# DYNAMIC TARGETING FOR CLOUD AVOIDANCE TO IMPROVE SCIENCE OF SPACE MISSIONS

**Alberto Candela, Jason Swope, and Steve Chien**

*Jet Propulsion Laboratory* California Institute of Technology, USA*

## ABSTRACT

Dynamic Targeting (DT) will enable future Earth Observing instruments to intelligently reconfigure and point instruments to dramatically enhance science return. In this work we present a simulation study of DT for cloud avoidance. To this end we have developed several algorithms from Operations Research and Artificial Intelligence/heuristic search. We benchmark these algorithms and show that DT is a powerful tool with the potential to significantly improve science yield.

Key words: Dynamic Targeting, Artificial Intelligence, Cloud Avoidance.

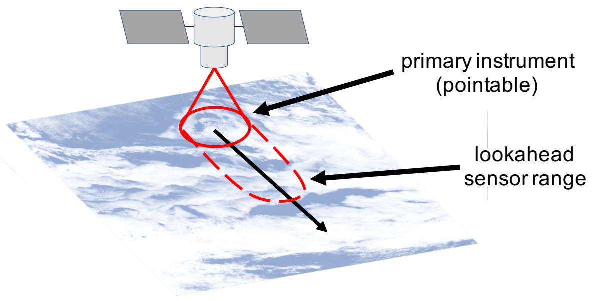

*Figure 1. Dynamic targeting uses information from a lookahead sensor to identify targets for the primary instrument in order to improve science yield.*

## 1. INTRODUCTION

While a new generation of unprecedented miniaturized Earth observing instruments has emerged, fundamental physics of remote sensing dictates that high spatial resolution at reduced size (and therefore power, cost) forces reduced swath. This places a premium on measurement on acquiring the highest science value data enabled by pointable instruments.

Dynamic targeting (DT) can improve the efficiency of conventional expensive narrow swath instruments. DT uses information from a lookahead sensor to identify targets for the primary sensor which can then be pointed or reconfigured to improve science yield (Figure 1). DT also addresses a major inefficiency in many Earth observing missions, where the majority of their data is not usable due to cloud cover and other poor observing conditions. Additionally, for other instruments that may be limited by energy, data volume, or configuration, DT can be used to best operate an instrument by turning on only when high value targets are detected (to conserve energy), varying compression/summarization (to conserve data volume), and control other instrument settings (gain, frequency, chirp rate, etc.).

DT is applicable across a wide range of missions and will enable far better coverage of transient phenomena. When overflying storm systems, a DT radar such as being developed in the SMICES project as part of the NASA Instrument Incubator Program (IIP) [1] could be directed towards the deep convective core to increase measurement of these targets of interest. The radar configuration could be adjusted based on the type of target and type of science study to further improve return. And the instrument could track a single cell for an entire overflight to provide fine temporal scale evolution for 8-10 minutes overflight in low Earth orbit. Alternatively, a forward looking sensor could detect areas of high gradient in temperature and moisture and direct lidar or radar to search for planetary boundary layer phenomena. When imaging the Earth's surface, or even an atmospheric column (such as OCO-2 and OCO-3), clouds can interfere with measurement. In these cases, lookahead cloud detection (e.g. such as Thompson et al. [2]) with DT could be used to target around clouds [3] to improve science return.

In this work we present a simulation study focused on using DT for cloud avoidance. To this end we have developed several algorithms that draw from a rich heritage of methods including Operations Research, as well as Artificial Intelligence/heuristic search methods. We benchmark these algorithms and show that DT is a powerful tool for improving science return.

*(c) 2022. California Institute of Technology. Government sponsorship acknowledged.

