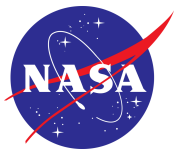
Cloud-Top Height Field Estimation from Aerial Imagery

Ian Kelk, Yash Nahar, Chandrakant Shendarkar, Christina Wang, Guilherme Zuanazzi



A Thesis in the Field of Data Science for

the Degree of Master of Liberal Arts in Extension Studies

Harvard University

December 2024

Disclaimer

This report has been refined and enhanced with the assistance of generative AI tools. These tools were utilized to provide suggestions on structuring, clarity, and coherence in the presentation of the research findings. While they contributed to refining the language and organization of the content, the underlying research, data analysis, and conclusions remain the original work of the author(s). The use of generative AI was limited to editorial assistance and did not influence the report's substantive content, research methodologies, or outcome interpretations.

GitHub Repository

The complete public code repository for this project can be found at:

https://github.com/cloud-2-cloud/c2c

Abstract

This project aimed to address a critical issue in extreme weather forecasting and guarantee accurate measurements of the earth's atmosphere. We developed predictive models for estimating cloud top heights using advanced deep learning, computer vision, and optical flow geometry techniques. Our objectives were to accurately predict cloud top heights and create a 3D mesh projection of the cloud height field. We utilized high-definition images from NASA's FEGS ER2 mission, complemented by LiDAR and aircraft metadata.

The ConvNext pre-trained model demonstrated robustness in feature extraction from sparse cloud images after appropriate augmentation during preprocessing. The combined CNN-RNN model effectively learned and predicted cloud top heights at the center. The RAFT model, employing optical flow, generated visually consistent height fields.

The optical flow geometry approach turned out to be a new and robust method for cloud height estimation. TV-L1 for optical flow tracking seemed to be the most robust in estimating pixel velocity for cloud top height predictions across entire frames. The optical flow geometry methods particularly did well at capturing variability in the heights at all parts of the image and not just the center.

These results validate the potential of advanced computer vision and machine learning techniques coupled with optical flow geometry as a promising framework for precise cloud top height prediction.

Acknowledgments

We extend our sincere gratitude to our esteemed professors, Dr. Bruce Huang and Dr. Stephen Elston, from Harvard University, for their invaluable guidance, insightful references, and meticulous reviews during this project. Their profound questions and key insights proved instrumental in addressing the complexities of the project and broadening our perspectives. The collaborative efforts with NASA researchers, Dr. Mason Quick and Dr. Timothy Lang, were pivotal to the project's success. Their subject matter expertise, consistent involvement, and comprehensive review of our work significantly contributed to our achievements.

Furthermore, we would like to acknowledge the fundamental support of all faculty members, colleagues, friends, and family who made our ALM journey at Harvard Extension School a reality.

Table of Contents

Chapter I.

Introduction

Measuring cloud-top heights and their evolution in time is extremely important for NASA. Accurate cloud height data enhances the calibration and validation of radiometric instruments on satellites, ensuring precise measurements of Earth's atmosphere. This in turn helps in understanding the interaction between clouds and radiation, which is needed for studying Earth's energy balance and climate. By accurately measuring cloud heights, NASA can refine radiometric algorithms, improve the quality of satellite data, and enhance the overall accuracy of atmospheric observations, resulting in better climate models and weather predictions.

NASA aims to reduce the cost of future missions but also solve limitations with cloud height measurements from LiDAR installations with limited fields of view. This project aims to develop a predictive model to estimate cloud height utilizing high-definition images captured from the Fly's Eye GLM Simulator (FEGS) along with Light Detection and Ranging (LiDAR) data.

1.1. Problem & Setting

The Geostationary Lightning Mapper (GLM) is a lightning detection sensor onboard the GOES-16 satellite. To validate the GLM, NASA developed the FEGS. The FEGS is a multispectral radiometer system that features 30 radiometers and an HD camera. It is mounted aboard the NASA ER-2 aerial laboratory, a plane that operates at altitudes ranging from 20,000

feet to 70,000 feet, above 99% of the Earth's atmosphere. During a flight campaign in 2017 to validate the recently launched GOES-16, the ER-2 flew a suite of instruments that included FEGS and the Cloud Physics LiDAR (CPL) for measuring the height of clouds. NASA's current objective is to reduce the cost of future aerial meteorological missions; cameras have lower costs, are more prevalent, and have much larger 90-degree fields of view.

While LiDAR is effective at accurately measuring the height of the cloud tops, it only provides a single point value corresponding to where the lasers are pointed. This can provide a ground truth for the center of the cloud-top heights, allowing us to train and validate a model to estimate cloud-top height using the HD images from FEGS.

The problem can be defined as: *How can we use high-definition images from the FEGS combined with LiDAR data to accurately estimate cloud-top heights and create a height field cloud projection?*

1.2. Subproblems

1. Adjust for lens distortion and viewing geometry: The FEGS uses an HD camera with a fisheye lens. It is necessary to undistort the images in order to properly apply geometry calculations and modeling.
2. Predictive Model for Cloud Height Estimation: The challenge was to use high-definition images captured from FEGS along with metadata about the aircraft and LiDAR data for the development of the predictive model for cloud-top height estimation.

3. Creating the height field: Once the cloud-top height is predicted, the challenge becomes to expand the prediction from the center to a height field, thus creating the height field.
4. Stitching consecutive height fields: After creating the height field for each image, the height fields can be stitched to each other, forming a complete map from the flight.

1.3. Assumptions, Delimitations, & Limitations

This project assumes that:

- Data Quality: High-definition images and LiDAR data captured by the NASA ER-2 aerial laboratory are of high quality and sufficient for training and validating the machine learning models.
- Timestamp Alignment: Timestamps for HD images and LiDAR measurements can be accurately synchronized to ensure precise alignment of the datasets.
- Data Source Consistency: Cloud structures captured during different flights exhibit enough consistency to allow for effective model training and validation.
- Field of View Requirements: The 90-degree field of view for the camera is sufficient to capture a comprehensive representation of the cloud structures needed for model training and validation.
- Flight Metadata Accuracy: Accurate metadata with respect to the ER-2 aircraft carrying the FEGS is made available, such as altitude, pitch, yaw, roll, heading, speeds (indicated, true, ground), bank, and aircraft vertical velocity.

The project focuses on data collected from the NASA ER-2 aerial laboratory, specifically the HD images and LiDAR data obtained during specific flight missions. The study only includes the geographical areas that the ER-2 flights covered, which might not encompass all cloud types and atmospheric conditions on a global scale. Furthermore, the project utilizes data collected over a defined period, limiting the analysis to specific seasonal and weather conditions.

Integrating HD images and LiDAR data presents technical challenges, including aligning data spatially and temporally as well as handling discrepancies in data quality and resolution. Noise is also a relevant factor, which can arise from motion blur, lighting issues, or inherent noise from LiDAR sensors, potentially affecting the data quality and model accuracy. The ability of the machine learning models to generalize to unseen data and different atmospheric conditions may be limited, impacting the robustness of point cloud generation. It is important to guarantee that computational resources necessary for processing sizable datasets and training computationally intensive machine learning models are available. Lastly, variations in environmental conditions, such as cloud density, rain, lightning, and weather patterns, may affect the parallax calculations and, consequently, the accuracy and consistency of the generated point clouds.

1.4. The Importance of the Study

Understanding cloud-top heights and their temporal evolution is vital to comprehending cloud-top processes and their interactions with local meteorology and climate. HD cameras, being cost-effective and simple to deploy, can be readily installed on CubeSats, weather satellites, and space stations, and they are already ubiquitous in many current systems. If HD image data could be used to estimate cloud-top height, this widespread deployment of cameras

will provide global monitoring capabilities for cloud tops and significantly enhance the accuracy of weather predictions by offering more detailed and frequent observations. It would also help to refine the radiometric algorithms that satellites, such as the GOES-16, use to gather data. Ultimately, this research will support better climate models, improved weather predictions—especially extreme weather forecasts—and more cost-effective future aerial meteorological missions.

Chapter II.

Literature Review

To properly consider this project within the broader field of atmospheric remote sensing and inform a methodological approach, we examine a range of relevant research papers. The selected papers cover important topics related to measuring cloud height, such as satellite imagery, stereoscopic and multi-angle imaging methods, LiDAR applications, and 3D cloud structure reconstruction methods. We both explore current capabilities and limitations in cloud height estimation, find new methodologies for data integration and validation, and identify key challenges that this project must address.

The review follows a thematic structure, concentrating on three areas:

1. Cloud Height Measurement and Estimation: Various techniques for estimating cloud heights using instruments such as LiDAR, all-sky cameras, and airborne sensors.
2. 3D Cloud Structure Reconstruction: Methods for reconstructing three-dimensional cloud structures from two-dimensional imagery.
3. Calibration and Validation: Ensuring the accuracy of cloud measurements.

2.1 Cloud Height Measurement and Estimation

2.1.1. Motivation and Challenges

Various applications in meteorology, aviation, and climate science require accurate cloud base and top height measurements. Mussa et al. (2022) and Bedka et al. (2007) both discuss the significance of precise cloud height data for improving the accuracy of atmospheric observations and models. Mussa et al. investigated cloud height and coverage because they affect the analysis

of cosmic rays at the Pierre Auger Observatory. Bedka et al. checked the accuracy of cloud-top pressure products made by satellites using LiDAR data from aircraft.

Borisov et al. (2023) and Nied et al. (2023) explore the use of artificial neural networks and convolutional neural networks, respectively, to estimate cloud heights from imagery. Borisov et al. (2023) introduce a novel approach using parallax effects observed with wide-angle cameras and neural networks to estimate cloud base heights, addressing the limitations of expensive and complex equipment in maritime conditions. Nied et al. (2023) focus on detecting high-level clouds above aircraft using CNNs, which is necessary to improve the accuracy of aerosol measurements in remote sensing. They also note that existing calculation methods are too slow to be used in near-real-time scenarios, such as where the aircraft must avoid flying through or below such clouds for aerosol measurement. Nied et al. (2023) also notes photo imagery as being superior to some other sensor methods in detecting the presence of clouds. Both papers demonstrate the effectiveness of machine learning techniques in overcoming the limitations of traditional methods and providing cost-effective and adaptable solutions for cloud height estimation.

McGill et al. (2002) developed advanced instruments for high-resolution cloud measurements. The paper discusses the Cloud Physics LiDAR (CPL), designed to overcome the limitations of previous LiDAR systems by using state-of-the-art technology. The paper emphasizes the challenges of high-altitude measurements and the need for high temporal and spatial resolution while maintaining sensitivity. The CPL's innovative design, featuring a high-repetition-rate laser and photon-counting detectors, addresses these challenges. demonstrating significant improvements in data accuracy and reliability.infrared methods, while Yu et al. (2019) point out the insufficient coverage of satellites like CloudSat and CALIPSO

provide. Hadjitheophanous et al. (2010) criticize software-based solutions for their high computational and power requirements, which are unsuitable for real-time applications.

2.2.2. Proposed Solutions

Hasler (1981), Yu et al. (2019), and Hadjitheophanous et al. (2010) offer novel approaches to overcoming the limitations of existing methods for 3D cloud structure reconstruction.

The solutions proposed in these studies involve using advanced observational techniques and hardware implementations. Hasler (1981) proposes the use of stereographic observations from geosynchronous satellites. This technique relies on straightforward geometric relationships to provide much higher horizontal resolution and accuracy compared to infrared methods. By capturing stereo images from geosynchronous satellites positioned at different longitudes, the approach leverages the parallax effect to measure cloud heights with greater precision. Yu et al. (2019) suggest using multi-angle, multi-spectral, and polarization data from the DPC onboard the GF-5 satellite. The DPC takes pictures of the same target from different angles. This lets cloud structures be reconstructed in 3D space using a ray casting algorithm. This algorithm figures out where cloud voxels are located by comparing where rays from different angles meet. Hadjitheophanous et al. (2010) suggest putting the 3D reconstruction algorithm on FPGA hardware, taking advantage of the algorithm's built-in parallelism. This approach includes a Sobel edge detection unit to reduce the amount of data processed, increase frame rates, and achieve real-time performance.

Each study successfully proves its claims through empirical verification and comparative analysis. Hasler (1981) demonstrates the accuracy of stereo height measurements by comparing

them with known altitudes of high-altitude mountain lakes, achieving accuracy within ±0.1–0.2 km near reference points and ±0.5 km for general cloud features. The application of these measurements to meteorological problems, such as severe thunderstorms and hurricanes, further substantiates the efficacy of stereographic observations. Yu et al. (2019) show that their 3D reconstruction method works by showing that it matches up with CALIOP data in terms of vertical profiles and cloud boundaries. They also give accurate measurements of the reconstructed clouds. Hadjitheophanous et al. (2010) achieve real-time performance with frame rates up to 75 FPS for 320x240 image pairs, demonstrating the system's efficiency across different parameter settings. The integration of the Sobel edge detector significantly enhances performance by reducing the amount of data processed.

2.2 Calibration and Validation

2.3.1 Motivation and Challenges

Bedka et al. (2007) focus on validating satellite-derived CTP products using aircraft-based CPL data. The problem here is ensuring the accuracy of CTH measurements, which are needed for meteorology, aviation safety, and climate studies. The difficulty lies in the discrepancies introduced by the differences in spatial and temporal resolution between satellite instruments and in situ measurements, along with the inherent complexity of cloud structures. Existing solutions often fall short due to these complexities.

Vaughan et al. (2010) address the calibration of the CALIOP 1064 nm LiDAR channel, which is necessary for reliable atmospheric measurements. The challenge here involves weaker signal levels, noise, and the natural variability in the backscatter color ratio of cirrus clouds.

Existing calibration methods based on potentially flawed assumptions can lead to significant errors.

2.3.2 Proposed Solutions

In 2007, Bedka et al. looked for and fixed problems in satellite data by comparing CTH products from the GOES-12 Imager and Sounder with CPL measurements taken during the ATReC field campaign. The study focuses on various cloud types and conditions and includes a brief comparison with MODIS-Aqua CTH retrievals. Their results indicate that the GOES-12 Imager generally agrees better with CPL measurements than the GOES-12 Sounder, particularly for mid-level and low clouds. The study clearly shows the pros and cons of current satellite measurements and backs up its claims with in-depth analysis and real-world examples. Using high-resolution LiDAR measurements provides a more accurate standard, showing that satellite-derived data could be more accurately validated.

Vaughan et al. (2010) calibrated the CALIOP 1064 nm LiDAR channel using cirrus clouds. They employed a measurement-based approach to determine the best estimate of the mean backscatter color ratio for cirrus clouds. They studied how cirrus cloud backscatter coefficients change with wavelength by examining more than 400 hours of LiDAR data from the CPL. They discovered backscatter color ratios for cirrus clouds as part of this study. To make sure the calibration process worked, they compared the CPL color ratios with the assumptions used in the CALIPSO calibration scheme. They accounted for aerosol loading in the normalization region using data from multiple satellites to apply a parameterized correction factor. Their findings showed that the best estimate for the backscatter color ratio of cirrus clouds is 1.01 ± 0.25, aligning with pre-launch assumptions and measurements. This result reassured the

validity of the CALIOP 1064 nm calibration algorithm and revealed the need for a large sample size to minimize calibration errors.

2.3 Relevance of Literature to this Project

The reviewed literature offers a thorough summary of prior research methodologies pertinent to this project. Mussa et al. (2022) and Bedka et al. (2007) discuss how important it is to get accurate measurements of cloud height and suggest ways to make them more accurate by using advanced instruments such as the flexible threshold-based algorithm for cloud detection and validation techniques using high-resolution LiDAR data. This is similar to our project's plan to combine HD images and LiDAR data to improve estimates of cloud height. Borisov et al. (2023) and Nied et al. (2023) show that ML techniques, especially artificial neural networks and CNNs in particular, are good at estimating cloud heights from images, and Nied et al. (2023) notes that HD imagery is superior to detecting cloud presence as opposed to some other sensors used. This supports our plan to use CNN and RNN models in this project and the focus on using camera imagery. The work of McGill et al. (2002) on high-resolution LiDAR systems provides a foundation for using LiDAR data as a benchmark for validating our predictive models. Hasler (1981), Yu et al. (2019), and Hadjitheophanous et al. (2010) all look into different ways to reconstruct 3D cloud structures while dealing with issues like data synchronization and computational efficiency. Finally, Bedka et al. (2007) and Vaughan et al. (2010) talk about calibration and validation, which stress how important it is that the cloud measurements we receive from NASA are reliable.

Chapter III.

Methodology

3.1 Infrastructure

We completed this project using Google Colab notebooks integrated with GitHub, combined with Azure Blob Storage for data storage. We have used Azure Blob Storage as the primary data storage tool for the entire machine learning pipeline, including model training and validation. The output from the various model runs has also been stored in Azure Blob in a dedicated output folder. Trained models have been archived in a trained_models folder in Google Drive. AWS S3 and Sagemaker were also used as a backup for storing data and training the ML models once academic cloud credits were made available. The main tools and instruments we used are listed below:

- Azure Blob Storage
- GitHub
- Python
- Google Colab Pro+ (~USD 50 per month per user)

The Google Colab instance with compute power requirements to support deep learning for this project is the following (only the traditional optical models had a lower requirement and could run on a T4 GPU with 16GB):

GPU: NVIDIA A100

Memory: 40 GB HBM2 @ 1.6 TBps

Interconnect: NVLink Full Mesh @ 600 GBps

3.2 Data

3.2.1. Data Sourcing and Collection

We used the following data sources obtained from NASA:

- FEGS Flight Video: Provided by NASA, uploaded to cloud storage in the specified Azure Blob Storage Account.
- ER2 Aircraft Metadata: https://asp-archive.arc.nasa.gov/N809NA/FY2017/ (plane speed, pitch, yaw, altitude, etc.)
- LiDAR Data: https://cmr.earthdata.nasa.gov/search/concepts/C1979112912-GHRC_DAAC.html
    - This data provides the cloud height at the center of each image frame from the HD camera.
- The data scope for all three sources from the NASA ER2 Missions spans the following dates and times:
    1. 18th April 2017, 17:57:06 - 23:08:11
    2. 22nd April 2017, 20:30:40 - 23:48:13
    3. 8th May 2017, 20:28:28 - 9th May 2017, 01:41:03
    4. 12th May 2017, 16:04:57 - 20:49:18
    5. 14th May 2017, 11:55:18 - 17:19:17

3.2.2. Exploratory Data Analysis

While analyzing the LiDAR data and after conversations with NASA scientists, it was discovered that large turns in the aircraft led to a consequent large delta in the LiDAR height.

This observation led to all data points in the LiDAR having a turn of over 10 degrees to the previous being dropped.

Illustration of Need to Drop Large Turn

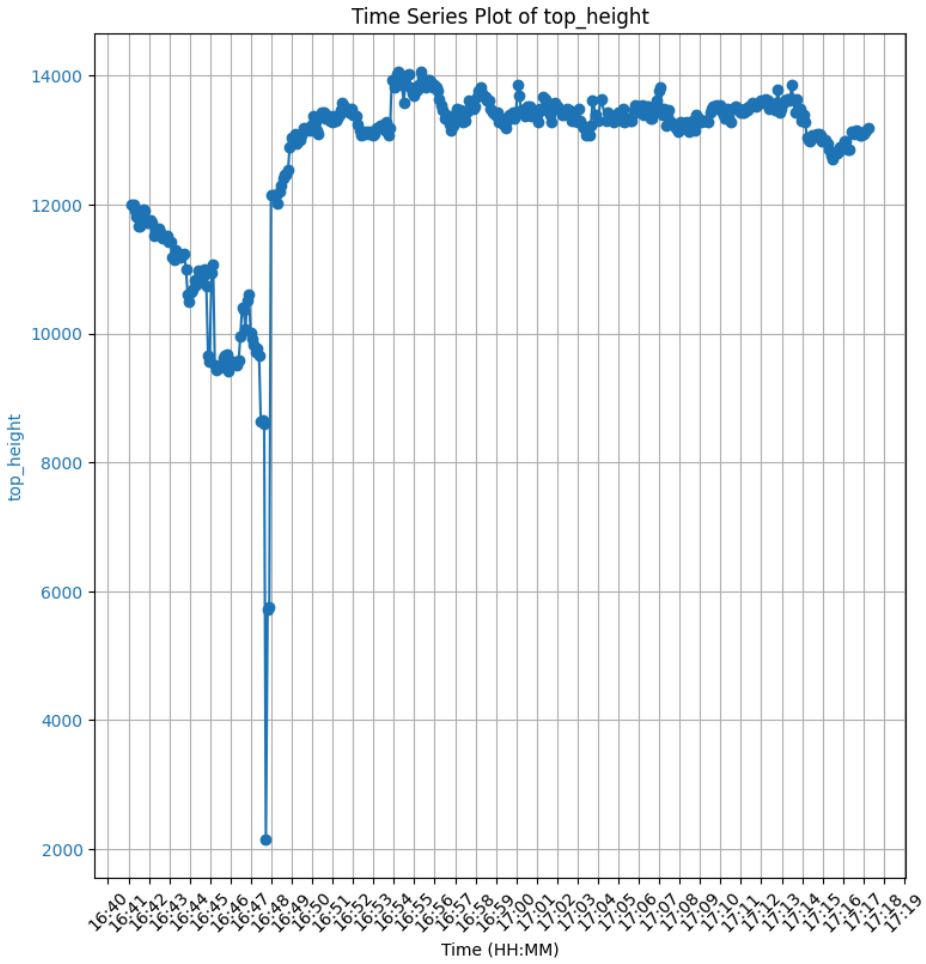


Figure 1. Illustration of Need to Drop Large Turn.

