# Enhanced Dynamic Targeting for the OPS-SAT Cubesat

**Juan Delfa Victoria, Alberto Candela, Steve A. Chien**

Jet Propulsion Laboratory, California Institute of Technology
4800 Oak Grove Dr,
Pasadena, CA 91109 USA
Juan.M.Delfa.Victoria@jpl.nasa.gov Alberto.Candela.Garza@jpl.nasa.gov Steve.A.Chien@jpl.nasa.gov

## Abstract

Dynamic Targeting (DT) is a concept that enables a spacecraft to exploit data gathered from on-board instruments to plan in advance the activities of these or other instruments with the intention of maximizing the science return.

The work presented in this paper aims to further extend the capabilities of DT, including two variants with and without target-pointing planning. The latter is currently being tested on the OPS-SAT Flatsat before being deployed on the actual spacecraft, marking a significant milestone in the application of DT technology. The pointing variant will be tested in the future on spacecraft with faster slewing capabilities or larger observation windows.

## Introduction

As the number of satellites and their capabilities increase, there is a growing demand for more efficient operations to maximize the scientific return. To address this challenge, Dynamic Targeting (DT) has emerged as a promising approach.

DT enables the decision-making process to be performed on-board, utilizing in-situ data collected by on-board instruments to assess whether the conditions are appropriate to use an instrument and to determine the optimal time for sampling. This technique has the potential to enhance scientific output in two distinct ways. Firstly, it allows for prompt response to serendipitous events without the need to wait for ground-based instructions. Secondly, it reduces the amount of limited resources such as energy and bandwidth wasted, which may occur in conventional approaches where random sampling is conducted even under inappropriate instrument usage conditions.

The use of DT is expected to become more widespread in the future, benefiting from more powerful on-board computing capabilities. In this paper, we present an improved version of DT which aims to address some limitations of previous efforts. The MIRE experiment was proposed and approved to test the algorithm on-board ESA's OPS-SAT Cubesat by acquiring DT-targeted images of clouds using a utility function and user-defined constraints. We finally present early results of our investigations, highlighting the benefits and limitations of this approach, and discussing future research directions.



## Previous Work

Previous work has been developed in the area of cloud detection to decrease on-board storage and bandwidth(Thompson et al. 2014), including a one year long experiment with intelligent targeting on-board the GOSAT-2 satellite(Suto et al. 2021).

Efforts to capture images of volcanoes and wildfires through the Earth Observation Autonomy (EOA) project aimed at a more challenging approach in which the on-board system would be capable of imaging, image analysis, operations scheduling, and re-imaging within a realistic flight software and hardware (Chien and Troesch 2015). A combination of CASPER (Chien et al. 2000) and Eagle Eye (Knight, Donnellan, and Green 2013) solvers was used for the planning and scheduling part in a time window between 5-8 minutes. Recent work has focused on developing more agile cloud-imaging scenarios for spacecraft endowed with a dedicated lookahead instrument that provides early information on upcoming science targets. The spacecraft is capable of quickly iterating in a matter of seconds between lookahead data acquisition and dynamically targeting the selected targets. To this end, greedy and graph search algorithms have been used to choose from a range of targets and maximize the scientific output. These algorithms have been demonstrated in a simulator (Candela et al. 2022) and mission concept (Swope et al. 2021).

## DT Approach

The work in this paper focuses on a reactive scenario in which the spacecraft needs to react in seconds to serendipitous events while not having a lookahead instrument: DT uses the primary instrument and autonomous re-pointing to swap between lookahead and science data sampling, those significantly increasing the number of mission scenarios that could benefit from it. It also allows to reconfigure the instrument depending on the activity using it.

The following sections present first the mission scenario that will be used as reference to test DT, followed by a description of the DT technical details.

## MIRE - A DT experiment on the OPS-SAT Cubesat

The forthcoming version of dynamic targeting (DT) will undergo testing on-board the European Space Agency's (ESA) OPS-SAT (ESA accessed on 2023-03), a 3U cubesat aimed at testing innovative operational technologies in orbit. OPS-SAT, illustrated in Figure 1, represents an ideal platform for DT testing purposes, given its robustness, powerful computing capabilities, and flexibility in programming languages.