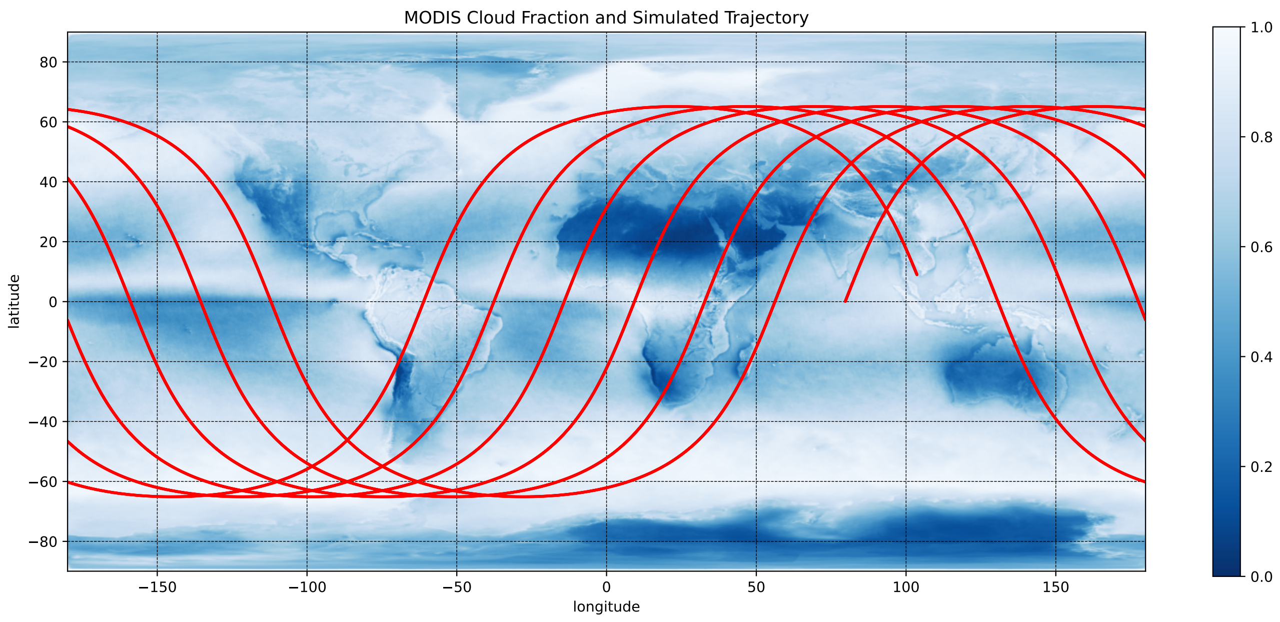

*Figure 2. MODIS cloud fraction dataset and simulated satellite orbit with a 65 degree inclination. The simulation consists of 36,000 time steps spanning 10 hours.*

## 2. RELATED WORK

Rapid cloud screening has been used onboard aircraft to remove and compress clouds to reduce data volume [2]. Another example of cloud avoidance work has been completed on TANSO-FTS-2 where intelligent targeting is utilized to minimize observations compromised by the presence of clouds [4]. Hasnain et al. use greedy and a graph-search based algorithms to select the most clear sections of sky during a flyover [3]. Similar work includes the Smart Ice Cloud Sensing (SMICES) smallsat concept, a radar application that intelligently targets storms and clouds [5, 1]. It operates in a similar manner by picking an area in the instrument field of view to analyze. SMICES targets images at a rapid rate, on the order of seconds. Moreover, SMICES uses multiple cloud labels to identify different targets instead of the labels used for cloud avoidance (cloud vs. clear). This gives SMICES flexibility on which targets to analyze and allows scientists to tailor its algorithm to target the clouds that best align with their scientific interests. Finally, Candela et al. [6] build upon the SMICES work for tracking storm features; they conduct a study with satellite mission analysis tools in conjunction with Global Precipitation Measurement (GPM) data products [7].

## 3. SIMULATION STUDY

The simulation study consists of an Earth science satellite whose mission is to minimize cloud obscured observations. The satellite has two onboard instruments: a radar with a narrow swath that serves as the primary instrument, and a secondary sensor with a wider field of view that can only be used for lookahead. General Mission Analysis Tool (GMAT) [8], an open-source space mission analysis tool, was used to simulate and generate realistic satellite trajectories. We simulated a low Earth orbit with a 65 degree inclination, a 400 km altitude, an approximate period of 95 minutes, and an eccentricity of 0 (Figure 2). Most of the previous parameters were chosen to resemble the ones for the SMICES satellite concept [5, 1]. In a similar way, the primary instrument swath is of 217 km (cross-scan is $\pm 15°$), while the lookahead sensor range is of 592 km (cross-scan is $\pm 45°$). We also used a duty cycle of 20% and each measurement from the primary instrument consumes 5% while the satellite battery recharges 1% at each time step. The experiment consists of 36,000 time steps at 1 seconds per time step, spanning a total of 10 hours. The simulations were conducted using a MacBook Pro 16 (2019) computer with an Intel i9 processor (2.3 GHz octacore) and 32 GB of RAM memory.

