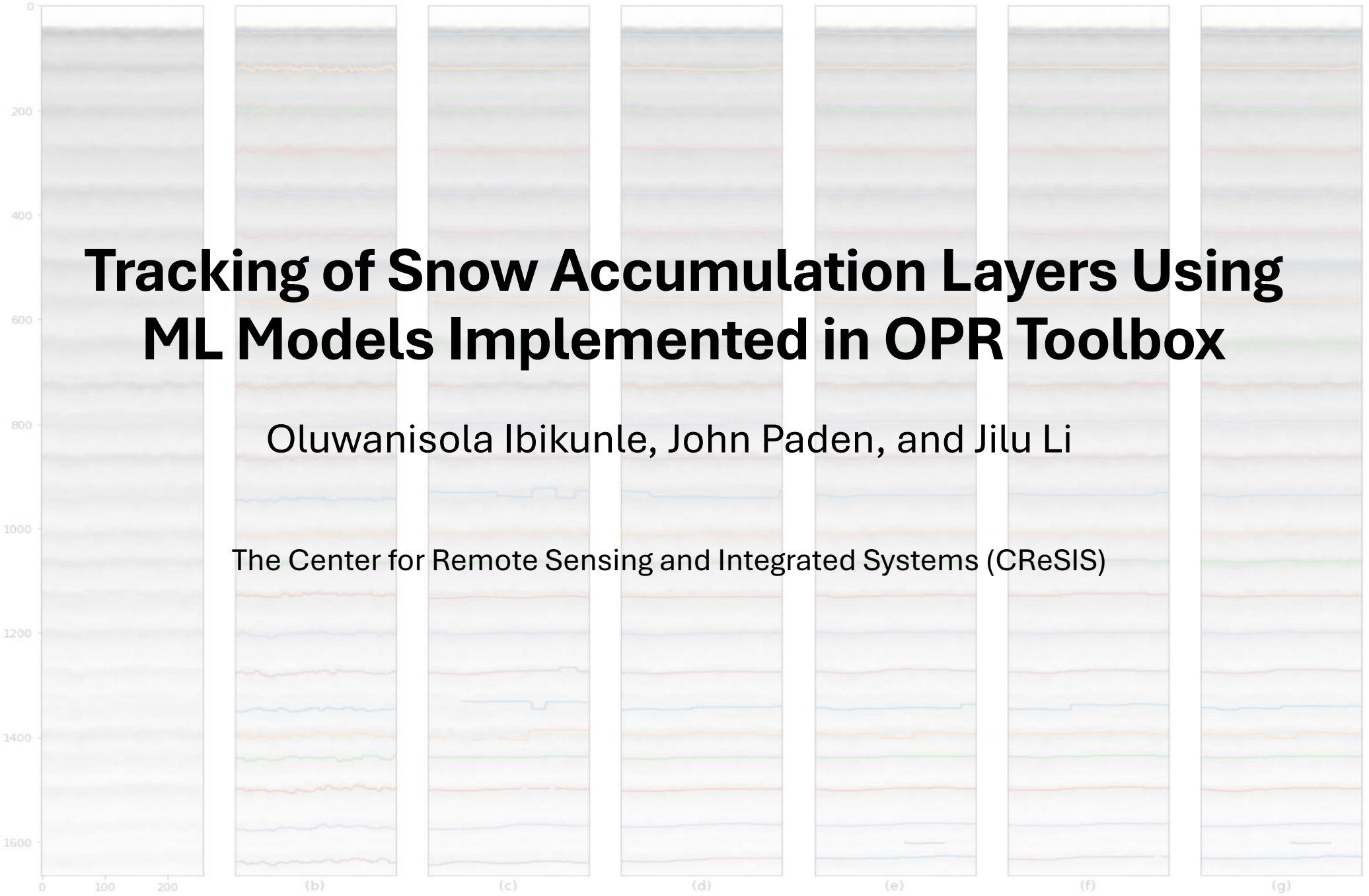


Tracking of Snow Accumulation Layers Using ML Models Implemented in OPR Toolbox

Oluwanisola Ibikunle, John Paden, and Jilu Li

The Center for Remote Sensing and Integrated Systems (CReSIS)

1. Overview

- Snow and ice layers of ice sheets and glaciers in radar data contain critical information of climate and ice sheet and glacier dynamics and mass balance.
- Tracking these layers in big data at ice sheet scale is a huge workload and challenging.
- Traditional manual and semi-automated methods are not efficient and expensive.
- Machine Learning (ML) algorithms have emerged as the efficient ways for fully automatic tracking of multiple snow and ice layers.
- NSF supported project BIGDATA: IA: Collaborative Research: Intelligent Solutions for Navigating Big Data from the Arctic and Antarctic (09/01/2018-08/31/2022: Maryam Rahnemoonfar and John Paden)
- CReSIS has compiled an AI-ready training dataset and implemented ML models in OPR toolbox for tracking snow layers (Oluwanisola Ibikunle).

