Using Intelligent Targeting to increase the science return of a Smart Ice Storm Hunting Radar

Jason Swope[1], **Steve Chien**[1], **Xavier Bosch-Lluis**[1], **Qinq Yue**[1], **Peyman Tavallali**[1], **Mehmet Ogut**[1], **Isaac Ramos**[1], **Pekka Kangaslahti**[1], **William Deal**[2], **Caitlyn Cooke**[2]
[1] Jet Propulsion Laboratory, California Institute of Technology
[2] Northrop Grumman
Jason.Swope@jpl.nasa.gov

Abstract

Smart Ice Cloud Sensing (SMICES) is a small-sat concept in which a radar intelligently targets ice storms based on information collected by a lookahead radiometer. Often space observations are performed by continuously collecting data from an instrument aimed at nadir (e.g. directly below the space platform). However, if the platform has the ability to assess science utility of features being overflown, an intelligent measurement scheme can improve science return. This can be achieved by controlling the on/off state of the instrument if it is not able to continuously operate (e.g. due to energy or thermal constraints), and by allowing the instrument to view off nadir if it has pointing capabilities. In the case of SMICES, power constraints and the rarity of storms means that with blind nadir targeting SMICES would collect a limited amount of ice storm radar data. The algorithms proposed acquire measurements to maximize acquired high interest storms while concurrently collecting a background sampling of all features. We use a cloud classification system to identify five different cloud types. Six algorithms ranging from "blind" to more selective are described and results from evaluation on a dataset of 13 ground swaths covering 72,399,600 km2 of data are presented. This data is from high quality science simulations that contain all five cloud types and multiple storms. When utilizing the radiometer's lookahead and the full range of the radar the results show a 23.7x and 1.9x increase over the base algorithm in the most and second most important cloud types respectively.

Introduction

High altitude ice clouds, covering more than 50% of the Earth's surface are often produced from high-impact deep convection events (Luo and Rossow 2004), and are strong modulators of Earth's weather and climate (Stephens 2005; Bony et al. 2006). High altitude ice clouds play a significant role in the Earth's energy balance and hydrologic cycle through their effects on radiative feedback and precipitation, and therefore crucial for life on Earth.

SMICES is designed to increase our knowledge of the phenomena by collecting data on the vertical resolution of the ice cloud particles. This information has never been analyzed through a global satellite. Instead, only in-situ ice particle size data has been available. Therefore SMICES will be able to provide an innovative path towards the quantification of how ice cloud radiative effects impact convective storm intensity, size, and track as well as constraining climate model simulations of ice cloud feedbacks and associated hydrological processes, contributing to reducing uncertainties of climate predictions.

This paper focuses on the targeting system of the SMICES radar. Current targeting systems for satellite imaging consist of continuously targeting nadir. If the instrument cannot be on for the entire duration of the orbit, it is randomly turned off to meet the energy restraints. This method ensures that the data collected will be either the ground-track of the satellite, or a random subsample of that. It also only takes images at nadir, which is the angle that returns the best data. This approach fails to address the problem that some parts of the sky are more intriguing than others. Smart targeting can be used to guide an instrument to focus its analysis on the more interesting areas of the sky as the instrument flies over.

SMICES is actively targeting clouds, specifically deep convective storms. Even though global cloud coverage can span roughly $2/3$ of the Earth (King et al. 2013), deep convective storms are far more rare than all clouds. While SMICES would fly over and collect data from some storms using the general image targeting technique, its performance can be improved with development in the pathing of its scientific instrument. SMICES intends on utilizing the knowledge gained from its cloud classification to allow its planning algorithms to target the most scientifically interesting clouds along its path. Another challenge that the SMICES algorithms must account for are the power constraints that do not allow continuous observation during the entirety of its mission. If a random schedule were to control when the radar is turned on, important clouds would be overlooked.

Related Work

We focus on the targeting of clouds to guide the radar through the storms. Similar work has been conducted on the inverse problem of cloud avoidance. Cloud screening onboard aircraft has worked to help cut out swaths of data compromised by cloud cover to reduce the amount of downlinked information (Thompson et al. 2014). While SMICES is trying to collect the most useful data, its algorithms are designed to control the data collection process instead of discard invalid portions of previously collected data.

Other implemented cloud avoidance work has been completed on TANSO-FTS-2 where intelligent targeting is utilized to minimize the amount of cloud coverage captured in its images (Suto et al. 2021). Other work that focuses on developing algorithms to achieve cloud avoidance is being done at NASA Jet Propulsion Laboratory where a greedy and a graph search based algorithm has been developed to select the most clear sections of sky during a flyover (Hasnain et al. 2021). This work is more similar to SMICES as its goal is to target its instrument at more scientifically relevant features during flight. However, the algorithms differ in their actual targets. Storms are significantly rarer than clear sky and are composed of different types of clouds. SMICES prioritizes two different types of clouds through its flight in contrast to the single feature of clear sky addressed in cloud avoidance.