In order to identify clear and cloudy skies at a global scale we used data from the Moderate Resolution Imaging Spectroradiometer (MODIS) [9]. Specifically, we employed cloud fraction products from the Aqua satellite [10]. Cloud fraction is the portion of each pixel that is covered by clouds. It is derived from the 1-km-pixel resolution cloud mask product made from radiance and reflectance measurements of Earth. These data are daily, global products with a $0.1°$ spatial resolution (Figure 2). Time interpolation was used to better capture the evolution of clouds over time. Similarly to cloud detection for Landsat 8 images [11], we use cloud fraction to define three different cloud categories: cloudy, mid-cloudy, and clear (Table 1).

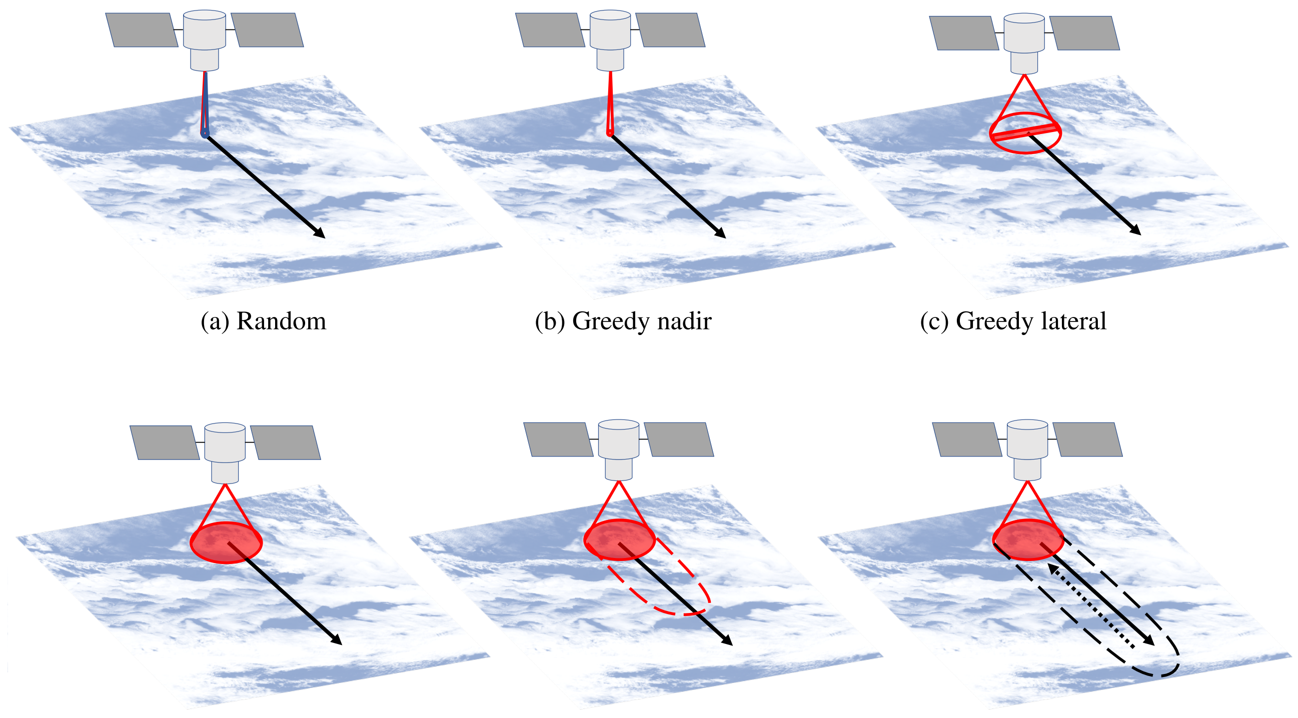

*Figure 3. Dynamic targeting algorithms for storm feature tracking. The random and greedy nadir algorithms are exclusively aimed at nadir (a and b). The greedy lateral algorithm can collect samples along the cross-path direction (c). The greedy radar algorithm has an even wider field of view, but is restricted by the primary instrument's swath (d). The greedy window algorithm leverages a lookahead sensor with a farther reach to better allocate resources for future measurements (e). The dynamic programming approach has a full lookahead (assuming the path is finite) and achieves optimality via backward induction, however it cannot be deployed using realistic instrument and computational resources (f).*

