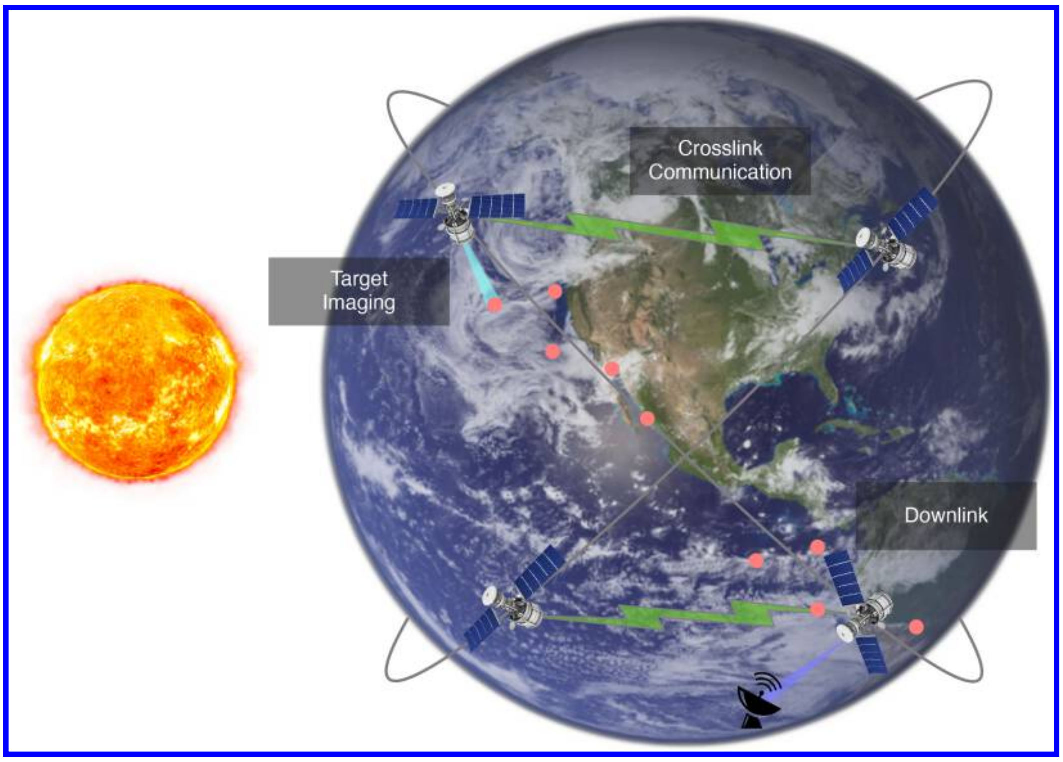


Fig. 2 Multi-satellite agile Earth-observing scheduling problem. A constellation of satellites attempts to maximize the local weighted sum of imaging targets imaged and downlinked.

for 1) environment modules, 2) dynamics modules, and 3) FSW modules. A single environment class is instantiated for the entire simulation, but dynamics and FSW classes are instantiated for each satellite. The architecture allows for the number of spacecraft to easily scale up and down using only a few lines of code.

The environment class contains modules for gravity (the Earth and sun), eclipse, each of the seven ground stations available for downlink, imaging targets, an atmospheric density model, and a random disturbance torque that helps build up momentum in the reaction wheels. Basilisk uses a spherical harmonic gravity model expanded to degree and order 10 for the Earth. The ground stations are selected from NASA's Near Space Network such that each satellite will encounter a ground station at least once over the planning horizon [36]. The dynamics class of each satellite contains modules for the power system (i.e., batteries, solar panels, instrument and transmitter power sinks, etc.), data management system (i.e., data buffer, instrument, and transmitter), and attitude control system (reaction wheels and thrusters). Furthermore, spacecraft location modules are instantiated for each satellite and connected to the other satellites to determine when line-of-sight access occurs. Finally, the flight software class of each satellite contains numerous tasks. The sunPoint, nadirPoint, locationPoint, and trackingError task provide an attitude reference to the mrpControl task, which uses a Modified Rodrigues Parameters (MRP)-based feedback control law to compute reaction wheel torques. Finally, the rwDesat task contains the modules that map reaction wheel momentum to thruster commands. The source code for the MSAEO and SSAEO scheduling problems may be found on the develop branch of the basilisk-gym-interface library[**] under the names multiTgtEarthEnvironment and multiSatMultiTgtEarthEnvironment. The satellite and simulation parameters are provided in Ref. [36].

III. Methods

This section provides an overview of how the decision-making agents are trained and deployed in the constellation. In the first step, the decision-making agents are trained in the single satellite environment. The trained policy is wrapped within a safety shield that prevents the decision-making agents from taking unsafe actions. Then, the policy is deployed on each satellite in a constellation defined using a set of Walker-delta parameters, imaging targets, and communication assumptions. After the policy is deployed in the environment, the performance subject to the Walker-delta parameters, distributed target set, and selected communication assumption is evaluated. This performance is then compared to the performance of a genetic algorithm for a subset of the Walker-delta parameters explored. The deployment pipeline is summarized in Fig. 4. This section details the policy training, policy deployment, and ground target generation blocks of the pipeline. The communication methods, which are contained within the constellation parameterization block, and the genetic algorithm used for comparison purposes are also described.

A. Single-Agent Training

The single-agent training process is described in detail in Refs. [36] and [37] but will be summarized here. A process known as MCTS-Train is used to generate a neural network approximation of the state-action value function $Q_\theta(s,a)$. The state-action value function following some policy is provided in Eq. (4), where γ is the discount factor:

$$Q^*(s,a) = \mathbb{E}_\pi\left[\sum_{k=0}^{\infty}\gamma^k r_{i+k+1}|s_i = s, a_i = a\right] \tag{4}$$

The state-action value function is the expected value of the sum of all future reward while following some policy. Effectively, it is a measure of *how good* a certain state-action value pair is following that policy. If one has computed the optimal state-action value function, the optimal policy can be computed as follows:

$$\pi^*(s) = \arg\max_a Q^*(s,a) \tag{5}$$

MCTS-Train uses Monte Carlo tree search (MCTS), an online tree search algorithm, to find optimal solutions to the planning problem.

**Data available online at https://bitbucket.org/avslab/basilisk-gym-interface.

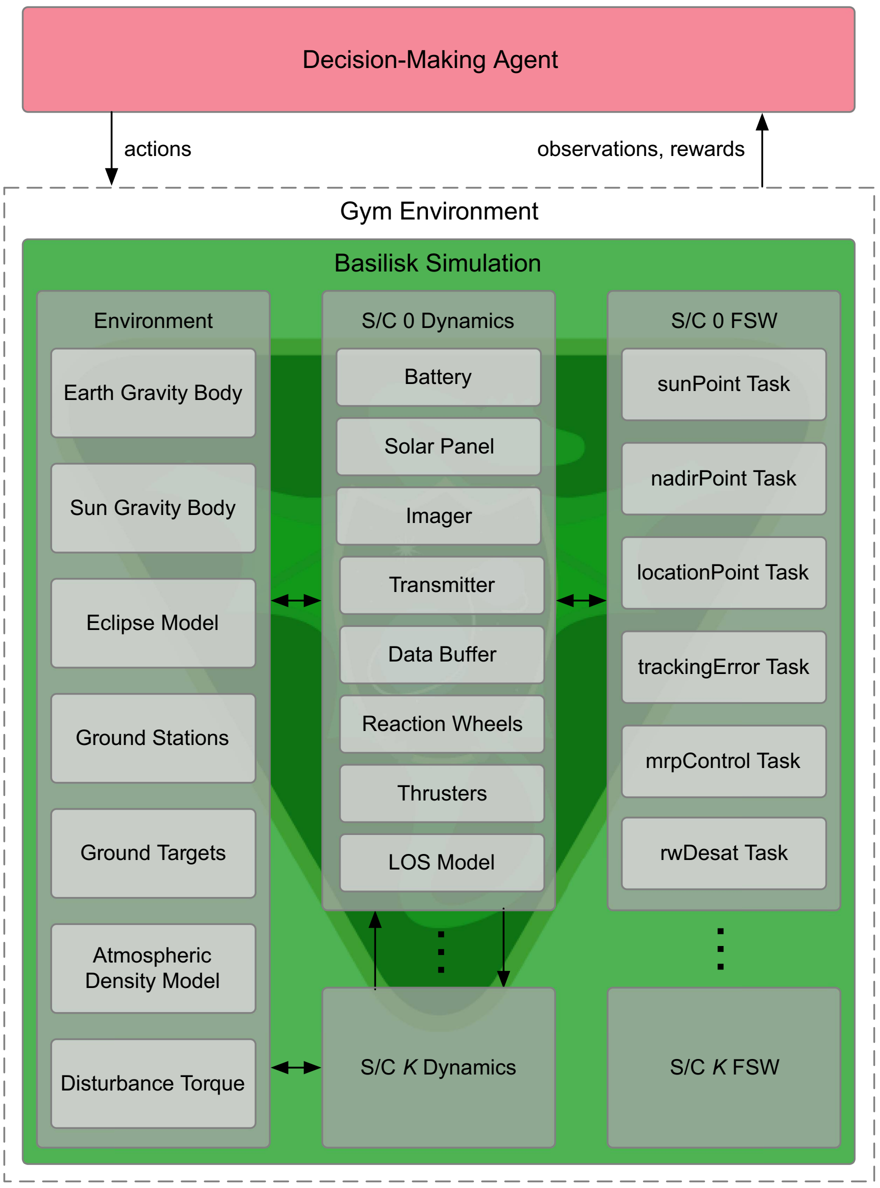


Fig. 3 Multi-satellite Basilisk simulation architecture.

At every step through the environment, MCTS runs a number of simulations to compute an estimate of the state-action value function. After all the simulations have been completed, the action that maximizes the state-action value function is selected. The environment transitions to the next state, and the process repeats until the end of the planning horizon. After the end of the planning horizon is reached, the state-action value estimates along the main trajectory of the search tree are collected and added to a data set. This data set is then regressed over with a feedforward neural network to compute a neural network approximation of the state-action value function $Q_\theta(s, a)$. The parameterized state action value function is used to create a parameterized policy $\pi_\theta(s)$ using Eq. (5).

B. Multi-Agent Deployment

The decision-making agents, or policies, are trained using the MCTS-Train pipeline. After training, the policies are deployed on board each satellite in the Walker-delta constellation. The policy on board each satellite is wrapped with a safety shield that ensures the decision-making agent only takes safe actions. For instance, if decision-making agent attempts to take an image when the data buffer is one image away from overfilling, the safety shield will override the decision with a safe action (i.e., downlink). A visual representation of the shield in action is provided in Fig. 5. Both the decision-making agent and the safety shield receive an observation from the environment. The decision-making agent passes an action to the safety shield, which evaluates the observation and action to ensure a safe action is passed to the environment. Note that the same shield is used within MCTS to guide the decisions during the search process.