Background

The SMICES problem is a continuous online planning problem implemented as an orbiting satellite where there is no set end to the imaging of the clouds. The parameters include an orbiting altitude of 400km and the setup is designed to allow its radar to slew 15° from nadir in all directions. The slewing is assumed to have instantaneous electronic movement. The radar is capable of targeting an area of roughly 4x4km whenever it is turned on. The power constraints of the vehicle mean that the targeting should reach a 20% duty cycle at any given time over the course of the flight. If these constraints were applied to current satellite targeting techniques, the result would be randomly sampling 20% of the clouds at nadir under the satellite. Identification of the clouds occurs within the radiometer's range, which sweeps at 45° ahead of nadir. This sweep covers 60° around the satellite and therefore covers more ground than the radar is capable of viewing.

When modelling the radar's viewpoint, shown in figure 1, we focus on the knowledge window, which is defined as the area from the back of the radar's reachability to the front of the radiometer sweep. The window is only as wide as the radar's movement since we are unable to target any clouds outside of this range. We investigate smart targeting in the context of this model.

Within our knowledge window it is necessary to define the priority of the different clouds that will be viewed. The classifier onboard is capable of distinguishing between five different cloud types, clear sky, thin cirrus, cirrus, rainy anvil (RA), and convection core (CC), in order of increasing interest. In particular, the rainy anvil and convection core clouds are the most important since they make up storms. The scientists also want to make sure that the radar is collecting data from all of the cloud types it passes over since there can be useful information in the non-storm clouds as well. To accomplish this, whenever an algorithm is not targeting rainy anvil and convection core clouds, the radar will be taking a random sampling of the clouds under nadir. To simplify, the algorithms will be created to prioritize convection core clouds, followed by rainy anvil clouds, and then a random sample of the clouds under nadir.

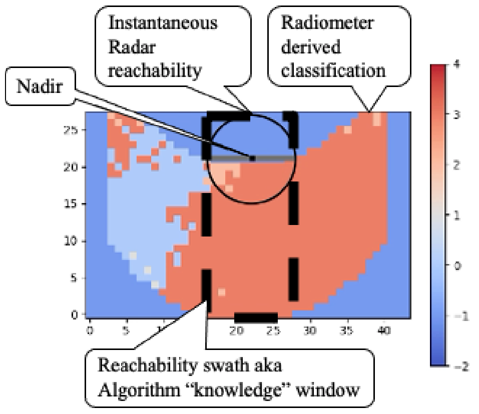

Instantaneous Radar reachability
Radiometer derived classification
Nadir
Reachability swath aka Algorithm "knowledge" window
0
5
10
15
20
25
0
5
20
25
30
35
40
4
3
2
1
0
−1
−2

Figure 1: Example of the field of view of the satellite. Identified storms are represented by each color: 4.0 = convection core, 3.0 = rainy anvil, 2.0 = cirrus, 1.0 = thin cirrus, 0.0 = clear, -2 = not analyzed. Radar's View: black circle, Nadir: black square, Knowledge Window: dotted rectangle

Multiple algorithms were created with an increasing field of view to demonstrate the improvements gained by utilizing more of the information available to the instrument. The random algorithm serves as the baseline comparison as it is representative of the results we would receive without targeting any clouds. The only algorithms that utilize the entirety of the knowledge window are the windowed smart and greedy path, which plan out the projected usage of power beyond the immediately reachable pixels.

Algorithms

The duty cycle of the algorithms is dictated by the state of charge (SOC) of its system. In order to ensure that the SMICES duty cycle of 20% is maintained the SOC is decreased by 4% when the algorithms analyze a pixel, and the SOC is increased by 1% when the radar is left off. The simulation begins with 0% power.

Throughout this section there are figures displaying how each algorithm would perform in a given knowledge window. In each figure nadir is represented by the black square and the radar's view is displayed by the circle. The different cloud types are represented on the color bar such that 4.0 = convection core, 3.0 = rainy anvil, 2.0 = cirrus, 1.0 = thin cirrus, 0.0 = clear. These values also represent the reward given for analyzing each cloud type. The state of charge for every figure is assumed to be 50

Random

The random algorithm (figure 2 and algorithm 1) targets the pixel under nadir 20% of the time to ensure that it meets the energy requirements for SMICES. It is representative of most targeting methods on current Earth Science satellites today. Its random nature means that it is indifferent to the clouds it is flying over and will miss some important clouds.