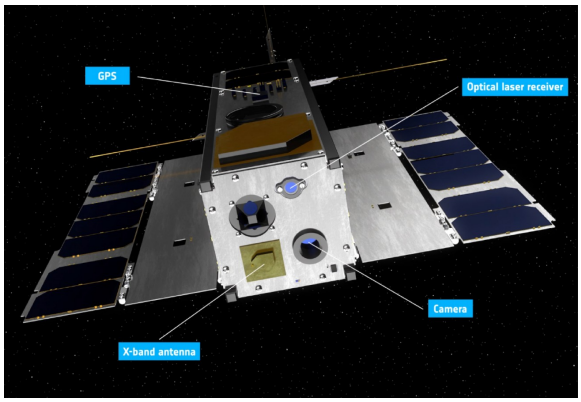


Figure 1: OPS-SAT Payload. Image courtesy of ESA

OPS-SAT is endowed with a dedicated High Performance Computer for experiments consisting of an Altera Cyclone V SoC with a 800 MHz ARM dual-core Cortex-A9 processor, 1GB DDR3 RAM and 256MB ECC RAM. Its payload includes a camera with a resolution of up to 80x80m/pixel as primary instrument, S-Band and X-Band communications and an Attitude Determination Control System (ADCS) consisting of gyros, accelerometers, magnetometers, reaction wheels, magnetorquers and a Star Tracker. The processing unit runs Linux and supports applications written in multiple languages such as C++, Python and Java, those greatly simplifying the transition from the laboratory to the space environment.

### Experiment Description

An experiment called MIRE (dynaMIc taRgeting Experiment) has been designed to acquire targeted images of clouds utilizing DT in OPS-SAT. The overall experiment sequence is presented in Algorithm 1. MIRE was submitted and approved on January 2023.

Algorithm 1: MIRE Experiment

1: **for** n iterations **do**
2: Capture a NADIR lookahead-image
3: Identify the clouds in (a region of) the image
4: Select the top n clouds as targets according to a utility function and a number of constraints defined by the user
5: Take a snap of each target
6: **end for**

Specifically, MIRE involves a sequence of iterations during which the system captures a NADIR lookahead-image, identifies clouds in the image, selects the top-ranked n clouds as targets based on a utility function and user-defined constraints, and takes a snapshot of each target. The performance of the DT algorithm will be evaluated based on the efficiency of the target selection process and the utility value of the acquired images. This experiment is expected to provide valuable insights into the capabilities and limitations of DT in space environments and its potential for enhancing operational capabilities for future space missions.

A number of constraints had to be incorporated in the mission approach and DT itself to comply with OPS-SAT characteristics:

- Payload: OPS-SAT does not have a lookahead instrument. In consequence, MIRE uses its camera also to gather lookahead data.
- Time constraints: OPS-SAT has a South-North Sun-Synchronous circular orbit at 490km altitude, 97.5° inclination. Its camera can take images with a size of 105 Km along track pointing at NADIR (in an estimated time of 2 seconds using BMP format). Hence, OPS-SAT takes under 14 seconds to move through the length of an image (see Figure 2). Considering that the Attitude Determination and Control Subsystem (ADCS) can take minutes to perform slew maneuvers in the range of 15 degrees, re-pointing to gather lookahead data or to capture centered target images is not feasible[1]. However, in our test we would need the ADCS to be able to re-point from the lookahead position to the target position (typically nearby NADIR) in few seconds.
- Considering a time window for each iteration equivalent to the time to traverse the area covered by the camera pointing at NADIR and the on-board computational capabilities available, DT had to be carefully designed to balance computational time and science acquisition time.