The details regarding the safety shield in this work are provided in Ref. [36]. Harris et al. [25] demonstrate the first application of shielded deep reinforcement learning for EOS scheduling. Shielded deep reinforcement learning uses a linear temporal logic (LTL) specification to monitor the MDP state and the actions output by the policy, overriding unsafe actions if they violate the LTL specification [40]. The safety shield for this problem is constructed using a low-dimensional safety MDP that considers the states related to resources on board the satellite. The state space of the safety MDP is $\mathcal{S}_{\text{safe}} = \text{tumbling} \times \text{saturated} \times \text{low power} \times \text{buffer overflow}$. Each state component of the safety MDP can take a value of 0 or 1. A 0 indicates that a safety limit has not been exceeded, and a 1 indicates that a safety limit has been exceeded. There are 16 possible states, which are provided in Table 2. Safe states include $s_{\text{safe}} = \{1, 0, 0, 0\}$ or $\{0, 0, 0, 0\}$. If a safe state occurs, the trained policy's action is passed directly through the shield without modification. If an unsafe

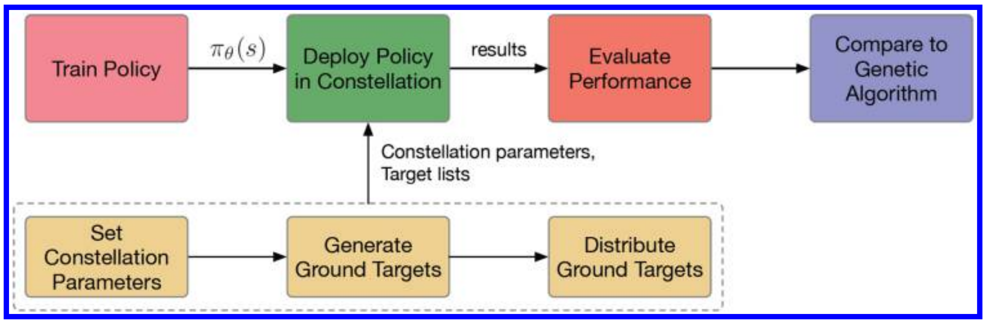

Train Policy
$\pi_\theta(s)$
Deploy Policy in Constellation
results
Evaluate Performance
Compare to Genetic Algorithm
Constellation parameters, Target lists
Set Constellation Parameters
Generate Ground Targets
Distribute Ground Targets

Fig. 4 Policy deployment pipeline.

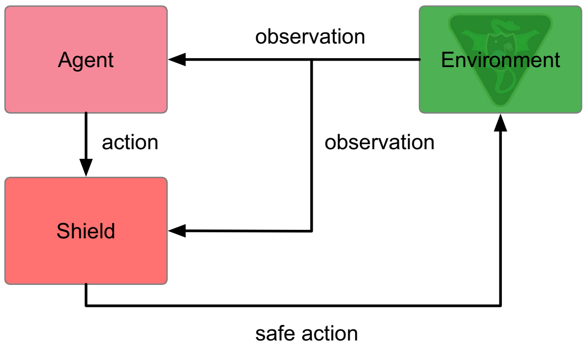


Fig. 5 Shielded agent deployment.

Table 2 Shield policy

Tumbling	Saturated	Low power	Buffer limit	Action
1	1	1	1	Charge
1	1	1	0	Charge
1	1	0	1	Desaturate
1	1	0	0	Desaturate
1	0	1	1	Charge
1	0	1	0	Charge
1	0	0	1	Downlink
1	0	0	0	— —
0	1	1	1	Desaturate
0	1	1	0	Desaturate
0	1	0	1	Desaturate
0	1	0	0	Desaturate
0	0	1	1	Charge
0	0	1	0	Charge
0	0	0	1	Downlink
0	0	0	0	— —

state occurs, the corresponding safety action, shown in Table 2, is passed to the environment instead.

C. Communication Methods

Communication between the satellites is used to locally update which targets have been imaged and downlinked to help ensure that the duplication of efforts is minimal. At the end of each decision-making interval, the satellites use the selected communication method to update their local lists of targets T_k. Four separate communication methods are implemented: no communication, single-degree line-of-sight communication, multi-degree line-of-sight communication, and free communication. These are displayed in Fig. 6.

1. No Communication

The no communication model assumes that the satellites do not update their local target lists. The local target sets T_k are never updated to include which targets have been imaged or downlinked by other satellites in the constellation.

2. Single-Degree Line-of-Sight Communication

The single-degree line-of-sight communication assumption is meant to represent a constellation with limited crosslink communication capabilities. Line-of-sight connectivity between the satellites is defined as a straight line connecting two satellites that does not intersect the Earth plus 100 km of atmosphere above the surface of the Earth. Each satellite updates its local list of imaging targets with the neighbors it is directly connected to using only one iteration of communication. Imagine a scenario in which spacecraft A has line-of-sight communication with spacecraft B but spacecraft B has line-of-sight communication with both spacecraft A and spacecraft C. Spacecraft A will not receive information about which targets spacecraft C has imaged and downlinked; it will only receive information about which targets spacecraft B has imaged and downlinked. This is demonstrated in Fig. 6b.

3. Multi-Degree Line-of-Sight Communication

The multi-degree line-of-sight communication assumption is meant to represent a constellation with near unlimited crosslink communication bandwidth. If the previous example is used again, spacecraft A will now receive information about which targets both spacecraft B and C have imaged and downlinked.

4. Free Communication

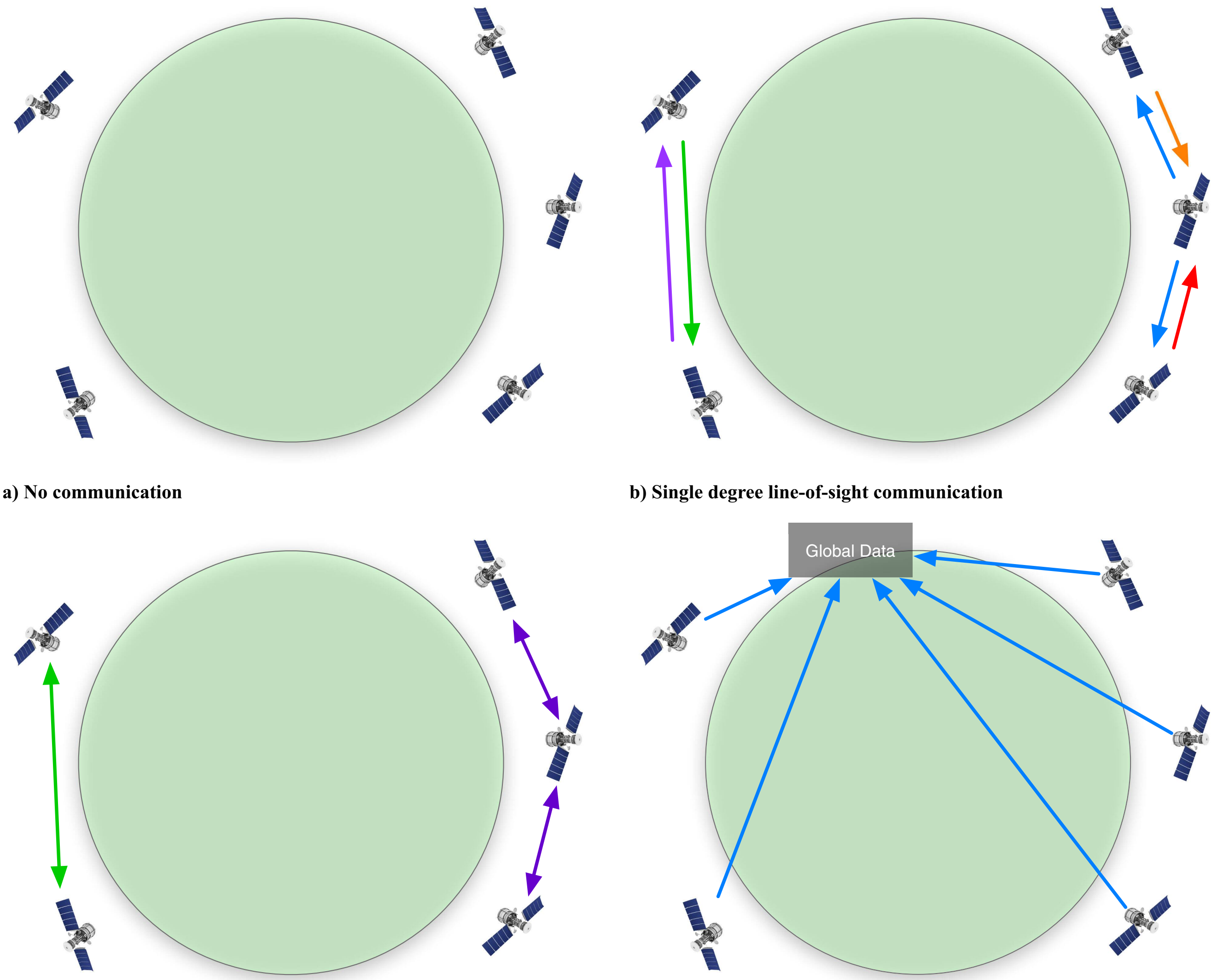
The free communication case is meant to represent a constellation with near constant access to communication resources, either ground or space based. This could include a large network of ground stations or a dedicated constellation for communication routing. In the free communication assumption, every satellite has access to which targets have been imaged and downlinked in the global target set M, and their local target lists are updated accordingly.

D. Ground Target Distribution

The set of M imaging targets is generated using uniformly distributed unit vectors projected onto the surface of the Earth. This work investigates two different methods for distributing the imaging targets between the satellites in the constellation, one centralized and one decentralized. The first method distributes the imaging targets solely based on access time in a first come, first served manner where imaging targets may be shared between satellites. The second method optimally distributes the targets using a mixed-integer program, and no imaging targets are shared between satellites.

1. Ordered Access Target Distribution

The first method of target distribution creates a set of local targets for each satellite ordered by the access time to that target. Targets may be shared between satellites using this method. The algorithm for this target distribution method is provided in Algorithm 1. Initial conditions are first generated for each satellite. Then, each satellite is looped through to create the list of local targets. The local target set is initialized, the spacecraft trajectory is propagated for the duration of the planning horizon, and the access times for each target are computed. Then, for each interval and each target, if the satellite has access to the target, the target is added to the local list.



a) No communication

b) Single degree line-of-sight communication

c) Multi-degree line-of-sight communication

d) Free communication

Fig. 6 Communication methods.

Algorithm 1 Ordered access target distribution

```
1: initialize set of initial conditions for $K$ spacecraft.
2: for spacecraft k = 1:K,
3:      initialize local target set $T_k$.
4:      propagate spacecraft trajectory.
5:      pull access times $o_{i,j,k}$.
6:      for $i \in I$,
7:          for $m \in M$,
8:              if $o_{i,m,k}$,
9:                  $T_k \cup \{m\}$ 9:
10:             end if
11:         end for
12:     end for
13: end for
14: assign local target sets to spacecraft initial conditions.
```

2. *Mixed-Integer Programming Target Distribution*

The second method of target distribution uses a mixed-integer program to generate an optimal distribution of targets. The purpose of using this method is to determine the impact that centralized coordination has on performance. The objective of the integer program is to maximize the weighted sum of targets distributed between the satellites. This objective function is provided in Eq. (6), where I is the set of decision intervals, M is the set of global targets, and K is the set of satellites; $x_{i,m,k} \in \{0, 1\}$ is the binary decision variable for whether or not a target $m \in M$ is assigned to spacecraft $k \in K$ at interval $i \in I$, and $p_m \in \mathbb{R}^+$ is the priority of target m; $o_{i,m,k} \in \{0, 1\}$ is a binary variable representing the access of spacecraft k to target m at interval i. The integer program includes several constraints. The first constraint, provided in Eq. (7), ensures that a ground target is collected no more than one time over the planning horizon. The second constraint, provided in Eq. (8), ensures that each satellite collects at most one target at every decision interval. Finally, the last constraint, Eq. (9), ensures that imaging targets are only collected when access is available:

$$\max \sum_{i\in I} \sum_{m\in M} \sum_{k\in K} \frac{x_{i,m,k}}{p_m} \tag{6}$$

such that

$$\sum_{i\in I} \sum_{k\in K} x_{i,m,k} \leq 1 \quad \forall\ m \in M \tag{7}$$

$$\sum_{m\in M} x_{i,m,k} \leq 1 \quad \forall\ i \in I, k \in K \tag{8}$$

$$x_{i,m,k} \leq o_{i,m,k} \quad \forall\ i \in I, m \in M, k \in K \tag{9}$$

Algorithm 2 Mixed-integer programming target distribution

```
1: initialize set of initial conditions for $K$ spacecraft.
2: for spacecraft k = 1:K,
3:     propagate spacecraft trajectory.
4:     pull access times $o_{i,j,k}$.
5: end for
6: construct mixed integer programming (MIP) formulation.
7: solve optimization problem.
8: construct local target sets $T_k$.
9: assign local target sets to spacecraft initial conditions.
```

Note that the target distribution program does not account for data buffer, reaction wheel speed, and power constraints. While data buffer and power constraints are straightforward to model with an integer program if linearity assumptions are made regarding their dynamics, reaction wheel speeds are not because of the highly nonlinear dynamics involved. With this target distribution method, the trained decision-making agents are in charge of resource management, and the mixed-integer program is in charge of supplying the decision-making agent with the next best target to image.