*Table 1. Cloud classification thresholds.*

| | |
|---|---|
| **Clear** | cloud fraction is less than 0.35 |
| **Mid-cloudy** | cloud fraction between 0.35 and 0.65 |
| **Cloudy** | cloud fraction is over 0.65 |

**Algorithm 1:** Random

```
input : Results: array of analyzed pixel values,
        Radar picture: knowledge window of the
        simulation, SOC: state of charge {0-100}
output: Results: array of analyzed pixel values,
        SOC: state of charge {0-100}
1 $i \leftarrow randomvalue(0 < i < 1)$
2 if $i \leq 0.2$ then
3     $results$ ←value of pixel at nadir
4     $SOC \leftarrow SOC - 4\%$
5 else
6     $results \leftarrow$ value of radar turned off
7     $SOC \leftarrow \min(SOC + 1\%, 100\%)$
8 return Results, SOC
```

## 4. DYNAMIC TARGETING ALGORITHMS

In this work we compare six different algorithms (Figure 3). Four of them are dynamic targeting algorithms that are based on greedy heuristics and can be easily deployed onboard aircraft and spacecraft. These are based on the intelligent targeting methods recently tested for SMICES [1]. The other two methods provide lower and upper bounds on performance.

### 4.1. Random

The random algorithm (1) targets the pixel under nadir 20% of the time to ensure that it meets energy requirements. It is representative of most targeting methods on current Earth Science satellites. It is indifferent to the clouds it is flying over and will most likely miss many clear pixels. It provides a lower bound on performance.

### 4.2. Greedy Nadir

The greedy nadir algorithm (2) improves upon the random algorithm by controlling when the primary instrument is turned on. It utilizes the system's current state of charge together with the cloud under nadir to determine when the primary instrument is turned on instead

**Algorithm 2:** Greedy Nadir

```
input : Results: array of analyzed pixel values,
        Radar picture: knowledge window of the
        simulation, SOC: state of charge {0-100}
output: Results: array of analyzed pixel values,
        SOC: state of charge {0-100}
if $SOC == 100\%$ then
    $results \leftarrow$ value of pixel under nadir
    $SOC \leftarrow SOC - 4\%$
else if $SOC \geq 4\%$ then
    if nadir pixel == clear then
        $results \leftarrow$ value of pixel under nadir
        $SOC \leftarrow SOC - 4\%$
    else if nadir pixel == mid-cloud then
        $r \leftarrow randomvalue(0 < i < 1)$
        if $r > 0.5$ then
            $results \leftarrow$ value of pixel under nadir
            $SOC \leftarrow SOC - 4\%$
        else
            $results \leftarrow$ value of radar turned off
            $SOC \leftarrow \min(SOC + 1\%, 100\%)$
    else
        $results \leftarrow$ value of radar turned off
        $SOC \leftarrow \min(SOC + 1\%, 100\%)$
else
    $results \leftarrow$ value of radar turned off
    $SOC \leftarrow \min(SOC + 1\%, 100\%)$
return Results, SOC
```

of making random decisions. This allows the system to save energy to collect clear sky measurements. Note that mid-clouds under nadir are not always sampled to save even more energy for clear observations. Additionally, the algorithm performs atmospheric profiling only if the state of charge (SOC) is 100% in order to reduce cloudy observations.

### 4.3. Greedy Lateral

The greedy lateral algorithm (3) improves on the greedy nadir algorithm by allowing the primary instrument to analyze pixels along the cross-path direction. The two important factors in determining when the primary instrument is turned on is the state of charge and the best pixel along the lateral band. This is resolved by searching for the highest valued cloud with a tiebreaker going to the pixel that is closest to nadir.

