SSC21-S1-01

MATA-Cloud: A Cloud Detection and Dynamic Attitude Correction Evaluation Software

Vanessa Tan, Julie Ann Banatao, John Leur Labrador, Lia Cristina Mabaquiao, Floyd Ferrant Fortes, Marc Caesar Talampas
Space Technology and Applications Mastery, Innovation and Advancement (STAMINA4Space) Program
Velasquez St., Diliman, Quezon City, Philippines; +63 8981-8500 loc 3305
vanessa.tan@eee.upd.edu.ph

ABSTRACT

With the increasing demand for high-resolution images from earth observation satellites, there is a need to optimize the usability of the images being downloaded in the ground stations. Most captured satellite images are not usable for certain applications due to high cloud cover percentage. To address this problem, this research demonstrates a cloud detection and dynamic attitude correction evaluation software. This software explores two key experiments. First is evaluating different image processing and machine learning-based approaches to detect cloud cover. The cloud detection algorithms were evaluated based on their accuracy, latency, and memory consumption. The second is exploring dynamic attitude correction to minimize the effect of cloud cover on captured images. Results show that our software can help test algorithms that increase the usability of captured images.

INTRODUCTION

The primary mission of most earth-observation satellites is to capture high-resolution images for mapping land cover, detecting vegetation, and mitigating disasters, among others. Satellite operators plan to capture such images in advance by considering the target area, time of capture, attitude maneuver, and weather forecast to create scheduled commands for these satellites. The weather is the most unpredictable element, especially since these commands are prepared days in advance. This unpredictability regularly leads to unusable images wherein clouds cover the target. The limited contact and revisit time for downlink adds to the problem as well, since it limits the operators' ability to recapture a target area in the event of a cloud-covered image and effectively wastes the opportunity for capture.[1] An emerging way to solve this problem is through on-board cloud detection.[2–4] Another possible solution is to have satellite image processing performed by the ground stations.[5–10]

This research leverages two key ideas: on-board image processing and attitude computation. Our system detects whether a captured image has an acceptable level of cloud cover. Otherwise, it will dynamically point the satellite to less cloudy areas near the original target. We explore image processing methods from traditional image processing to deep learning techniques for cloud detection. The cloud detection algorithms are evaluated on three different datasets to demonstrate that they can generalize on different optical payloads. The first dataset is a publicly available collection of Landsat 8 Level-1 images.[8] On the other hand, the second dataset is a collection of Middle Field Camera (MFC) images from the Philippine Diwata satellites.[11] Lastly, we train our algorithm on synthetic images from our Mission, Attitude, and Telemetry Analysis (MATA) simulation software.[12] The MATA simulator is developed from the telemetry data of the Diwata satellites and can simulate optical payload views over an earth model with controllable cloud cover.

After determining the cloud detection algorithm, we explore an appropriate control method through the MATA simulator. The output of our research will enable future satellite operations to have more usable data. Furthermore, it will enable the exploration of using machine learning for on-board computation. In summary, the main contributions of this paper are the following:

- exploration of highly accurate and low latency cloud detection algorithms
- demonstration of generating synthetic satellite images for cloud detection using the MATA simulation software
- implementation of a dynamic attitude target pointing using the MATA simulator

CLOUD DETECTION ALGORITHMS

Four cloud detection algorithms are explored for this research. These algorithms are to be implemented in the satellite's hardware, hence, they should be highly accurate, adaptable, and have low latency. The input of these algorithms are the RGB composites since these bands are available to most satellites.[13,14] Details of these algorithms are discussed in the next sections.