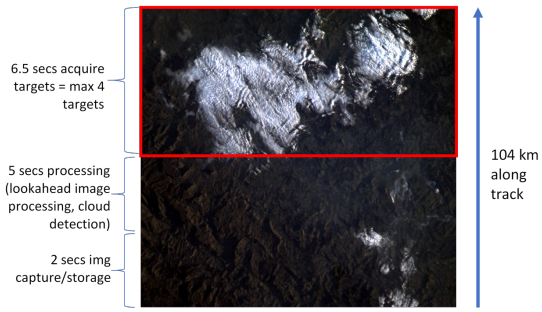


Figure 2: Time window for each lookahead/science gathering iteration

### Gathering Lookahead Information

The first step of the loop consist of gathering lookahead information to decide which clouds to target. DT allows

[1]Information about temporal performance of OPS-SAT subsystems provided by the OPS-SAT team at ESOC

to perform slew manoeuvers, e.g. to point forward on the track direction for those spacecraft that do not have a dedicated instrument. However, this is not an option for this test with OPS-SAT due to the ADCS performance as mentioned above, effectively forcing our mission to focus on a constant NADIR pointing. This fact combined with the lack of a dedicated instrument limits the available lookahead data to NADIR images.

Maintaining a high cadence of observations and targeting cycles is of utmost importance for most spacecraft, and this is especially true for OPS-SAT considering the 14 seconds per cycle cadence. To achieve it, it is necessary to optimize the dynamic targeting (DT) algorithm and reduce the processing time of the target detection/classification process, which is the primary computational time consumer in DT. To accelerate this process, two measures have been implemented.

The first measure consist of down-sampling the image resolution. This technique can reduce significantly the processing and observation time as a consequence of the reduction in the number of pixels and low level details in the image.

The second measure involves the extraction of a percentage of the image from the top as lookahead data. The percentage of the image to be extracted must be carefully selected, as a smaller value decreases the area in which science targets can be detected (and therefore the processing time), increasing the time available to perform the actual scientific observations.

This approach ensures that the lookahead step is optimized to achieve the desired high cadence of observations and targeting cycles while maintaining the required level of scientific output.

## Cloud Detection

Considering that the primary objective of this study is to demonstrate dynamic targeting rather than cloud classification, a rudimentary threshold-based cloud detection approach was chosen for its simplicity and expeditious development. However, it is worth noting that this methodology may encounter challenges in detecting clouds in low-light images (see Figure 3), while also exhibiting a propensity to produce false positives in bright images. While such errors are tolerable within the scope of this project and may be mitigated by carefully selecting favorable study areas, they would not be suitable for a rigorous scientific endeavor. The Algorithm 2 summarizes the key steps.

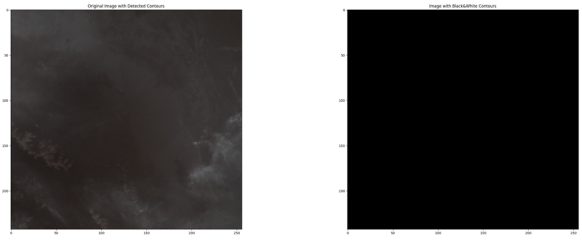


Figure 3: False negative in dark image

Algorithm 2: Cloud Detection

1: Convert the image to grayscale
2: Apply a Gaussian blur to the image to reduce noise
3: Apply thresholding to the image to convert it into a binary image
4: Apply morphology operations to remove small objects and fill gaps
5: Find white-coloured contours in the binary image
6: Calculate rewards for all the contours
7: Sort contours in decreasing order of rewards
8: Get the centroids of the n highest-reward contours
9: Return the contours

The algorithm uses the OpenCV2 library. It first converts the image to grayscale. Next, it applies a Gaussian blur to the image to reduce noise and thresholding to the image to convert it into a binary image. It then applies morphology operations to remove small objects, fill gaps and find contours in the binary image. For each contour found, the function calculates a reward and sorts the contours in decreasing order of rewards. It then gets the centroids of the n largest contours (n being a user-defined parameter), saves the image and return the centroids.

## Dynamic Targeting

The final stage of the iteration entails the selection of the targets to be sampled and subsequently capturing an image of each of them.