The LiDAR is able to measure clouds at different layers. Since the only metric of interest is the cloud top height, LiDAR only needs to detect the first available layer of cloud.

After dropping the turns and assembling the dataframe for all five flights, we found that the dataset contained 68,040 images taken at 1 FPS from the videos (note the optical flow geometry model in section 3.3.1.1 used frame rates up to 60 FPS). These 68,040 images contain

12,522 LiDAR height measurements, so roughly 5.4 images per sequence lead to a ground truth measurement. The mean value of the LiDAR is 12696.3 meters, the maximum height is 16,009 m, and the minimum height is 1079 m. However, despite this appearing like we have sufficient variability in the data to train a model for this range, the actual distribution is much poorer. The overwhelming majority of the cloud heights are from roughly 10,500 meters to 14,000 meters, which later leads to our deep learning models struggling to predict outlier heights.

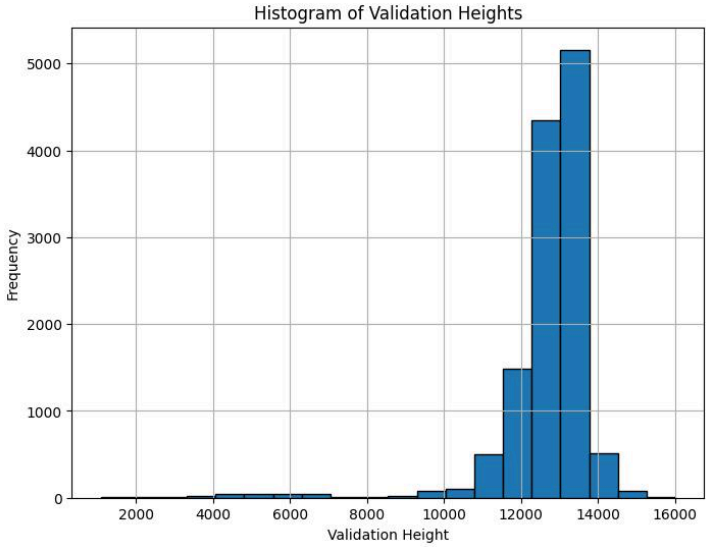


Figure 2. Histogram of Validation Heights

Considering the FEGS video, plotting the histogram and cumulative distributions of the images extracted revealed a lower range of minimum and maximum pixel values, i.e., low contrast. Thus we employed a range of correction techniques, including histogram equalization and adaptive histogram equalization, to improve the image contrast.

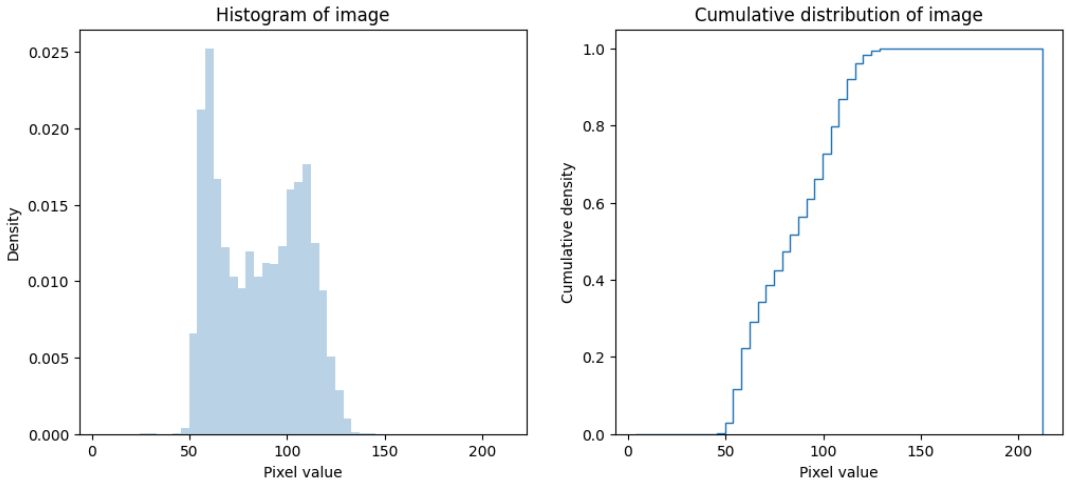


Figure 3. Illustration of Histogram and Cumulative Distribution of Original vs. Augmented Image to Demonstrate Effect of Augmentation

3.2.3. Data Processing

3.2.3.1 ER2 Aircraft Metadata Processing

We first processed the ER2 Aircraft Metadata data to filter it based on the start and end of the associated FEGS Video timestamps. Temporal alignment was implemented to match timestamps of the frames extracted from the video. We have dropped duplicates and kept the first records in case of multiple entries within 1 second. After processing, the features made available apart from metadata (timestamp, etc.) columns are `"Lat", "Lon", "GPS_MSL_Alt", "WGS_84_Alt", "Press_Alt", "Radar_Alt", "Grnd_Spd", "True_Airspeed", "Indicated_Airspeed", "Mach_Number", "Vert_Velocity", "True_Hdg", "Track", "Drift", "Pitch", "Roll", "Side_slip", "Angle_of_Attack", "Ambient_Temp", "Dew_Point", "Total_Temp", "Static_Press", "Dynamic_Press", "Cabin_Pressure", "Wind_Speed", "Wind_Dir", "Vert_Wind_Spd", "Solar_Zenith", "Sun_Elev_AC", "Sun_Az_Grd", "Sun_Az_AC".`

3.2.3.2 LiDAR Data Processing

LiDAR Data Availability is at ~1 measurement per 5 seconds.The LiDAR data has been processed to first filter it based on the start and end of the associated FEGS Video timestamps. The LiDAR data has a layer discriminator for each layer: 1->PBL; 2->Elevated aerosol; 3->Cloud; 4->Undetermined. The LiDAR data was processed to include the height of the first layer where clouds were found. Finally, large turns in the aircraft add noise to the data; therefore, LiDAR measurements that had over a 10-degree delta over the previous measurement were dropped from the analysis.

3.2.3.3. FEGS ER2 Video processing

Data processing for the FEGS ER2 video followed the steps described below:

Figure 4. Chart of steps for ER2 video processing

Frame extraction

All videos were processed to extract JPEG image frames at the rate of 1 FPS and 60 FPS, and the frames were stored.

Fisheye Correction for the Lens Distortion

The lens used by NASA in the FEGS ER2 mission is a 1⁄2" Format CMount Fisheye Lens 1.4mm FL. It is essential to correct the fisheye distortion before processing the image further. A fisheye lens is an ultra-wide-angle lens used to attain a larger field of view but, in the process, produces strong visual distortion intended to create a wide panoramic image. These distortions can lead to problems during the modeling phase. During pre-processing and augmentation, this could lead to non-linear transformations. During feature extraction using CNN, this may lead to incorrect deductions about the spatial features in the cloud. Parameters from lens specifications, derivations, and scripts used to correct the fisheye distortion are detailed in the Appendix.

Image Cropping and Correction

The undistorted fisheye image is scaled out and has a significant unusable area that needs cropping. We realign the undistorted fisheye image so that its center aligns with the original frame center. Finally, it is cropped to take the maximum square from the lens that shows most of the cloud without any lens boundary being visible. The resultant corrected and cropped image is 800x800.

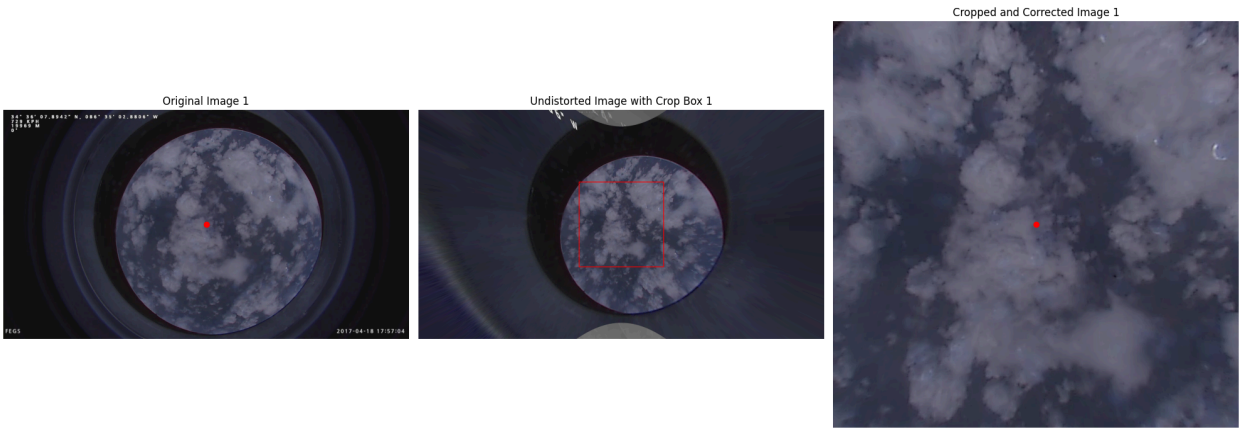


Figure 5. Illustrations of the original image, fisheye-corrected image, and cropped and corrected image.

Original Image: 1920x1080

Cropped and corrected image: 800x800

Image Augmentation

We convert the original RGB image to grayscale to prepare it for augmentation. We adjust the brightness and contrast of the grayscale image using cv2.convertScaleAbs(img, alpha=contrast_factor, beta=brightness_beta). Here a contrast factor of 1.5 and a brightness beta of 30 are used.

$$g(i,j) = \alpha \cdot f(i,j) + \beta = 1.5 \cdot f(i,j) + 30$$

Next, we apply histogram equalization to the image to further improve the contrast of the images by uniformly distributing the intensities on the histogram. Then, we use CLAHE, a variant of adaptive histogram equalization (AHE) that takes care of over-amplification of the contrast. CLAHE operates on small tiles in the image, rather than the entire image. Bilinear interpolation then combines the neighboring tiles to eliminate the artificial boundaries.

Parameters used: `clipLimit=2.0, tileGridSize=(8, 8)`

Finally, the image is sharpened using a convolutional kernel using the `cv2.filter2d` function. The function applies an arbitrary linear filter to an image. The function supports in-place operation. When the aperture is partially outside the image, the function interpolates outlier pixel values according to the specified border mode. The function does actually compute correlation, not the convolution.

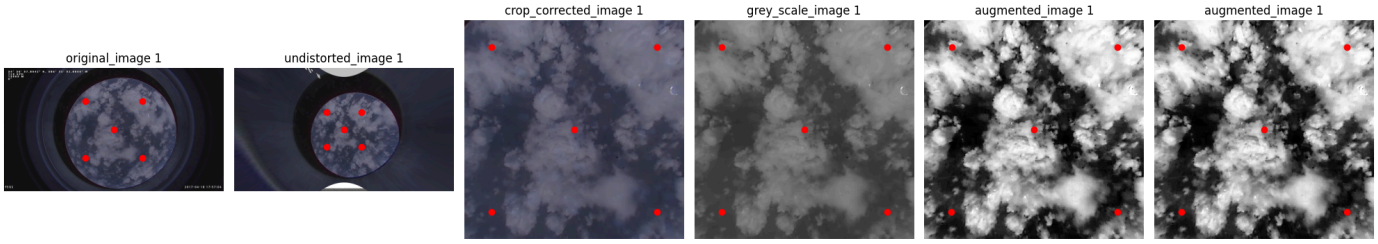


Figure 6. Illustrations of augmented images.

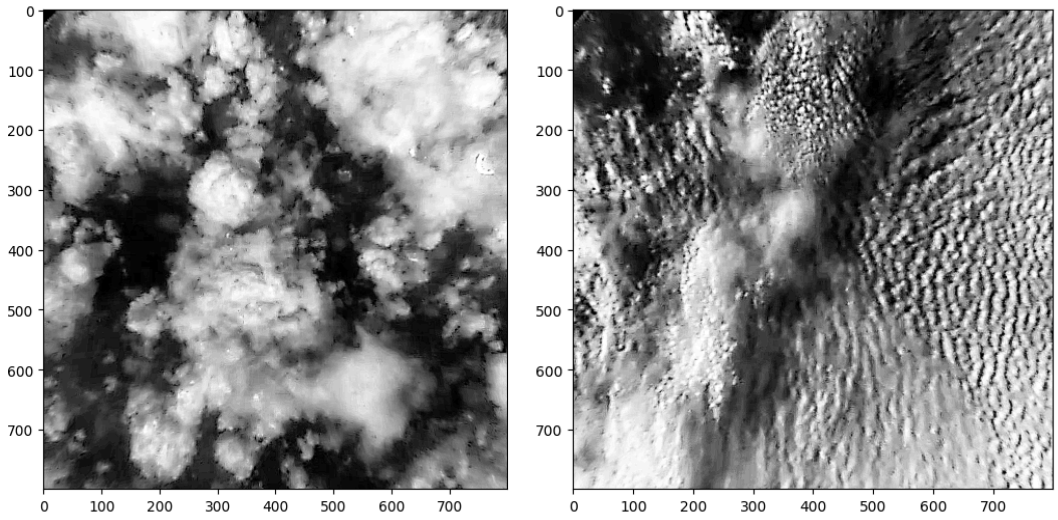


Figure 7. Chain of Image Processing.
Original >> Fisheye Corrected >> Crop Corrected >> Grayscale >> Augmented

3.2.3.4. Combined Dataset

We individually process each dataset from each of the three sources, then combine them into a flat dataset. The FEGS Video Frames and ER2 Aircraft Metadata have a 1:1 mapping, and

both are available at 1 datapoint per second based on the frames and metadata extracted between the start and end time for each day of the video. The LiDAR data, on the contrary, has <= 1 datapoint per 5 data points of the frames/metadata. This is due to the inherent lower frequency of measurement and drop in records for large turns or clouds not found.

We performed a left join between the FEGS Video Frame paths and the ER2 Aircraft Metadata, retaining the number of rows. The resultant data frame is again joined with LiDAR data, and data points that do not have a corresponding LiDAR measurement are filled with NaN. We also add a frame length to indicate the duration of the time-series until a LiDAR measurement is added. Thus the final dataset has the frame paths, corresponding aircraft metadata, and corresponding lidar validation data.

3.3 Feature Engineering

3.3.1 Feature Extraction from Augmented Images Using ConvNext CNN Pretrained Models

Feature extraction is one of the key tasks of CNNs. Feature maps depict the output activations of the convolutional layers, giving insights into edges, patterns, textures, and other features and facets of the image that the CNN has learned. It is how CNNs recognize patterns to be able to classify them or perform various computer vision tasks. In “A ConvNet for the 2020s,” the researchers presented ConvNeXts, a fully convolutional neural network (CNN) model that can effectively compete with state-of-the-art hierarchical vision transformers across multiple computer vision benchmarks. Notably, ConvNeXts retains the simplicity and efficiency of conventional CNNs.

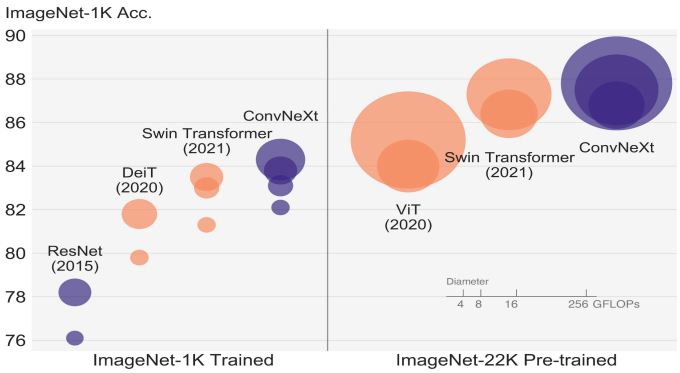


Figure 8. Chart comparing ConvNeXts to other models.

We therefore chose ConvNext as our preferred choice of pre-trained CNN for feature extraction. We compared all the instances of the ConvNext pretrained models. In the face of the challenge of some flight missions being under mostly cloudy skies and very little visual changes to extract features from, it was essential to have a robust CNN model for feature extraction. The convnext-large-224 and convnext-large-384 were found to be the best performing.

The fisheye-corrected, cropped, and aligned grayscale image is first converted back to RGB to prepare it for feature extraction. The RGB image is then passed to the `AutoImageProcessor` of ConvNext that transforms the image to align with the requirements of the input image of the chosen model. The transformations applied include resizing, resampling, rescaling, and normalization.

Next, we load the pretrained model. A single forward pass of the image is done through the model, and the last hidden state is extracted to produce the feature map.

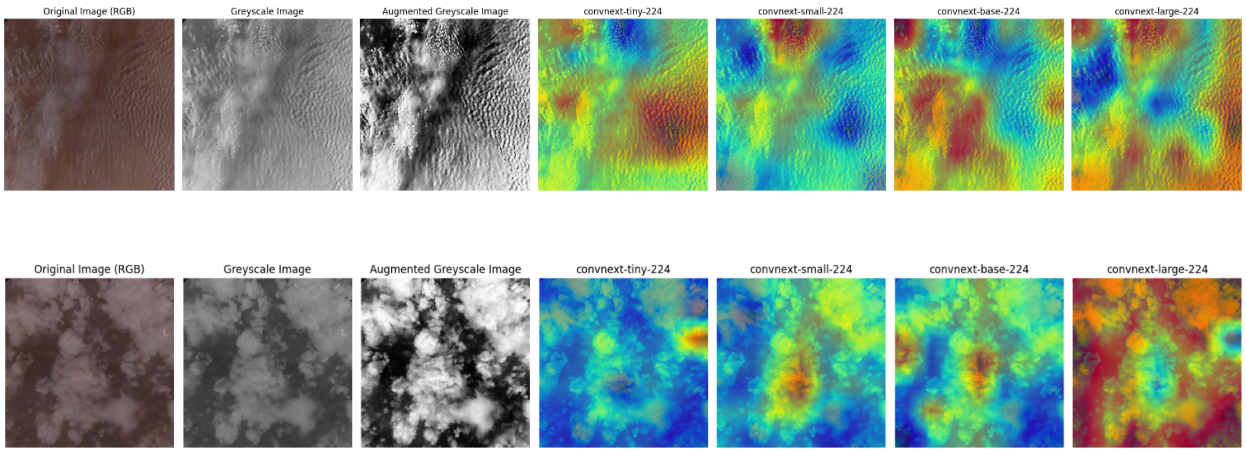


Figure 9. Illustrations of image transformation and overlay of feature map on the transformed image across ConvNext models

3.3.2 Feature Selection from Aircraft Metadata

Based on our research, SME from scientists from NASA and some iteration of model training, the following features were chosen from aircraft metadata to complement the feature maps of the images:

1. Geographic Parameters: GPS_MSL_Alt - GPS Altitude, Mean Sea Level (MSL) in m, Lat - Aircraft latitude in degrees, Lon - Aircraft longitude in degrees
2. Aircraft Motion Parameters: Drift Angle in degrees, Pitch in degrees -90 to 90, Roll in degrees -90 to 90, Vert_Velocity - Aircraft vertical velocity in m/s; Grnd_Spd - Aircraft speed over the ground in m/s.
3. Environmental Parameters: Ambient temperature, total temperature, pressure measurements, wind conditions (speed and direction).
4. Solar Parameters: sun position relative to aircraft, helping in understanding lighting conditions that affect image quality and interpretation.