The algorithm for the MIP target distribution method is provided in Algorithm 2. Similar to the ordered access distribution, the set of initial conditions is first generated. Then, the spacecraft trajectory is propagated, and the access times are computed. The mixed-integer program is then implemented using the Python-MIP[††] optimization package and solved using the default branch and cut algorithm. After an optimal solution is generated, the target sets are assigned to each satellite.

E. Genetic Algorithm

A genetic algorithm is implemented to compare the solution method presented in this work to a method that optimizes over the entire action space, $\mathcal{A} = \mathcal{A}^1 \times \cdots \times \mathcal{A}^k$, of the MSAEO scheduling problem. A genetic algorithm is a metaheuristic optimization algorithm that searches for an optimal solution to a problem through the biological processes of evolution and natural selection. The algorithm begins by initializing a population of individuals, which are sequences of actions for the MSAEO scheduling problem. Each individual within the population is then evaluated using a fitness function. For the MSAEO scheduling problem, each sequence of actions is simply input into the MSAEO simulator, and the reward function in Eq. (2) is evaluated. The population of individuals then *mates*, or combines their sequences of actions, to produce a population of offspring, which then mutate to produce a new population of individuals. The offspring are then evaluated using the fitness function and added to the overall population. The selection operator is applied to select the best members of the population. This entire process then repeats for the specified number of generations. The pseudocode for the genetic algorithm is provided in Algorithm 3.

The genetic algorithm is implemented using the Distributed Evolutionary Algorithms in Python (DEAP) evolutionary computational framework.[‡‡] The framework has predefined operators available for selection, mating, and mutation. DEAP also offers the ability to define custom operators. The selection operator used in this work is the selection tournament which returns the best individual from a set of three individuals. The mating strategy used is a one-point crossover operator with a probability of 0.25. The population mutates using a uniform mutation operator where a sequence of actions has a probability of 0.25 to mutate, and each action in a mutating sequence has a probability of 0.3 to mutate.

[††]Data available online at https://www.python-mip.com/.

[‡‡]Data available online at https://deap.readthedocs.io/en/master/index.html.

Algorithm 3 Genetic algorithm

```
1: initialize population.
2: evaluate fitness of each individual in population
3: for generation 1:$N$,
4:     generate offspring by mating and mutating population.
5:     evaluate offspring.
6:     add offspring to population.
7:     perform selection on population and offspring.
8: end for
```

IV. Results

A. Communication Methods

To determine how assumptions regarding communication between satellites impact performance, several benchmark experiments are performed for each communication method deployed in different Walker-delta constellation designs. In the first experiment, each communication method is tested in a constellation of K satellites that reside in a single orbital plane. In the second experiment, each communication method is tested in a constellation of satellites distributed among P orbital planes.

1. Single-Plane Results

In the single-plane experiments, a constellation of $K = \{4, 7, 10, 15, 20, 30, 40\}$ satellites is deployed in a single plane at 45 deg inclination. The purpose of this experiment is to determine how performance of the trained agent depends on intraplane communication. Analytical predictions about this performance can be readily made based solely on whether or not the satellites in the orbital plane can communicate with one another or not. First and foremost, it is always expected that the free communication assumption will outperform the no communication assumption. The reason for this is obvious. However, the performance of the line-of-sight communication assumption is not as easily predicted. There will be some critical number of satellites K^* that determines whether or not the Earth occludes communication. For $K < K^*$, there will never be line-of-sight communication between the satellites, and the performance of the line-of-sight communication assumptions should match that of no communication. For $K > K^*$, the satellites will always maintain line-of-sight communication with one another, and the performance of the line-of-sight communication assumption should match that of free communication.

Assuming a semi-major axis of 6871 km and occlusion occurring for altitudes less than 100 km (due to atmospheric interference), the critical number of satellites K^* may be computed as follows, where θ^* is the angle between the right triangle formed by the satellite's radius and the occluding radius:

$$K^* = \frac{\pi}{\theta^*} = \frac{\pi}{\cos^{-1}(R_E + 100\ \text{km}/R_E + 500\ \text{km})} = 9.2\ \text{satellites} \tag{10}$$

A diagram for this is provided in Fig. 7. Therefore, for nine or fewer satellites, line-of-sight communication is identical to no communication. For ten or more satellites, line-of-sight communication will approximate free communication.

An experiment is performed for the described Walker-delta constellations; 16 samples are generated for each combination of constellation design and communication model. The results of this experiment are provided in Figs. 8–10. In Fig. 8, two-dimensional and three-dimensional views of the global and local reward are plotted. Global reward is the sum of reward across all satellites, and local reward is the average reward of each satellite following Eq. (2). The first observation to note is that the analytical prediction regarding when line-of-sight communication approximates no communication or free communication matches the experimental results. For the blue and orange curves (4 and 7 satellites), line-of-sight communication approximately matches no communication. For the green, red, purple, brown, and pink curves (greater than or equal to ten satellites), line-of-sight communication

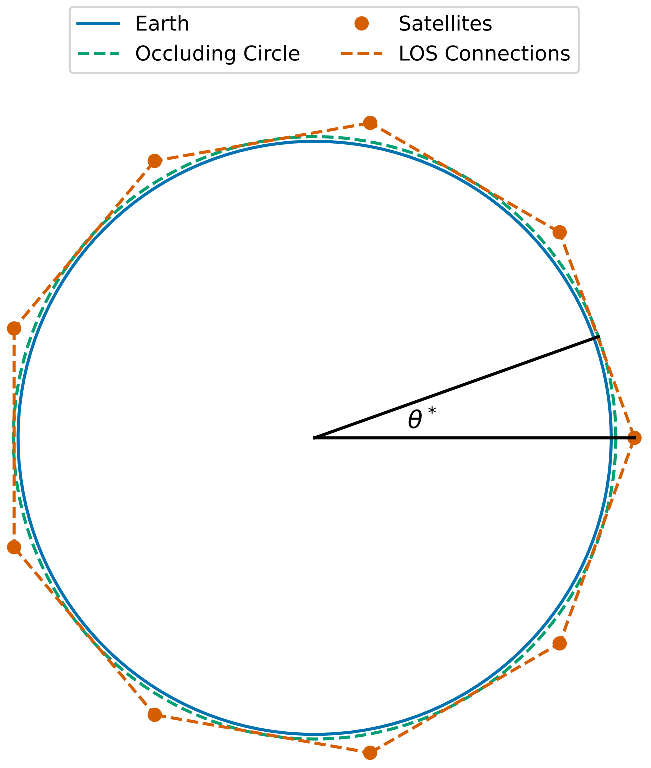


Fig. 7 The critical angle θ^*, which determines when LOS communication is not possible.

approximately matches free communication. The second observation to note is that the performance difference between the single- and multi-degree line-of-sight communication assumptions is not discernible. The two communication methods perform approximately the same. This is due to the fact that one-way information sharing between neighbors (i.e., single degree line-of-sight communication) is sufficient to ensure neighboring satellites do not duplicate another satellite's efforts.

In addition to observations regarding the performance of each communication method, observations can be made about the dependency of global and local reward on the size of the constellation and the number of global targets. In general, more global targets correlates with higher reward. For smaller constellations, the satellites become saturated with imaging tasks, and the reward plateaus. Furthermore, more satellites generally results in higher global reward. This intuitively makes sense. As satellites are added to the constellation, there are more resources available to image and downlink targets. However, local reward decreases as more satellites are added to the constellation. Because there is more competition for the available imaging targets, and because many of the satellites share the same imaging targets, local reward decreases with an increase in satellites.

In Fig. 9, the performance of the deployed agents is displayed in terms of the unique number of imaged and downlinked targets. Note that the shape of the unique number of imaged and downlinked targets match one another as well as the shape of the reward function, shape of the curves.

In Fig. 10, the percent of the collected imaging targets that are unique is plotted. As expected, the no communication assumption results in a low percentage of images that are unique. For a large constellation of 40 satellites with only 200 imaging targets, less than 10% of the imaged targets are unique. For a small constellation of four satellites with 3200 imaging targets, about 90% of the collected imaging targets are unique. The percentage of unique targets for the free communication case is heavily dependent on the size of the constellation as well. Only about 50–80% of the targets imaged assuming free communication in a constellation of 40 satellites are unique. The reason for this is that the satellites are not coordinating during a given decision-making interval. It is possible, and, depending on the priority of the target likely, that more than one satellite will attempt to image the same ground target at the same decision-making interval.

2. *Multi-Plane Results*

Duplicate benchmarks are performed for a Walker-delta constellation with multiple planes. A set of $P = \{1, 3, 5, 7, 9\}$ planes with four satellites in each plane at a 45 deg inclination is benchmarked for $M = \{200, 800, 1200, 1600, 2400\}$ global imaging targets. A phasing factor of 0 is used. Similar to the single-plane experiments, there exists some number of planes P^* where two satellites in adjacent planes at the equator can communicate with one another. Using the same assumption as the single-plane case, this number is computed as follows:

$$P^* = \frac{\pi}{2\cos^{-1}(R_E + 100\ \text{km}/R_E + 500\ \text{km})} = \frac{1}{2}K^* = 4.6\,\text{planes} \tag{11}$$