Otsu

The first cloud detection algorithm is Otsu, a classical thresholding method. The researchers opted to use this method since it is widely used in remote sensing images and has low complexity.[5,10,15,16] Otsu was also used in Diwata, the base satellite for this research.[13,14]

Similar to Otsu implementations for Diwata images,[13,14] rather than just using the RGB images as input, an HSI (Hue, Saturation, Intensity) color model was utilized. After converting the RGB image to HSI, a significance map is then constructed to highlight the differences between the cloud regions and non-cloud regions.[14,15] The significance map is computed as follows:

$$W = \frac{I + \epsilon}{H + \epsilon} \tag{1}$$

where I and H refer to the intensity and hue of the image, and ϵ is the amplification factor.[14,15] The amplification factor for this method is set to 1. After extracting the significance map, it is then fed to the Otsu segmentation to create the output cloud masks. The Otsu method assumes that the image has two pixel classes or a bi-modal histogram.[15,17] It then computes for the optimal threshold by maximizing their inter-class variance.[15] Fig. 1 shows the process for the first cloud detection algorithm.

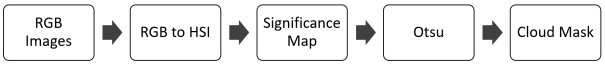

Figure 1: Otsu Method

K-means + Otsu

The second method is a variation of the first method as shown in fig. 2. Rather than directly applying Otsu to the extracted significance map, it utilizes the K-means clustering algorithm. K-means is an unsupervised learning algorithm which partitions input data into k clusters.[13] In order to partition the data, this algorithm minimizes the squared error distance of the input data and the cluster's centroid.[13,18] Since Otsu assumes bi-modal histogram, it has a tendency to misclassify snow or smog as clouds. The clustering algorithm could help improve the algorithm to classify non-clouds and eliminate interference.[10,19] The elbow method is used to get the optimal K for clustering. The output image clusters are then binarized using the threshold from the Otsu algorithm.

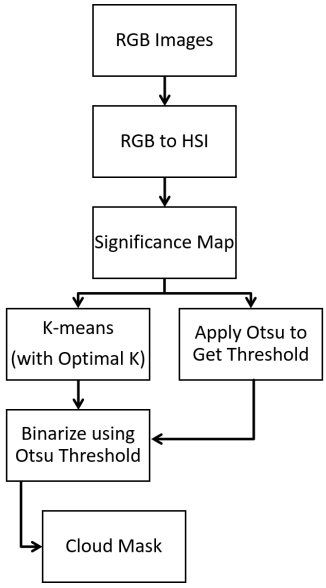

Figure 2: K-means + Otsu Method

Deep Learning

With the rapid improvements in utilizing deep learning architectures in embedded systems, the researchers explored lightweight architectures for image segmentation. Most image segmentation architectures have two major components: the encoder and the decoder.[4,11,20–22] The encoder transforms the input into small representation or features using stacked convolutional layers while the decoder recovers the original input size using transposed convolutional layers.[22]

Since Diwata satellites have a small number of annotated images for cloud segmentation, the researchers used the concept of transfer learning to improve the algorithm's performance. Transfer learning is a method where a model is trained for an initial task with a high number of data points. The model architecture is then reused for another application with a similar task.[1,13] For this research, the baseline encoder is the MobileNet architecture since it has proven to be lightweight and can be implemented in embedded systems.[23]

The MobileNet is initially trained in the ImageNet dataset which contains 14,197,122 different images.[24] Since most of the images in the ImageNet dataset are RGB images, the input satellite images were not converted to the HSI color model. The encoder block of the MobileNet is reused to serve as the feature extractor of the deep learning architecture. The decoders implemented in this research are the most popular image segmentation architectures: the UNet and the SegNet.[20,21] A visualization of the deep learning architectures for cloud segmentation is shown below.

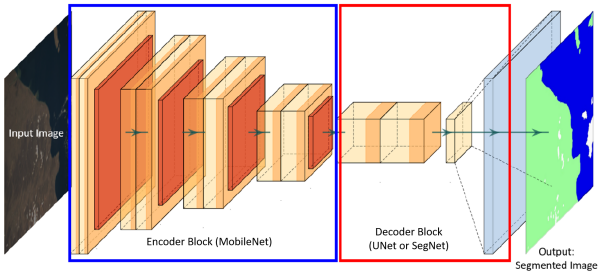

Figure 3: Deep Learning Architecture for Cloud Segmentation[11,25]

DYNAMIC TARGET POINTING USING THE MATA SIMULATOR

The Mission, Attitude, and Telemetry Analysis (MATA) software is a simulation tool which generates or reads satellite telemetry.[12] The kinematics and dynamics of the software are based on the Philippine Diwata satellites. This tool is also used by satellite engineers in a Hardware-In-the-Loop (HIL) mode to test the components of an engineering model. The interface of the MATA simulator is shown in Fig. 4.