3.3.3 Random Scaling and Cropping Strategy

The cloud top height from LiDAR is used for validation, and during training we use a random scaling and cropping strategy to move it around the image. This data augmentation strategy aims to improve the model's ability to estimate cloud heights when the LiDAR measurement point is in different locations; LiDAR measurements always being at the exact center of each image could lead to the model overfitting to center-focused features. The strategy maintains a consistent final image size (384x384), preserves original aspect ratios, and maintains relative spatial relationships. During validation, no random cropping is applied, keeping the center point fixed. This allows for a fair comparison with ground truth data.

Consider an original 800x800 image with the LiDAR measurement point at the exact center (400, 400). For the random crop, let's say we choose a crop size of 500x500 and a top-left position of (220, 150).

In the crop transformation step, the new center in the cropped image becomes (180, 250). When we resize to 384x384, we use a scale factor of 384/500 = 0.768, resulting in a final center position of (138, 192).

The result is a 384x384 image where the LiDAR point is no longer at the center but at (138, 192). This represents a large horizontal shift (54 pixels) from the center while maintaining the vertical position.

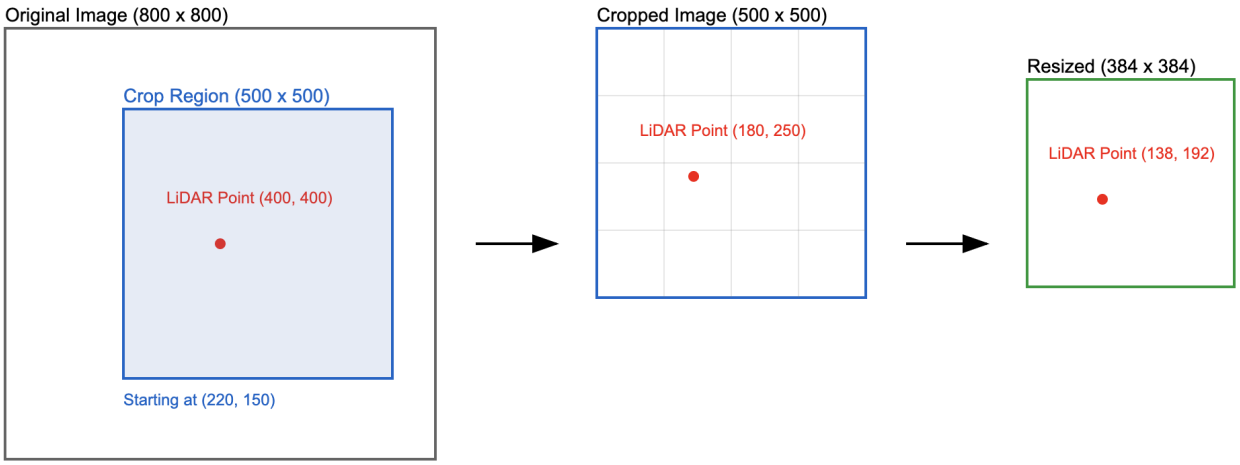


Figure 10. Result: resized and cropped image after transformations.

3.4 Modeling

3.3.1 Proposed Models

The implemented models can be divided into two fundamental categories based on their approach to height field generation:

Full Field Generation Models:

1. Optical Flow Geometry: Uses traditional computer vision techniques to calculate motion fields and derive heights through geometric relationships.
2. MiDAS: Employs monocular depth estimation to generate complete height fields from single images.
3. RAFT: Utilizes advanced optical flow estimation to create motion fields, which are then converted to height fields using parallax principles.

Single-Point Prediction Model:

4. CNN-RNN: Focuses solely on predicting the height at the image center where LiDAR measurements are available.

The key limitation of the CNN-RNN approach emerged from having ground truth height measurements only at a single point (the LiDAR measurement) every five frames. When the CNN-RNN generated complete height fields, it would produce plausible-looking variations in cloud height across each field, but these patterns would be identical between different images except at the center point—the only location where the model received training supervision. This behavior indicated that while the model learned to generate realistic-looking cloud height variations, it wasn't learning to interpret the actual cloud features in different images to produce unique height fields. This was also our initial motivation for using random cropping and scaling to move the center point around in an attempt to circumvent this behavior.

Rather than abandon the CNN-RNN approach, we adapted its role to complement the RAFT model's capabilities. The CNN-RNN became specialized in center-point height prediction, providing calibration data for the RAFT model's full field estimates.

The fifth methodology, field stitching, serves as an integration layer, combining these different approaches to produce coherent, large-scale height maps.

3.3.1.1. Optical Flow Geometry and Algorithms

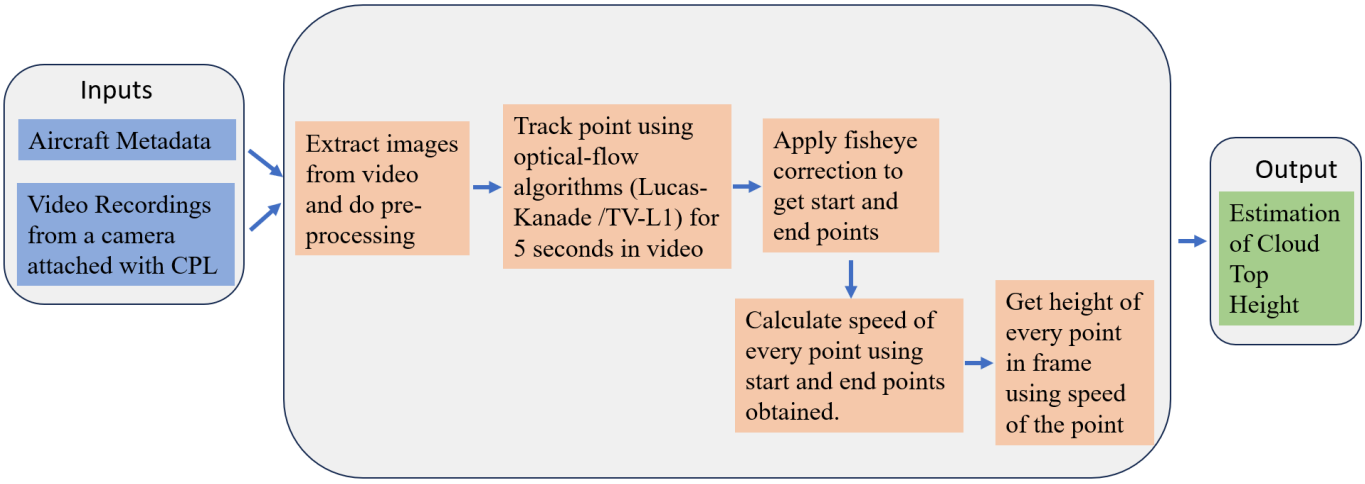


Figure 11. Chart with optical flow macro steps.

The speed of a given point in a camera frame depends on three parameters: the speed of the point in the real world, the speed of the camera, and the distance of the point from the camera. We can assume clouds are stationary and the airplane is moving; therefore, we have moving camera scenes and static scenes.

Let's consider the relative motion of clouds with respect to the camera. In this case, the camera will appear fixed and clouds are moving. The distance of any point from the camera will depend on its speed in the camera frame; we can use this speed in the camera frame to calculate its distance from the camera.

Height of a point from speed of the point in camera frame

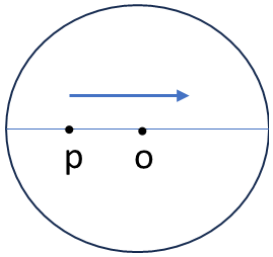


Figure 12(a).

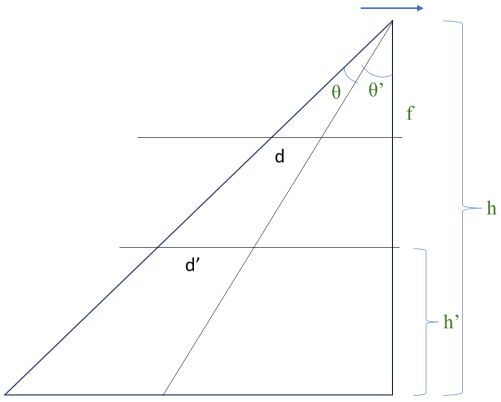


Figure 12(b).

Consider a point P on a line passing through the center of the frame and in the direction of camera motion (Figure 12(a)).

In Figure 12(b), below are parameters,

h - Height of airplane (camera) from ground.
h' - Height of point from ground.
d - Distance travelled by point P in camera frame
d' - Distance travelled by point P with respect to the camera in the real world
θ - Angle made by starting and end position of the point P in the camera frame
θ' - Angle made by end position of the point P and center point in the camera frame
f - Focal length of the camera
u - Speed of the point in the camera frame
v - Velocity of the airplane (camera) w.r.t. ground

The distance d travelled by the point in a frame is given by:

$$d = f \cdot (\tan(\theta + \theta') - \tan(\theta'))$$

The distance d' traveled by the point in the real world is given by:

$$d' = (h - h') \cdot (\tan(\theta + \theta') - \tan(\theta'))$$

The time taken by the point to travel the distance in the real world is given by:

$$t = \frac{d'}{v}$$

The speed of the point in the camera frame is:

$$\begin{aligned} u &= \frac{d}{t} \\ &= \frac{f \cdot (\tan(\theta + \theta') - \tan(\theta'))}{\frac{(h-h') \cdot (\tan(\theta+\theta') - \tan(\theta'))}{v}} \\ &= \frac{f \cdot v}{h - h'} \end{aligned}$$

Rearranging for h'

$$\begin{aligned} h - h' &= \frac{f \cdot v}{u} \\ h' &= h - \frac{f \cdot v}{u} \quad (1) \end{aligned}$$

The above formula (1) gives the height of the point from the ground. This formula doesn't depend on θ and θ'. Hence, all points on the line passing through the center of the frame and in the direction of the camera motion with the same speed u will have the same height. Conversely, all points having the same height from the ground will move with the same speed.

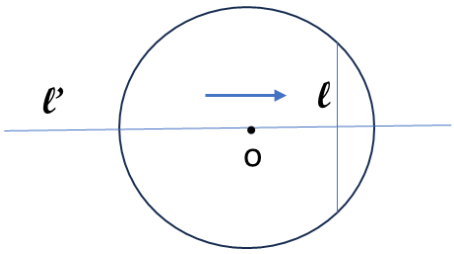


Figure 13.

Consider a plane $\mathcal{P}$ parallel to the ground and at height h. Let ℓ' be a line in the plane $\mathcal{P}$ such that it is parallel to the direction of the camera motion and its image in the camera frame

passes through the center of the camera frame. Consider a line ℓ in the plane $\mathcal{P}$ that is perpendicular to the line ℓ' (Figure 13). All straight lines are preserved in the projective geometry. Therefore, all points on this line will move with the same speed in the camera frame. (2)

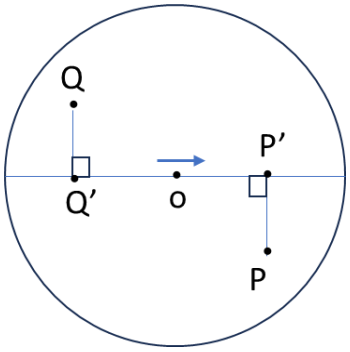


Figure 14.

Now consider the below points (Figure 14).

P - A random point in plane $\mathcal{P}$ that moves in camera frame with a speed u
P' - A point in plane $\mathcal{P}$ on a line ℓ' such that PP' is perpendicular to ℓ
Q - A random point in plane $\mathcal{P}$ that moves in camera frame with a speed u
Q' - A point in plane $\mathcal{P}$ on a line ℓ' such that QQ' is perpendicular to ℓ

From (2), points P and P' will move with the same speed u in the camera frame. Similarly, points Q and Q' will move with the same speed in the camera frame. From (1), points P' and Q' will have the same height above the ground. Hence points P and Q are on the same height above the ground. This shows that all camera plane points moving at the same speed are the same height from the ground in the real world.

Finding the ground position of a point in a camera frame

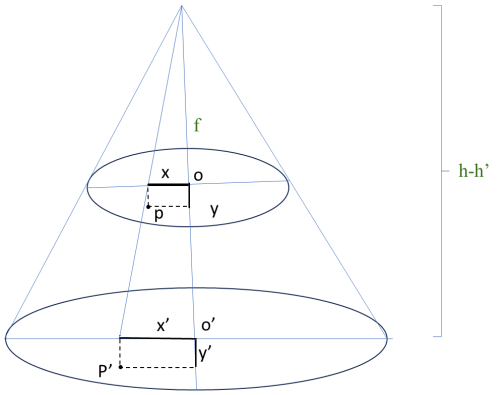


Figure 15.

The position of a point in a camera frame with respect to the ground depends on the height of the point from the ground. Let the center of the camera frame be origin O(0,0). Consider a point P in a camera frame having coordinates (x, y) and height h' above the ground. The origin has the same position on the ground. Let's calculate the x-coordinate of the corresponding point in the ground plane. From the above diagram:

$$\frac{(h - h')}{x'} = \frac{f}{x}$$

$$x' = \frac{(h - h') \cdot x}{f} \tag{2a}$$

Similarly,

$$y' = \frac{(h - h') \cdot y}{f} \tag{2b}$$

Finding the ground position of all points in the video

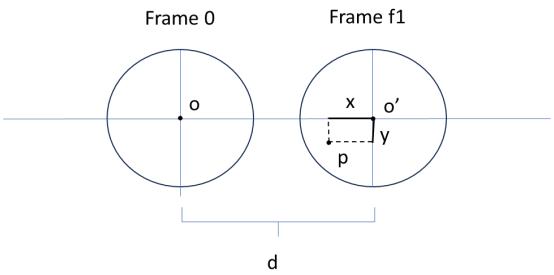


Figure 16.

To find the position in the real world of each point in the video, we can use the latitude and longitude. With this information, coupled with the latitude and longitude for the center point of each frame, we can use formula (1) to calculate the real-world position of each point in the video.

Let's assume that the plane is moving in a straight line. Consider *frame 0* and *frame f1* of the video. We can create a coordinate system for the entire video with the center at point ***O,*** which is the center of *frame 0*. We can create another coordinate system for the frame f1 with the center at point ***O'*** of the *frame f1*.

In Figures 15 and 16 we have the following parameters:

O	Center of *frame 0*
O'	Center of *frame f1*
P	Point in *frame f1* with ground position coordinates (x,y)
d	Distance between ***O*** and ***O'***

Let P have coordinates (x', y') in the coordinate system of the entire video. From Figures 15 and 16,

$$x' = x + d \qquad (3a)$$
$$y' = y \qquad (3b)$$

Using the above formulas, we can get the position of all pixels in the video, and we can create a height field of the entire cloud with viewing points along the line of the flight.

We can use optical flow algorithms to calculate the speed of the point in the camera frame. Below are the algorithms we used.

Lucas-Kanade: Lucas-Kanade is an optical flow algorithm that estimates per-pixel motion estimation between two consecutive frames. It assumes that the objects do not move significantly between frames. The method was documented by B.D. Lucas and T. Kanade (1981). The Lucas-Kanade method is considered "sparse" because it calculates optical flow only for a select set of points in an image, typically features like corners, rather than computing the flow for every single pixel, which would be considered "dense" optical flow. The main idea of this method is based on a local motion constancy assumption, where nearby pixels have the same displacement direction. We primarily used this method for center point tracking and validation against the lidar dataset. Since it computes the motion vector for the specific set of objects (for example, detected corners in an image), it requires some preprocessing to extract trackable corner features from the image, which will be the base for the optical flow calculation.

TV-L1 Method: TV-L1 is a dense optical flow algorithm used to calculate per-pixel motion estimation between two consecutive frames. This means that unlike sparse optical flow methods, this algorithm can calculate a motion vector for every pixel in the image. This method was first described by Zach, C., Pock, T., and Bischof, H. (2007), and it preserves discontinuities in the flow field and is more robust against illumination issues and noise. For this paper, this model was used to generate cloud height fields for the entire image instead of being limited to just the center trackable points as with Lucas-Kanade.

3.3.1.2. MiDaS

MiDaS is a model for monocular depth estimation developed by Birkl, Work, and Müller at Intel Labs (2023). Depth estimation means regressing dense depth solely from a single input image. MiDaS is a pre-trained model on 12 different datasets with multi-objective optimization. By using this pre-trained model, we can generate height fields using each cloud image as an input; however, given that cloud imagery would be considerably out of domain with this model's training data, coupled with the lack of distinct features in clouds, we were skeptical of success with this method.

3.3.1.3. RAFT + Height Calculator

The Recurrent All-Pair Field Transforms (RAFT) model is a deep network architecture for optical flow. RAFT was developed by Zachary and Deng at Princeton University (2020). By extracting pixel features and building 4D correlation volumes for pixels, it creates a flow field. For a full technical overview of our implementation with all algorithms explained, please see: https://github.com/cloud-2-cloud/c2c/blob/main/RAFT/raft_appendix.pdf

RAFT Model and Motion Field Generation

We use RAFT for optical flow estimation. RAFT processes sequences of high-definition aerial images to detect cloud motion. It generates dense optical flow fields that capture the movement of each pixel between consecutive frames. This fine-tuned model is specifically optimized to identify motion patterns characteristic of different cloud types and altitudes, providing a robust foundation for subsequent height calculations.

Bidirectional Flow Analysis and Motion Field Refinement

To ensure reliability, we perform bidirectional motion analysis, calculating flow fields in both forward and backward directions. This dual-direction approach helps validate motion estimates—true cloud movements should be consistent when analyzed in either direction. When discrepancies are detected, we can identify potential errors or occlusions, helping filter out spurious motion estimates.

The refined motion fields undergo statistical analysis to identify and preserve significant features while reducing noise. We pay particular attention to preserving motion anomalies that might indicate high-altitude clouds, which may only appear between 2 frames in the sequence and not appear in others. A confidence mapping process evaluates the reliability of each motion estimate based on consistency and magnitude.

Parallax-Based Height Calculation

As previously discussed, height estimation relies on the parallax effect, which is the apparent difference in movement between closer and more distant objects when viewed from a moving platform. In this context, higher clouds appear to move faster across the image than lower clouds when observed from an aircraft. We analyze the motion fields produced by RAFT and apply parallax principles to convert these 2D motion observations into 3D height estimates. This technique allows for height estimation across a wide field of view without relying solely on direct distance measurements.

To calibrate the height calculator, a calibration step uses the single LiDAR measurement at the center of each sequence to convert relative height estimates into absolute measurements. We maintain three distinct scales, each with a specific purpose:

Motion Scale

This scale converts pixel motion to physical distances by comparing expected motion (based on aircraft speed and LiDAR height) with observed pixel motion at the image center. The motion scale uses Median Absolute Deviation (MAD) to identify and reject outliers before updating the running average. This helps prevent spurious measurements from destabilizing the scale.

The motion scale uses asymmetric smoothing:

$$\text{scale}_{\text{new}} = (1 - \alpha)\text{scale}_{\text{old}} + \alpha(\text{scale}_{\text{update}})$$

where:

$$\alpha = \begin{cases} 0.05 & \text{if } \text{scale}_{\text{update}} > \text{scale}_{\text{old}} \\ 0.01 & \text{if } \text{scale}_{\text{update}} \leq \text{scale}_{\text{old}} \end{cases}$$

Height Scale

The height scale converts relative motion to absolute heights using the ratio between LiDAR height and aircraft altitude. It uses simple exponential smoothing without MAD filtering, as the input measurements (aircraft altitude and LiDAR) are generally stable.

Vertical Extent Scale

This scale maintains reasonable vertical ranges for cloud features, limiting them to either 30% of the cloud-top height or 3000 meters. Like the height scale, it uses simple smoothing without MAD filtering.

Integration with Aircraft Metadata

Accurate height calculation requires knowledge of the aircraft's position and motion. We integrate real-time metadata from the aircraft, including altitude, speed, and orientation. This information is used to calibrate the scale of the observed motion, converting pixel movements in the image to real-world distances and heights.

Confidence Mapping for Measurement Reliability

To account for uncertainties in the estimation process, we generate confidence maps for each height measurement. These maps assess the reliability of height estimates based on factors such as the clarity of detected motion, consistency across multiple frames, and agreement with surrounding estimates. Confidence mapping provides context for interpreting the height data and can be used to weight measurements in subsequent processing steps or to identify areas that may require additional validation.

Height Field Generation and Validation

The system combines individual height estimates to create continuous height fields representing the cloud top surface across the entire observed area. We use the LiDAR to calibrate and adjust the height estimation algorithms in real time.

Uncertainty Estimation and Quality Control

Quality control measures are applied to ensure the reliability of the generated height fields. This process involves statistical analysis of the height estimates, direct comparison with

LiDAR measurements, and assessment of temporal consistency across image sequences. Uncertainty estimation provides error bounds for each height measurement, allowing users to understand the confidence level of different areas within the height field.