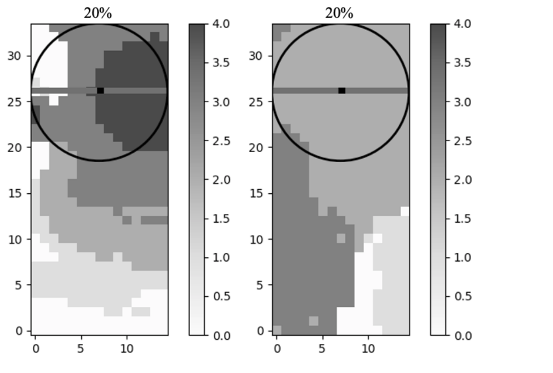


Figure 2: Chance that the radar analyzes the pixel under the random algorithm

Algorithm 1: Random Algorithm

```
output: Results: array of analyzed pixel values
input : Results: array of analyzed pixel values,
        Picture: knowledge window of the
        simulation
1 $i \leftarrow randomvalue(0 < i < 1)$
2 if $i \leq .2$ then
3     $results \leftarrow$value of pixel at nadir
       // add the value of pixel under
      nadir to results
4 end
5 return $Results$
```

On/Off

The on/off algorithm (figure 3 and algorithm 2) improves the random algorithm by controlling when the radar is turned on. It utilizes the system's current energy state and the cloud type under nadir to determine when the radar is turned on instead of using a random generator. This allows the system to save energy when there are no interesting clouds, and use the stored energy when there are. It also mimics random by taking the value under nadir when the SOC is high.

Lateral

The lateral algorithm (figure 4 and algorithm 3) improves on the on/off algorithm by allowing the radar to analyze pixels along the cross-path direction. This is symbolized in the knowledge window graphic by the band that crosses nadir. The two important factors in determining when the radar is turned on is the state of charge of the vehicle 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.

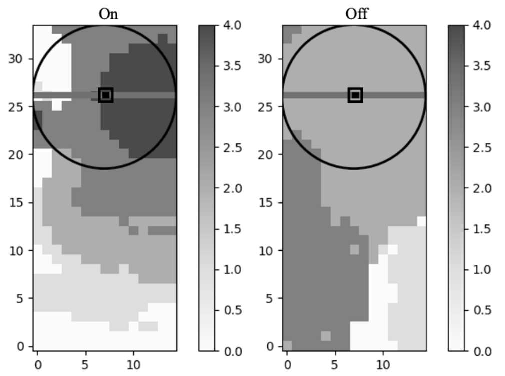


Figure 3: Left: radar is on because a convection core is under nadir. Right: radar is off because a cirrus cloud is under nadir. The radar turns on when it sees good pixels under nadir in the on/off algorithm

Algorithm 2: On/off Algorithm

```
output: Results: array of analyzed pixel values,
        power: power state of the system, sample:
        boolean to sample if there are no high
        priority targets
input : Results: array of analyzed pixel values,
        Picture: knowledge window of the
        simulation, Power: power state {0-100},
        Sample
1  if $power > 60$ then
2      $results \leftarrow$ value of pixel under nadir
3      $power \leftarrow power - 4$
4      $sample \leftarrow True$
5  else if $power > 40$ and sample then
6      $results \leftarrow$ value of pixel under nadir
7      $power \leftarrow power - 4$
8      $sample \leftarrow True$
9  else if $power > 4$ then
10     if Pixel under nadir == (CC or RA) then
11         $results \leftarrow$ value of pixel under nadir
12         $power \leftarrow power - 4$
13     else
14         $results \leftarrow$ value of radar turned off
15         $power \leftarrow power + 1$
16     end
17     $sample \leftarrow False$
18 else
19     $results \leftarrow$ value of radar turned off
20     $power \leftarrow power + 1$
21     $sample \leftarrow False$
22 return $Results$
```

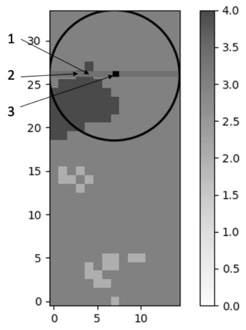


Figure 4: Top three prioritized pixels based on the lateral algorithm

Smart

The smart algorithm (figure 5 and algorithm 4) expands its view along the path of the satellite to include the entirety of the radar's reachability. This area is signified by the black circle in the graphic. When deciding which pixel to analyze for a given time step, the smart algorithm follows steps similar to the lateral algorithm. The state of charge determines which cloud types are able to be analyzed, and a search inside of the radar's reachability finds the highest valued cloud with a tiebreaker going to the pixel that is closest to nadir.