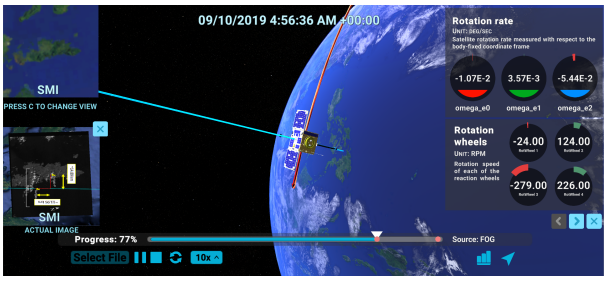

Figure 4: MATA Simulator Interface[12]

MATA has two main features that can be leveraged in machine learning applications. The first feature is its capability to simulate different optical payload views in realtime. Currently, MATA can simulate three optical payload views of Diwata satellites: the Middle Field Camera (MFC), Wide Field Camera (WFC) and the Spaceborne Multispectral Imager (SMI). Its second feature is the earth model with a controllable cloud cover. With this feature, it has the capability to generate synthetic images with cloud annotations. This is a powerful feature for machine learning since manual annotations are tedious and are prone to error.

The machine learning algorithms are trained using the synthetic images generated by the MATA simulator. The best cloud detection algorithm is then integrated in the simulator via sockets. The cloud segmentation and dynamic target pointing is implemented in realtime.

The dynamic target pointing for this paper is a simple method to demonstrate the simulator's capability in testing attitude correction algorithms. The satellite starts image capture using the MFC at nadir or off-nadir position. The cloud detection algorithm then segments the captured image and divides it into quadrants. The satellite is then pointed to the least cloudy quadrant and captures the area using the SMI. It then captures a new MFC image and repeats the process. A visualization of the MATA simulator with cloud detection and dynamic target pointing is shown in Fig. 5.

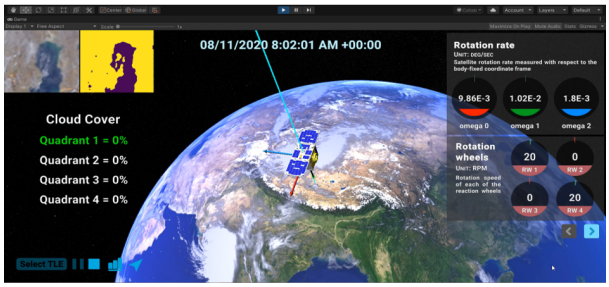

Figure 5: MATA - Cloud Interface

EVALUATION

This section discusses the different datasets utilized, the implementation details for the deep learning architectures, and the evaluation metrics for the cloud detection algorithms.

Datasets

The datasets were divided into training, validation, and test sets. The input RGB images were also resized to 224 x 224. The details of the datasets are shown below.

- 95-Cloud Dataset:[6–8] consists of 75 Landsat 8 Collection 1 Level-1 scenes which were cropped to non-overlapping patches; this dataset uses

the natural false-color of the satellite images (12,301 train; 9,201 valid; 9,201 test)

- MFC Dataset:[11,25] consists of Middle Field Camera (MFC) images from Diwata Satellites (70 train; 23 valid; 23 test)
- Synthetic Dataset: consists of simulated MFC images from the MATA simulator (306 train; 102 valid; 102 test)

Implementation Details

The batch size of the deep learning architecture was set to 16, running for 150 epochs with early stopping. The optimizer is ADAM with a learning rate of 0.001. Random geometric data augmentations such as flipping, scaling, and rotating of images were also utilized. The network was implemented in Keras with Tensorflow as backend using a NVIDIA RTX 2080 GPU.