RAFT Fine-tuning

Since we don't have any complete height fields to supervise the fine-tuning of the RAFT model, we created a loss function with multiple components that combines temporal consistency constraints with LiDAR-based supervision.

The motion consistency loss consists of three complementary terms that enforce different aspects of physical plausibility:

Photometric Loss

The photometric loss enforces temporal coherence by verifying that predicted motion fields can accurately reconstruct subsequent frames:

$$L_{photo} = \sum_{t=1}^{T-1} \|I_{t+1} - \text{warp}(I_t, F_t)\| \cdot V_t$$

Where I_t represents frame t, F_t is the estimated motion field, and V_t is a visibility mask that handles cloud occlusions. This term helps distinguish between actual cloud motion and illumination changes.

Smoothness Loss

To enforce physically plausible cloud motion patterns, we penalize rapid spatial variations in the motion field. This term encourages uniform, fluid-like motion patterns.

$$L_{smooth} = \sum_{t=1}^{T-1} \|\nabla F_t\|$$

Flow Consistency Loss

We ensure bidirectional consistency by verifying that forward and backward motion estimates are compatible:

$$L_{consist} = \sum_{t=1}^{T-1} \|F_t + \text{warp}(F_t', F_t)\|$$

This bidirectional verification helps distinguish between actual cloud motion and apparent motion caused by viewing angle changes.

3.3.1.4. CNN + RNN models (Single-point predictions only)

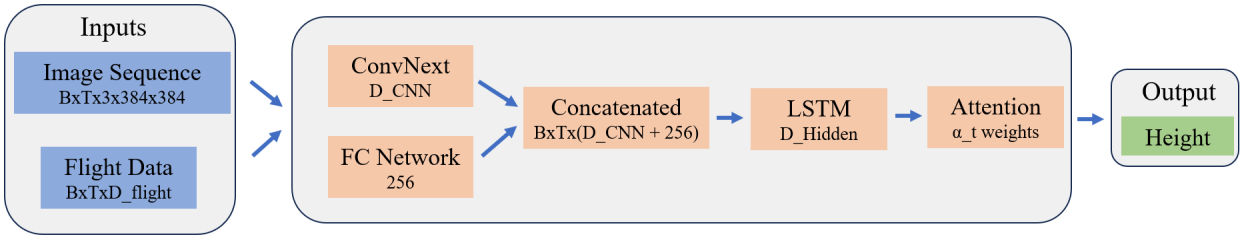


Figure 17. Workflow chart for CNN + RNN models.

CNNs are popular and commonly used models for image classification, regression, object detection, and other computer vision applications. They use convolution layers to apply convolutional filters to the input images for feature extraction. Pooling layers then reduce the dimensionality of the feature maps, making the computation more efficient and reducing the risk of overfitting.

The CNN-RNN CloudHeightEstimator is a hybrid neural network architecture designed to predict cloud-top heights from sequences of aerial imagery and flight metadata. The model combines computer vision and temporal sequence analysis through three main components: a

CNN for spatial feature extraction, a flight data processing network, and a long short-term memory (LSTM) network for temporal pattern recognition. For a full technical overview of our implementation with all algorithms explained along with the code, see this notebook, which is readable in GitHub through a browser: https://github.com/cloud-2-cloud/c2c/blob/main/cnn_rnn/final_CNN_RNN_LiDAR_point.ipynb

Image Processing

The first step is the ConvNext-large model, pretrained on ImageNet and optimized for 384x384 pixel inputs. This CNN processes each frame in the temporal sequence, extracting spatial features that characterize cloud formations and their structural properties. The ConvNext-large provides substantial model capacity while maintaining computational efficiency through a modern architecture.

The image processing pipeline handles sequences of 2-5 consecutive frames, with each frame passing through multiple convolutional stages before producing a dense feature vector. This sequential processing enables the model to capture both spatial patterns within individual frames and temporal evolution of cloud structures across frames.

Flight Data Integration

The model incorporates a dedicated processing branch for flight metadata. This branch consists of two fully connected layers that transform 29 raw flight parameters into a learned representation. The first layer maps the input features to a 128-dimensional space with ReLU activation, followed by a second layer that expands this to 256 dimensions. This allows the

network to learn complex relationships between different flight parameters while maintaining computational tractability.

The flight data includes measurements such as aircraft altitude, position, orientation, and environmental conditions. These parameters provide context for interpreting the visual data and help constrain the height predictions based on the aircraft's position relative to the clouds.

Temporal Integration and Attention

After parallel processing of images and flight data, the model concatenates the features from both branches into a unified representation. This combined feature vector captures both spatial and contextual information for each timestep in the sequence. The temporal dynamics are then processed through a two-layer LSTM network with a hidden size of 256 units, allowing the model to learn patterns across the sequence.

To handle varying sequence lengths and focus on the most relevant temporal information, the model employs an attention mechanism. This attention layer computes importance weights for each timestep, enabling the model to learn which frames in the sequence are most informative for height prediction. The attention weights are computed through a learned linear transformation followed by softmax normalization, with padding masks handling variable-length sequences.

Output Generation

The final stage of the model applies dropout regularization ($p=0.3$) to the attention-weighted features before passing them through a final linear layer that outputs the height prediction.

CNN training, validation, and target performance metrics

Training and validation set features are presented on the comparative table below:

Training Set Features	**Validation Set Features**
Random cropping and scaling	No random cropping (center crop only)
Data augmentation	Limited augmentation
Shuffled sequences	Sequential processing
Calculates normalization parameters	Uses the training set's normalization parameters

The model is trained using Huber loss, which combines the benefits of L1 and L2 losses to provide robustness against outliers, since some images are blurry and covered with water droplets.

$$L_\delta(y, \hat{y}) = \begin{cases} \frac{1}{2}(y - \hat{y})^2 & \text{for } |y - \hat{y}| \le \delta \\ \delta|y - \hat{y}| - \frac{1}{2}\delta^2 & \text{otherwise} \end{cases}$$

Training employs the AdamW optimizer with a learning rate of 1e-4 and weight decay of 0.01, which helps prevent overfitting. The learning rate is dynamically adjusted using a ReduceLROnPlateau scheduler, which reduces the learning rate by half when validation performance plateaus with a minimum learning rate of 1e-8.

Batch processing is managed through a custom collation function that handles variable-length sequences and ensures proper padding and packing for efficient computation. We employ gradient clipping with a maximum norm of 1.0 to prevent gradient explosions.

The validation framework for the CloudHeightEstimator model uses a separate validation and test dataset to test how well the model generalizes to new data. The test data comprises a

single section of one of the flights and is removed prior to splitting and set aside to use at test time. When splitting the data, we split by complete sequences rather than individual frames. Each sequence contains 2-5 consecutive frames taken roughly 5 seconds apart. We keep these sequences intact because randomly splitting individual frames would break the temporal relationships within each sequence.

All normalization parameters used during validation come from the training set; this includes both the height normalization and flight data parameters. This reflects real-world usage where we won't have normalization parameters for new data and avoids data leakage.

Since the model works with normalized heights during training (scaled to between 0 and 1), these predictions need to be converted back to real heights in meters for evaluation. This conversion uses the same scaling factors that were calculated from the training data to ensure consistency. The same process applies to the flight metadata—we use the training set's normalization parameters to denormalize all values back to their original ranges.

The main metric we use to evaluate performance is the Root Mean Squared Error (RMSE) between the predicted heights and the actual LiDAR measurements. RMSE is useful because it gives us the error in meters and emphasizes larger mistakes more than small ones. For validation sequences, we don't apply random cropping or data augmentation—these are only used during training.

3.3.1.5 Field Stitching Method

Coordinate System Creation

The stitching process begins by establishing a unified coordinate system based on GPS data from each sequence. This accounts for aircraft movement and aligns overlapping sequences and converts geographic coordinates into a common pixel space.

Weighted Field Merging

Where sequences overlap, the system employs a weighted averaging scheme to merge height estimates. Weights are calculated based on multiple factors, including measurement confidence, distance from LiDAR points, and local height field consistency. This weighted approach creates smooth transitions between sequences. It considers confidence values to avoid over-smoothing in areas with high-confidence measurements while applying increased smoothing to transitions in regions of lower confidence.

Final Height Map Generation

The final output is a combined cloud height map that stitches together data from multiple sequences. Both the broad-scale cloud field structure and fine local details are preserved, providing information about cloud topology across the entire area. Each final height map also includes associated confidence information, communicating the reliability of different regions in the reconstruction.

3.5 Ethical, Legal, and Privacy Considerations

Considering data privacy and access, the data used to train and validate the models are publicly available datasets from NASA. Since NASA used LiDAR and HD cameras with their ER-2 aircraft to collect the data, it is assumed that it is reliable and trustworthy.

The models (either pre-trained or developed using libraries) implemented in the project are open source. Pre-trained models (such as MiDaS) and original methods (such as Lucas-Kanade) have their original authorship properly referenced in this report.

It is important to state that the NASA mission was performed in the US, which may create a bias in terms of geographical region and climate conditions. Thus, before extrapolating the model for other regions in the world, it is important to be aware of this potential bias.

Chapter IV.

Analysis

4.1 Optical Flow Geometry and Algorithms: Modeling, Analysis, and Results

With the implemented height formula derivations described in Chapter 3, we can now calculate cloud height per pixel if we can get the pixel movement velocity. The next step is to select an optical flow algorithm to track cloud pixels and get distance over the time of the selected video. For the first method, we used the Lucas-Kanade sparse optical flow algorithm to track pixels near the center to validate against the LiDAR height, which is also limited to the center. For the second, we use the TV-L1 dense optical flow method, which is able to track all pixels within a given image.With the TV-L1 method, we can generate a full height field of the entire cloud.

4.1.1. Lucas-Kanade For Height Calculation of Central 48 x 48 Pixels

Firstly, we have used the Lucas-Kanade derivation logic to predict the height and implemented a pipeline to focus on the central 48 x 48 pixels for frames with lidar truth. From those pixels we select suitable candidates for Lucas-Kanade optical flow tracking.

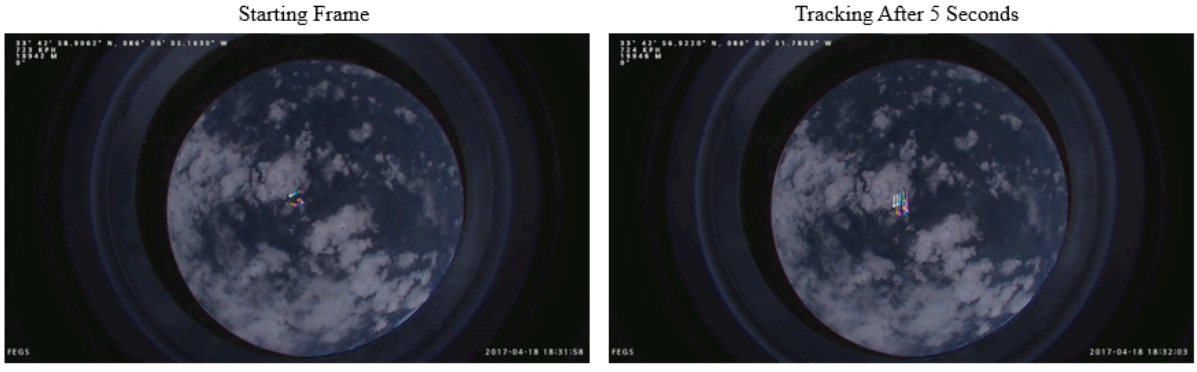


Figure 18. Starting Frame and Tracking After 5 Seconds.

On the left-hand side, we are starting Lucas-Kanade tracking for frame number 125640 from the 18th April FEGS video from 17:57:06 hours to 18:33:28 hours. The multi-colored dots within the center 48 x 48 pixels of the image represent different cloud "corners" identified as suitable for tracking by the sparse optical flow method. On the right-hand side, the multicolored lines are the tracked movements of those respective points after 5 seconds, i.e., 300 frames (60 FPS).

For Figure 18 on the left-hand side, we show an example of "corner" pixels selected as suitable for tracking by the Lucas-Kanade sparse optical flow algorithm. On the right, we show the colored tracks of the movement of pixels in successive frames for a 5-second video duration. In total, 10 pixels are successfully tracked in this example. Then, we translate this pixel velocity (pixels/second) into actual cloud height using the aircraft data at that time and the camera lens information as explained earlier. For Lucas-Kanade, we took the mean of trackable pixels in the 48 x 48 center point for any given frame as our predicted center height for validation against the LiDAR data. Note, we did not track the centermost pixels directly, given that this tracking method can only work with corners.

In terms of what specific criteria we used to identify good pixels to track for Lucas-Kanade, we used the Shi-Tomasi scoring method:

$$R = \min(\lambda 1, \lambda 2)$$

If the minimum eigenvalue is greater than a threshold value, it is considered a corner. Initially, for proof of concept, we had a list of good, Lucas-Kanade trackable pixels that were hand-selected. In order to find the optimal threshold value, we iterated from 0.5 down to 0.07, stopping at 0.07 when 100% of our hand-labeled good trackable pixels were auto-selected using Shi-Tomasi scoring.

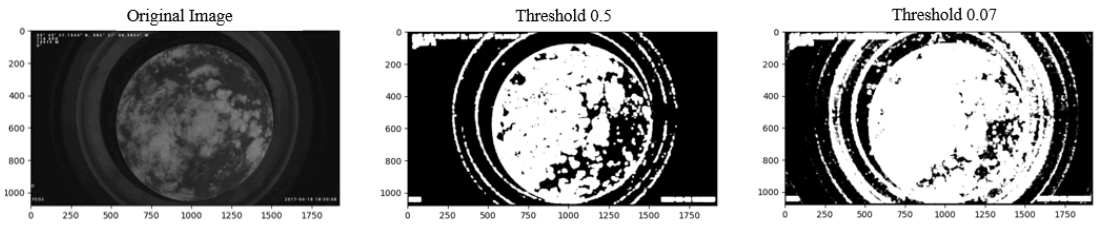


Figure 19. Original Image vs. Threshold 0.5 and 0.7.

We use Shi-Tomasi corner detection to identify "corner" pixels suitable for tracking. In the above images, from left to right, we have the original frame and then 2 grayscale frames representing different thresholds for Shi-Tomasi. White pixels represent the locations in the original image where the function has identified suitable pixels for Lucas-Kanade tracking based on the structure tensor and the specified threshold. Black pixels represent locations where the pixels are not suitable for tracking. You can see that they often select pixels that outline the cloud against the sky. Threshold 0.5 covered only 60% of the hand-labeled trackable pixels, whereas the threshold 0.07 covered 100%.

4.1.2. Time-Series Analysis for Predicted vs. LiDAR Validation Center Height Comparisons

The following table shows Lucas-Kanade and TV-L1 results of height estimation at the center using frames extracted at 60 FPS from the 18th April FEGS video from 17:57:06 hours to 18:33:28 hours. Points with predicted heights > 15000m or < 10000m were dropped from analysis since optical flow was not accurate—either facing tracking algorithm constraints and/or sudden observed camera movement possibly due to turbulence (see Future Work section).

Optical Flow Algorithm	Mean Absolute Error (MAE)	Root Mean Squared Error (RMSE)
Lucas-Kanade	601.46	804.94
TV-L1	748.59	1029.57

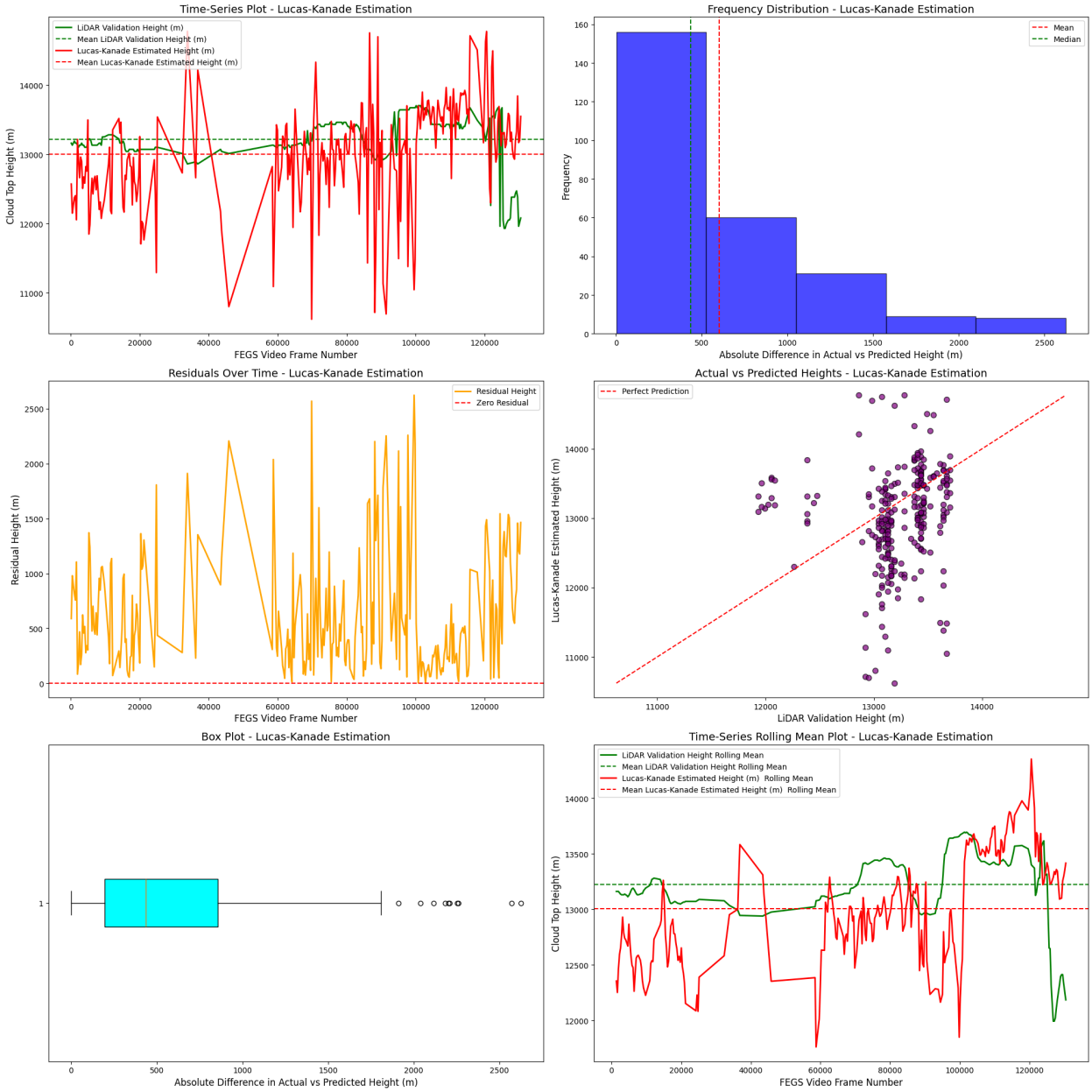


Figure 20. Lucas-Kanade Plots

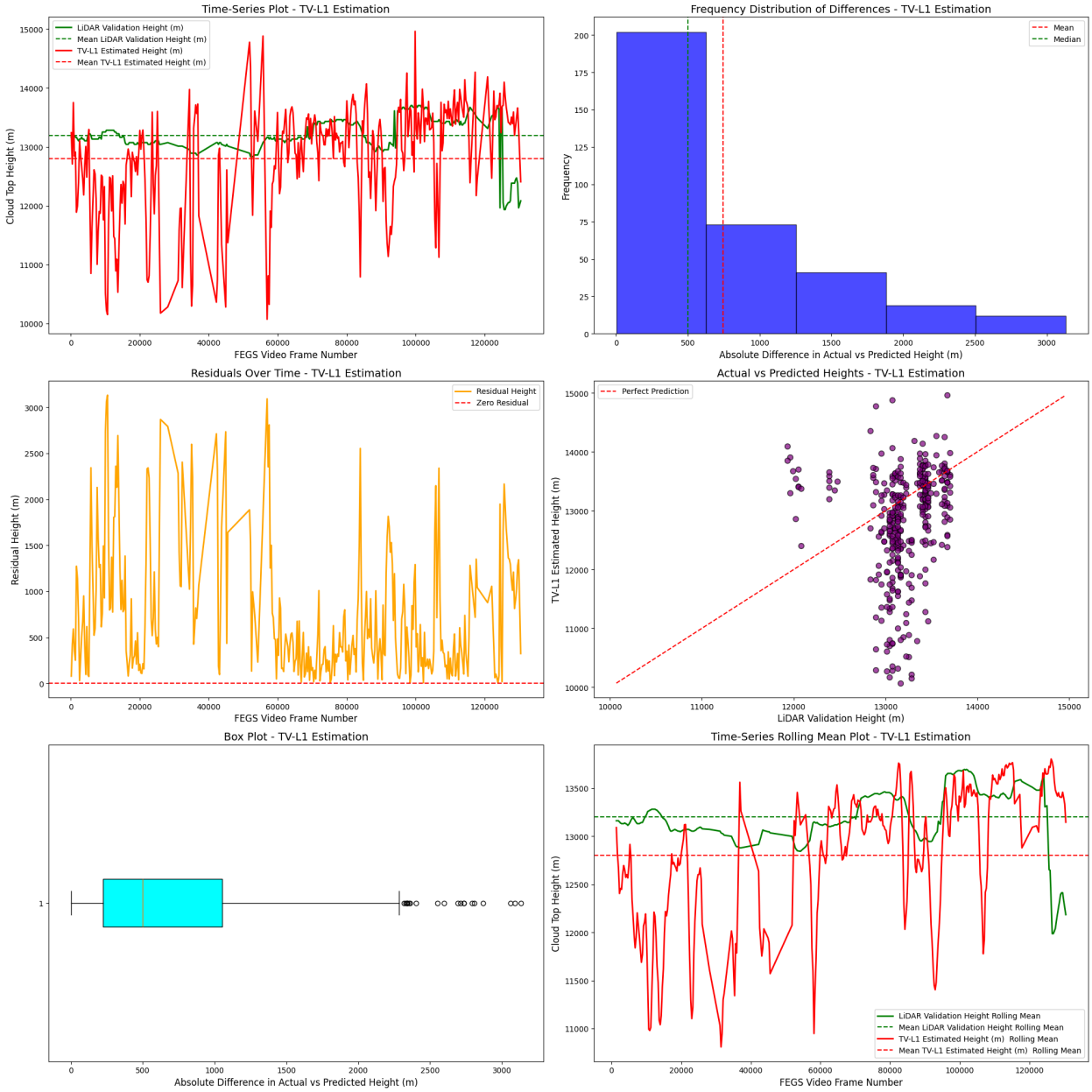


Figure 21. TV-L1 Plots

The above time-series plots for TV-L1 and Lucas-Kanade estimations compare the actual LiDAR measured height versus predicted height across the test video dataset after removing the anomalies where the optical flow was unable to track properly. The residual plot shows the residual getting very close to 0 for many points in the video. The box plot of absolute differences