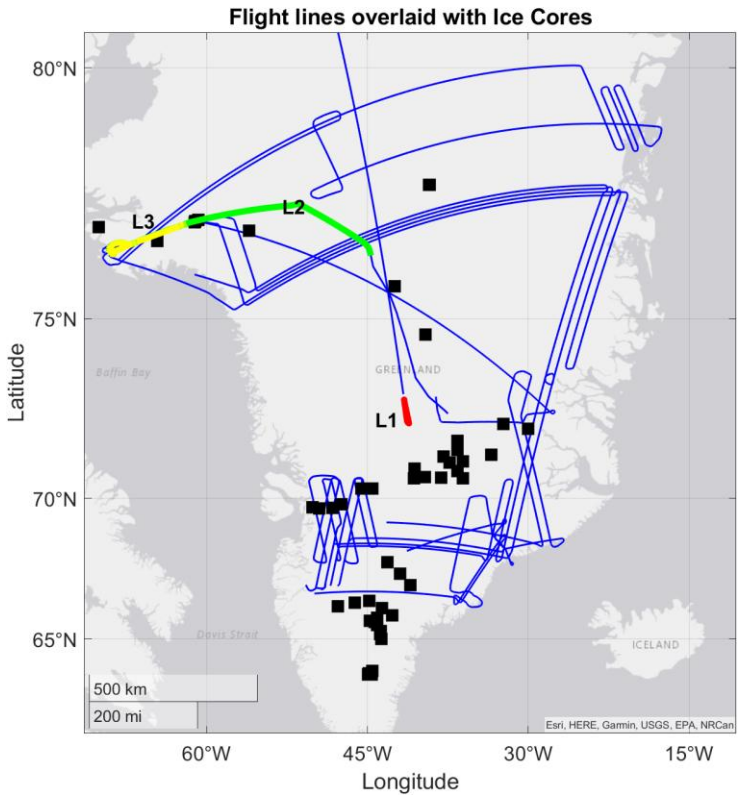


Fig. 1 An AI-ready training dataset

https://data.cresis.ku.edu/data/temp/bigdata/NASA_OIB_test_files/image_files/Dataset/snow/SR_Dataset_v1/

2. ML_tracker setup

- ML tracker Location in OPR toolbox: ~/scripts/opr/python/ML_tracker
- ML models were developed using Python 3.9, required packages to be installed include:

  Anaconda or miniconda

  TensorFlow

  PyTorch
- Small_requirements.txt in ML_tracker folder include a full list of required packages and their versions.
- Follow the step-by-step setup instructions on OPR wiki page:

  https://gitlab.com/openpolarradar/opr/-/wikis/ML-Tracker-setup

3. Test of virtual python environment setup

>python -m test_installation
Installation works perfectly!
Echogram layers have been tracked

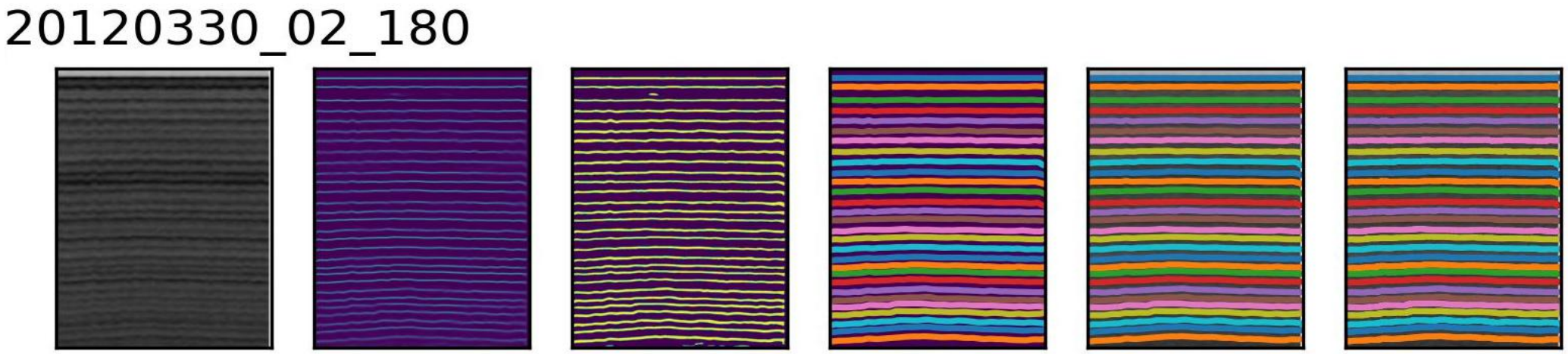


Fig. 2 Image from setup testing by running test_installation.py using sample_echogram.mat