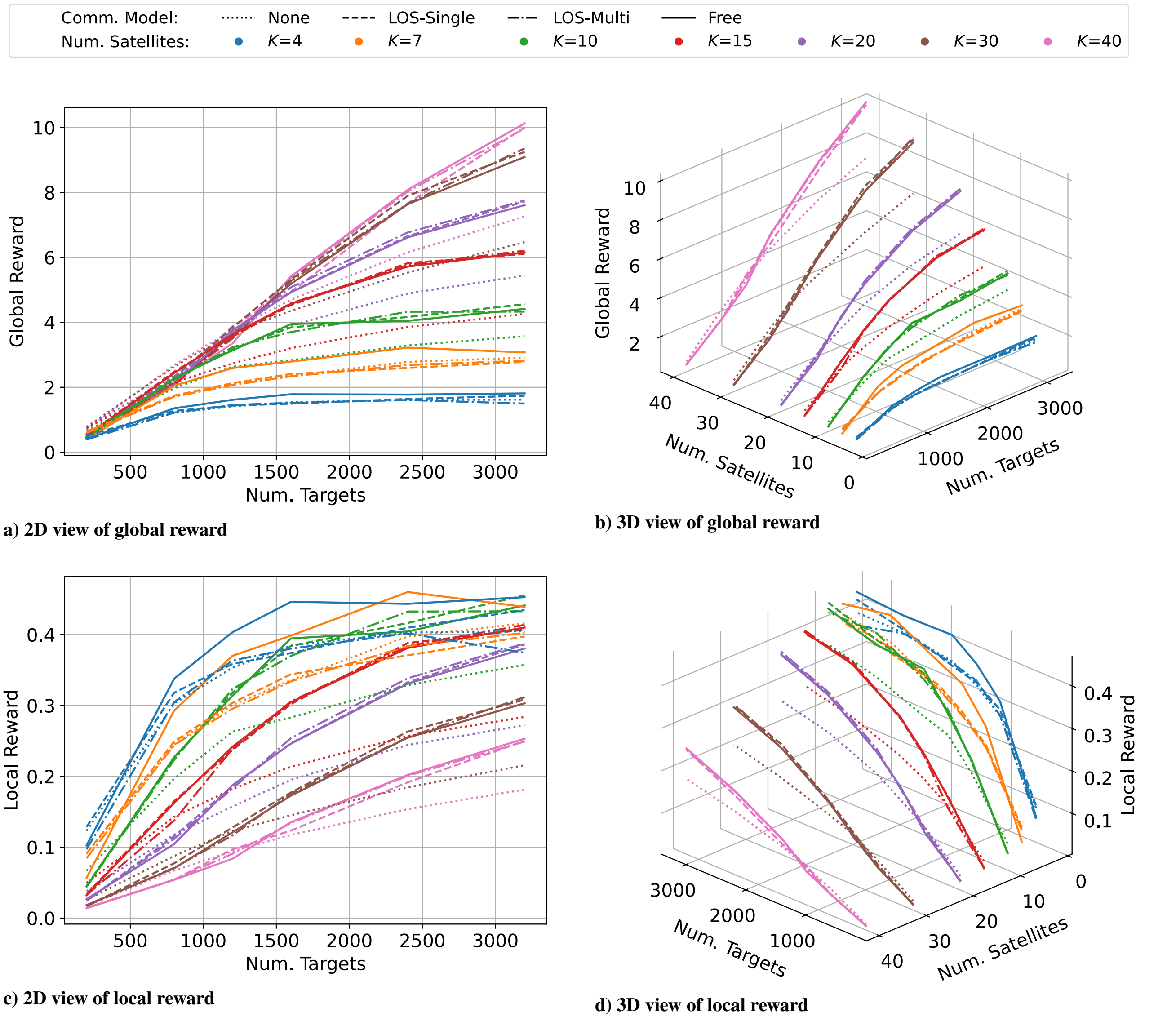
Therefore, four or fewer planes will result in no communication at the equator. Five or more planes will result in communication at the equator. To determine how the number of planes impacts performance, four satellites per plane are selected such that there is no intraplane communication. The global and local reward of the experiment is provided in Fig. 11. The local reward plots in Figs. 11c and 11d demonstrate the impact of the interplane communication on performance. For $P = \{1, 3\}$ planes, line-of-sight communication more closely matches that of no communication. However, because there is intermittent communication where orbit lines intersect, the line-of-sight communication assumption does not match the no communication assumption like it did for the single-plane experiments.

The number of unique imaged and downlinked targets is provided in Fig. 12. As expected, the no communication assumption results in the fewest amount of uniquely imaged and downlinked targets. The free communication assumption results in the most amount of uniquely imaged and downlinked targets.

Finally, the percent of imaged targets that are unique are provided in Fig. 13. These experimental results match that of the single-plane experiment. No communication results in the lowest percentage of targets that are unique. Furthermore, more planes and more satellites results in fewer uniquely imaged and downlinked targets because the probability that two satellites will image the same target at the same decision-making interval increases. This probability decreases as the number of global targets increases, but the trend is still present.

3. *Communication Method Discussion*

Several insights may be drawn from the results of the communication experiments. First, it is evident that the communication assumption has a significant impact on the performance of the constellation. However, there is little to no difference between the free and line-of-sight communication assumptions if there are enough satellites in a single plane to communicate with one another. The more satellites available in a plane, the more targets that can be imaged and downlinked. However, the probability that two satellites will image the same target increases as well, and the overall efficiency of the constellation is decreased, as evidenced by Fig. 10. If there are enough satellites in the constellation such that line-of-sight (LOS) communication approximates free communication, then there will be an overlap in the targets available to neighboring satellites (and even nonneighboring satellites if the plane is dense enough). The proposed decentralized decision-making architecture cannot ensure that the satellites do not image the same targets, as the individual decision-making agents do not communicate intent to one another, but only communicate the targets they have already imaged and downlinked. Therefore, the satellites will not know what targets their neighbors are planning to image. This is a significant limitation of the proposed decentralized decision-making architecture. However, the results of the communication experiments show that this limitation is not a significant issue if the constellation is small enough. In that case, performance is limited by whether or not the satellites can communicate at all.



a) 2D view of global reward

b) 3D view of global reward

c) 2D view of local reward

d) 3D view of local reward

Fig. 8 Global and local reward for the single-plane experiment.

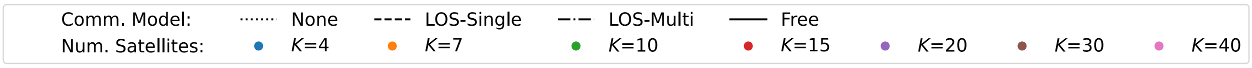
Another insight deals with the number of planes. When an equal number of satellites are distributed among planes such that no intraplane communication is available, but interplane communication is available, the performance of the constellation improves significantly. This is because the satellites within the plane do not have to worry about imaging the same targets as their neighbors and benefit from information being propagated between the planes. When comparing the efficiency metrics from the single- and multi-plane results (Figs. 10 and 13), it is evident that the multi-plane results are significantly better. In this case, a coordinated approach would likely have less of a potential to improve the results, unless the use of coordination and intent communication can help improve resource management for a single satellite, which can better prepare it for future science opportunities.

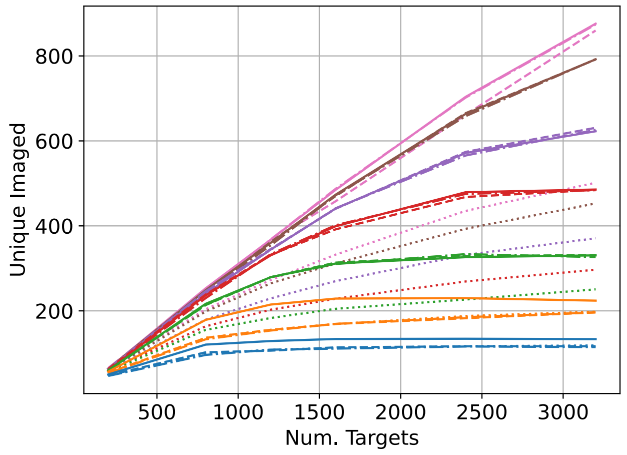
B. Ground Target Distribution Comparison

The last set of experiments addresses the question of how best to distribute targets. Two approaches are taken: one centralized and one decentralized. The centralized approach formulates a MIP optimization problem to distribute the targets between satellites such that no targets are shared. The decentralized approach assigns imaging targets to satellites as they are available, and the first satellite to capture and downlink the ground target receives the reward. For each comparison, the centralized approach uses a neural network trained with a $|U_k| = 1$. The decision-making agent only looks one target ahead. The decentralized approach uses a neural network trained with a $|U_k| = 3$. The decision-making agent looks three targets ahead. The reason for this is that the centralized distribution approach effectively prunes the list of targets to provide the next best one to the decision-making agent. Therefore, the lookahead capabilities provided by $|U_k| = 3$ are not necessary for the centralized approach.

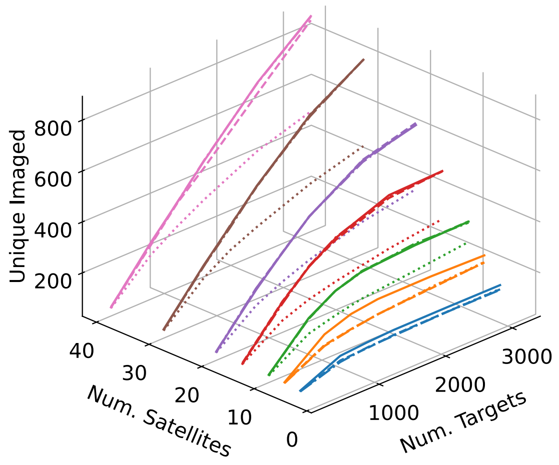
1. Single-Plane Results

The first set of ground target distribution experiments compares each method for varying numbers of satellites in a single plane. The parameters of the experiment are the exact same as the parameters of the communication experiments. For the decentralized distribution approach, free communication between satellites is assumed. For the centralized distribution, no communication between satellites is necessary. For each initial condition, provided that the access intervals are already computed, it takes on average 0.73 s to solve the MIP for the smallest case of satellites (K = 4) and imaging targets (200). However, it takes an average of 170 s to solve for the largest case of satellites (K = 40) and imaging targets (3200). The global reward for each method is provided in Fig. 14. A general trend can be observed with these plots. For large constellations (greater than or equal to 30 satellites), the decentralized target distribution approach performs best for all numbers of global targets. For small constellations (fewer than or equal to 10) satellites, the centralized distribution method performs better for almost all numbers of imaging targets. For constellations of 15–20

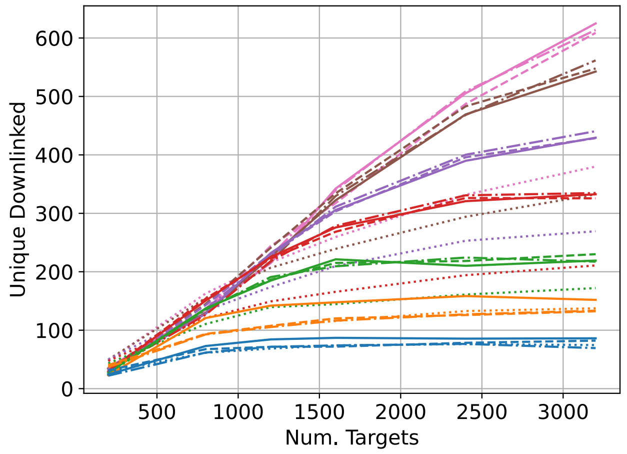




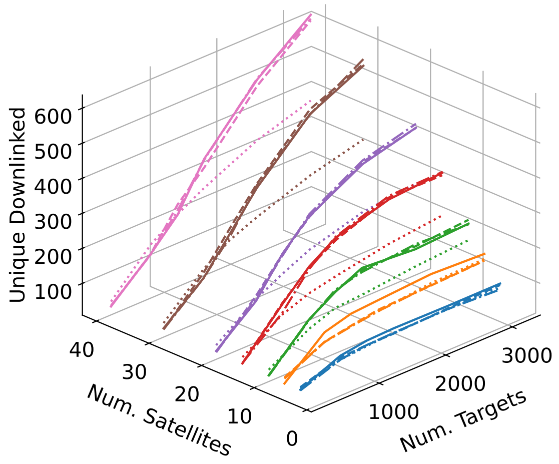
a) 2D view of imaged targets



b) 3D view of imaged targets

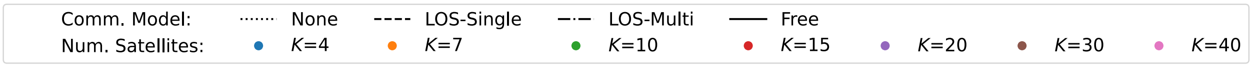


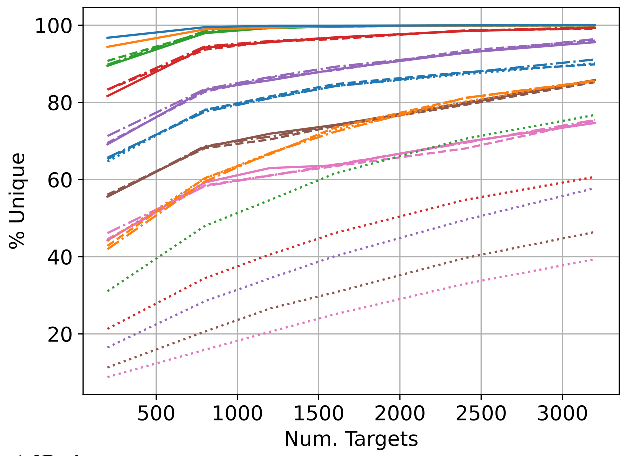
c) 2D view of downlinked targets



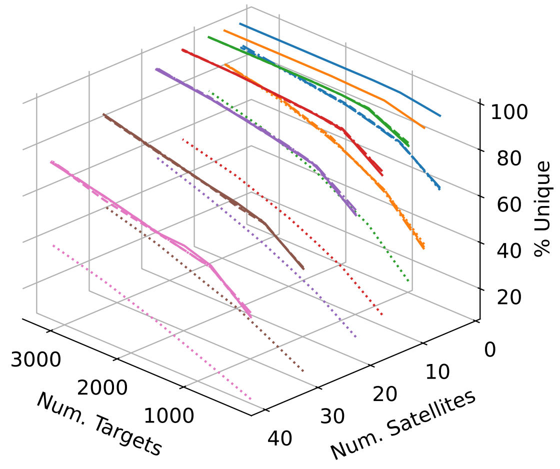
d) 3D view of downlinked targets

Fig. 9 Global numbers of unique imaged and downlinked targets for the single plane experiment.