between actual and predicted heights show the mean differences being ~600m with some outliers to the right of the box plot. This means that the majority of our predictions are off by 600m or less. The scatter plot reaffirms that most estimated height values are around the ~13,000m mark, which makes sense considering most LiDAR validation heights were around this height (see EDA section 3.2.2). Given that the mean value of the LiDAR is 12,696.3 meters across our entire dataset with the heights largely skewed in that vicinity, it is especially important to see whether the predictions can fall outside the 13,000 meter mark—which it does. In the scatter plot, we can see that the predictions do range from 10-15,000 meters; however, they could be more closely coupled to the red-dotted line of perfect predictions. In general, both methods seem to do better for capturing heights between ~12,000-14,000 meters, but not so much outside this range. What is promising is that in general, when looking at the rolling mean plots, especially for Lucas-Kanade in Figure 20, you can see that when ignoring noise, the predicted heights do follow the same visual trends as the actual LiDAR with hills and valleys in the same general timespans.

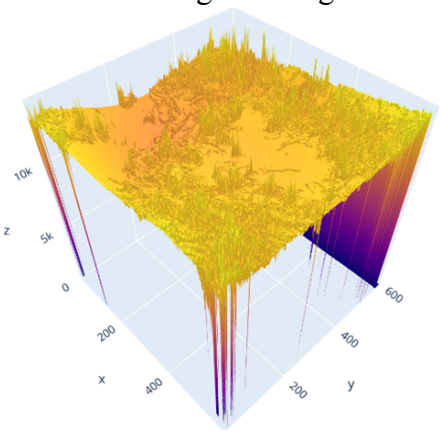
4.1.3. TV-L1 Results of Height Estimation for Entire Frame (3D Height Field Mapping)

We have chosen 430 frames from the 18th April FEGS video and calculated the height field for the 600x600 pixel region around the center of the frame. The time duration between two chosen frames is 5 seconds. Below are the sample results from the run. The first image is the original frame from the video; the second image is a cropped image of size 600x600 around the center; the third is the side view of the plot of the height field calculated by the algorithm, and the fourth is the top view of the height field. From the results, it can be seen that the algorithm is successfully calculating the height fields for the frames (see more examples in Appendix 1).

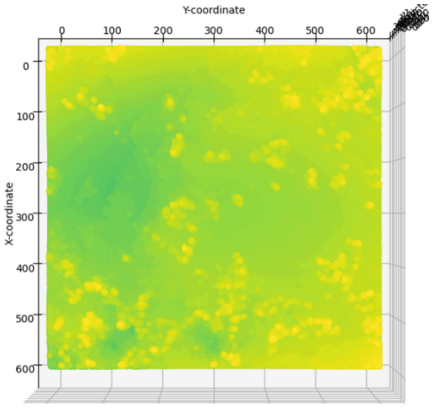
18th April FEGS video 18:32:55

Original Image

Cropped Image (600x600)



Side View of Cloud Height Plot



Top View of Cloud Height Plot

Figure 22. Results example from TV-L1 height estimations

Creating a height field of the entire video using TV-L1 results of height estimation

Once we have height fields for all frames of video, we can stitch them together using estimated height and formulas (3a) and (3b). This will give us a 3D view of the cloud with points of view on line along the aircraft route and passing through the center of the video frame. Figure 22 is a height field generated by stitching height fields of frames from 18:01:04 to 18:01:49.

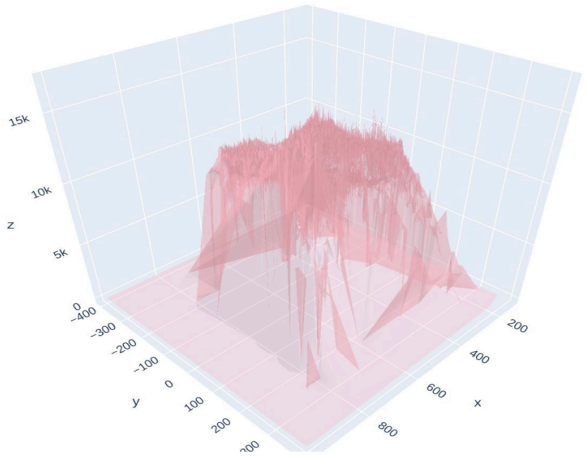


Figure 23. The generated height field.

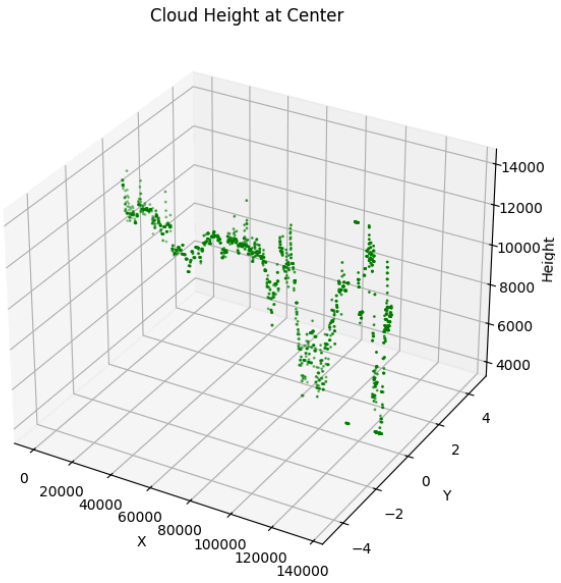


Figure 24. Cloud height at center 3D plot

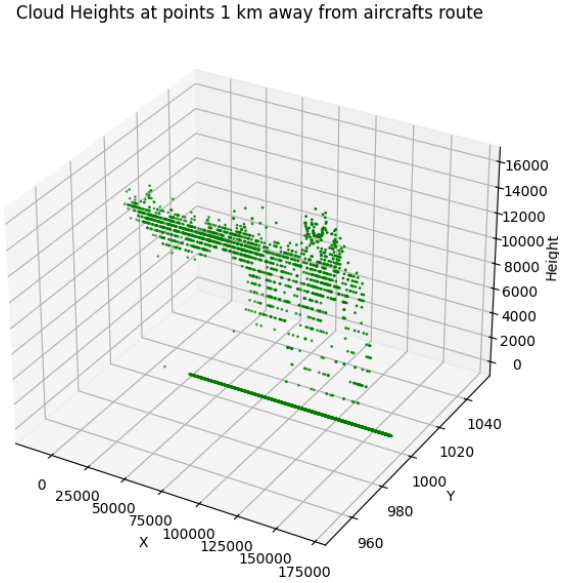


Figure 25. Cloud height at points 1 km away

Figure 24 and Figure 25 show heights calculated by stitching height fields for frames from 17:57:04 to 18:09:35. Figure 24 shows heights at the center line along the aircraft route. We can see that the algorithm is able to estimate the top view of the cloud around the center. Figure 25 shows cloud heights at points 1 km away from the aircraft's route. We can see that the algorithm is able to estimate the side view of the cloud, which is not in the center of the frame. From the above results, we can conclude that using the algorithm, we can generate consecutive stitched height fields of the cloud with viewpoints along the aircraft route and passing through the center of the frames.

4.1.4. Challenges

TV-L1 Performance Challenges and Discussion

Surprisingly, the optical flow geometry method using Lucas-Kande was found to have a better Mean Absolute Error (MAE) than TV-L1, which is generally accepted to be the more accurate tracking algorithm. There are several possible reasons why:

1. First of all, only 294 out of 432 total LiDAR frames in the test dataset FEGS video were able to have any center pixels tracked by Lucas-Kanade sparse optical flow. TV-L1 meanwhile, being a dense optical flow method, was more robust as it was designed to be able to track all pixels within a frame.
2. For center point height comparison with LiDAR, we were able to directly grab the height output for the center-most pixel per frame for TV-L1. However, for Lucas-Kanade, given our limitations, we had to expand to the centermost 48x48 pixels in order to find trackable "corners." The averaging of heights from trackable "corners" could also smooth out the height results.
3. There were per-pixel distance accumulation challenges with TV-L1 that were not present for the sparse optical flow Lucas-Kanade algorithm. For available open-source (OpenCV) packages for Lucas-Kanade, it was relatively simple to calculate the distance of the ~5-10 drawn tracks to find "corner" pixel distance traveled. However, for TV-L1, we can get movement between frames for every pixel, but this movement was usually fractional and not a complete whole number update. Thus, we had to experiment with different frame rates besides the default 60 FPS and frame comparison skips in order to get a movement large enough for a full pixel update, e.g., (x + 1, y + 0) instead of (x + 0.1, y + 0.03). Given potential rounding errors and sampling combinations, it is possible that our

accumulated per-pixel travel distance was not as accurate as it could be. It should be noted that another challenge in refining the optical flow geometry approach using TV-L1 was the runtime speed. It wasn't until the latest updates towards the end of this semester for NVIDIA GPU support that we could run TV-L1 practically enough to experiment with different combinations for optimal frame rates. One of our intriguing observations was that if the framerate were lowered beyond 20 FPS, the predicted heights would be lower across the board for the entire frame. For this report's results, we sampled 20 FPS for a 5-second tracking duration and compared the current frame with 10 successive frames (at 20 FPS) to find the comparison that gave the nearest whole number pixel movement.

General Challenges

If the frame is densely clouded and uniform, this poses a challenge for both the "corner" sensitive Lucas-Kanade and also sometimes the TV-L1 algorithms as well. It would be worthwhile to explore experimenting with enhancement, parameter regularization, or, in general, other optimal flow algorithms (see section 5.4 Future Work).

If there are water droplets or condensation on the camera lens, the algorithm fails to calculate correct motion for those areas. But if a point under the water droplet comes out of it in other frames, it is possible to potentially get the correct motion and backfill those heights (see Future Work section 5.4).

Even after dropping turns as described in the EDA section, it was observed that some of the predicted height outliers manually revisited included segments where the camera appears to wobble—either to missed portions of a turn or simply turbulence (refer to the Future Work

section 5.4). This makes the tracked pixel lines zig-zag, creating issues with calculated pixel velocity.

4.2 MiDaS: Implementation and Results

During the first CNN-RNN model training, the height field was being predicted as equal for all sample images. This challenge arose because the loss function only applied to the center of each image, leaving the model with no incentive to learn the other parts. After reflection on this problem, we decided to implement a pre-trained monocular depth model to generate the relative height fields.

Figure 26. Examples of the types of scenes where MiDAS succeeds

source: original MiDaS documentation: https://pytorch.org/hub/intelisl_midas_v2/

Although it performed well in other scenarios as shown in the example above, MiDAS did not perform very well when applied to the cloud images. Generally speaking, monocular depth models lack absolute scale reference and are not trained for aerial perspectives or cloud-specific features. They struggle with the homogeneous textures of clouds and the absence of a ground plane. These models also cannot incorporate additional metadata or a ground truth. The height fields created often had little resemblance to reality, and it was clear that any model

would need to incorporate image sequences (likely multi-view geometry), temporal information, and flight metadata to generate realistic predictions.

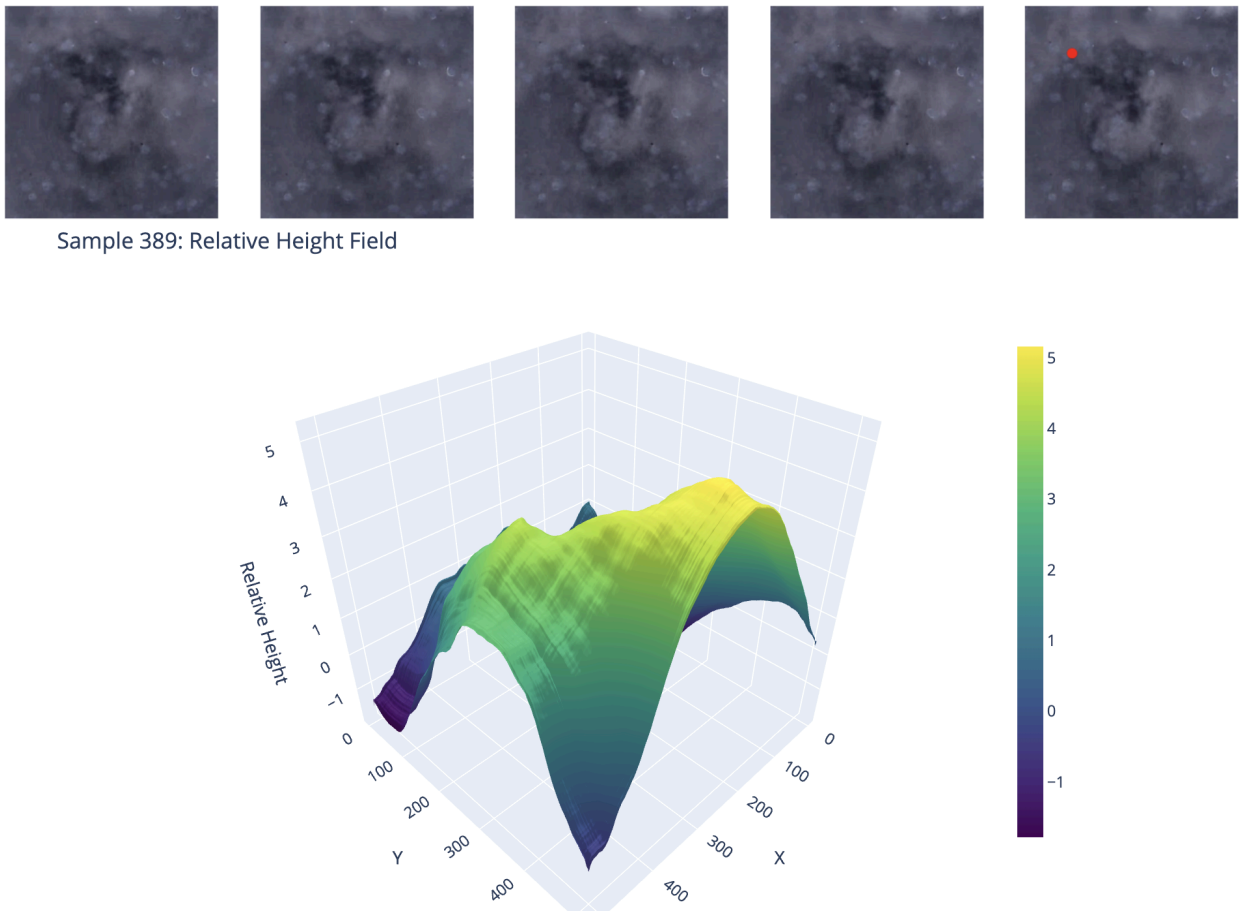


Figure 27. MiDAS model attempting to create a height field for a feature-sparse cloud. The red dot in the final image is the LiDAR location.

4.3 RAFT: Modeling, Analysis, and Results

Here are the outcomes of the RAFT-based cloud height estimation. The following images illustrate key aspects of the system's performance and accuracy with respect to the motion field predictions, the performance of fine-tuned and original RAFT models, and the resulting 3D height field reconstructions.

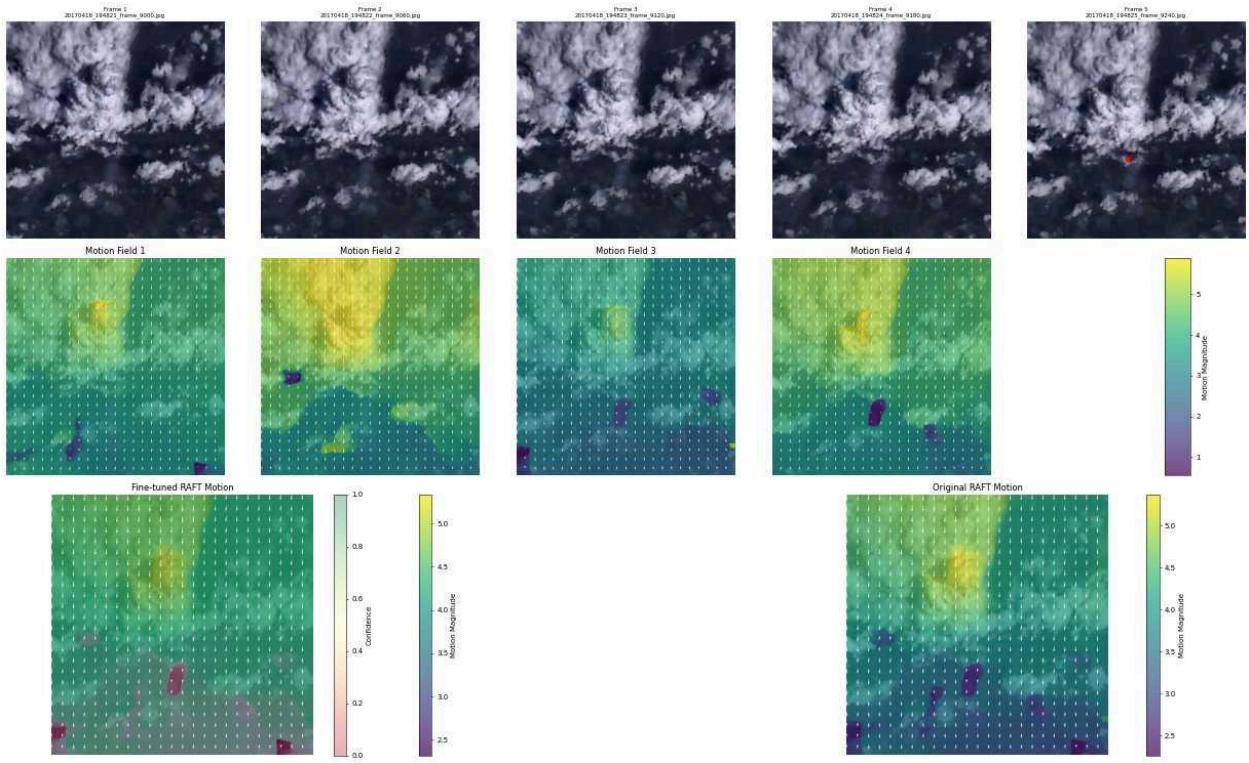


Figure 28. Cloud sequence analysis and motion field comparison.