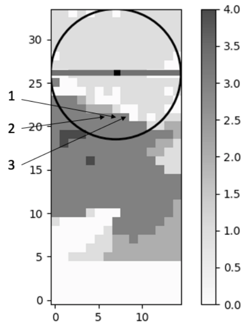


Figure 5: Top three prioritized pixels based on the smart algorithm

Windowed Smart

The windowed smart algorithm (figure 6 and algorithm 5) expands its view to include the entire knowledge window of the simulation. This increased view now exceeds the radar's reachability, meaning that the algorithm is able to account for future clouds along the radar's path. The algorithm first calculates how many clouds can be analyzed based on the current state of charge. It then counts the number of convection core and rainy anvil clouds present within the knowledge window. The power is then allocated for all of the convection core pixels, followed by the rainy anvil pixels, and then any leftover power is reserved as free. The highest valued pixel within the radar's view that has allocated power is

Algorithm 3: Lateral Algorithm

```
output: Results: array of analyzed pixel values,
        power: power state of the system {0-100},
        sample: boolean to sample if there are no
        high priority targets
input : Results, Picture: knowledge window of the
        simulation, Power, Sample
1  $clouds \leftarrow$ pixels that make up lateral band across
     nadir within radar's view
2  $best \leftarrow$ lat_search($clouds$)    // Returns the
     best pixel in the lateral field of
     view that is closest to nadir
3  if $power > 60$ then
4      if best == RA or CC then
5          $results \leftarrow$ value of best
6      else
7          $results \leftarrow$ value of pixel under nadir
8      end
9      $power \leftarrow power - 4$
10     $sample \leftarrow True$
11 else if $power > 40$ and sample then
12     if best == RA or CC then
13         $results \leftarrow$ value of best
14     else
15         $results \leftarrow$ value of pixel under nadir
16     end
17     $power \leftarrow power - 4$
18     $sample \leftarrow True$
19 else if $power > 4$ then
20     if best == (CC or RA) then
21         $results \leftarrow$ value of best
22         $power \leftarrow power - 4$
23     else
24         $results \leftarrow$ value of radar turned off
25         $power \leftarrow power + 1$
26     end
27     $sample \leftarrow False$
28 else
29     $results \leftarrow$ value of radar turned off
30     $power \leftarrow power + 1$
31     $sample \leftarrow False$
32 return Results, Power, Sample
```

imaged. The tiebreaker still goes to the pixel closest to nadir. The pixel under nadir is imaged if neither a convection core or rainy anvil pixel are within the radar's view, there is free power, and there is a sufficient SOC.

Algorithm 4: Smart Algorithm

```
output: Results: array of analyzed pixel values,
        power: power state of the system {0-100},
        sample: boolean to sample if there are no
        high priority targets
input : Results, Picture: knowledge window of the
        simulation, Power, Sample
1  $radar\_view \leftarrow$ pixels that make up radar's range of
     possible targets
2  $best \leftarrow$ smart_search(radar_view, view_radius)
     // Returns the best pixel in the
     radar's field of view that is
     closest to nadir
3  if $power > 60$ then
4      if best == RA or CC then
5          $results \leftarrow$ value of best
6      else
7          $results \leftarrow$ value of pixel under nadir
8      end
9      $power \leftarrow power - 4$
10     $sample \leftarrow True$
11 else if $power > 40$ and sample then
12     if best == RA or CC then
13         $results \leftarrow$ value of best
14     else
15         $results \leftarrow$ value of pixel under nadir
16     end
17     $power \leftarrow power - 4$
18     $sample \leftarrow True$
19 else if $power > 4$ then
20     if best == (CC or RA) then
21         $results \leftarrow$ value of best
22         $power \leftarrow power - 4$
23     else
24         $results \leftarrow$ value of radar turned off
25         $power \leftarrow power + 1$
26     end
27     $sample \leftarrow False$
28 else
29     $results \leftarrow$ value of radar turned off
30     $power \leftarrow power + 1$
31     $sample \leftarrow False$
32 return Results, Power, Sample
```

Greedy Path

Greedy path (figure 7 and algorithm 6) improves upon the windowed smart algorithm by ranking the priority of each convection core and rainy anvil pixel in the knowledge window. The algorithm begins by collecting the locations of all these pixels and calculating the available radar cycles based on the state of charge. Once collected, the two pixel types are sorted independently by their lateral distance to nadir. This means that a newly scanned convection core pixel that will eventually cross nadir will have a higher priority than an off-nadir convection core pixel within the radar's view. The sorted list of rainy anvil pixels is then concatenated to

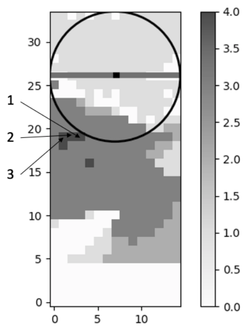


Figure 6: Top three prioritized pixels based on the windowed smart algorithm

the end of the sorted list of convection core pixels to create a priority queue. Greedy path then assigns one radar cycle to the highest priority pixel and checks if it is within the radar's view. If it is viewable, the pixel is analyzed. Otherwise, it continues until the free cycles run out or the priority queue ends. If free cycles are left over after the end of the priority queue and the SOC is sufficient, the algorithm will analyze nadir.