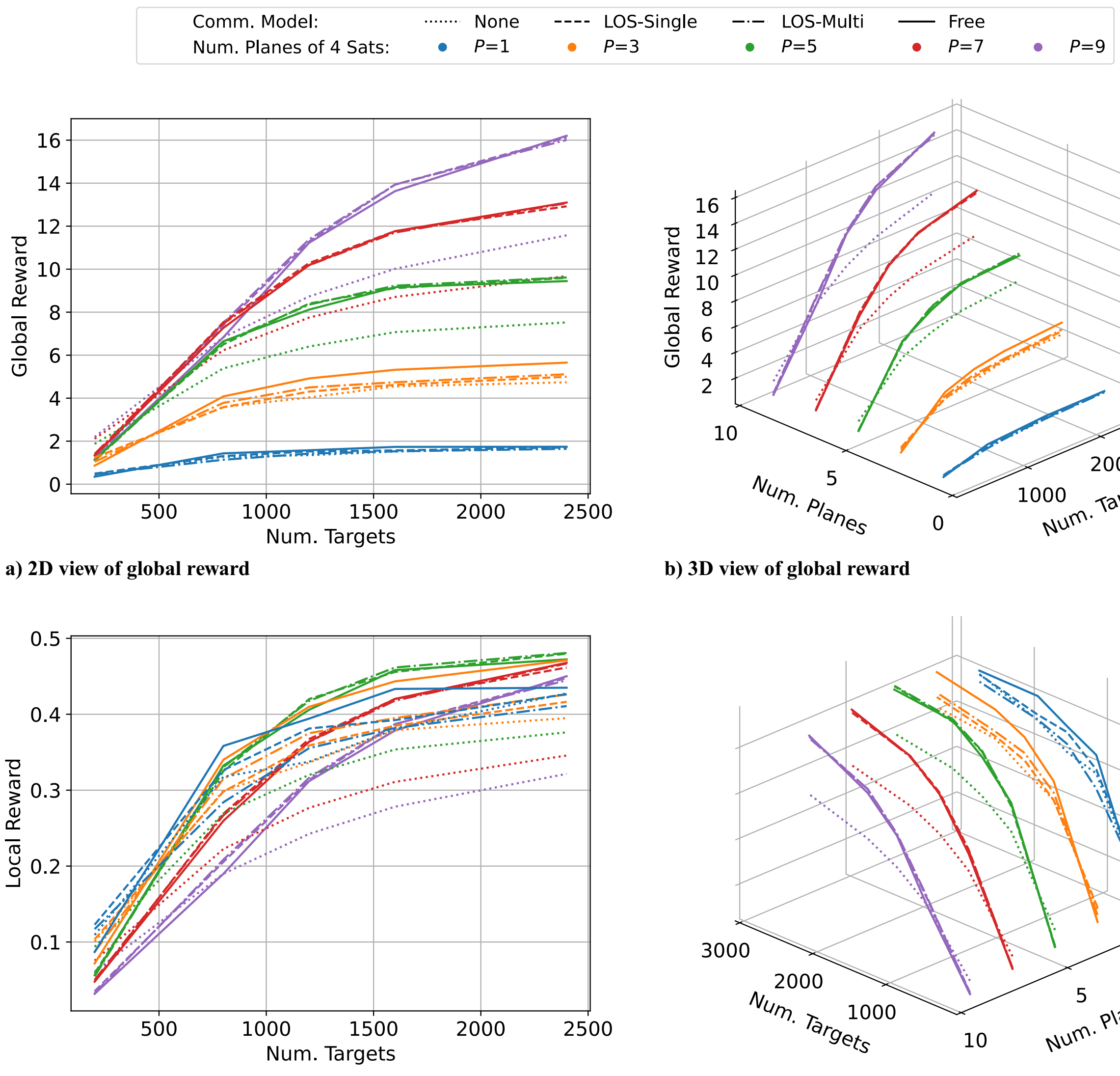


a) 2D view



b) 3D view

Fig. 10 Percent of imaged targets that are unique for the single-plane experiment.



a) 2D view of global reward

b) 3D view of global reward

c) 2D view of local reward

d) 3D view of local reward

Fig. 11 Global and local reward for the multi-plane experiment.

satellites, the centralized approach performs best for large numbers of targets but worse for small numbers of targets.

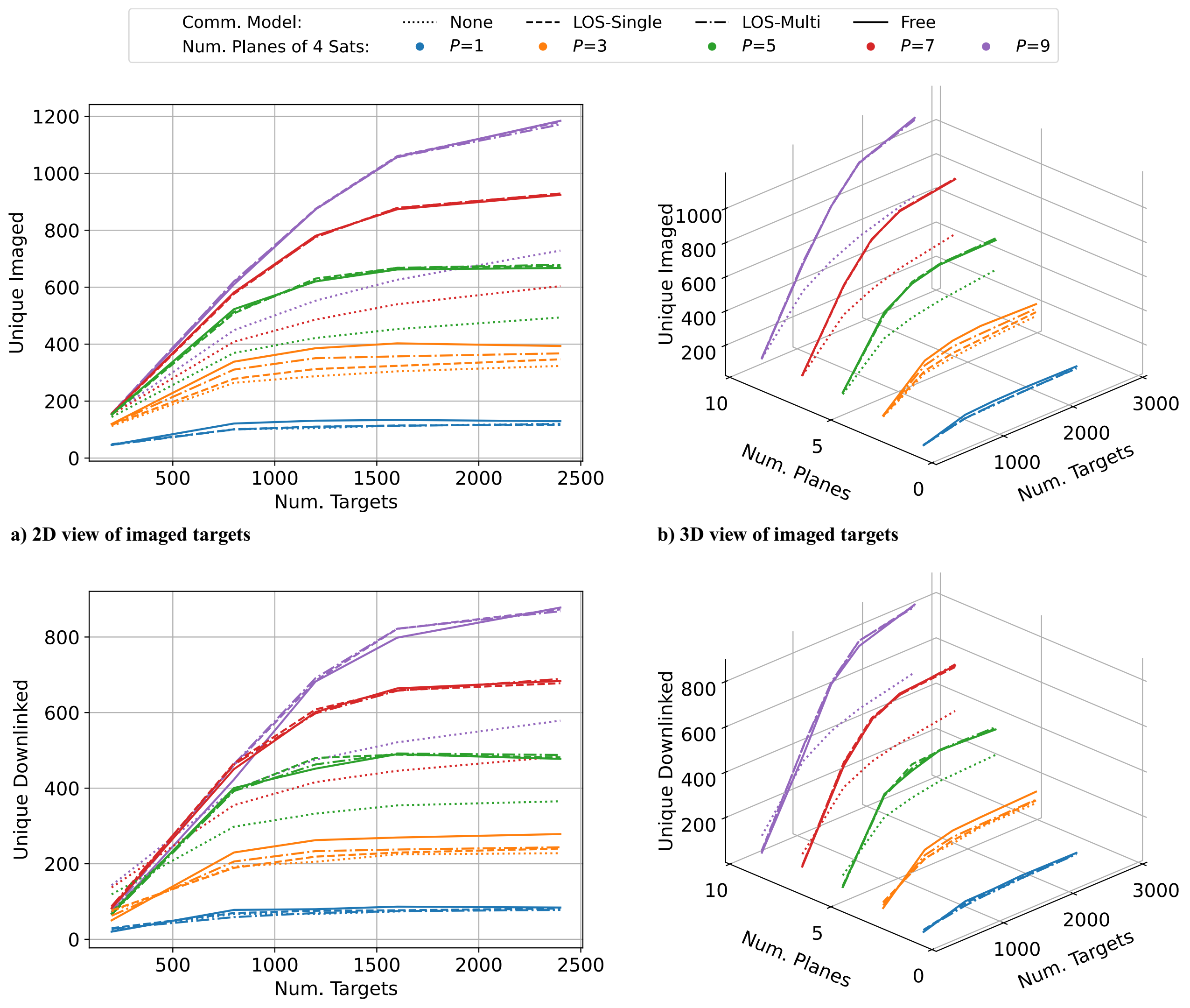
In Fig. 15, more specific metrics are plotted to better understand where and why each method performs best. In Fig. 15a, the percent difference in reward between the decentralized and centralized methods is plotted. Blue indicates that the centralized approach is better, and red indicates that the decentralized approach is better. In Fig. 15b, the average number of targets in T_k divided by the total number of steps is plotted for the centralized method. A value of 1 indicates that each satellite has on average one target available for imaging at each time step. Finally, Figs. 15c and 15d plot the percent difference in the number of unique imaged and downlinked targets.

Several observations may be made regarding these plots. First, in Fig. 15b, the centralized approach has almost one target on average per decision interval in the bottom right region, which occurs when small numbers of satellites and large numbers of targets are present. Furthermore, this region of high target assignment shows up in Fig. 15a. When the centralized approach has almost one target available each decision-making interval, it performs better than the decentralized approach. An interesting observation can be made when low numbers of targets are available (fewer than 1000). In this region, the centralized MIP approach results in fewer than 0.5 targets distributed to each satellite at each step on average. However, the centralized MIP approach performs better than the decentralized approach, which seems counter to the previous argument. An explanation for this can be found in Fig. 10. For low numbers of targets, the decentralized approach, even with free communication, results in a relatively high amount of duplication because there is more competition for a given ground target. The centralized approach ensures duplication is not an issue, so for very low numbers of targets, the centralized approach outperforms the decentralized approach.

In Fig. 15c, it is shown that the decentralized approach always results in more unique targets imaged. Two exceptions to this are in the aforementioned bottom right region, where the centralized distribution method has about the same number of uniquely imaged targets, and the left-hand region, where the centralized region has few targets distributed but performs better. Finally, Fig. 15d provides the rest of the story. In the areas where the percent difference in reward favors the centralized approach, the centralized approach is shown to have downlinked more targets on average. Intuitively, this makes sense because the largest component of reward is downlink.