The top row shows the input image sequence with the red LiDAR dot showing where the measurement was taken (after cropping and resizing). Middle rows display frame-by-frame motion fields. The bottom row compares fine-tuned RAFT (left) and original RAFT (right) motion estimations.

The sequence shows a distinct pattern with denser clouds in the upper portion and scattered clouds in the lower portion. The motion fields (rows 2-5) show consistent patterns across frames, with higher motion (yellow areas) primarily in the upper cloud regions. The fine-tuned RAFT motion field appears more refined, with clearer definition of motion boundaries and less noise in low-motion areas compared to the original RAFT.

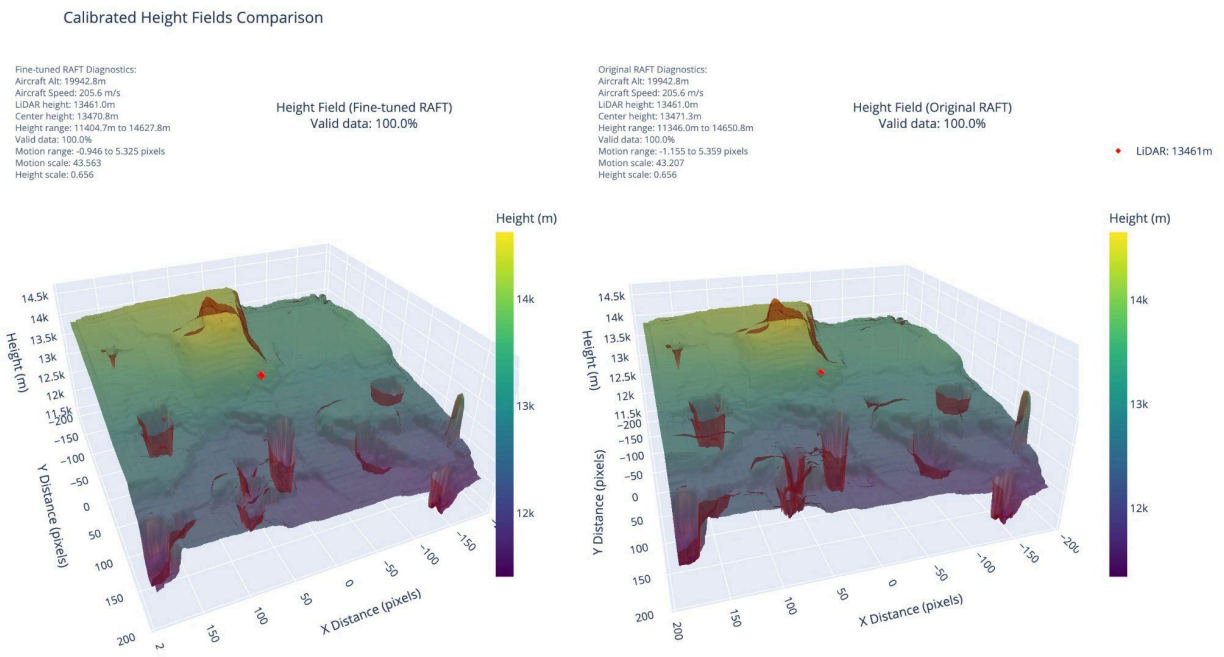


Figure 29. 3D visualization of calibrated height fields. Left: Fine-tuned RAFT results. Right: Original RAFT results. *Both include diagnostic information and LiDAR reference points.*

Both models produce very similar overall cloud structures, indicating consistency in height estimation. The LiDAR height (13461m) is used as a reference point for both models, and we can see that the height fields are indeed well-centered around this value. The fine-tuned RAFT model estimates a slightly wider height range (11404.7m to 14627.8m) compared to the original RAFT (11346.0m to 14650.8m). The motion scales are slightly different (43.563 for fine-tuned vs. 43.207 for original), suggesting minor calibration differences resulting from the fine-tuning.

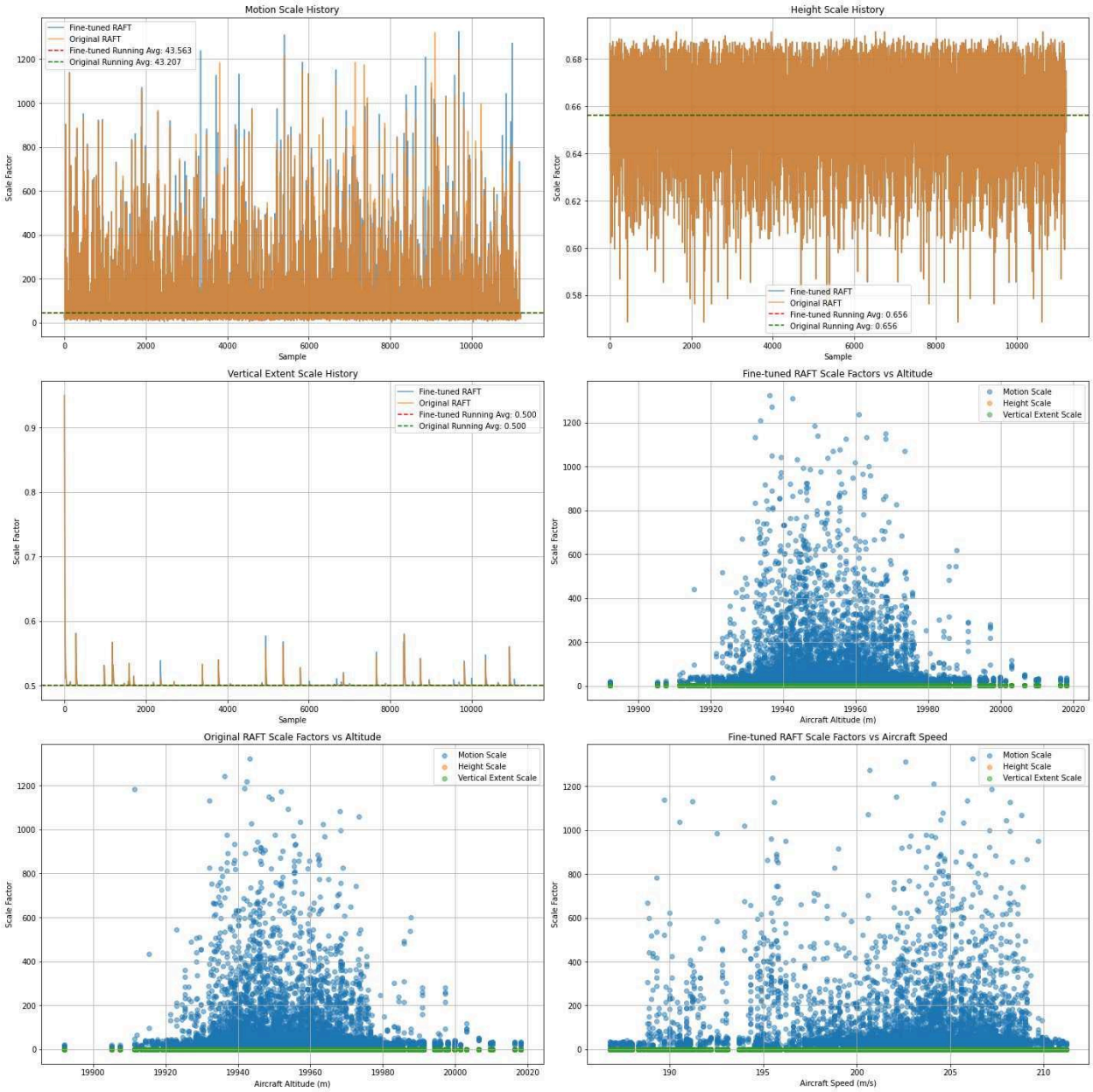


Figure 30. Scale Factor Analysis for Cloud Height Estimation System.

- The Motion Scale's stable running average (around 43) demonstrates the MAD-based outlier rejection. Frequent upward spikes in the graph are preserved by the asymmetric smoothing, biased towards larger scales.

- The Height Scale History graph shows stability, with a consistent running average of 0.656. This directly results from the slow update mechanism. Small fluctuations are visible but quickly dampened.
- The Vertical Extent Scale History graph clearly shows the implemented lower bound of 0.5. Occasional spikes above this level represent instances where the system expanded the vertical range.

4.3.1. Challenges

It's important to note that a significant challenge in assessing the accuracy of our cloud height estimation system is the lack of comprehensive ground truth data for the entire cloud field. While we have precise LiDAR measurements at specific points every 5 seconds, these represent only a small fraction of the total area being analyzed. This limitation necessitates a more subjective evaluation approach for the broader cloud field; that is, we have to literally create visualizations of the cloud heights and decide if they make sense.

We rely on visual inspection of the generated height fields, consistency checks between frames, and comparison of fine-tuned and original RAFT models to gauge performance. Additionally, we examine the coherence of motion fields with visible cloud movements in the image sequences. While these methods provide valuable insights into the system's performance, they cannot offer the same level of quantitative certainty as a full ground truth dataset would. Despite this limitation, our approach represents the best available method for evaluating such a wide-area cloud height estimation system under real-world conditions. Future work could benefit from more extensive validation data, perhaps through coordinated multi-instrument campaigns or advanced simulation techniques.

LiDAR Supervision Component

While the motion consistency terms ensure physically plausible motion fields, converting these to absolute heights requires calibration. We use the periodic LiDAR measurements at the image center for this purpose, comparing the expected motion from the ground truth LiDAR to the predicted height and then scaling the motion to calculate the loss. The process involves:

1. Converting predicted motion to relative motion:

$$r_{actual} = \frac{\|F_{pred}(x_c, y_c)\|}{v_{aircraft}}$$

2. Comparing with expected motion from LiDAR:

$$r_{expected} = \frac{h_{lidar}}{h_{aircraft}}$$

3. Applying calibration through scaling:

$$F_{scaled} = F_{pred} \cdot \frac{r_{expected}}{r_{actual}}$$

The LiDAR supervision loss is then:

$$L_{lidar} = L_{smooth\_l1}(F_{scaled}, F_{pred})$$

Combining Motion and LiDAR Loss

The final combined loss combines these components with dynamic weighting:

$$L_{total} = L_{motion} + W_{lidar} \cdot L_{lidar} \cdot \alpha$$

We train the model for 5 epochs with a low 10e-7 learning rate and a scheduler to only slightly affect the RAFT predictions while also monitoring the original RAFT model alongside it for comparison.

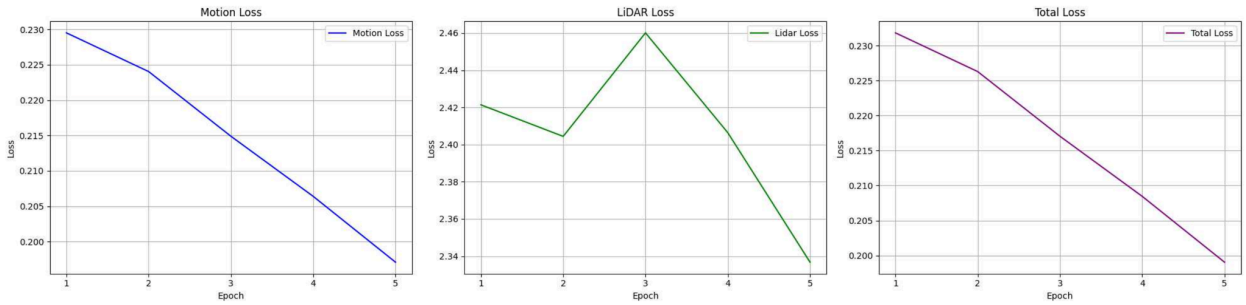


Figure 31. Training loss curves over 5 epochs during RAFT fine-tuning.

The motion loss (left) shows steady improvement as the model learns to better predict cloud motion patterns. The LiDAR loss (middle) exhibits some variation but generally decreases, indicating improving alignment between predicted heights and LiDAR measurements. The total loss (right) demonstrates consistent overall improvement in the model's performance, which is a weighted combination of both the motion consistency and height calibration loss.

Lens Condensation at Extreme Altitudes

The NASA ER-2 aircraft operates at an exceptional altitude of 20,000 meters. At this altitude, well into the lower stratosphere, the aircraft encounters temperatures at about -56.5°C (-69.7°F). These severe conditions make it extraordinarily difficult to prevent condensation on the camera lenses used for cloud observation.

The extreme cold at this altitude causes any moisture in the air to rapidly condense and freeze on surfaces, including camera lenses. Despite best efforts to control the environment around the imaging equipment, the formation of water droplets or ice crystals on the lens is sometimes unavoidable. These droplets are visible in the left image as small, circular distortions scattered across the frame.

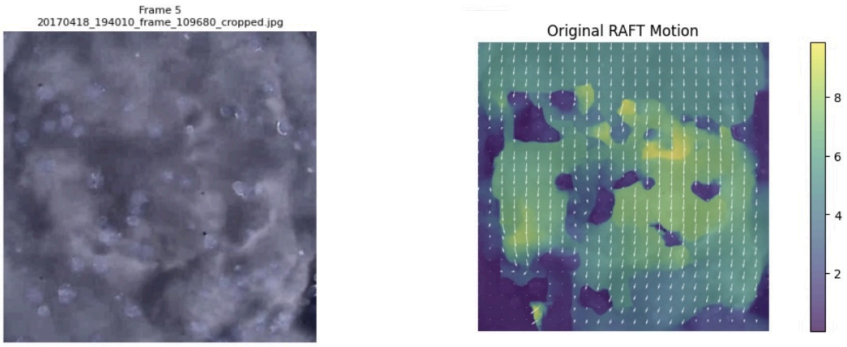


Figure 32. Processed cloud image (left) and motion field (right) that both illustrate the problems of water droplets on the camera lens.

The presence of these water droplets on the lens creates a problem for parallax analysis. As the ER-2 moves at high speed through the stratosphere, the clouds below show apparent motion relative to the aircraft. However, the water droplets on the lens remain stationary in the camera's field of view.

The parallax-based analysis interprets this lack of motion as indicating objects very far away. Since the camera is pointed downwards from the aircraft, the system calculates these static points as being far below the actual cloud tops, resulting in the appearance of what seem to be extremely deep "holes" in the cloud structure.

In the motion field visualization (right image above), these misinterpreted areas appear as dark purple regions, indicating minimal detected motion. These areas correspond directly to the positions of the water droplets visible in the original image, resulting in a motion field and subsequent height map that show artificial variations in cloud height.

Addressing this issue at its source may require innovative solutions, such as advanced lens heating systems or specialized coatings to repel moisture, but they fall outside the scope of the computer vision solution. Many of the possible ways of dealing with this as an image

processing problems require some degree of estimation; we can smooth over what seem to be obvious droplet “holes” in the cloud, but that could potentially smooth over holes that actually exist. However, one technique in the stitching process could potentially solve this problem, which we discuss in the next section.

4.4 CNN + RNN: Modeling. Analysis and Results

After more than 10 hours of training using an NVIDIA GPU A100, the RMSE on the test set was 334.65 meters. Below are the training and validation results from the CNN-RNN model on the center point:

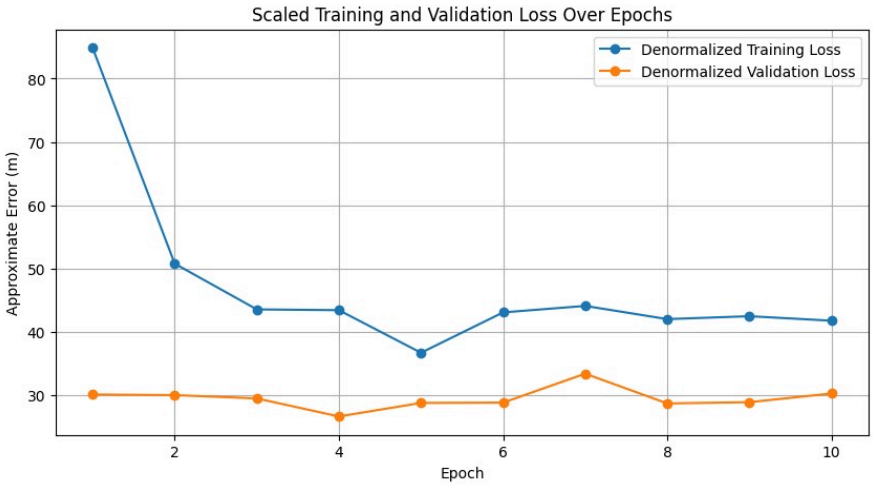


Figure 33. CNN–RNN scaled training and validation loss over 10 epochs.

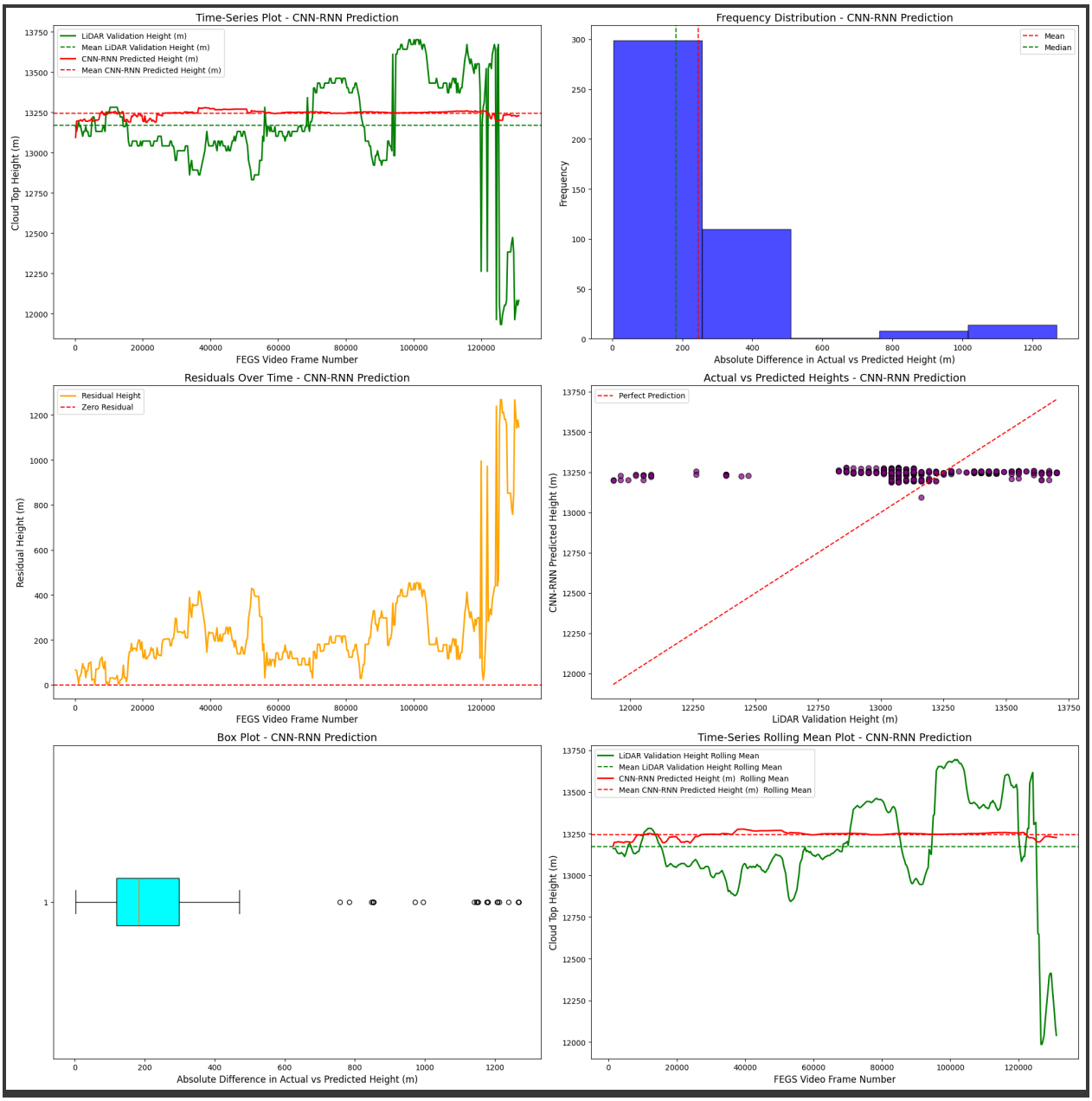


Figure 34. Plots for CNN-RNN results.

The above time-series plot reveals the actual and predicted height across the time frame for real values and rolling heights for the CNN–RNN model. The model seems to have fit well but captures less variability in the cloud top height. This is especially noticeable in the first half of the plot, where the red line has similar movement, albeit diminished, compared to the green line.