Since the annotations for cloud and non-cloud regions are highly imbalanced, the binary focal loss is used as the network's loss functions. This loss function is an extension of the cross entropy and has been proven effective in imbalanced data sets for object detection[26] and source separation.[22] The focal loss is given by the equation below.

$$Loss = -(1 - p_t)^{\lambda} log(p_t) \tag{2}$$

p_t is defined as:

$$p_t = \begin{cases} p & \text{if } y = 1 \\ 1 - p & \text{otherwise} \end{cases} \tag{3}$$

where p is the probability for the label y and λ a positive tunable focusing parameter.[26]

Metrics

The evaluation metrics for the cloud detection algorithms are computed as follows:[6–8]

$$Jaccard = \frac{TP}{TP + FP + TN} \tag{4}$$

$$Precision = \frac{TP}{TP + FP} \tag{5}$$

$$Recall = \frac{TP}{TP + FN} \tag{6}$$

$$Specificity = \frac{TN}{TN + FP} \tag{7}$$

$$F1\ Score = \frac{2 * Precision * Recall}{Precision + Recall} \tag{8}$$

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{9}$$

where TP is the True Positive, TN True Negative, FP False Positive, and FN False Negative.

On the other hand, the dynamic target pointing algorithm has two evaluation metrics: the cloud percentage and the usability percentage, shown in equations 10 and 11. For the usability metric, the captured images from the simulator were manually selected based on the image usability criteria from Banatao et al.[13]

$$Cloud\ \% = \frac{no.\ of\ predicted\ cloud\ pixels}{total\ no.\ of\ pixels} \tag{10}$$

$$Usability\ \% = \frac{no.\ of\ manually\ selected\ usable\ images}{total\ no.\ of\ captured\ images} \tag{11}$$

RESULTS AND ANALYSIS

This section discusses the performance of the cloud detection algorithms. Other than the image segmentation metrics, their latency and memory consumption were also evaluated. Lastly, the evaluation of the attitude correction algorithm in the MATA simulator is presented.

Cloud Detection

As shown in the tables below, the MobileNet + UNet has the best overall metrics for the three datasets. It can also be seen that deep learning algorithms can generalize to different satellite images than the traditional image processing methods. As shown in the sample qualitative results in Fig. 6, the K-means + Otsu algorithm has a pixel-like segmentation while the MobileNet + SegNet has a blob-like segmentation. It can also be seen that the Otsu and MobileNet + UNet have more detailed segmentation results for the Diwata and MATA images. Moreover, the traditional image processing algorithms could not properly segment natural false-color satellite images like the 95-cloud dataset.

Table 1: Results for 95-Cloud Dataset

Metrics	Otsu	K-means + Otsu	**MobNet + UNet**	MobNet + Segnet
Jaccard	0.0223	0.0051	**0.7142**	0.7108
Precision	0.5303	0.6156	**0.8105**	0.8015
Recall	0.0275	0.0054	0.8347	**0.8370**
Specificity	0.9915	**0.9991**	0.9704	0.9678
F1 Score	0.0523	0.0107	**0.8224**	0.8189
Accuracy	0.6881	0.6880	**0.9495**	0.9490

Table 2: Results for Diwata MFC Images

Metrics	Otsu	K-means + Otsu	**MobNet + UNet**	MobNet + Segnet
Jaccard	0.6698	0.6561	**0.8066**	0.7521
Precision	**0.8887**	0.8728	0.8813	0.8123
Recall	0.7304	0.723	**0.8641**	0.8340
Specificity	0.7798	0.7363	**0.9114**	0.8391
F1 Score	0.7452	0.7333	**0.8682**	0.8186
Accuracy	0.7148	0.7051	**0.9102**	0.8953