2. *Multi-Plane Results*

The second set of ground target distribution experiments repeats the comparison between the centralized and decentralized distribution methods for Walker-delta constellations with multiple planes. The Walker-delta and ground target parameters are the same as those used in the communication experiment. The global reward comparing the two methods is provided in Fig. 16. The same trends observed for the single-plane experiments are shown here as well. For large constellations (more than or equal to nine planes, 36 satellites), the decentralized target distribution approach performs best for all numbers of global



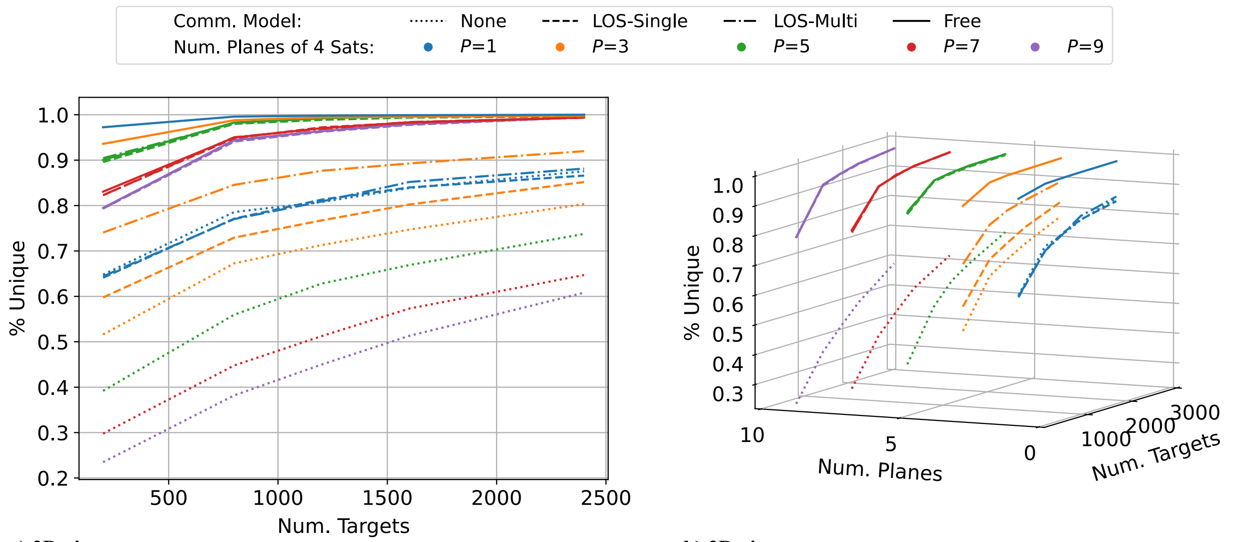
a) 2D view of imaged targets

b) 3D view of imaged targets

c) 2D view of downlinked targets

d) 3D view of downlinked targets

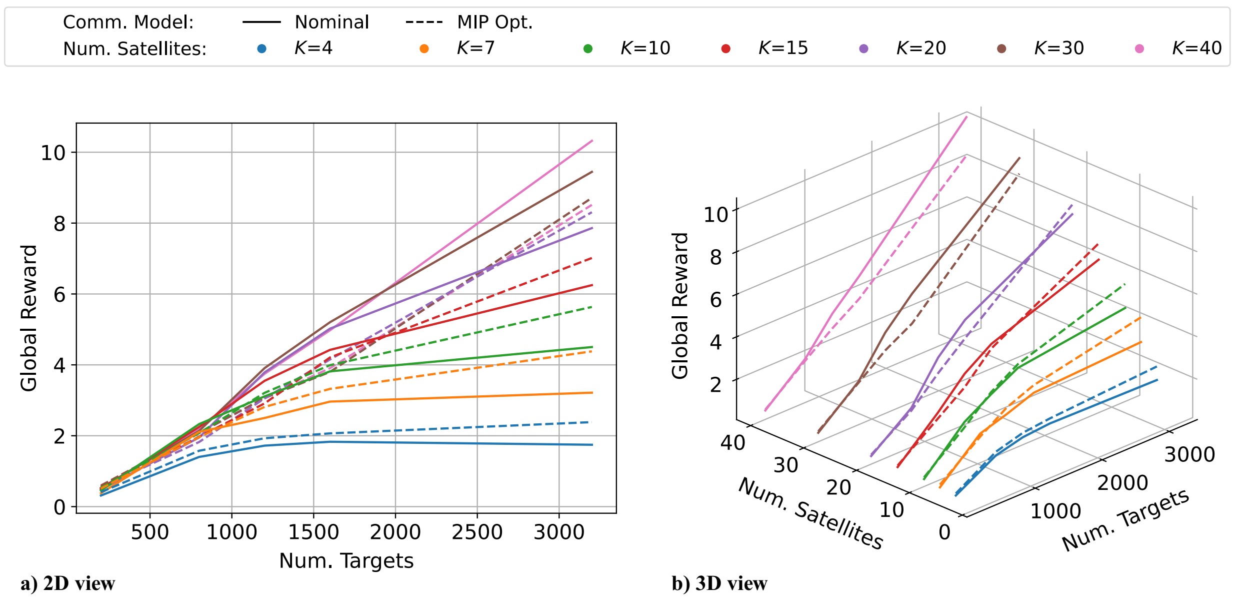
Fig. 12 Global numbers of unique imaged and downlinked targets for the multi-plane experiment.



a) 2D view

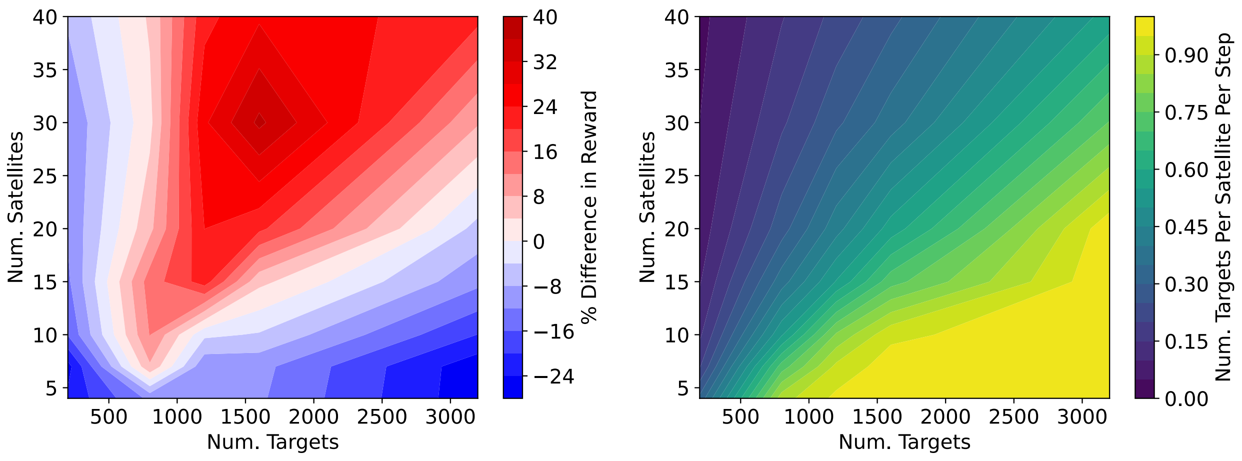
b) 3D view

Fig. 13 Percent of imaged targets that are unique for the multi-plane experiment.

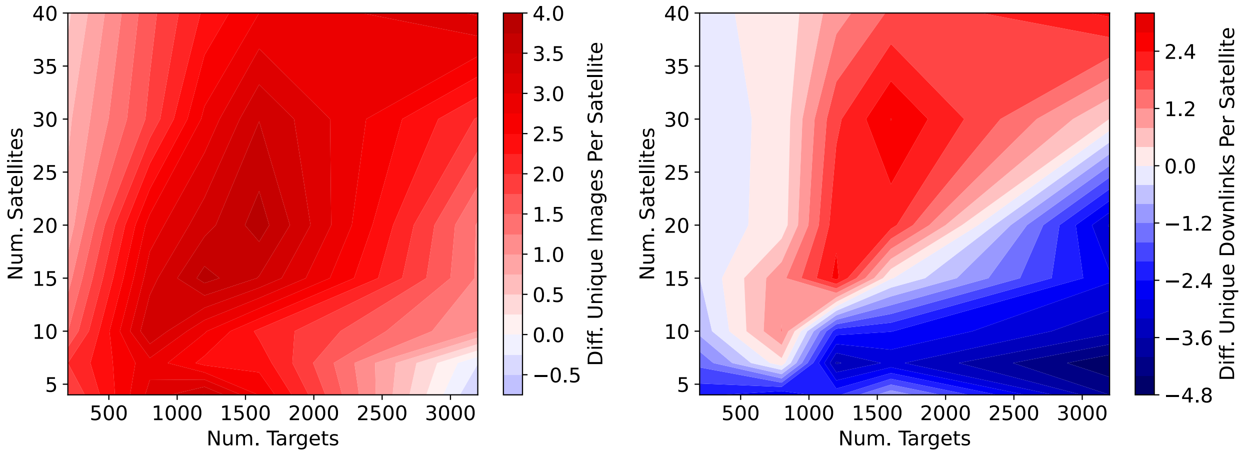


a) 2D view **b) 3D view**

Fig. 14 A comparison of global reward for nominal vs MIP target distribution methods.



a) Percent difference in reward **b) MIP target distribution statistics**

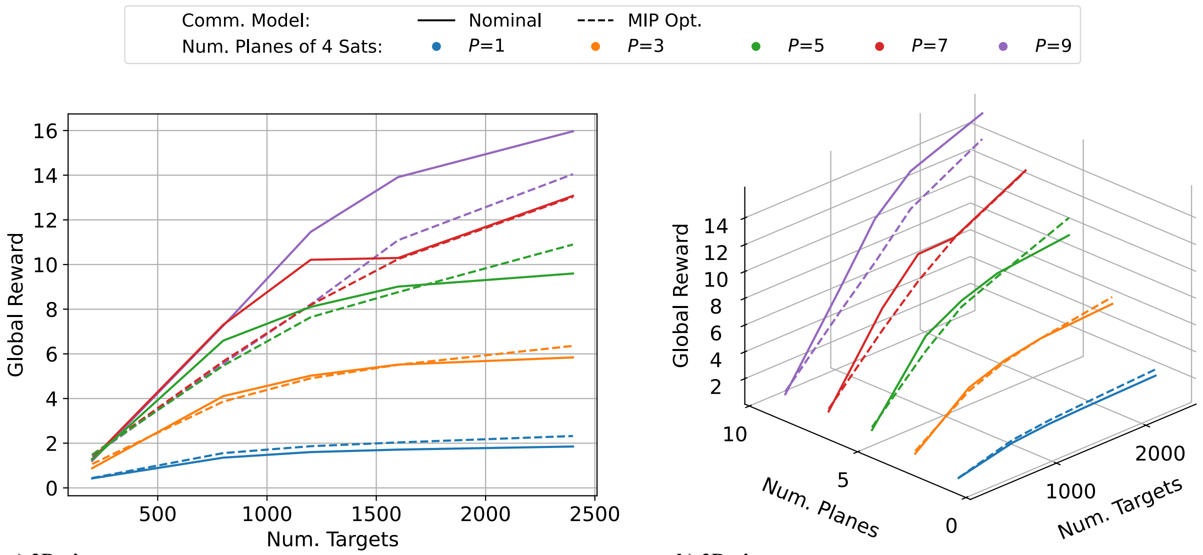


c) Difference in unique images per satellite **d) Difference in unique downlinks per satellite**

Fig. 15 Nominal and MIP target distribution comparison metrics.

targets. For small constellations (fewer than or equal to one plane, four satellites), the centralized distribution method performs better for almost all numbers of imaging targets. For constellations with three to seven planes, the centralized approach performs best for large numbers of targets but worse for small numbers of targets.