The algorithm first calculates a path between the centroids provided by the cloud classificator while satisfying a number of constraints, including:

- Each target is visited at most once: This constraint is actually mission-dependent. Its relaxation to allow multiple observations is subject to future work.
- The total cost derived from performing all the observations must be within a maximum boundary. It is calculated by a cost function that typically models a resource utilization such as energy. In case multiple plans satisfy the cost constraint, the plan with lower cost is selected. That captures the general desire to preserve resources for future observations.
- There is one starting node (the current spacecraft attitude) and one final node (spacecraft attitude to image the final target) which are different, that is, the spacecraft doesn't need to re-point to its original attitude.

This problem can be formulated as an optimization problem with the objective of maximizing the total utility value of a sequence of targets, subject to a budget constraint. Specifically, the problem seeks to find a subset of targets that maximizes the sum of their associated utility values while maintaining the total cost of visiting the targets below a predetermined threshold.

Formally, let $T = t_1, t_2, ..., t_n$ be the set of $n$ targets, where each target $t_i$ has a known utility value $u_i$. The traveler seeks to find a subset $S \subseteq T$ of targets to visit that

maximizes the total utility value of the sequence:

$$f_{utility} = \max_{s_1,s_2,\ldots,s_n} \sum_{i=1}^{n} u_i s_i$$

subject to a budget constraint:

$$f_{cost} = \sum_{i=1}^{n} c_i s_i \leq C$$

where $s_i$ is a binary variable that takes the value 1 if target $t_i$ is visited and 0 otherwise, $c_i$ is the cost of visiting target $t_i$, and $C$ is the total budget constraint.

We call this problem the "Dynamic Pointing Target Selection Problem" or "DP-TSP"[2].

The problem of selecting which targets to acquire can be modelled as a complete graph[3] where the nodes are the targets, each with a utility value, and the edges represent the cost to go from one node to another.

DP-TSP resembles the n-cities Open Travel Salesman Problem or n-OSTP(Chieng and Wahid 2014). The utility function, not considered in the n-OSTP problem, is captured in the Prize Collecting TSP or PCTSP(Balas 2007). It is important to note that, unlike in these TSP variants, the position of the traveler changes in DP-TSP after every observation and the extent of this change is directly influenced by the time required for the previous observation. Additionally, the plan in DP-TSP does not affect the spacecraft position, which is fully defined by its orbital properties, but instead specifies the attitude at which the spacecraft should point and the time range during which the pointing should occur.

Unlike in classical TSP, which static nature allows the problem to be solved with well studied algorithms such as integer linear programming or evolutionary algorithms, solving DP-TSP requires algorithms that can handle the dynamic nature of the problem. This can be challenging since the optimal solution at one point in time may not be optimal at a later time due to changes in the cost between nodes. Considering that the classical TSP can be solved in $O(n^4)$ using Christofides Algorithm(Neoh et al. 2020), DP-TSP would require $O(n^4T)$, where n is the number of nodes, and T is the number of time steps. This time complexity arises from computing the minimum cost for each combination of a node, time step, and subset of nodes visited so far. Therefore, DP-TSP can become computationally infeasible for large instances with many nodes and time steps. In practice, heuristics and approximation algorithms can be used to find near-optimal solutions.

We have tested a relaxed version of DP-TSP considering as cost function the accumulated slew angle and fixing the spacecraft position under the premise that the impact of its position change is negligible given the limited time window to perform all the observations, modelling it with integer programming. However, the small area of observations considered in MIRE also means that the benefit of minimizing the accumulated slew angle has little benefit. On the contrary, the computational time spent could be better used to take more observations. As a result, the OPS-SAT variant of DT forgoes the use of TSP target-planning capabilities and instead employs a simpler approach where targets are arranged based on their latitude. Notice that this is a particularity of the OPS-SAT scenario, while the more generic DP-TSP problem is subject of on-going work. Moreover, given that the time required for slewing maneuvers in OPS-SAT exceeds the time window available for observations (below 4 seconds), another relaxation of the problem requires to consider the spacecraft permanently pointing at NADIR. While these two relaxations represent an important departure of the generic DP-TSP problem, the resulting scenario is still relevant to test the capability of DT, namely to dynamically select on-board the next set of targets to be sampled.