### 4.4. Greedy Radar

The greedy radar algorithm (4) expands its view along the path of the satellite to include the entirety of the primary instrument's reachability. The SOC determines which pixels are able to be analyzed, and a simple greedy search inside of the primary instrument's reachability finds the clearest pixel with a tiebreaker going to the one that is

**Algorithm 3:** Greedy Lateral

```
input : Results: array of analyzed pixel values,
        Radar picture: knowledge window of the
        simulation, SOC: state of charge {0-100}
output: Results: array of analyzed pixel values,
        SOC: state of charge {0-100}
$lateral\_view \leftarrow$ pixels that make up lateral band
  across nadir within radar's view
$best \leftarrow$ lat_search($lateral\_view$)    // best
  pixel in the lateral field of view
if $SOC == 100\%$ then
    $results \leftarrow$ best
    $SOC \leftarrow SOC - 4\%$
else if $SOC \geq 4\%$ then
    if best == clear then
        $results \leftarrow$ best
        $SOC \leftarrow SOC - 4\%$
    else if best == mid-cloud then
        $r \leftarrow randomvalue(0 < i < 1)$
        if $r > 0.5$ then
            $results \leftarrow$ best
            $SOC \leftarrow SOC - 4\%$
        else
            $results \leftarrow$ value of radar turned off
            $SOC \leftarrow \min(SOC + 1\%, 100\%)$
    else
        $results \leftarrow$ value of radar turned off
        $SOC \leftarrow \min(SOC + 1\%, 100\%)$
else
    $results \leftarrow$ value of radar turned off
    $SOC \leftarrow \min(SOC + 1\%, 100\%)$
return Results, SOC
```

closest to nadir. This method also performs atmospheric profiling only when the SOC is 100%.

### 4.5. Greedy Window

The greedy window algorithm (5) expands its view using the lookahead sensor, meaning that it is able to account for future clouds along the radar's path. The algorithm first calculates how many pixels can be analyzed based on the current state of charge. It then counts the number of clear pixels present within the knowledge window. The power is then allocated for all of the clear pixels, followed by mid-cloudy, and then any leftover power is reserved as free. The highest valued pixel within the radar's view that has allocated power is imaged. The tiebreaker still goes to the pixel closest to nadir. The pixel under nadir is imaged if only clouds are within the radar's view, there is free power, and there is a sufficient SOC (100%).

### 4.6. Dynamic Programming

The dynamic programming (DP) algorithm (6) is optimal and provides an upper bound on performance for the

**Algorithm 4:** Greedy Radar

```
input  : Results: array of analyzed pixel values,
         Radar picture: knowledge window of the
         simulation, SOC: state of charge {0-100}
output: Results: array of analyzed pixel values,
         SOC: state of charge {0-100}
1  $radar\_view \leftarrow$ pixels that make up radar's range of
     possible targets
2  $best \leftarrow$ radar_search(radar_view)        // best
     pixel in the radar's field of view
3  if $SOC == 100\%$ then
4      $results \leftarrow$ best
5      $SOC \leftarrow SOC - 4\%$
6  else if $SOC \geq 4\%$ then
7      if $best == clear$ then
8          $results \leftarrow$ best
9          $SOC \leftarrow SOC - 4\%$
10     else if $best == mid\text{-}cloud$ then
11         $r \leftarrow randomvalue(0 < i < 1)$
12         if $r > 0.5$ then
13             $results \leftarrow$ best
14             $SOC \leftarrow SOC - 4\%$
15         else
16             $results \leftarrow$ value of radar turned off
17             $SOC \leftarrow \min(SOC + 1\%, 100\%)$
18     else
19         $results \leftarrow$ value of radar turned off
20         $SOC \leftarrow \min(SOC + 1\%, 100\%)$
21 else
22     $results \leftarrow$ value of radar turned off
23     $SOC \leftarrow \min(SOC + 1\%, 100\%)$
24 return Results, SOC
```