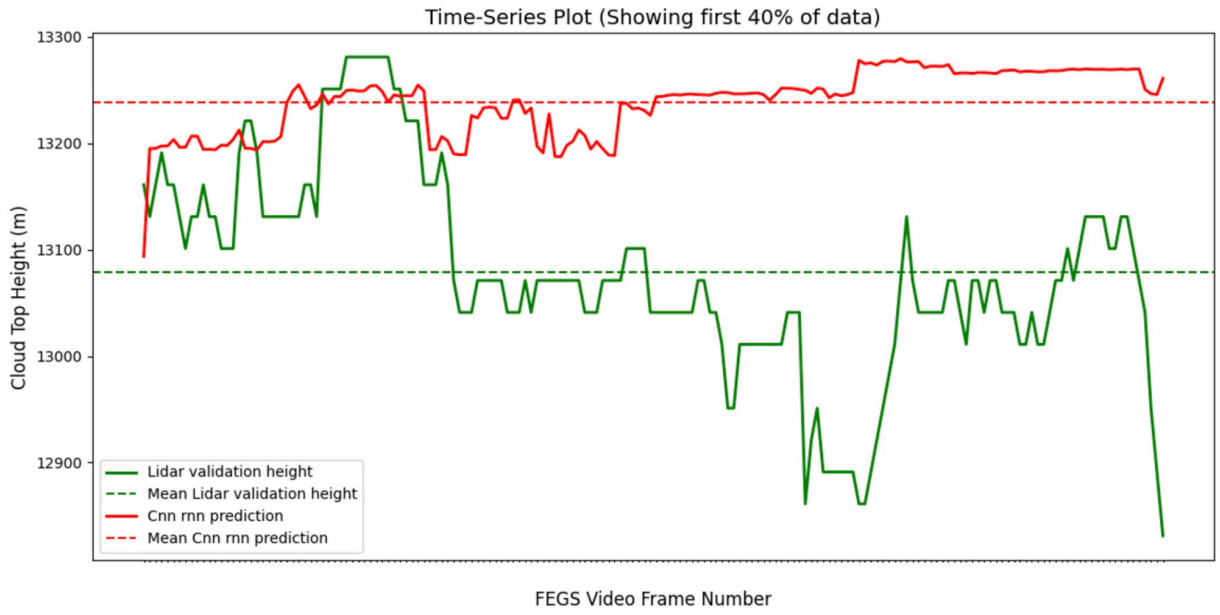


Figure 35. The zoomed-in version of Figure 34's top left plot for CNN-RNN results shows that the variability does indeed correlate with the LiDAR for the first 40% of the time series. While the scale differs, they are actually reasonably close together (the Y-axis above only ranges over 3% of the true altitude).

The frequency distributions of the differences in actual and predicted heights indicate the majority of the differences being in the range of 0 - 200. The residual plot also shows the residual getting very close to 0 for many points in the video. The box plot of differences is suggestive of the mean differences being ~250 with some outliers to the right of the box plot. The scatter plot reaffirms that most predicted values are around the ~13000m mark, reflecting lesser variability in the predictions.

4.5 Field Stitching: Implementation and Results

The stitching implementation as previously described (global coordinate grid creation, weighted averaging where confidence is lower, and heat map generation) works quite well. Due to using the latitude and longitude measurement of the plane, it is able to place the individual

height fields precisely before connecting them. Since the plane is not always traveling in a direction that lines up with the top or sides of the height fields, we needed to account for this direction in the stitched height fields. Below is an example using a tower formation in a storm cloud taken in a diagonal direction.

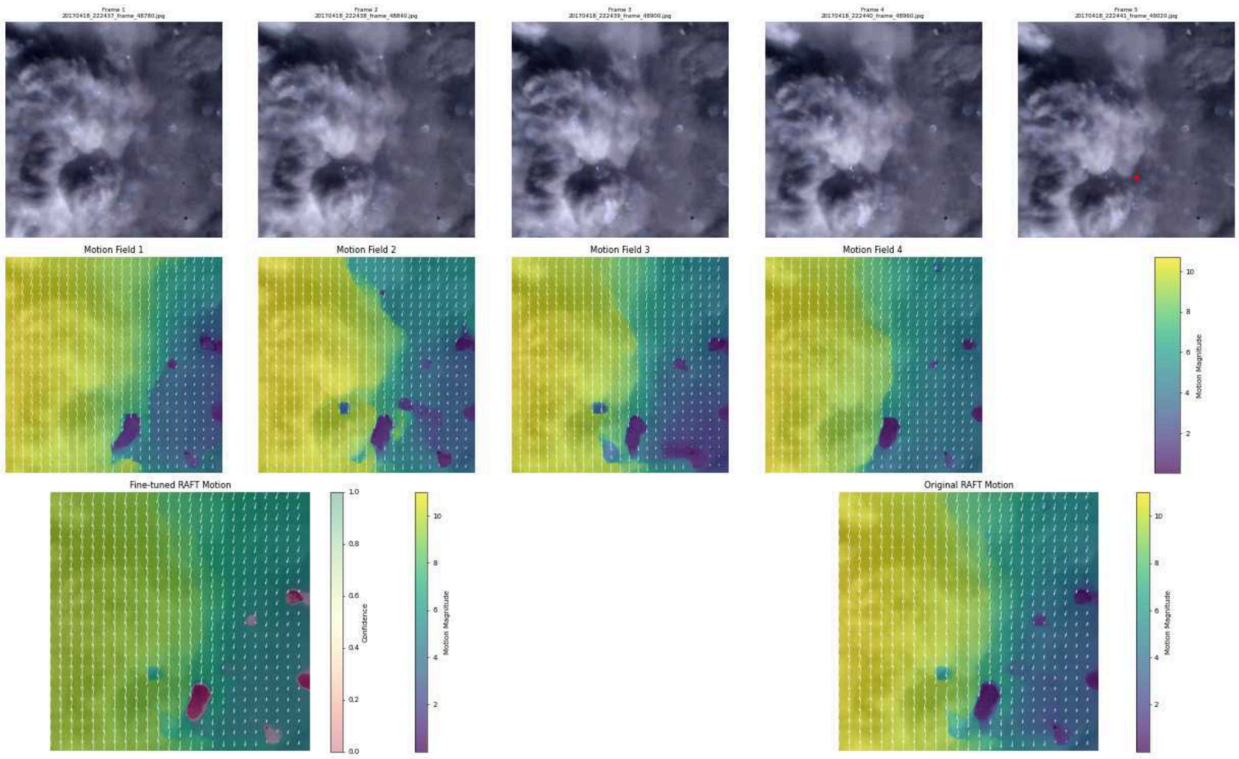


Figure 36. Cloud sequence analysis and motion field comparison of towering formation.

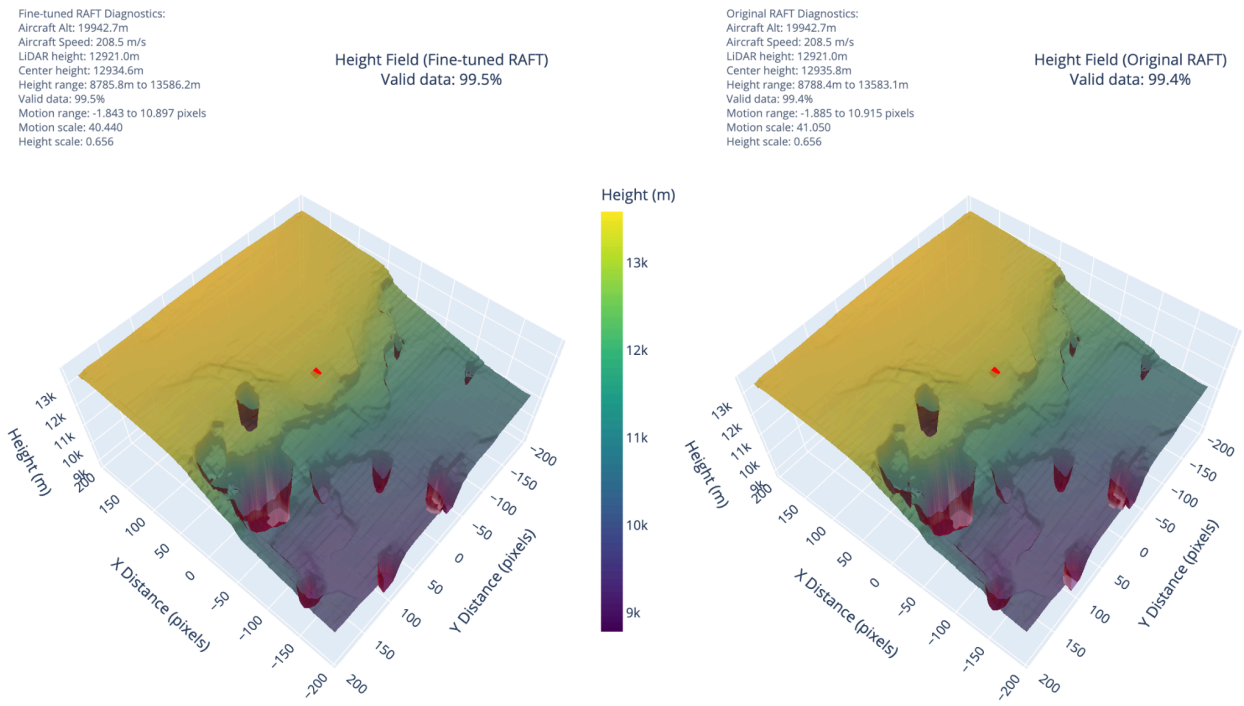


Figure 37. 3D visualization of single calibrated height fields

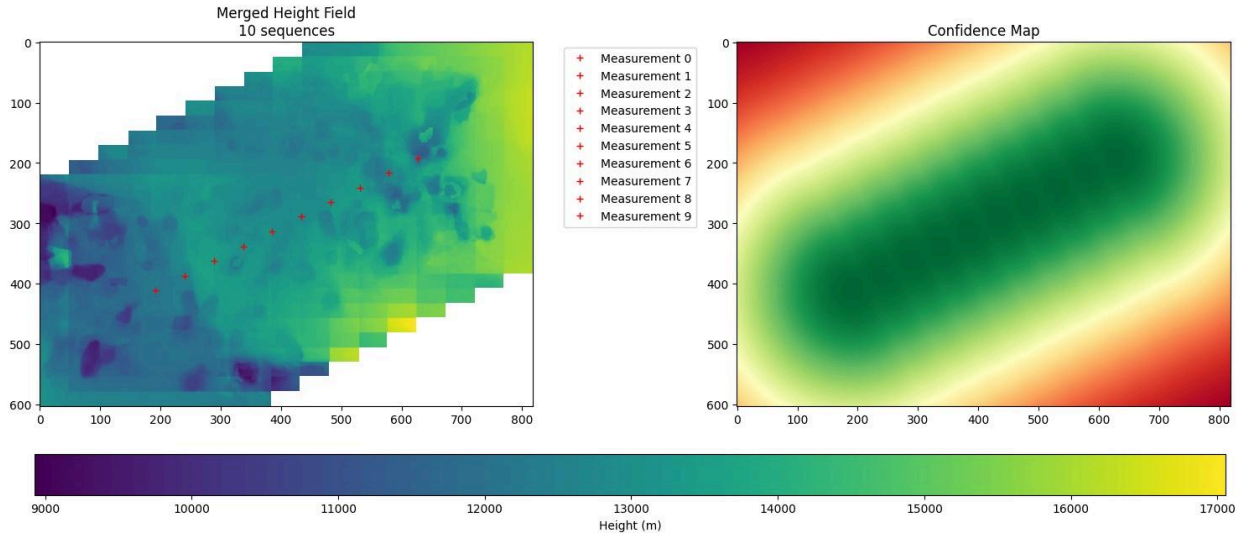


Figure 38. 2D visualization of 10 stitched height fields with LiDAR point in red showing the path of the plane.

The confidence map on the right shows how it falls off as we move towards the edges.

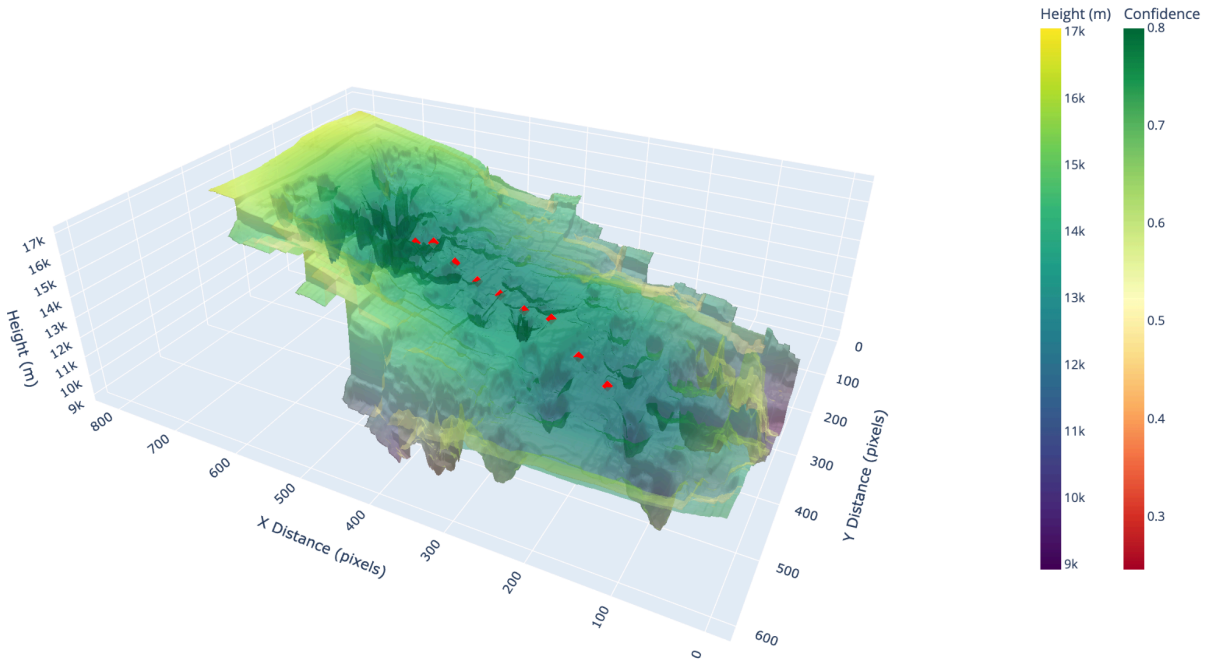


Figure 39. 3D visualization of 10 stitched height fields showing the path of the plane with the LiDAR dots. *Each sequence represents 5 seconds, so this formation was in view for 50 seconds.*

The confidence is generated using a Gaussian falloff from the LiDAR points, which appear as red crosses in the image. The confidence decreases exponentially with distance from each measurement point. Since the LiDAR follows a line down the center of the stitched image, the confidence naturally falls off towards the edges where we're farther from any measurement point.

This was designed intentionally to reflect the fact that we have less certainty about the cloud heights farther from where we have actual LiDAR measurements to validate them.

4.5.1 Water droplet removal

During the stitching, we attempted to identify and remove water droplets on the camera lens based on the principle that water droplets would appear as stationary features in the frame.

while actual cloud features would exhibit parallax motion. The method looked for anomalously low height values that remained in consistent positions across multiple frames, using a combination of dilation operations to account for small positional variations and local height statistics to identify potential droplets.

However, when analyzing the motion fields generated by the RAFT model across sequences, the detected low-height regions showed significant inconsistency in their positions. While water droplets should appear as fixed artifacts in the camera frame, the motion fields failed to capture this consistency. This likely arose because the RAFT model may interpret the sharp boundaries and optical distortions created by water droplets differently across frames, leading to varying motion estimates even for physically stationary droplets.

Chapter V.

Findings, Conclusions, Implications, and Future Work

Results described in the previous chapter validate the potential of computer vision techniques in order to predict cloud top height through HD camera images. The techniques developed and implemented during this project are not isolated and can be combined to achieve better results in the future.

5.1 Findings

The methodology encompassed exploratory data analysis to correct, crop, and augment the images employing computer vision techniques. Lens distortion was properly implemented during data processing. Feature extraction was achieved through convolutional neural networks (CNNs). Time-series modeling was accomplished using recurrent neural networks (RNNs). Additionally, height fields were generated outside the cloud center using rapid atmospheric field transformation (RAFT). Optical flow geometry also involved employing methods such as Lucas-Kanade and TVL1 for pixel tracking on top of our own derived height formulas to adopt a non-predictive approach for estimating cloud top heights.

Comparative analysis of error for the modeling approaches for the 18th April FEGS video from 17:57:06 hours to 18:33:28 hours:

Modeling Approach	Lucas-Kanade	TV-L1	CNN–RNN
MAE	601.46	748.59	245.65
RMSE	804.94	1029.57	334.65

In addition to predicting the cloud top height, the methods were able to generate the height field for the entire image instead of just the center. The field stitching method performed greatly for combining the images to create a full height map for the flight.

5.2 Conclusions

Though Lucas-Kanade has a lower RMSE than TVL1, the TV-L1 algorithm is more reliable than Lucas-Kanade for cloud top height estimation using optical flow geometry methods due to its tracking capability for entire frames. Optical flow geometry methods are more reliable outside the center of the image across the height fields since they do not depend on ML-based training.

The ConvNext class of CNN models performs well for feature and gradient extraction from clouds, given the sparsity of features in the images captured.

The CNN–RNN model outperforms TV-L1 and Lucas-Kanade + derived height formulas in terms of MAE and RMSE for the center predictions. However, it captures less variability in the heights than the optical flow methods.

MiDaS, which is a pre-trained model with proven performance for height estimation of urban images (indoor scenes such as offices, outdoor scenes such as streets and buildings, and human-centric scenes with people performing various activities), did not perform well. Clouds

are far too different from anything in its training domain, and we do not currently possess the training data to fine-tune it.

The quality of the height fields created by the RAFT has been the best of all the deep learning models we've experimented with. While the motion fields it creates suffer from inaccuracies in scaling and depth estimation and require training additional parameters to get reasonable results, its performance has been surprisingly good given the challenges that cloud images present.

One standout observation is that much of the data contains images that do not contain enough features to properly train the model, or they are inundated with water droplets and blurry to the point of not being usable. This is a data problem, not a modeling one, and likely requires human intervention to curate the data.

The final field stitching implementation is agnostic and can be used in combination with any prediction model.

5.3 Implications

Our findings and inferences introduce new modeling approaches for cloud top height estimation. NASA can use this research to augment their capabilities of cloud top height estimation and 3D height field generation of clouds. This advancement will help NASA and other space & weather agencies to obtain precise measurements of the Earth's atmosphere and gain a deeper understanding of clouds. This solution can potentially solve a key aspect of extreme weather forecasting. Since the solution relies majorly on the video captured by the HD camera aboard FEGS ER2 missions, it can save significant costs and reduce dependence on complex apparatus.

5.4 Future Work

Some parts of the ER2 mission flights were through fully cloudy skies for long periods of time where clouds appeared more uniform. The feature extraction was relatively less evident in these instances. Future research can focus on trying techniques to enhance the image augmentation for such images.

Surprisingly, the optical flow geometry method using Lucas-Kande was found to have a better mean absolute error (MAE) than TV-L1, which is generally accepted to be the more accurate tracking algorithm. As discussed, there were per-pixel distance accumulation challenges with TV-L1 that were not present for the sparse optical flow Lucas-Kanade algorithm. Future work would include revisiting the pixel distance accumulation across the selected video segment. to have potentially more accurate estimations. Also, in general, better methods of identifying failed tracking should be explored to avoid extreme predicted height outliers.

Despite having a greater MAE than the later CNN-RNN model, the optical flow geometry calculations described in section 4.1 were found to better capture variability in the cloud top heights. To improve MAE for this promising method, it would be worthwhile to replace Lucas-Kanade or TV-L1 with one of the latest optical flow methodologies such as RAFT to see if it would perform better.

The CNN–RNN model was found to capture relatively less variability in the cloud top heights. Future work can attempt to train the model using different transformations of the cloud top height, like squared transformation, to allow the model to capture the slight variability.

A lot of the images from the FEGS video have condensation on the lens, and this leads to noise in the image. Future research can attempt to tackle condensation on the lens dynamically. In the RAFT system, a more promising approach for stitched droplet removal

would be to perform droplet detection directly on the raw image sequences rather than their derived motion fields. This method would look for pixel patterns that remain identical across frames, as water droplets should appear as nearly identical dark or distorted regions in the same position across multiple images. By working with the raw imagery, the system could avoid the additional complexity and potential inconsistencies introduced by motion field estimation, focusing instead on the direct visual signature of water droplets on the lens. For the optical geometry approach, backwards tracking work should be done to see if stationary droplet voids can be backfilled with later frames where the cloud pixels are no longer obscured.