In consequence, the OPS-SAT variant of DT doesn't use the TSP target-planning capabilities in favor of a more simple approach in which the targets are simply ordered by their latitude

Figure 4 illustrates the sequence of images obtained during a simulation of DT using the Consumer Test Tool (CTT) simulator provided with the OPS-SAT framework. It shall be highlighted that CTT is configured so that it provides random images without taking into account the impact of the spacecraft attitude on the camera's field of view.

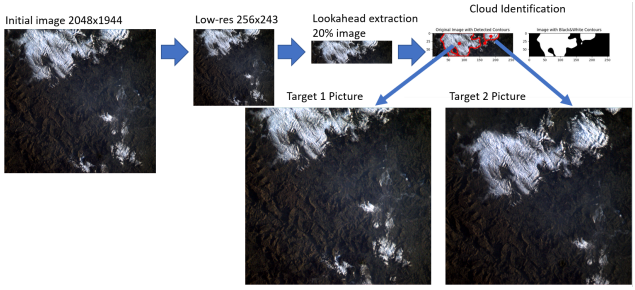


Figure 4: Sequence of images captured with DT executed in OPS-SAT CTT simulator

## Interfacing with OPS-SAT

There are two ways of developing software for OPS-SAT. The first method involves using the NanoSat Mission Operations Framework (NMF) SDK, which is written in Java and provides high level libraries to access most of the platform functionalities. The second method involves developing customized software using a C++ library, which interfaces with the payload systems.

MIRE's layer interfacing with OPS-SAT has been developed in Java using NMF, while the core DT layer is written in Python. A high level view of its architecture is presented in Figure 5.

MIRE implementation follows two key principles. The first is to encapsulate and isolate the DT algorithm from

[2]The name emphasizes that the selection of pointing targets is a dynamic process. The inclusion of "TSP" in the name also highlights the similarity to the classic Traveling Salesman Problem (TSP)

[3]The graph is complete because each pair of targets are connected (by a single node). This is not the case in the classical TSP problem in which two cities might not be connected.

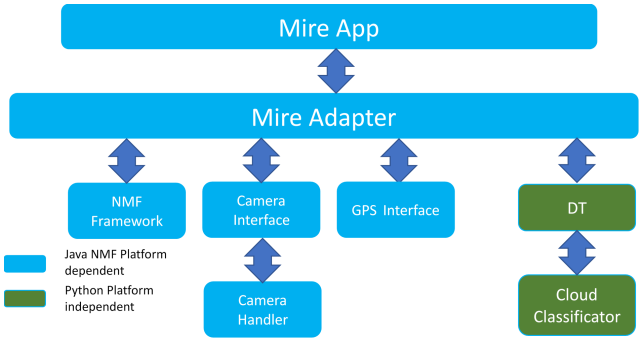


Figure 5: MIRE Architecture

spacecraft (S/C) details, ensuring that it can be easily reused in other platforms. The second is to enable users to fine-tune the DT algorithm without requiring modifications to the code. The main loop of the application is presented in Algorithm 3, which involves loading user parameters, resetting ADCS attitude, choosing the slew strategy (either pointing to the targets or permanently to NADIR), capturing a lookahead image, calling the DT algorithm to obtain a "tour" of targets, initiating pointing for each target and finally taking a picture. The main loop repeats for a number of iterations specified by the user and exits after storing all images in a designated folder to be retrieved by Ground for later analysis.

MIRE's OPS-SAT layer interfaces with three spacecraft subsystems: the GPS to gain information about the spacecraft attitude, the ADCS for slew maneuvers (if pointing is enabled) and the camera to take images. As the code is fully autonomous, we didn't use the ACTION paradigm provided in NMF to implement behaviours as there were no apparent benefit. Instead, MIRE directly command and query the subsystems via NMF services.