**Algorithm 5:** Greedy Window

```
input  : Results: array of analyzed pixel values,
         Radar picture: knowledge window of the
         simulation, Lookahead picture: secondary
         instrument's observation, SOC: state of
         charge {0-100}
output: Results: array of analyzed pixel values,
         SOC: state of charge {0-100}
1  $best\_clear, best\_mid \leftarrow$ radar_search(radar_view)
     // Returns the best clear pixel
     and the best mid-cloud pixel in
     the radar's field of view that are
     closest to nadir
2  $free\_cycles \leftarrow$ SOC / 4
3  $clears \leftarrow$ number of clear pixels in radar picture
4  $view\_clear \leftarrow clears > 0$
5  $clears \leftarrow clears +$ number of clear pixels in
     lookahead
6  $midclouds \leftarrow$ number of midcloud pixels in radar
     picture
7  $view\_mid \leftarrow midclouds > 0$
8  $midclouds \leftarrow midclouds +$ number of clear pixels
     in lookahead
9  if $cycles \geq 1$ then
10     if $view\_clear$ then
11         $results \leftarrow best\_clear$
12         $SOC \leftarrow SOC - 4\%$
13     else
14         $cycles \leftarrow \max(0, cycles - clears)$
15         if $cycles \geq 1$ then
16             $r \leftarrow randomvalue(0 < i < 1)$
17             if $view\_mid \ \& \ r > 0.5$ then
18                 $results \leftarrow best\_mid$
19                 $SOC \leftarrow SOC - 4\%$
20             else
21                 if $SOC == 100\%$ then
22                     $results \leftarrow$ value of pixel under
                         nadir
23                     $SOC \leftarrow SOC - 4\%$
24                 else
25                     $results \leftarrow$ value of radar turned
                         off
26                     $SOC \leftarrow \min(SOC + 1\%, 100\%)$
27         else
28             $results \leftarrow$ value of radar turned off
29             $SOC \leftarrow \min(SOC + 1\%, 100\%)$
30 else
31     $results \leftarrow$ value of radar turned off
32     $SOC \leftarrow \min(SOC + 1\%, 100\%)$
33 return Results, SOC
```

previous sampling methods. Its lookahead comprises the whole path to be traversed, which is assumed to be finite. The states are given by the location of the satellite and the SOC, while the actions consist of the pixels that are within the primary instrument's reach. This algorithm uses backward induction to determine the optimal sequence of actions. The objective function is additive and the reward values for each cloud type are as follows: no sample 0, cloudy 1, mid-cloudy $1 \times 10^4$, and cloudy $1 \times 10^8$. These values virtually eliminate tradeoffs among different cloud types. Finally, the tiebreaker goes to pixels closest to nadir. Unfortunately, this algorithm cannot be deployed in most cases onboard aircraft or spacecraft for the following reasons. First, it is computationally expensive and planning requires minutes or hours depending on the total path length; second, a lookahead sensor with such range is unrealistic. However, this algorithm is valuable for evaluation and comparison purposes.

## 5. RESULTS

Overall we observe that DT methods are good at choosing when to save energy and when to collect measurements (Figure 4). Results indicate that DT delivers a significant increase in performance (Table 2). The baseline random algorithm is always outperformed and tends to observe too many clouds because it does not use any auxiliary information. The greedy nadir approach has a better performance, but it is still very constrained. Greedy radar does a much better job since it can sample from pixels, however it samples too many mid-clouds. Greedy window samples fewer mid-clouds because it exploits the

**Algorithm 6:** Dynamic Programming Algorithm

```
input  : Full path: knowledge window of the
         simulation, $N_{steps}$: steps in the full path,
         $N_{actions}$: number of pixels in radar's view,
         $SOC_0$: initial state of charge {0-100}
output: Results: array of analyzed pixel values
   // Initialization
1  $Q \leftarrow \mathbf{0}_{N_{steps} \times 101 \times N_{actions}}$      // zeros array
   // Compute optimal plan
   // Move backward
2  for $step \leftarrow N_{steps} - 1$ to $0$ do
3      for $SOC \leftarrow 0$ to $100$ do
4          for $action \leftarrow 0$ to $N_{actions} + 1$ do
               // Simulate
5              $value, SOC', reward \leftarrow$
                 $sample(step, SOC, action)$
               // Update Q
6              $Q(step, SOC, action) \leftarrow$
                 $reward + \max_a Q(step + 1, SOC', a)$
7          end
8      end
9  end
   // Execute optimal plan
10 $SOC \leftarrow SOC_0$      // set to initial SOC
   // Move forward
11 for $step \leftarrow 0$ to $N_{steps}$ do
       // Select optimal action
12     $action^\star \leftarrow \arg\max_a Q(step, SOC, a)$
       // Simulate
13     $value, SOC, reward \leftarrow$
         $sample(step, SOC, action^\star)$
       // Store results
14     $results \leftarrow value$
15 end
16 return $Results$
```