Greedy path has two variations, greedy path and greedy path wide. The difference is in how the pixels are prioritized. Clouds that run directly under nadir are within the radar's field of view for significantly longer than the clouds that run just under the edge. Greedy path wide takes this into account and always chooses the cloud that is laterally farther away from nadir if there is a tie in priority. The logic is that it will be able to analyze more high priority clouds by choosing the ones that are within its field of view for the shortest time. In contrast, the normal greedy path targets the cloud that is laterally closer to nadir in the case of a tie.

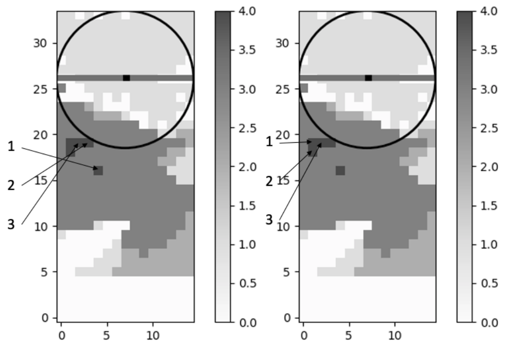


Figure 7: Left: Top three prioritized pixels based on the greedy path algorithm. Right: Top three prioritized pixels based on the greedy path wide algorithm

Algorithm 5: Windowed Smart Algorithm

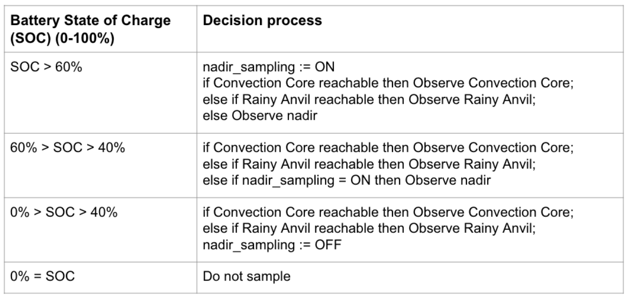
```
output: Results: array of analyzed pixel values,
        power: power state of the system {0-100},
        sample: boolean to sample if there are no
        high priority targets
input : Results, Picture: knowledge window of the
        simulation, Power, Sample
1  $free\_cycles \leftarrow$ power / 4
     // free_cycles is set to the total
     number of times the radar can be
     turned on at the current power
     state
2  $storms \leftarrow$ dictionary mapping the storm types to
     their occurrence in the knowledge window
3  $best\_cc, best\_ra \leftarrow$ smart_search(radar_view,
     view_radius)    // Returns the best CC
     pixel and the best RA pixel in the
     radar's field of view that are
     closest to nadir
4  $radar\_view \leftarrow$ pixels that make up radar's range of
     possible targets
5  if $power > 60$ then
6      sample = True
7  else if $power < 40$ then
8      sample = False
         // set the sampling variable
9  if CC in storms then
10     $cc \leftarrow$ number of CC pixels in Picture
11 if RA in storms then
12     $ra \leftarrow$ number of RA pixels in Picture
13 if $free\_cycles \leq cc$ then
14     $cc \leftarrow$ free_cycles
15     $free\_cycles \leftarrow 0$
16 else
17     $free\_cycles \leftarrow$ free_cycles - cc
18 end
     // set cc to the total number of
     CC pixels that can be pictured
19 if $free\_cycles \leq ra$ then
20     $ra \leftarrow$ free_cycles
21     $free\_cycles \leftarrow 0$
22 else
23     $free\_cycles \leftarrow$ free_cycles - ra
24 end
     // set ra to the total number of
     RA pixels that can be pictured
25 if $cc > 0$ and best_cc exists then
26     if $cc > 0$ then
27         $results \leftarrow$ value of best_cc exists
28         $power \leftarrow power - 4$
29     else if $ra > 0$ and best_ra exists then
30         $results \leftarrow$ value of best_ra
31         $power \leftarrow power - 4$
32     else if $free\_cycles > 0$ and sample then
33         $results \leftarrow$ value of pixel under nadir
34         $power \leftarrow power - 4$
35     else
36         $results \leftarrow$ value of radar turned off
37         $power \leftarrow power + 1$
38 return Results, Power, Sample
```