Algorithm 3: MIRE Main loop

```
Load User Parameters
for n iterations do
    Reset ADCS Attitude
    Point to NADIR and take lookahead image
    Call DT and obtain "tour" of targets
    for each target do
        Initiate pointing
        (optional) Wait for target alignment
        Take Picture and Store in designated folder
    end for
end for
```

## Initial Results

We conducted tests on various combinations of two parameters: the percentage of the lookahead image used for target identification and the application of image compression. Each configuration was run multiple times to obtain average durations and eliminate sporadic outliers. The goal of these simulations is to gain insight into the time consumption distribution of the key DT steps. While the absolute values are not relevant due to the use of a non-representative platform[4], the results provide valuable knowledge for advancing to the next phase, which involves testing in the FlatSat and deploying in OPS-SAT.

As depicted in Figure 6, the classification step requires most of the planning time, followed by image compression, while sorting the targets by latitude has negligible duration.

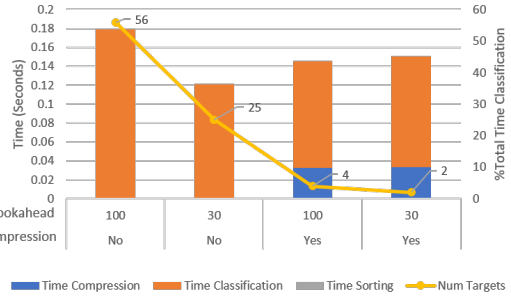


Figure 6: Time consumption for target detection, classification, and prioritization using four combinations of lookahead image percentage and image compression

Although image compression may not yield significant improvements in computing time for small observation windows, it offers a notable advantage (approximately $30\%$) when the number of potential targets increases, as demonstrated when processing the entire image. This is attributed to the reduction in the number of pixels that require processing and the elimination of small-sized targets that would otherwise need to be filtered out at a later stage in the detection process. Our experiments indicate an average reduction of over $90\%$ in the number of targets detected by the algorithm. Figure 7 provides a visual representation of this effect.

With respect to the target selection and path planning step, its execution time is expected to increase from the current $O(nlog(n))$ to $O(n^4T)$ (if TSP is used as discussed previously) for scenarios in which we can plan the spacecraft pointing maneuvers. This step has the highest temporal complexity of all the steps in the planning side and need to be carefully analyzed for each mission. In case the spacecraft has a dedicated lookahead instrument and can perform planning and instrument sampling in parallel, the planning step can take as much time as the observation time while keeping $100\%$ coverage of the terrain. The opposite case in which there is no lookahead instrument and/or planning and observations need to be interleaved, requires to consider the time balance between each step. Consider the scenario illustrated in Figure 8 where the planning time grows at a quadratic rate $O(n^4)$ while the observation time grows lin-

[4]The experiment was executed on an Ubuntu 22.04.2 LTS Virtual Machine, deployed on VirtualBox on a Windows 10 operating system. The Virtual Machine was configured with 32GB of RAM and allocated two cores of the 11th Gen Intel(R) Core(TM) i9-11950H @ 2.60GHz.

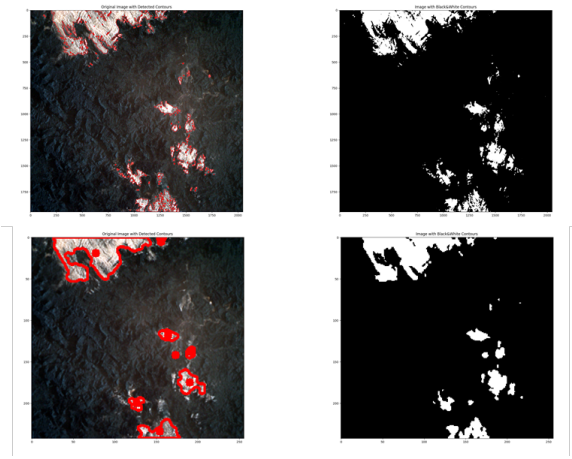

Figure 7: Impact of compression in noise reduction and target detection. The uncompressed image at a resolution of 2048x1944 result in 141 targets detected while the compressed one with a resolution of 256x248 results in 8 targets

early with respect to the number of targets[5]. If there are 24 targets, we will require an overall window of almost 10 minutes for the entire cycle, with the planning step consuming more time than the observations. If the available time window is smaller than 10 minutes, the maximum number of targets should be limited so that computation time is not wasted planning observations that cannot be satisfied.