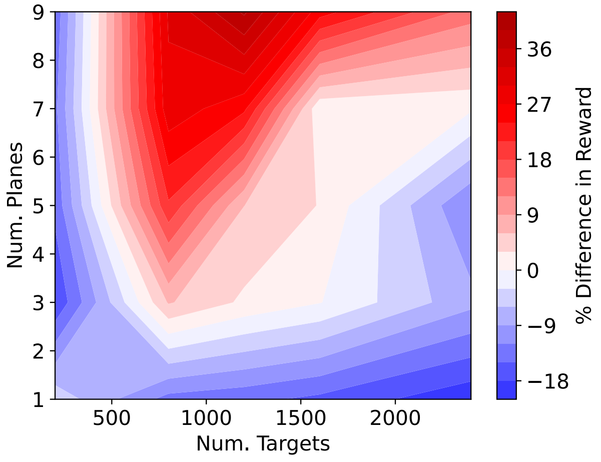
Metrics regarding the percent difference in reward, the average number of distributed targets per step, the difference in number of unique images, and difference in number of unique downlinks are plotted in Fig. 17. The trends present in the single-plane experiment are also present here. When the centralized distribution approach



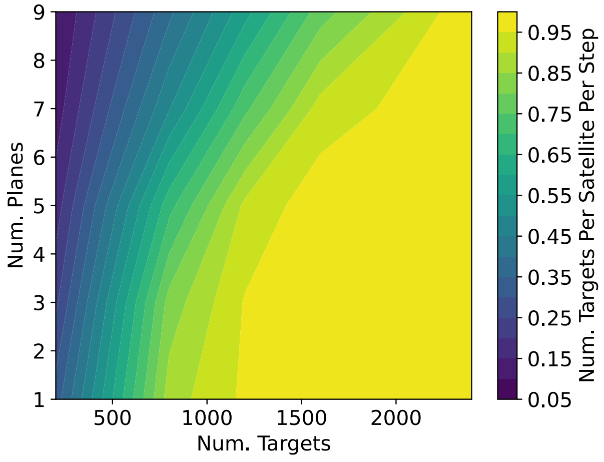
a) 2D view

b) 3D view

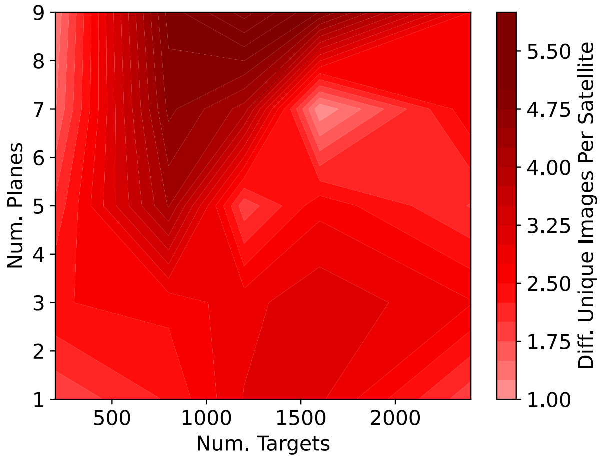
Fig. 16 A comparison of global reward for nominal vs MIP target distribution methods.



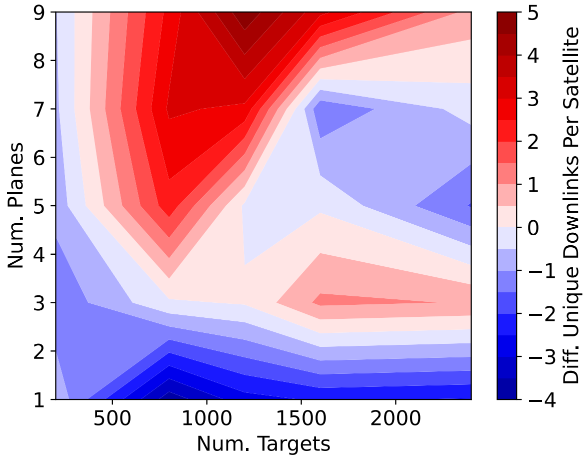
a) Percent difference in reward



b) MIP target distribution statistics



c) Difference in unique images per satellite



d) Difference in unique downlinks per satellite

Fig. 17 Nominal and MIP target distribution comparison metrics.

results in roughly one target distributed to each satellite at each decision-making interval, the centralized distribution method either performs the same as or better than the decentralized distribution method. The centralized approach also outperforms the decentralized approach for low numbers of targets because it eliminates the issue of target duplication. In all other regions, the decentralized approach outperforms the centralized approach because of its ability to capture missed targets. Finally, the decentralized distribution method results in more unique targets imaged, but the centralized distribution method results in more unique targets downlinked in regions where there centralized target assignment averages are close to 1 or where duplication following the decentralized approach is relatively high.

3. *Target Distribution Discussion*

The difference in the performance between the two target distributions highlights the power of the approach taken in this work. The centralized method of target distribution typically only improves performance for constellations with small numbers of targets or constellations with small numbers of satellites. In larger constellations with many targets, the ordered access distribution algorithm performs better because it provides more flexibility in regard to which satellite collects the target. If one satellite must perform resource management activities and miss a collection opportunity, another satellite can collect the target later. This method is also more robust to additions to target lists from other satellites within the constellation. If the satellites in the constellation fly science detection algorithms that can detect new targets, the satellites can communicate the existence of the new targets to the other satellites in the constellation. In this case, the other satellites simply need to determine when they can access the new targets and add them to their target lists accordingly. This is a much simpler process than the centralized approach, which would need to reoptimize the target distribution problem.

C. Genetic Algorithm Comparison

To compare the selected solution method to other state-of-the-art methods, a genetic algorithm is used to solve the MSAEO scheduling problem. For this experiment, two Walker-delta configurations are investigated. To keep the required computation to a minimum, each Walker-delta configuration contains only four spacecraft. With four spacecraft and $|U| = 3$, the size of the action space is $(3 + |U|)^4 = 1296$. In the first experiment, each of the four satellites is located in a single plane and has 3200 imaging targets available. In the second experiment, the four satellites are distributed among four planes and have 2400 targets available. The inclination of each satellite is 45 deg. Again, a phasing factor of 0 is used.

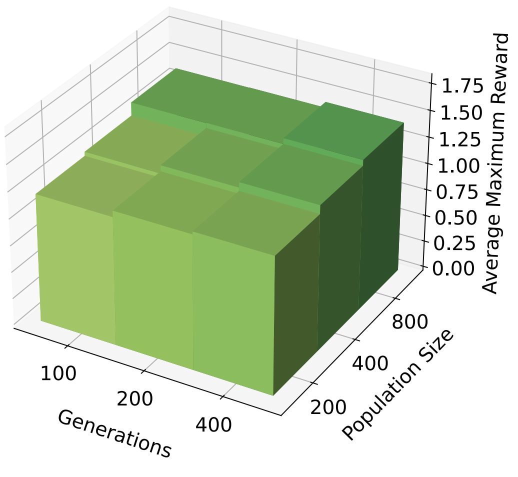
The results of the experiment are presented in Fig. 18. The genetic algorithm is benchmarked for different numbers of population size and generations. Each bar in the figure is the average of the GA performance for 20 different trials, where each trial is a different initial condition. The maximum of the z axis is set to the maximum average reward of the RL algorithm for four satellites in a single plane. This is done to provide a direct comparison between the two methods. The results here show that the GA, using the selected hyperparameters, performs worse than the RL methodology even though the GA is searching for a globally optimal solution. The GA likely needs more generations and a larger population size to converge to the globally optimal solution. However, this would come at a huge cost in terms of computation. For reference, the GA takes an average of 1.23 h to complete 100 generations with a population size of 200 using 64 cores of an AMD Milan CPU with 240 GB of RAM. At 400 generations and a population size of 800, this increases to an average of 16.97 h. The GA would need days of computation to converge to the solution found by the RL method for a single initial condition. For reference, the RL method takes 1–2 days to generate a neural network that can generalize to any initial condition, depending on how MCTS is parameterized and how many initial conditions are solved for.

V. Conclusions

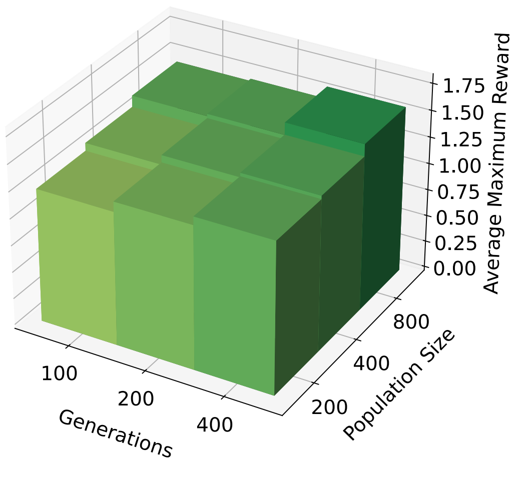
This work explores the application of single-agent reinforcement learning to multi-satellite constellation operations. To explore the performance and scalability of this method, performance benchmarks are collected for different communication assumptions and different methods of target distribution in different Walker-delta constellation designs. Four communication models are tested: no communication, single-degree line-of-sight communication, multi-degree line-of-sight communication, and free communication. The free communication model always outperforms the no communication model because satellites following the no communication model are constantly imaging and downlinking the same targets, receiving no reward for the duplication of efforts. The performance of the single- and multi-degree line-of-sight communication assumptions converge to either free communication or no communication, depending on how frequently the satellites may communicate. To test the impact on performance when coordination is used, two separate target distribution methods are also tested. The default target distribution method takes a decentralized approach where satellites are assigned targets based on their access to the imaging targets. The second target distribution method takes a centralized approach where a mixed-integer program is formulated, and the targets are optimally distributed among the satellites based on their access. The centralized target distribution method performs better than the decentralized approach when 1) the centralized approach results in 0.9–1.0 targets distributed to each satellite at each decision-making interval on average and 2) there are few targets, which results in high amounts of duplication for the decentralized approach. In the majority of other cases, the decentralized approach is best because it allows satellites to share imaging targets. If one satellite misses a ground target because it must perform resource management activities, another satellite may image that target. Finally, the RL-based solution method is compared to the performance of a genetic algorithm. The RL-based solution method is shown to outperform the GA for a fraction of the computation cost.

Acknowledgments

This work is partially supported by the NASA Space Technology Research Graduate Opportunity under grant number 80NSSC20 K1162 and is partially supported by the Air Force Research Laboratory grant number FA9453-22-2-0050. This work utilized the Alpine high performance computing resource at the University of Colorado Boulder.



a) 4 satellites. Single plane. 3200 global targets



b) 4 satellites. 4 planes. 2400 global targets

Fig. 18 Average reward of the genetic algorithm under various hyperparameters. 20 trials each.

Alpine is jointly funded by the University of Colorado Boulder, the University of Colorado Anschutz, Colorado State University, and the National Science Foundation (award 2201538). The Summit supercomputer is a joint effort of the University of Colorado Boulder and Colorado State University.

References

[1] Shah, V., Vittaldev, V., Stepan, L., and Foster, C., "Scheduling the World's Largest Earth-Observing Fleet of Medium-Resolution Imaging Satellites," *International Workshop on Planning and Scheduling for Space*, 2019, pp. 156–161, https://sites.google.com/view/iwpss/proceedings.

[2] Cappaert, J., Foston, F., Heras, P. S., King, B., Pascucci, N., Reilly, J., Brown, C., Pitzo, J., and Tallhamm, M., "Constellation Modelling, Performance Prediction and Operations Management for the Spire Constellation," *SmallSat Conference*, SmallSat, 2021, Paper SSC21-I-13.