Algorithm 6: Greedy Path Algorithm

```
output: Results: array of analyzed pixel values,
        power: power state of the system {0-100},
        sample: boolean to sample if there are no
        high priority targets
input : Results, Picture: knowledge window of the
        simulation, Power:, Sample
1  $free\_cycles \leftarrow$ power / 4
     // free_cycles is set to the total
     number of times the radar can be
     turned on at the current power
     state
2  if $power > 60$ then
3      sample = True
4  else if $power < 40$ then
5      sample = False
         // set the sampling variable
6  $cc\_pixels \leftarrow$ empty array
7  $ra\_pixels \leftarrow$ empty array
8  for Each pixel in Picture do
9      if pixel is CC then
10         $cc\_pixels \leftarrow$ location of the pixel
11     if pixel is RA then
12         $ra\_pixels \leftarrow$ location of the pixel
13 end
14 $cc\_pixels \leftarrow$ sort(cc_pixels)
15 $ra\_pixels \leftarrow$ sort(ra_pixels)        // sort
     cc_pixels and ra_pixels in order
     of how laterally close they are to
     nadir
         // for greedy path wide the
     sorting would be on how laterally
     far they are from nadir
16 $interesting\_pixels \leftarrow$ cc_pixels + ra_pixels
17 $checks \leftarrow 0$
18 $undecided \leftarrow True$
19 while $checks < free\_cycles$ and undecided do
20     if checks <length of interesting_pixels then
21         $point \leftarrow interesting\_pixels[checks]$ if
            point is within the radar's view then
22             $undecided \leftarrow$ False
23             $results \leftarrow$ value of pixel at point
24             $power \leftarrow power - 4$
25     else
26         $undecided \leftarrow$ False
27         if sample then
28             $results \leftarrow$ value of pixel under nadir
29             $power \leftarrow power - 4$
30         else
31             $results \leftarrow$ value of radar turned off
32             $power \leftarrow power + 1$
33         end
34     end
35     $checks \leftarrow checks + 1$
36 end
37 if undecided then
38     $results \leftarrow$ value of radar turned off
39     $power \leftarrow power + 1$
40 return Results, Power, Sample
```

Energy Level Thresholds

After the random algorithm the pixel selection is dictated by the energy level instead of a random generator. At full power the system should always be analyzing a cloud, but as the power drains the list of valid clouds to analyze becomes more constrained.



Battery State of Charge (SOC) (0-100%)	Decision process

SOC > 60%	nadir_sampling := ON
	if Convection Core reachable then Observe Convection Core;
	else if Rainy Anvil reachable then Observe Rainy Anvil;
	else Observe nadir

60% > SOC > 40%	if Convection Core reachable then Observe Convection Core;
	else if Rainy Anvil reachable then Observe Rainy Anvil;
	else if nadir_sampling = ON then Observe nadir

0% > SOC > 40%	if Convection Core reachable then Observe Convection Core;
	else if Rainy Anvil reachable then Observe Rainy Anvil;
	nadir_sampling := OFF

0% = SOC	Do not sample

Table 1: Energy level thresholds for viewing pixels

Table 1 highlights the thresholds used in the previously stated algorithms. When the state of charge is greater than 60%, the algorithms will either be view a convection core or rainy anvil pixel, or sample nadir. The sampling variable is also set to "on" in this threshold. If the power is between 40% and 60% and the sampling variable is on, then the algorithms perform the same as when the power is above 60%. If the variable is off, then the algorithms will only view convection core and rainy anvil pixels. Once the power drops below 40% the algorithms also only view rainy anvil and convection core pixels. The sampling variable is set to "off" at this threshold. The purpose of the sampling variable is to randomly sample nadir between the 60% and 40% SOC until high priority clouds come into view.

One important feature basing cloud selection on the available energy is its flexibility. The 60% and 40% decision boundaries are not set in stone, and can be changed to better fit a scientist's interest. If the boundaries were to be shifted down, then the algorithm would start randomly sampling more, and the results would be a more even distribution across different cloud types. Inversely, if the boundaries were each shifted up, then less random sampling would occur. This would cause the results to skew heavier to the convection core and rainy anvil clouds. Additionally, this algorithm can be adapted to focus on any types of identified classes such as cirrus and thin cirrus.