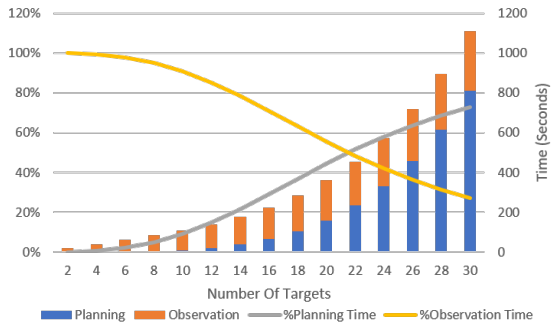


Figure 8: Planning/Observation time ratio for different temporal windows

In the particular case of the OPS-SAT scenario, the planning step only accounts for a small fraction of the total time during each lookahead/science cycle, ranging from 1% to 6%, as shown in Figure 9. The remaining time is consumed interfacing with various OPS-SAT subsystems (namely GPS, ADCS, and Camera). However, this ratio is expected to be re-balanced in the operational environment, with the computationally-intensive Python code slowing down due to the flight processor while the subsystem operations remain at similar values as their performance is limited by hardware rather than computer capability.

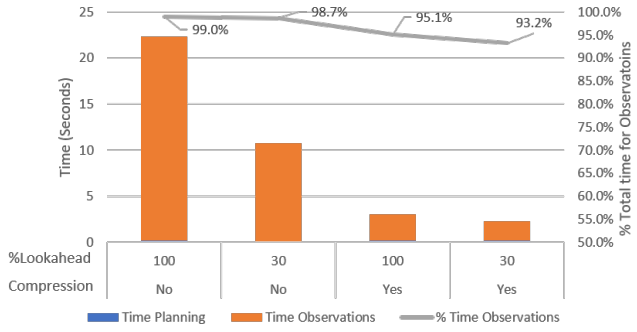


Figure 9: Overall time requirement during an interation lookahead/science

## Conclusions And Future Work

This paper presents an Enhanced Dynamic Targeting (DT) solution for the OPS-SAT CubeSat, which aims to address the limitations of previous work by enabling planning and sampling of multiple targets for each lookahead cycle and a customizable lookahead/science time window. The MIRE experiment was proposed and approved to test the algorithm on-board OPS-SAT by acquiring DT-targeted images of clouds using a utility function and user-defined constraints. DT trials with the OPS-SAT simulator have been completed with satisfactory results and is currently in the process of being tested in the FlatSat. At the same time, the DT configuration with active pointing using a slew cost function is also ready and waiting to identify an adequate platform to be tested.

The limitations of the OPS-SAT platform, namely the ADCS performance and the lack of a dedicated lookahead instrument prevented us from deploying and testing additional capabilities such as parallel lookahead and scientific data sampling or a full-fledge on-board planner/scheduler to choose the targets and optimize the pointing maneuvers.

While some capabilities designed specifically for OPS-SAT, such as target detection on a sub-region of the lookahead image for performance enhancement, may seem domain-specific, others like image down-sampling prior to processing could be applied to other spacecraft.

Furthermore, deploying and testing DT on OPS-SAT will provide valuable insights into its capabilities in space environments and potential for enhancing operational capabilities. MIRE represents the first deployment of DT in a spacecraft, paving the way for future tests on more advanced spacecraft. Additionally, we expect to demonstrate that DT can improve a mission's scientific return in a simple and cost-effective manner.