For all the models, some of the outlier predictions were found to be points where the plane appears to begin or end a turn, or otherwise have a significant wobble possibly due to turbulence, that was missed by the initial EDA to remove turning frames. More work can be done to revisit the root cause of the significant camera movement and refine the identification and dropping of these frames.

As mentioned in the conclusion, human curation of the dataset would likely improve the performance of all the models. This is, of course, time-consuming; there are more than 68,000 sequences, so we would likely need to scrub the videos and note the timestamps where the clearest images can be obtained.

References

Bedka, S., Feltz, W., Schreiner, A., & Holz, R. (2007). Satellite-derived cloud top pressure product validation using aircraft-based cloud physics lidar data from the ATReC field campaign. *International Journal of Remote Sensing, 28*(10), 2221–2239. https://doi.org/10.1080/01431160500391965

Borisov, M., Krinitskiy, M., & Tilinina, N. (2024). Estimating cloud base height from all-sky imagery using artificial neural networks. *Moscow University Physics Bulletin, 78*, S85–S95. https://doi.org/10.3103/S0027134923070020

Hadji Theophanous, S., Ttofis, C., Georghiades, A., & Theocharides, T. (2010). Towards hardware stereoscopic 3D reconstruction: A real-time FPGA computation of the disparity map. In *2010 Design, Automation & Test in Europe Conference & Exhibition (DATE 2010)* (pp. 1743–1748). IEEE. https://doi.org/10.1109/DATE.2010.5457096

Hasler, A. (1981). Stereographic observations from geosynchronous satellites: An important new tool for the atmospheric sciences. *Bulletin of the American Meteorological Society, 62*(2), 194–212. https://doi.org/10.1175/1520-0477(1981)062<0194:SOFGSA>2.0.CO;2

McGill, M., Hlavka, D., Hart, W., Scott, V., Spinhirne, J., & Schmid, B. (2002). Cloud Physics Lidar: Instrument description and initial measurement results. *Applied Optics, 41*(18), 3725–3734. https://doi.org/10.1364/AO.41.003725

Mussa, R., & Pallotta, J. (2022). Measuring cloud base height and cloud coverage using elastic multiangle lidars at Pierre Auger Observatory. *Journal of Physics: Conference Series, 2398*, 012018. https://doi.org/10.1088/1742-6596/2398/1/012018

Nied, J., Jones, M., Seaman, S., Shingler, T., Hair, J., Cairns, B., Van Gilst, D., Bucholtz, A., Schmidt, S., Chellappan, S., Zuidema, P., van Diedenhoven, B., Sorooshian, A., & Stamnes, S. (2023). A cloud detection neural network for above-aircraft clouds using airborne cameras. *Frontiers in Remote Sensing, 4*, 1118745. https://doi.org/10.3389/frsen.2023.1118745

Vaughan, M., Liu, Z., McGill, M. J., Hu, Y., & Obland, M. (2010). On the spectral dependence of backscatter from cirrus clouds: Assessing CALIOP's 1064 nm calibration assumptions using cloud physics lidar measurements. *Journal of Geophysical Research: Atmospheres, 115*, D00H24. https://doi.org/10.1029/2009JD013086

Yu, H., Ma, J., Ahmad, S., Sun, X., Li, Z., & Hong, Y. (2019). Three-dimensional cloud structure reconstruction from the Directional Polarimetric Camera. *Remote Sensing, 11*(24), 2894. https://doi.org/10.3390/rs11242894

Zach, C., Pock, T., & Bischof, H. (2007). A duality based approach for realtime TV-L1 optical flow. In F. A. Hamprecht, C. Schnörr, & B. Jähne (Eds.), *Pattern Recognition. DAGM 2007. Lecture Notes in Computer Science (Vol. 4713, pp. 214–223)*. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-540-74936-3_22

Lucas, B. D., & Kanade, T. (1981). An image registration technique with an application to stereo vision. In *Proceedings of the Image Understanding Workshop* (pp. 121–130). Arlington, VA: DARPA.

Birkl, R., Wofk, D., & Müller, M. (2023, July 26). *MiDaS v3.1: A model zoo for robust monocular relative depth estimation.* Intel Labs.

Teed, Z., & Deng, J. (2020). RAFT: Recurrent all-pairs field transforms for optical flow. *arXiv*. https://doi.org/10.48550/arXiv.2003.12039

Liu, Z., Mao, H., Wu, C.-Y., Feichtenhofer, C., Darrell, T., & Xie, S. (2022). A ConvNet for the 2020s. *arXiv:2201.03545 [cs.CV]*. https://arxiv.org/abs/2201.03545

Appendix 1.

Fisheye Correction for Lens Distortion

The lens used by NASA in the FEGS ER2 mission is a 1⁄2" Format CMount Fisheye Lens 1.4mm FL. It is essential to correct the fisheye distortion before processing the image further. A fisheye lens is an ultra-wide-angle lens used to attain a larger field of view but, in the process, produces strong visual distortion intended to create a wide panoramic image. These distortions can lead to problems during the modeling phase. During pre-processing and augmentation, this could lead to non-linear transformations. During feature extraction using CNN, this may lead to incorrect deductions about the spatial features in the cloud.

We used the following parameters from lens specifications, derivations, and script to correct the fisheye distortion:

$f = 1.4$ mm (focal length of lens)

$\mu = 2.8 \times 10^{-3}$ mm (pixel pitch)

$s = 2$ (Image scale factor)

$D = [\,0.01166363,\ -0.04819808,\ 0.07918044,\ -0.037572\,]$ (Distortion coefficients)

$H = 1920$ (Height in pixels)

$W = 1080$ (Width in pixels)

$$c_x = \frac{W-1}{2} \quad ; \quad c_y = \frac{H-1}{2}$$ (Image center coordinates)

$$K = \begin{bmatrix} \frac{f}{\mu} & 0 & c_x \\ 0 & \frac{f}{\mu} & c_y \\ 0 & 0 & 1 \end{bmatrix}$$ (Camera intrinsic matrix)

$$c_{px} = \frac{W \cdot s - 1}{2} \quad ; \quad c_{py} = \frac{H \cdot s - 1}{2}$$

$$P = \begin{bmatrix} \frac{f}{\mu} & 0 & c_{px} \\ 0 & \frac{f}{\mu} & c_{py} \\ 0 & 0 & 1 \end{bmatrix}$$

$$R = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$ (Rectification matrix)

Finally the image was undistorted using a combination of two functions from the cv2 library:

1. `cv2.fisheye.initUndistortRectifyMap`
    a. It computes undistortion and rectification maps for image transformation.

b. `map1, map2 = cv2.fisheye.initUndistortRectifyMap(K=K, D=coeffs, R=R, P=P, size=[W * S, H * S], m1type=cv2.CV_16SC2)`

2. `Cv2.remap`

   a. It applies a generic geometrical transformation to an image.

   b. The function remap transforms the source image using the specified map obtained from the above function's output:

      `dst(x,y)=src(mapx(x,y),mapy(x,y))`

   c. `undistorted_image = cv2.remap(distorted_image, map1, map2, interpolation=cv2.INTER_LINEAR, borderMode=cv2.BORDER_TRANSPARENT)`

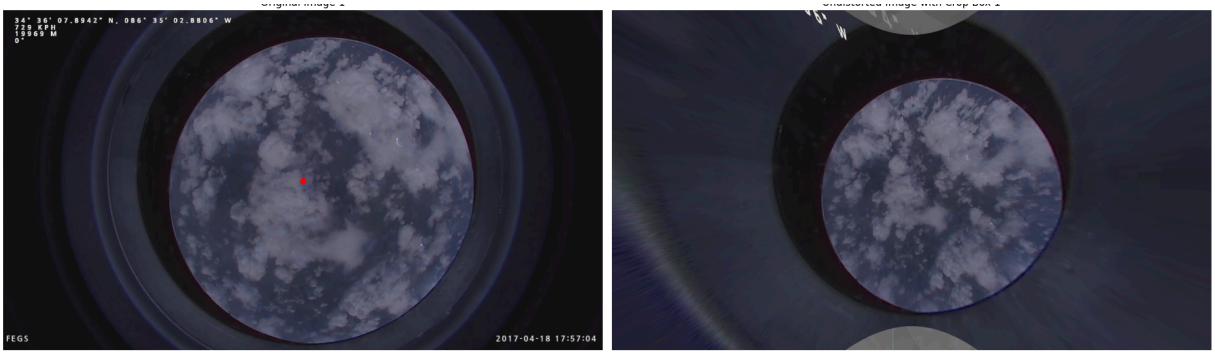


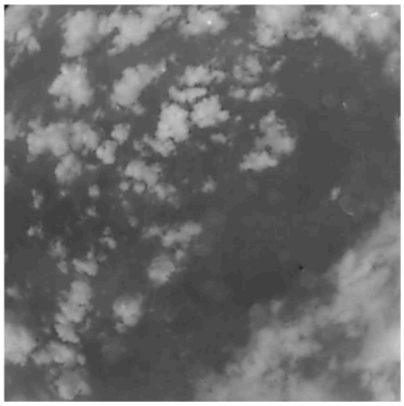
Figure 40. Illustration of the original image and the fisheye-corrected image.

Appendix 2.

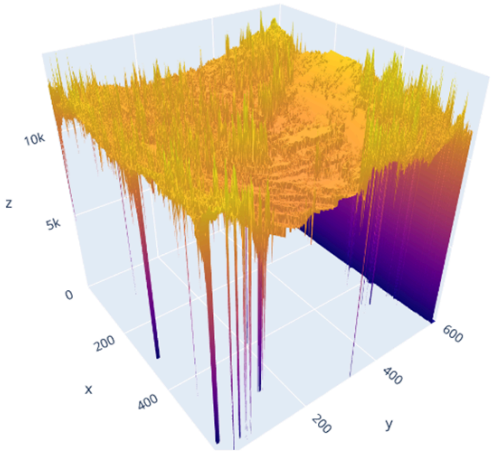
Optical Flow Results Examples

18th April FEGS video 18:01:15

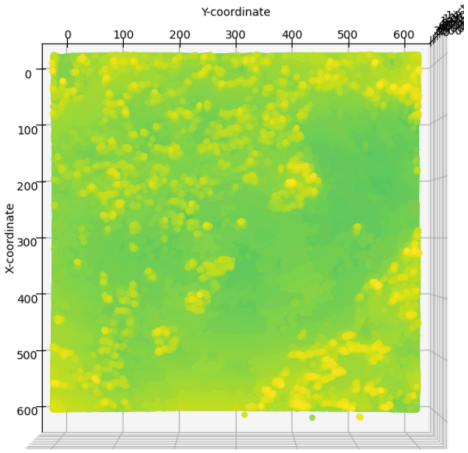
Original Image



Cropped Image (600X600)



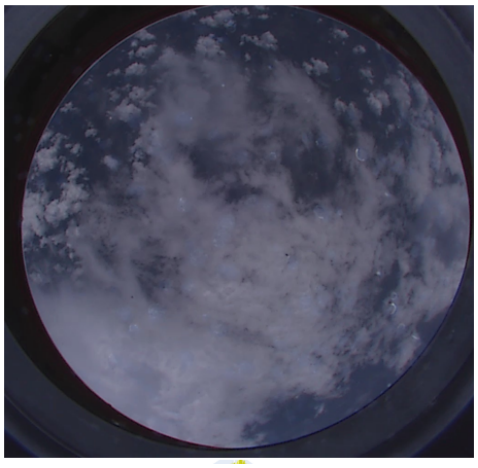
Side View of Cloud Height Plot



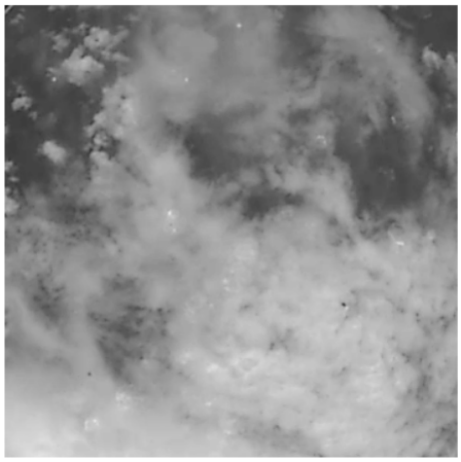
Top View of Cloud Height Plot

Figure 41. Optical flow example results

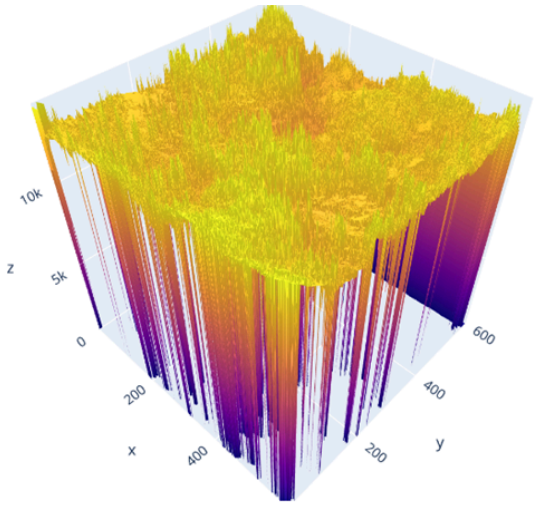
18th April FEGS video 18:24:00



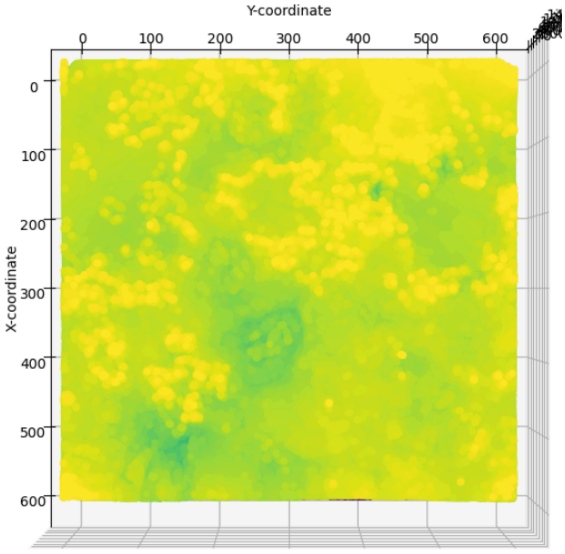
Original Image



Cropped Image (600X600)



Side View of Cloud Height Plot

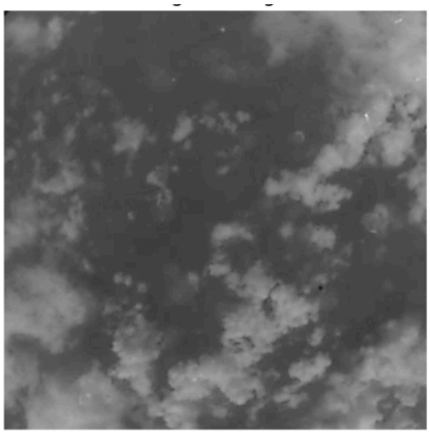


Top View of Cloud Height Plot

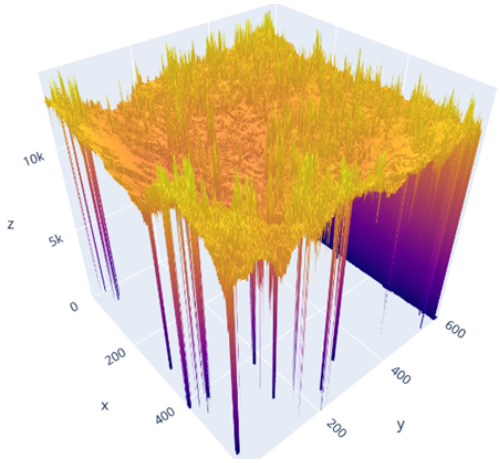
Figure 42. Optical flow example results

18th April FEGS video 17:59:09

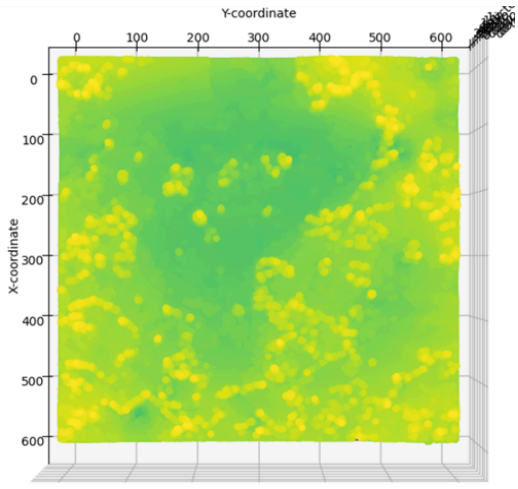
Original Image



Cropped Image (600x600)



Side View of Cloud Height Plot



Top View of Cloud Height Plot

Figure 43. Optical flow example results

Appendix 3.

Fisheye Correction Test with Lucas-Kanade Tracking

As a visual test that the fish-eye lens correction was working properly, we plotted Lucas-Kanade tracking on a sample original test dataset video segment and then again on the same segment but fisheye corrected.

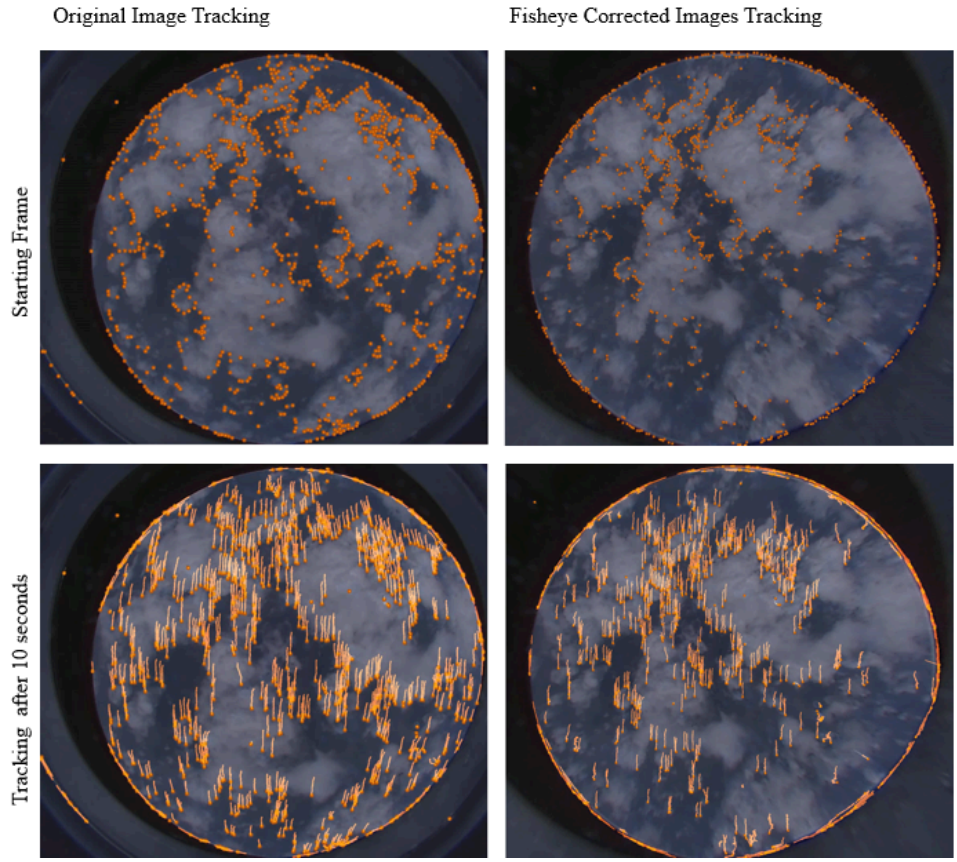


Figure 44. Original Image Tracking and Fisheye Corrected Images Starting Frame VS 10 Seconds

The left-hand side shows Lucas-Kanade tracking on the entire viewing window outside the Fly's Eye. You can see the camera lens fisheye distortion by the increasingly curved pixel tracking further out from the center. The right-hand side shows the same tracking paths are straight, which implies successful fisheye correction. The clouds themselves appear flattened and less globe-like as well. The top images are the starting frames, and the bottom shows lines after 10 seconds of tracking.

As expected, the left-hand image shows the cloud tracks as curved on the edges, whereas on the right-hand side, we can see that the tracks have straightened post-correction. Since we placed greater importance on the x-axis (image vertical direction) pixel movement, fisheye correction was especially important for accurate height calculations.