Table 3: Results for Synthetic MATA Images

Metrics	Otsu	K-means + Otsu	**MobNet + UNet**	MobNet + Segnet
Jaccard	0.8799	0.8697	**0.8926**	0.8491
Precision	**0.9533**	0.9338	0.9072	0.8755
Recall	0.9263	**0.9320**	0.8969	0.8617
Specificity	0.8531	0.8457	**0.9934**	0.9744
F1 Score	**0.9255**	0.9187	0.9019	0.8685
Accuracy	0.9110	0.9152	**0.9908**	0.9784

Other than the image segmentation metrics, the latency and the memory consumption are evaluated for these algorithms to be integrated in the satellite's hardware. As shown in table 4, the fastest algorithm to segment an image is the Otsu ($\approx 4.3ms$) while the slowest algorithm is the MobileNet + UNet ($\approx 176.4ms$).

The Otsu and K-means algorithms do not consume memory, hence, only the deep learning algorithms were evaluated for the storage metric. The deep learning models were converted to Tensorflow Lite, a format which can be integrated in embedded systems like the Raspberry Pi and Google Coral TPU.[27] As shown in table 4, the MobileNet + SegNet has a smaller memory consumption than the MobileNet + UNet.

Table 4: Latency and Storage Evaluation

Method	Inference Time (s)	Memory (MB)
Otsu	**0.0043**	-
K-means + Otsu	0.1089	-
MobNet + UNet	0.1764	6.3
MobNet + Segnet	0.1377	**5.56**

As shown in the results, the cloud detection algorithms have different advantages and disadvantages. Researchers need to prioritize the metric which fits their system. The Otsu is the recommended algorithm if the cloud detection is to be implemented on-board the satellite with dynamic target pointing. It has the lowest memory consumption and inference time. For this paper, the MobileNet + UNet algorithm was utilized to provide input for the dynamic target pointing since the main focus is to develop a cloud attitude correction software. An inaccurate cloud detection algorithm would result to improper testing of the satellite's dynamic control. A future work for this research is the improvement of deep learning models to be faster and more lightweight.

Dynamic Target Pointing

Different quadrant angles were explored for the attitude correction algorithm. The quadrant angle is the angle from the center of the image to the center of the target quadrant as shown in Fig. 7. Table 5 shows the attitude correction results when the satellite starts at nadir position. As shown in these results, the quadrant control with 5° improved the usability of the captured MFC and SMI images. On the other hand, when the satellite starts in an off-nadir position, the quadrant control with 1° slightly improved the performance of the image capture. Qualitative results for the dynamic target pointing algorithm is shown in Fig. 8.

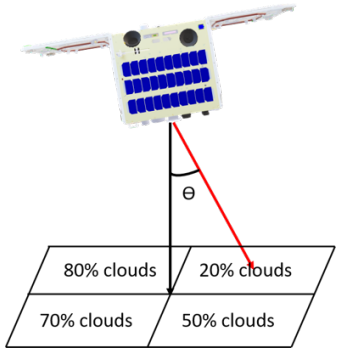

Figure 7: Dynamic Target Pointing Visualization (black line for origin, red line for target)

Table 5: Attitude Evaluation (Nadir)

Method	**Cloud % (MFC)**	**Usability % (MFC)**	**Cloud % (SMI)**	**Usability % (SMI)**
Nadir	19.1892	46.4865	23.7649	47.0270
Quad (0.5°)	17.7027	46.4865	20.3000	44.0540
Quad (1°)	18.0946	44.3243	21.0324	44.3243
Quad (3°)	18.5081	46.2162	21.2973	47.2973
Quad (5°)	**17.4459**	**55.9460**	**18.2135**	**52.9730**