information from the lookahead sensor to save energy more effectively to sample more clear pixels. DP, as expected, outperforms the rest as it is omniscient and optimal. Nonetheless, despite a substantially more limited lookahead, greedy window has a performance that is decently close to the optimum ($\sim 80\%$). Regarding the algorithms' computation times, we see they are quite fast since they are in the order of microseconds (Table 3). The notable exception is DP as it is slower by several orders of magnitude, which is to be expected since it is optimal and uses an unrealistically far lookahead.

## 6. CONCLUSIONS AND FUTURE WORK

This work discusses DT as a powerful approach that leverages lookahead sensor data to optimize the utilization of a primary sensor, commonly subject to operation constraints, and thus improve science return. We describe several DT algorithms and test them via a realistic simulation study that involves cloud avoidance. The experimental results indicate that DT is a very promising approach. When comparing the best performing algorithm, greedy window, against the baseline random algorithm, significantly fewer clouds are sampled while respecting energy constraints. Also, its computation time is quite fast. Furthermore, it tends to have a competitive performance when compared to the optimal DP method.

Future work will keep improving the realism of our simulation study; for instance, we plan to capture more physical constraints such as off-nadir measurements with deteriorating quality. Further research will continue to inves-

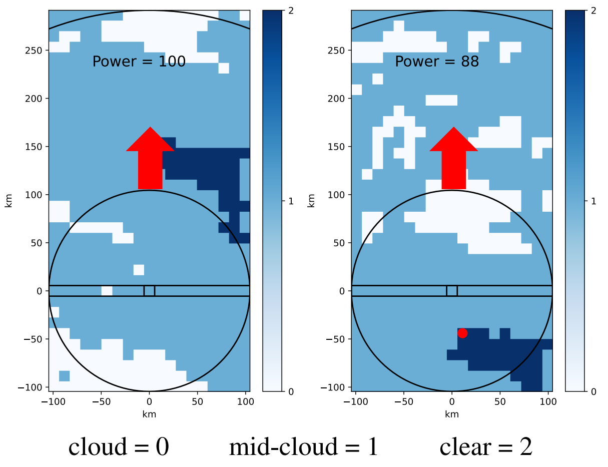

*Figure 4. Example of the greedy window algorithm for cloud avoidance. Left: It saves energy for clear observations in the near future, in this case within the lookahead sensor range. Right: A few time steps later, the algorithm uses the saved energy to collect measurements now within the primary instrument's reach.*

*Table 2. DT algorithms' performance as percentages of analyzed clouds per class. Random and DP provide lower and upper bounds on performance, respectively.*

| Algorithm | Off | Cloud | Mid-Cloud | Clear |
|---|---|---|---|---|
| Random | 80.52% | 12.93% | 5.55% | 0.99% |
| G. Nadir | 80.00% | 11.20% | 7.66% | 1.14% |
| G. Lateral | 80.00% | 10.35% | 8.06% | 1.59% |
| G. Radar | 80.00% | 9.81% | 8.07% | 2.12% |
| G. Window | 80.00% | 9.83% | 7.57% | 2.61% |
| DP | 79.91% | 9.73% | 7.08% | 3.28% |

*Table 3. Average and maximum computation times in microseconds per time step for each algorithm.*

| | Random | Nadir | Lateral | Radar | Window | DP |
|---|---|---|---|---|---|---|
| Average Time ($\mu$s) | 1.9 | 2.1 | 4.7 | 5.7 | 10.2 | 162658.6 |
| Max. Time ($\mu$s) | 9.5 | 13.0 | 24.8 | 26.8 | 27.5 | 197998.7 |

tigate the advantages of DT using other cloud and storm data sets. Finally, working closely with application scientists and specialists, we will refine use cases and quantify performance improvement for other application domains such as tracking volcanic targets.