Experimental Design

The dataset used to evaluate the algorithm was created through the Global Weather Research and Forecasting (GWRF) model (Skamarock et. al. 2019). The GWRF is a state of the art physics based weather model. It is used to create computationally expensive datasets that we use as a digital twin to real climate data. In our case the model generated the brightness temperatures for different cloud types along the bands of Tb250+0.0, Tb310+2.5, Tb380-0.8, Tb380-1.8, Tb380-3.3, Tb380-6.2, Tb380-9.5, and Tb670+0.0 as well as the scientific variables of ice water path, median particle size, and median cloud top height. These additional variables were used in the identification of the different cloud types throughout the dataset.

The dataset used in this study was generated as a tropical dataset of the Caribbean. The data contains 13 images that are 119x208 pixels with a pixel size of 15km for a spatial extent of 1,785km x 3,120km. Each image is a snapshot of the same area in one hour intervals. During the simulation the along track length is 1,785km and the across track length is 3,120km.

To prepare the images for the algorithms, each pixel is passed through a trained random decision forest that identifies the type of cloud represented by the pixel. It labels CC pixels with 89% accuracy, RA pixels with 75% accuracy, and clear, thin cirrus, and cirrus pixels with 92% accuracy when the expected radiometer noise is incorporated. Clear, thin cirrus, and cirrus is considered once class because the algorithms never actively target any of those cloud types.

The algorithms are then run over the classified image through non overlapping vertical paths from the top to the bottom of the image. The first path is at the far left of the image so that the left side of the radar's view matches with the left hand border of the scene. Once that run is complete the second run is shifted right by the radius of the radar's view so that no pixels overlap in between the runs. Each run progressively shifts right until it is impossible to fit another run without going over the boundary of the image or repeating pixels. The runs begin with nadir at the top of the image and end when nadir reaches the bottom.

We assume that the images are snapshots and do not change relative to the overflight time. Given the flight speed of 7.5km/s each run down the image only takes roughly four minutes. We are able to fit 13 runs per image, bringing the traversal time for an entire snapshot to roughly 52 minutes. The algorithm's are able to analyze one pixel or turn off the radar during each time step of the simulation. Each timestep is the traversal time of one pixel, and given the pixel size of 15km each timestep in the simulation represents roughly 2 seconds. It is important to note that this dataset contains a significantly higher number of storms than we would predict to see in a real flight. This is expected to skew the results of the algorithm towards greater performance since there are more storms to target.

Result

Table 2 shows that over the course of the runs it is clear that the dynamic targeting delivers a significant increase in performance. In the on/off scenario there was a slight increase in the number of convection core pixels analyzed, and the number of rainy anvil pixels evaluated effectively doubled. The number of these pixels chosen continues to rise as the view of the algorithm increases to the size of the radar's view. These increases can be explained by the improved possibility of seeing high priority pixels as well as a longer amount of time to analyze the pixels as the field of view expands along the radar's path.

A more interesting change is found when the algorithm's field of view expands from just the radar's view in the smart

	Off	Clear	Thin Cirrus	Cirrus	Rainy Anvil	Convection Core
Random (20% on)	16079 79.95%	1084 26.88%	439 10.89%	1099 27.26%	1367 33.90%	43 1.07%
On/Off	16097 80.04%	720 17.94%	142 3.54%	405 10.09%	2694 67.12%	53 1.32%
Lateral (1 DoF + on/off)	16094 80.03%	482 12.00%	111 2.76%	153 3.81%	3088 76.87%	183 4.56%
Smart (2 DoF + on/off)	16092 80.02%	352 8.76%	95 2.36%	86 2.14%	3107 77.31%	379 9.43%
Windowed Smart	16093 80.02%	313 7.79%	64 1.59%	43 1.07%	2850 70.93%	748 18.62%
Greedy Path	16093 80.02%	313 7.79%	64 1.59%	43 1.07%	2579 64.19%	1019 25.36%
Greedy Path Wide	16094 80.03%	411 10.23%	53 1.32%	80 1.99%	2415 60.12%	1058 26.34%

Table 2: The table contains the results of running the algorithms over all 13 images of the dataset. The runs were organized by shifting nadir to the East by the diameter of the radar view after every run. Each image contains 13 runs. 20,111 timesteps were taken over the entire dataset. During each time step an algorithm is capable of analyzing one pixel or turning off the radar. Percentages inside of the box are normalized over the time the radar was on during the runs. Greedy path found 23.7x more convection core than random, 1.9x more rainy anvil than random.

algorithm to the entire knowledge window in the windowed smart algorithm. When the two algorithms are compared the windowed smart algorithm views less rainy anvil pixels and more convection core pixels than the smart algorithm.

This is largely attributed to the distribution of clouds across the sky. Clouds are not randomly distributed. Instead, they cluster around storms. In particular, the convection core clouds serve as the center of most storms with rainy anvil clouds surrounding the center. The clustering means that the most important clouds for the algorithms are always found together. This problem is then exacerbated by the fact that the most scientifically interesting clouds, convection cores, are inside of large clusters of also highly prioritized rainy anvil clouds.