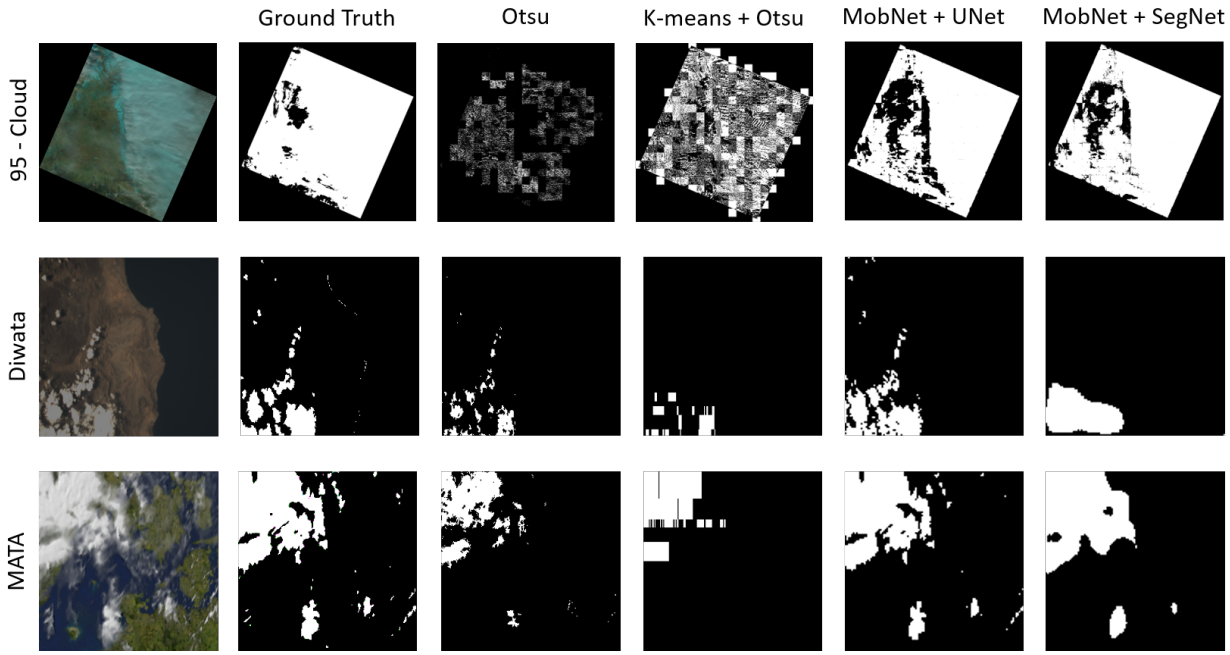

Figure 6: Sample Qualitative Results for the Cloud Detection Algorithms

Table 6: Attitude Evaluation (Off-Nadir)

Method	Cloud % (MFC)	Usability % (MFC)	Cloud % (SMI)	Usability % (SMI)
Off-Nadir	10.7822	69.1111	9.7822	70.6667
Quad (0.5°)	11.3311	**70.6667**	10.4556	70.6667
Quad (1°)	**9.5756**	**70.6667**	**9.1911**	**75.5556**
Quad (3°)	10.3956	69.3333	10.1711	67.5556
Quad (5°)	12.0644	64.0000	11.2267	67.1111

CONCLUSION AND FUTURE WORK

This paper presents MATA-Cloud, a cloud cover attitude correction evaluation software. Four cloud detection algorithms were explored for this research. Results show that Otsu, a traditional thresholding algorithm for image segmentation is the fastest algorithm. Meanwhile, deep learning models are the most accurate and adaptable algorithms to different datasets. Deep learning architectures still need to further develop for them to be integrated on-board a satellite with dynamic target pointing capability.

The MATA-Cloud also demonstrated how it can be a test bed for attitude correction algorithms. Other dynamic target pointing algorithms can be integrated in the simulator for future work. With the simulator's capability to capture images, attitude determination modules which depend on images such as star trackers could also be explored.

Acknowledgments

This research is funded by the Department of Science and Technology (DOST) under the Philippine Council for Industry, Energy, Emerging Technology, Research and Development (PCIEERD). The authors would like to thank the PHL-50 team for their support and contributions.

References

[1] Banatao J.A., Sakamoto Y., and Yoshida K. Realtime dynamic target pointing using on-board image processing of cloud cover for earth observation microsatellites. In *12th IAA Symposium on Small Satellites for Earth Observation*, May 2019.