4. How to run ML_tacker

- The trained ML models are accessed by executing python scripts (return_tracked_layers_argp.py & return_few_layers_argp.py) in return_ML_layers.m

- To ML tracker is run with the following settings in run_layer_tracker.m

  track.method = 'ML' *% hard coded*

  track.layer_names = {'ML_layer'} % user changeable

  param_override.layer_tracker.echogram_source = 'CSARP_post/qlook'; *% user changeable*

  param_override.layer_tracker.layer_params(layer_idx).layerdata_source = 'CSARP_post/layer'; *% user changeable*

  param_override.layer_tracker.layer_params(layer_idx).layerdata_destination = '~/ML_prediction'

  % hard coded, the location for saving the layer files with tracked layers by ML tracker. If it was not setor empty, the source layer files will be updated.

- The following functions will be called in sequence:

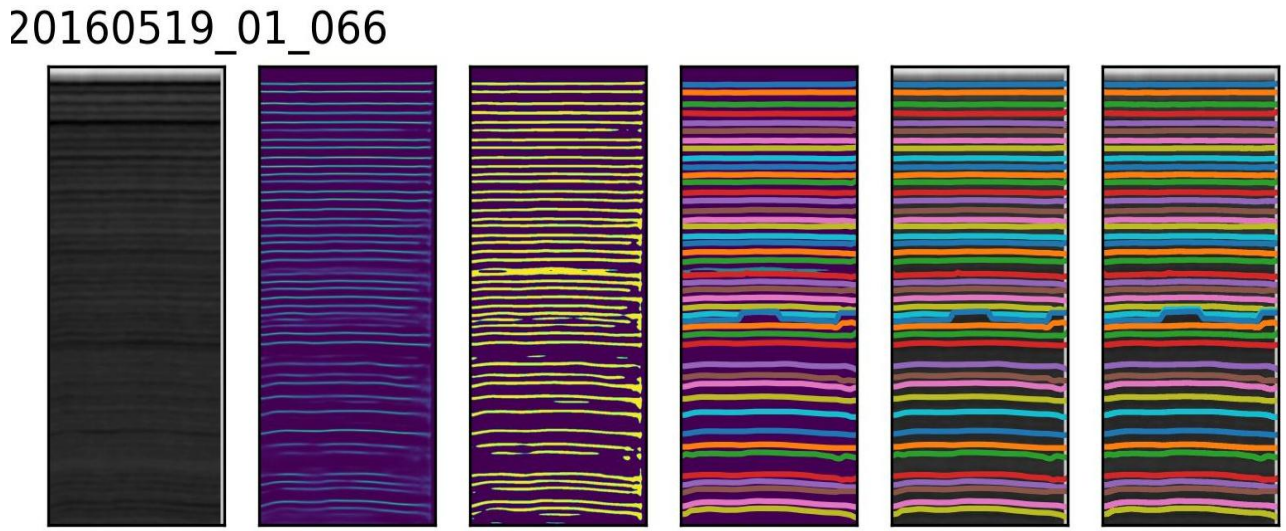
  run_layer_tracker.m, layer_tracker.m, layer_tracker_task.m

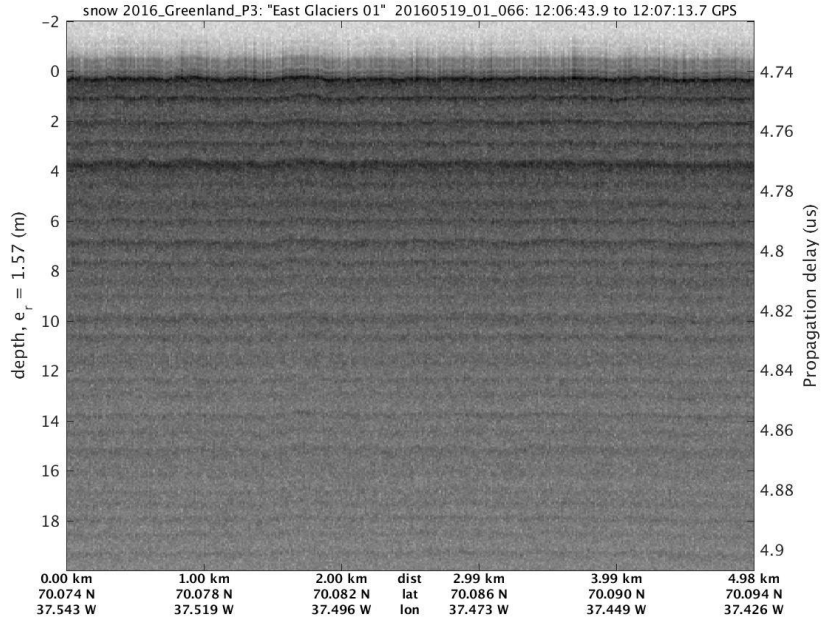
  return_ML_layers.m, return_tracked_layers_argp.py, return_few_layers_argp.py