Future work in DT generalization has already begun, focusing on multi-target planning for the DP-TSP problem, defining reward functions that consider inter-dependency between observations, and multi-agent DT. These areas hold

[5]In this example assumes a constant factor of 1 millisecond to evaluate each node and 10 seconds to slew and take an image of each target

great promise for enhancing the capabilities of DT and expanding its potential applications.

**Acknowledgments.** The research was carried out at the Jet Propulsion Laboratory, California Institute of Technology, under a contract with the National Aeronautics and Space Administration (NASA) (80NM0018D0004) and supported by the Earth Science and Technology Office (ESTO), NASA.

We would also like to thank ESA-ESOC OPS-SAT team: Dominik Marszk, Sam Bammens, Vladimir Zelenvskiy and David Evans for their constant support to this experiment. Without their valuable contribution, this experiment would not have been possible.

## References

Balas, E. 2007. *The Prize Collecting Traveling Salesman Problem and its Applications*. Boston, MA: Springer US. ISBN 978-0-306-48213-7.

Candela, A.; Swope, J.; Chien, S.; Su, H.; and Tavallali, P. 2022. Dynamic Targeting for Improved Tracking of Storm Features. In *International Geoscience and Remote Sensing Symposium (IGARSS 2022)*. Kuala Lumpur, Malaysia.

Chien, S.; Knight, R.; Stechert, A.; Sherwood, R.; and Rabideau, G. 2000. Using Iterative Repair to Improve Responsiveness of Planning and Scheduling. In *International Conference on Artificial Intelligence Planning Systems (AIPS 2000)*. Breckenridge, CO.

Chien, S.; and Troesch, M. 2015. Heuristic Onboard Pointing Re-scheduling for an Earth Observing Spacecraft. In *International Workshop on Planning and Scheduling for Space (IWPSS 2015)*. Buenos Aires, Argentina.

Chieng, H. H.; and Wahid, N. 2014. A Performance Comparison of Genetic Algorithm's Mutation Operators in n-Cities Open Loop Travelling Salesman Problem. In Herawan, T.; Ghazali, R.; and Deris, M. M., eds., *Recent Advances on Soft Computing and Data Mining*, 89–97. Cham: Springer International Publishing. ISBN 978-3-319-07692-8.

ESA. accessed on 2023-03. OPS-SAT web. https://www.esa.int/Enabling_Support/Operations/OPS-SAT.

Knight, R.; Donnellan, A.; and Green, J. 2013. Mission Design Evaluation Using Automated Planning for High Resolution Imaging of Dynamic Surface Processes from the ISS. In *International Workshop on Planning and Scheduling for Space (IWPSS 2013)*. Moffett Field, CA.

Neoh, A.; Wilder-Smith, M.; Chen, H.; and Chase, C. 2020. *An Evaluation of the Traveling Salesman Problem*. Master's thesis, California State Polytechnic University, Pomona.

Suto, H.; Kataoka, F.; Kikuchi, N.; Knuteson, R. O.; Butz, A.; Haun, M.; Buijs, H.; Shiomi, K.; Imai, H.; and Kuze, A. 2021. Thermal and near-infrared sensor for carbon observation Fourier transform spectrometer-2 (TANSO-FTS-2) on the Greenhouse gases Observing SATellite-2 (GOSAT-2) during its first year in orbit. *Atmospheric Measurement Techniques*, 14(3): 2013–2039.

Swope, J.; Chien, S.; Bosch-Lluis, X.; Yue, Q.; Tavallali, P.; Ogut, M.; Ramos, I.; Kangaslahti, P.; Deal, W.; and Cooke, C. 2021. Using Intelligent Targeting to increase the science return of a Smart Ice Storm Hunting Radar. In *International Workshop on Planning & Scheduling for Space (IWPSS)*.

Thompson, D. R.; Green, R. O.; Keymeulen, D.; Lundeen, S. K.; Mouradi, Y.; Nunes, D. C.; Castaño, R.; and Chien, S. A. 2014. Rapid Spectral Cloud Screening Onboard Aircraft and Spacecraft. *IEEE Transactions on Geoscience and Remote Sensing*, 52(11): 6779–6792.