[2] Giuffrida G., Diana L., Gioia F., Benelli G., Meoni G., Donati M., and Fanucci L. Cloudscout: A deep neural network for on-board cloud detection on hyperspectral images. *Remote Sensing*, 12:2205, July 2020.

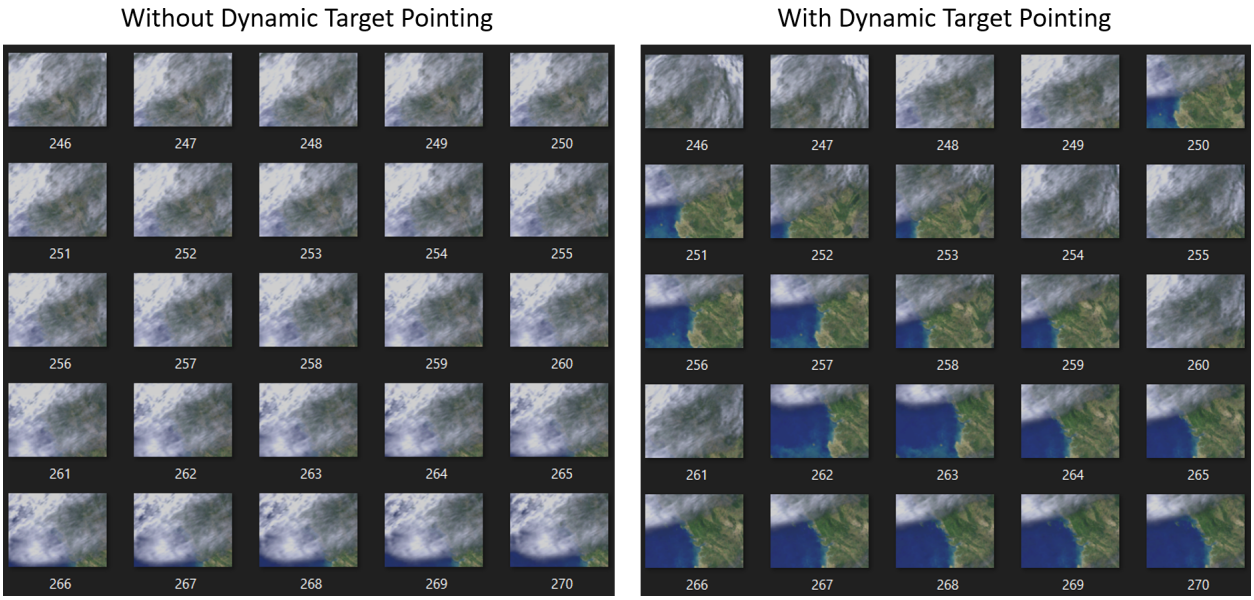

Figure 8: Sample Qualitative Results for the Dynamic Target Pointing

[3] Algra T. On the effectiveness of cloud cover avoidance methods in support of the super-spectral mission for land applications. In *IEEE International Geoscience and Remote Sensing Symposium*, volume 2, pages 982–985 vol.2, 2002.

[4] Bahl G., Daniel L., Moretti M., and Lafarge F. Low-power neural networks for semantic segmentation of satellite images. In *2019 IEEE/CVF International Conference on Computer Vision Workshop (ICCVW)*, pages 2469–2476, 2019.

[5] Mahajan S. and Fataniya B. Cloud detection methodologies: variants and development - a review. *Complex and Intelligent Systems*, pages 1–11, December 2019.

[6] Mohajerani S. and Saeedi P. Cloud-net: An end-to-end cloud detection algorithm for landsat 8 imagery. In *IGARSS 2019 - 2019 IEEE International Geoscience and Remote Sensing Symposium*, pages 1029–1032, July 2019.

[7] Mohajerani S., Krammer T. A., and Saeedi P. "A Cloud Detection Algorithm for Remote Sensing Images Using Fully Convolutional Neural Networks". In *2018 IEEE 20th International Workshop on Multimedia Signal Processing (MMSP)*, pages 1–5, Aug 2018.