[3] Stringham, C., Farquharson, G., Castelletti, D., Quist, E., Riggi, L., Eddy, D., and Soenen, S., "The Capella X-Band SAR Constellation for Rapid Imaging," *IGARSS 2019—2019 IEEE International Geoscience and Remote Sensing Symposium*, IEEE Publ., Piscataway, NJ, 2019, pp. 9248–9251.
https://doi.org/10.1109/IGARSS.2019.8900410

[4] Wang, X., Wu, G., Xing, L., and Pedrycz, W., "Agile Earth Observation Satellite Scheduling over 20 Years: Formulations, Methods, and Future Directions," *IEEE Systems Journal*, Vol. 15, No. 3, 2020, pp. 3881–3892.
https://doi.org/10.1109/JSYST.2020.2997050

[5] Globus, A., Crawford, J., Lohn, J., Morris, R., and Clancy, D., "Scheduling Earth Observing Fleets Using Evolutionary Algorithms: Problem Description and Approach," *International Conference on Space Mission Challenges for Information Technology*, 2003.

[6] Spangelo, S., Cutler, J., Gilson, K., and Cohn, A., "Optimization-Based Scheduling for the Single Satellite, Multi-Ground Station Communication Problem," *Computers and Operations Research*, Vol. 57, May 2015, pp. 1–16.
https://doi.org/10.1016/j.cor.2014.11.004

[7] Cho, D.-H., Kim, J.-H., Choi, H.-L., and Ahn, J., "Optimization-Based Scheduling Method for Agile Earth-Observing Satellite Constellation," *Journal of Aerospace Information Systems*, Vol. 15, No. 11, 2018, pp. 611–626.
https://doi.org/10.2514/1.I010620

[8] Kim, J., Ahn, J., Choi, H.-L., and Cho, D.-H., "Task Scheduling of Agile Satellites with Transition Time and Stereoscopic Imaging Constraints," *Journal of Aerospace Information Systems*, Vol. 17, No. 6, 2020, pp. 285–293.
https://doi.org/10.2514/1.I010775

[9] Chen, X., Reinelt, G., Dai, G., and Spitz, A., "A Mixed Integer Linear Programming Model for Multi-Satellite Scheduling," *European Journal of Operational Research*, Vol. 275, No. 2, 2019, pp. 694–707.
https://doi.org/10.1016/j.ejor.2018.11.058

[10] Monmousseau, P., "Scheduling of a Constellation of Satellites: Creating a Mixed-Integer Linear Model," *Journal of Optimization Theory and Applications*, Vol. 191, Nos. 2–3, 2021, pp. 846–873.
https://doi.org/10.1007/s10957-021-01875-2

[11] Kim, J., and Ahn, J., "Integrated Framework for Task Scheduling and Attitude Control of Multiple Agile Satellites," *Journal of Aerospace Information Systems*, Vol. 18, No. 8, 2021, pp. 539–552.
https://doi.org/10.2514/1.I010910

[12] Eddy, D., and Kochenderfer, M., "Markov Decision Processes for Multi-Objective Satellite Task Planning," *2020 IEEE Aerospace Conference*, IEEE Publ., Piscataway, NJ, 2020, pp. 1–12.
https://doi.org/10.1109/AERO47225.2020.9172258

[13] Belotti, P., Kirches, C., Leyffer, S., Linderoth, J., Luedtke, J., and Mahajan, A., "Mixed-Integer Nonlinear Optimization," *Acta Numerica*, Vol. 22, May 2013, pp. 1–131.
https://doi.org/10.1017/S0962492913000032

[14] Valicka, C. G., Garcia, D., Staid, A., Watson, J.-P., Hackebeil, G., Rathinam, S., and Ntaimo, L., "Mixed-Integer Programming Models for Optimal Constellation Scheduling Given Cloud Cover Uncertainty," *European Journal of Operational Research*, Vol. 275, No. 2, 2019, pp. 431–445.
https://doi.org/10.1016/j.ejor.2018.11.043

[15] Kennedy, A. K., "Planning and Scheduling for Earth-Observing Small Satellite Constellations," Ph.D. Dissertation, Massachusetts Inst. of Technology, Cambridge, MA, 2018.

[16] Dahl, M., Chew, J., and Cahoy, K., "Optimization of SmallSat Constellations and Low Cost Hardware to Utilize Onboard Planning," *ASCEND 2021*, AIAA Paper 2021-4172, 2021.

[17] Chien, S., Sherwood, R., Tran, D., Cichy, B., Rabideau, G., Castano, R., Davis, A., Mandl, D., Frye, S., Trout, B., Shulman, S., and Boyer, D., "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.
https://doi.org/10.2514/1.12923

[18] Chien, S., Tran, D., Rabideau, G., Schaffer, S., Mandl, D., and Frye, S., "Timeline-Based Space Operations Scheduling with External Constraints," *Proceedings of the International Conference on Automated Planning and Scheduling*, Vol. 20, No. 1, 2021, pp. 34–41.
https://doi.org/10.1609/icaps.v20i1.13410

[19] Chien, S. A., Davies, A. G., Doubleday, J., Tran, D. Q., Mclaren, D., Chi, W., and Maillard, A., "Automated Volcano Monitoring Using Multiple Space and Ground Sensors," *Journal of Aerospace Information Systems*, Vol. 17, No. 4, 2020, pp. 214–228.
https://doi.org/10.2514/1.I010798

[20] Chien, S., Mclaren, D., Doubleday, J., Tran, D., Tanpipat, V., and Chitradon, R., "Using Taskable Remote Sensing in a Sensor Web for Thailand Flood Monitoring," *Journal of Aerospace Information Systems*, Vol. 16, No. 3, 2019, pp. 107–119.
https://doi.org/10.2514/1.I010672

[21] Wei, L., Chen, Y., Chen, M., and Chen, Y., "Deep Reinforcement Learning and Parameter Transfer Based Approach for the Multi-Objective Agile Earth Observation Satellite Scheduling Problem," *Applied Soft Computing*, Vol. 110, Oct. 2021, Paper 107607.
https://doi.org/10.1016/j.asoc.2021.107607

[22] Haijiao, W., Zhen, Y., Wugen, Z., and Dalin, L., "Online Scheduling of Image Satellites Based on Neural Networks and Deep Reinforcement Learning," *Chinese Journal of Aeronautics*, Vol. 32, No. 4, 2019, pp. 1011–1019.
https://doi.org/10.1016/j.cja.2018.12.018

[23] Zhao, X., Wang, Z., and Zheng, G., "Two-Phase Neural Combinatorial Optimization with Reinforcement Learning for Agile Satellite Scheduling," *Journal of Aerospace Information Systems*, Vol. 17, No. 7, 2020, pp. 346–357.
https://doi.org/10.2514/1.I010754

[24] He, Y., Xing, L., Chen, Y., Pedrycz, W., Wang, L., and Wu, G., "A Generic Markov Decision Process Model and Reinforcement Learning Method for Scheduling Agile Earth Observation Satellites," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, Vol. 52, No. 3, 2020, pp. 1463–1474.
https://doi.org/10.1109/TSMC.2020.3020732

[25] Harris, A., Valade, T., Teil, T., and Schaub, H., "Generation of Spacecraft Operations Procedures Using Deep Reinforcement Learning," *Journal of Spacecraft and Rockets*, Vol. 59, No. 2, 2021, pp. 611–626.
https://doi.org/10.2514/1.A35169

[26] Hovell, K., and Ulrich, S., "Deep Reinforcement Learning for Spacecraft Proximity Operations Guidance," *Journal of Spacecraft and Rockets*, Vol. 58, No. 2, 2021, pp. 254–264.
https://doi.org/10.2514/1.A34838

[27] Qu, Q., Liu, K., Wang, W., and Lü, J., "Spacecraft Proximity Maneuvering and Rendezvous with Collision Avoidance Based on Reinforcement Learning," *IEEE Transactions on Aerospace and Electronic Systems*, Vol. 58, No. 6, 2022, pp. 5823–5834.
https://doi.org/10.1109/TAES.2022.3180271

[28] Gaudet, B., Linares, R., and Furfaro, R., "Adaptive Guidance and Integrated Navigation with Reinforcement Meta-Learning," *Acta Astronautica*, Vol. 169, April 2020, pp. 180–190.
https://doi.org/10.1016/j.actaastro.2020.01.007

[29] Takahashi, S., and Scheeres, D. J., "Autonomous Reconnaissance Trajectory Guidance at Small Near-Earth Asteroids via Reinforcement Learning," *Journal of Guidance, Control, and Dynamics*, Vol. 46, No. 7, 2021, pp. 1–18.
https://doi.org/10.2514/1.G007043

[30] Sutton, R. S., and Barto, A. G., *Introduction to Reinforcement Learning*, 1st ed., MIT Press, Cambridge, MA, 1998.

[31] Cui, K., Song, J., Zhang, L., Tao, Y., Liu, W., and Shi, D., "Event-Triggered Deep Reinforcement Learning for Dynamic Task Scheduling in Multi-Satellite Resource Allocation," *IEEE Transactions on Aerospace and Electronic Systems*, Vol. 59, No. 4, 2022, pp. 3766–3777.
https://doi.org/10.1109/TAES.2022.3231239

[32] Dalin, L., Haijiao, W., Zhen, Y., Yanfeng, G., and Shi, S., "An Online Distributed Satellite Cooperative Observation Scheduling Algorithm Based on Multi-Agent Deep Reinforcement Learning," *IEEE Geoscience and Remote Sensing Letters*, Vol. 18, No. 11, 2020, pp. 1901–1905.
https://doi.org/10.1109/LGRS.2020.3009823

[33] Oliehoek, F. A., and Amato, C., *A Concise Introduction to Decentralized POMDPs*, Springer–Verlag, Berlin, 2016, pp. 39–40.

[34] Kochenderfer, M. J., Wheeler, T. A., and Wray, K. H., *Algorithms for Decision Making*, MIT Press, Cambridge, MA, 2022, p. 548.

[35] Boutilier, C., "Sequential Optimality and Coordination in Multi-Agent Systems," *Proceedings of the 16th International Joint Conference on Artificial Intelligence — Volume 1 (IJCAI'99)*, Morgan Kaufmann Publishers Inc., San Francisco, CA, 1999, pp. 478–485.

[36] Herrmann, A. P., and Schaub, H., "Monte Carlo Tree Search Methods for the Earth-Observing Satellite Scheduling Problem," *Journal of Aerospace Information Systems*, Vol. 19, No. 1, 2021, pp. 70–82.
https://doi.org/10.2514/1.I010992

[37] Herrmann, A., and Schaub, H., "Reinforcement Learning for the Agile Earth-Observing Satellite Scheduling Problem," *IEEE Transactions on Aerospace and Electronic Systems*, 2023, pp. 1–13.
https://doi.org/10.1109/TAES.2023.3251307

[38] Wertz, J. R., Everett, D. F., and Puschell, J. J., *Space Mission Engineering: The New SMAD*, Microcosm Press, Hawthorne, CA, 2011.

[39] Kenneally, P. W., Piggott, S., and Schaub, H., "Basilisk: A Flexible, Scalable and Modular Astrodynamics Simulation Framework," *Journal of Aerospace Information Systems*, Vol. 17, No. 9, 2020, pp. 496–507.
https://doi.org/10.2514/1.I010762

[40] Alshiekh, M., Bloem, R., Ehlers, R., Könighofer, B., Niekum, S., and Topcu, U., "Safe Reinforcement Learning via Shielding," *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 32, AAAI Press, 2018, pp. 2669–2678.

K. Cahoy
Associate Editor