The distribution is a problem due to the energy constraints of the SMICES radar. Because both cloud types are scientifically significant, the targeting algorithms use any remaining power to analyze either of these clouds when they fall within the radar's field of view. This can lead to all of the available power being used to target the surrounding rainy anvil clouds and leave no power once the convection core clouds come into view. As the knowledge window increases, the algorithms are able to budget their remaining power by allocating resources to the future convection core clouds that will eventually enter the radar's view. The windowed smart algorithm is able to budget its power to save for future convection core pixels while the smart algorithm is not. This budgeting of power explains the increase in convection core performance at the expense of rainy anvil.

This feature of budgeting power is further developed in the greedy path algorithms, which selectively choose only the best rainy anvil and convection core pixels out of its knowledge window. The development is reflected in the results which show that the greedy path algorithm analyzes a decreased number of rainy anvil pixels and an increased number of convection core pixels when compared to the windowed smart algorithm.

These results also demonstrate that the wide approach continues this trend and analyzes slightly more convection core pixels at the cost of some rainy anvil pixels. Unfortunately, looking at clouds that are farther off of nadir returns worse data due to the increased distance to the imaging target. More analysis on how off nadir the analyzed clouds are and how severely the image quality degrades per angle off nadir is necessary to deduce whether this algorithm is actually superior. For this paper comparisons will be made to the greedy path algorithm since it prioritizes the clouds closer to nadir.

Future Work

It is important to run the simulation over a dataset that is representative of what SMICES would see in a real flight on future iterations of this work. This would give a better understanding of the improvements that we would expect to see through the targeting algorithms. Also, analyzing how the misclassifications of the classifier affect the scientific gain would be important to more accurately assess the performance of the algorithms. Further research also needs to be conducted on the impact that off-nadir analysis has on the data quality for SMICES. Understanding this impact along with how off-nadir the data collected from each algorithm is will allow for a more informed decision between the wide and non-wide variation of the greedy algorithm.

Conclusion

The results of the trial are very promising for the effectiveness of smart targeting. When comparing the best performing algorithm, greedy path, with the baseline random algorithm, there is a 23.7x increase in the number of convection core pixels analyzed and an almost 2x increase in rainy anvil pixels analyzed. It is important to note that these results are skewed by the dataset used, which contains a much higher proportion of clouds than we would expect to see in the real world. A dataset closer to a real-world scenario would result in a smaller increase in performance.

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.

References

Bony, S.; Colman, R.; Kattsov, V.; Allan, R.; Bretherton, C.; Dufresne, J.; Hall, A.; Hallegatte, S.; Holland, M.; Ingram, W.; Randall, D.; Soden, B.; Tselioudis, G.; and Webb, M. 2006. How Well Do We Understand and Evaluate Climate Change Feedback Processes? *Journal of Climate* 19(15): 3445 – 3482. doi:10.1175/JCLI3819.1.

Hasnain, Z.; Mason, J.; Swope, J.; Vander Hook, J.; and Chien, S. 2021. Agile Spacecraft Imaging Algorithm Comparison for Earth Science. *Under review, IWPSS 2021* .

King, M. D.; Platnick, S.; Menzel, W. P.; Ackerman, S. A.; and Hubanks, P. A. 2013. Spatial and Temporal Distribution of Clouds Observed by MODIS Onboard the Terra and Aqua Satellites. *IEEE Transactions on Geoscience and Remote Sensing* 51(7): 3826–3852. doi:10.1109/TGRS.2012.2227333.

Luo, Z.; and Rossow, W. B. 2004. Characterizing Tropical Cirrus Life Cycle, Evolution, and Interaction with Upper-Tropospheric Water Vapor Using Lagrangian Trajectory Analysis of Satellite Observations. *Journal of Climate* 17(23): 4541 – 4563. doi:10.1175/3222.1.

Stephens, G. L. 2005. Cloud Feedbacks in the Climate System: A Critical Review. *Journal of Climate* 18(2): 237 – 273. doi:10.1175/JCLI-3243.1.

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. doi:10.5194/amt-14-2013-2021. URL https://amt.copernicus.org/articles/14/2013/2021/.

Thompson, D.; Green, R.; Keymeulen, D.; Lundeen, S.; Mouradi, Y.; Nunes, D.; Castaño, R.; and Chien, S. 2014. Rapid Spectral Cloud Screening Onboard Aircraft and Spacecraft. *Geoscience and Remote Sensing, IEEE Transactions on* 52: 6779–6792. doi:10.1109/TGRS.2014.2302587.