## ACKNOWLEDGEMENTS

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). This work was supported by the Earth Science and Technology Office (ESTO), NASA.

## REFERENCES

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

[2] D. R. Thompson, R. O. Green, D. Keymeulen, S. K. Lundeen, Y. Mouradi, D. C. Nunes, R. Castaño, and S. A. Chien. Rapid spectral cloud screening onboard aircraft and spacecraft. *IEEE Transactions on Geoscience and Remote Sensing*, 52(11):6779–6792, 2014. doi: 10.1109/TGRS.2014.2302587.

[3] Z. Hasnain, J. Mason, J. Swope, J. Vander Hook, and S Chien. Agile Spacecraft Imaging Algorithm Comparison for Earth Science. In *International Workshop on Planning and Scheduling for Space (IWPSS)*, 2021.

[4] H. Suto, F. Kataoka, N. Kikuchi, R. O. Knuteson, A. Butz, M. Haun, H. Buijs, K. Shiomi, H. Imai, and A. Kuze. 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 Measurements Techniques*, 14, 2021. doi: 10.5194/amt-14-2013-2021.

[5] Xavier Bosch-Lluis, Mehmet Ogut, Sidharth Misra, Pekka Kangaslahti, Jonathan Jiang, Erich Schlecht, and William Deal. The Smart Ice Cloud Sensing (SMICES) Smallsat Concept. In *IEEE International Geoscience and Remote Sensing Symposium (IGARSS)*, 2020.

[6] Alberto Candela, Jason Swope, Steve Chien, Hui Su, and Peyman Tavallali. Dynamic Targeting for Improved Tracking of Storm Features. In *IEEE International Geoscience and Remote Sensing Symposium (IGARSS)*, 2022.

[7] Huffman, G.J. and others. GPM IMERG Final Precipitation L3 Half Hourly 0.1 degree x 0.1 degree V06. `https://disc.gsfc.nasa.gov/datasets/GPM_3IMERGHH_06/summary`, 2019. Greenbelt, MD, Goddard Earth Sciences Data and Information Services Center (GES DISC). Accessed: 2021-11-01.

[8] NASA Goddard Space Flight Center, USA. Design And Integration Tools. General Mission Analysis Tool (GMAT) v.R2016a. (GSC-17177-1). `https://software.nasa.gov/software/GSC-17177-1`, 2016. Reference Number: GSC-17177-1. Accessed: 2022-04-01.

[9] C.O. Justice, E. Vermote, J.R.G. Townshend, R. Defries, D.P. Roy, D.K. Hall, V.V. Salomonson, J.L. Privette, G. Riggs, A. Strahler, W. Lucht, R.B. Myneni, Y. Knyazikhin, S.W. Running, R.R. Nemani, Zhengming Wan, A.R. Huete, W. van Leeuwen, R.E. Wolfe, L. Giglio, J. Muller, P. Lewis, and M.J. Barnsley. The moderate resolution imaging spectroradiometer (modis): land remote sensing for global change research. *IEEE Transactions on Geoscience and Remote Sensing*, 36(4):1228–1249, 1998. doi: 10.1109/36.701075.

[10] Platnick, S. and Ackerman, S. and King, M. and others. MODIS Atmosphere L2 Cloud Product 06 L2. `http://dx.doi.org/10.5067/MODIS/MOD06_L2.006`, 2015. NASA MODIS Adaptive Processing System, Goddard Space Flight Center, USA. Accessed: 2022-04-01.

[11] Steve Foga, Pat L. Scaramuzza, Song Guo, Zhe Zhu, Ronald D. Dilley, Tim Beckmann, Gail L. Schmidt, John L. Dwyer, M. Joseph Hughes, and Brady Laue. Cloud detection algorithm comparison and validation for operational landsat data products. *Remote Sensing of Environment*, 194: 379–390, 2017. ISSN 0034-4257. doi: https://doi.org/10.1016/j.rse.2017.03.026. URL `https://www.sciencedirect.com/science/article/pii/S0034425717301293`.