[8] Mohajerani S. and Saeedi P. Cloud-Net+: A Cloud Segmentation CNN for Landsat 8 Remote Sensing Imagery Optimized with Filtered Jaccard Loss Function. volume 2001.08768, 2020.

[9] Luotamo M., Metsämäki S., and Klami A. Multiscale cloud detection in remote sensing images using a dual convolutional neural network. *IEEE Transactions on Geoscience and Remote Sensing*, pages 1–12, 2020.

[10] Xiang P. A cloud detection algorithm for modis images combining kmeans clustering and otsu method. *IOP Conference Series: Materials Science and Engineering*, 392:062199, Aug 2018.

[11] Banatao J.A., Sakamoto Y., and Yoshida K. Convolutional neural network for automated image processing of earth observation microsatellite images. In *71st International Astronautical Congress (IAC)*, Oct 2020.

[12] Tan V., Labrador J.L., and Talampas M.C. Mata: Mission, attitude, and telemetry analysis software for micro-satellites. In *2020 IEEE REGION 10 CONFERENCE (TENCON)*, pages 614–619, 2020.

[13] Banatao J.A., Sakamoto Y., and Yoshida K. Automated image processing module for images captured by earth observation microsatellite diwata-1 as support for ground station operations. In *32nd International Symposium on Space Technology and Science*, June 2019.

[14] Ramos M.K., Jiao B.J., Aranas R.K., Magallon B.J., Amado J., Tupas M.E., and Tamondong A. Automated calculation of cloud cover from rgb composite of landsat 8 and diwata-1 satellite imagery. In *ACRS Proceedings*, 2016.

[15] Zhang Q. and Xiao C. Cloud detection of rgb color aerial photographs by progressive refinement scheme. *IEEE Transactions on Geoscience and Remote Sensing*, 52(11):7264–7275, 2014.

[16] Sehgal S., Kumar S., and Bindu M. H. Remotely sensed image thresholding using otsu differential evolution approach. In *2017 7th International Conference on Cloud Computing, Data Science Engineering - Confluence*, pages 138–142, 2017.

[17] Otsu N. A threshold selection method from gray-level histograms. *IEEE Transactions on Systems, Man, and Cybernetics*, 9(1):62–66, 1979.

[18] Parsian M. *Data Algorithms: Recipes for Scaling Up with Hadoop and Spark*. O'Reilly Media, Inc., 1st edition, 2015.

[19] Song W.G., Liu S.X., Zhang Y.M., Zheng H.Y., and Tian W. A cloud detection algorithm for modis images combining kmeans clustering and multi-spectral threshold method. *Guang pu xue yu guang pu fen xi = Guang pu*, 31:1061–4, Apr 2011.

[20] Ronneberger O., Fischer P., and Brox T. U-net: Convolutional networks for biomedical image segmentation. *CoRR*, abs/1505.04597, 2015.

[21] Badrinarayanan V., Kendall A., and Cipolla R. Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *CoRR*, abs/1511.00561, 2015.

[22] Tan V. and de Leon F. Multi-task learning for detection, recovery, and separation of polyphonic music. In *2020 IEEE REGION 10 CONFERENCE (TENCON)*, pages 1112–1117, 2020.

[23] Howard A., Zhu M., Chen B., Kalenichenko D., Wang W., Weyand T., Andreetto M., and Adam H. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *CoRR*, abs/1704.04861, 2017.

[24] Deng J., Dong W., Socher R., Li L.J., Kai L., and Li F.F. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, 2009.

[25] Banatao J.A. Autonomous image processing and target pointing for an earth observation microsatellite. submitted, July 2020.

[26] Lin T.Y., Goyal P., Girshick R., He K., and Dollár P. Focal loss for dense object detection. *CoRR*, abs/1708.02002, 2017.

[27] David R., Duke J., Jain A., Reddi V.J., Jeffries N., Li J., Kreeger N., Nappier I., Natraj M., Regev S., Rhodes R., Wang T., and Warden P. Tensorflow lite micro: Embedded machine learning on tinyml systems. *CoRR*, abs/2010.08678, 2020.