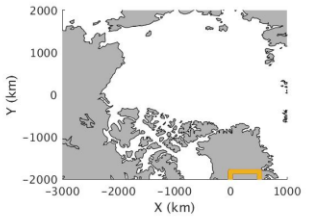
  layer_tracker_combine_task.m, runOpsCopyLayers.m

5. Sample tracking results

- Greenland







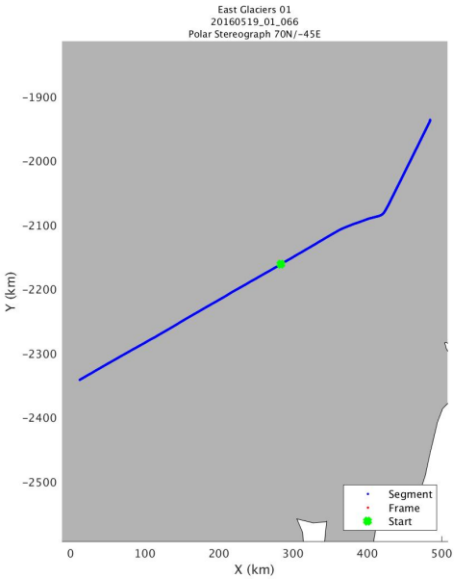
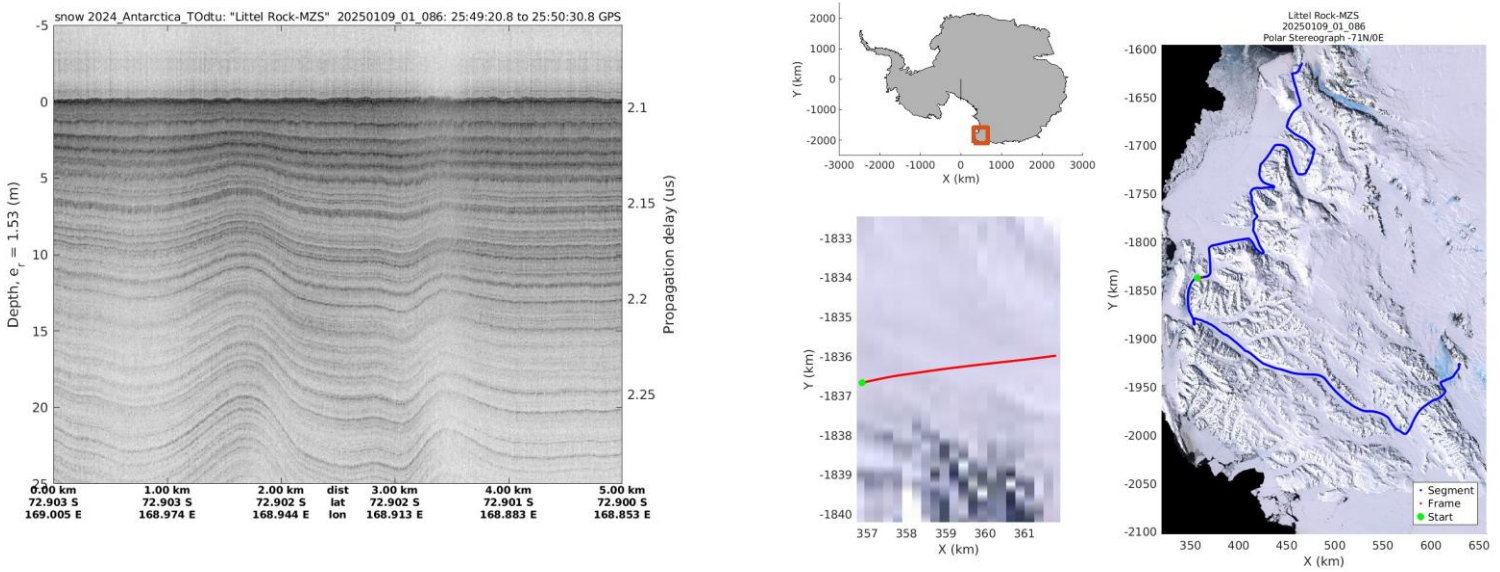


Fig. 3 Sample tracking result from Greenland data in 2016. From left to right, each panel corresponds to 1) input image; 2) probability map; 3) lowest threshold filtered image; 4) sorted label matrix; 5) tracked layers; 6) final filtered tracked layers

- Antarctica



20250109_01_086

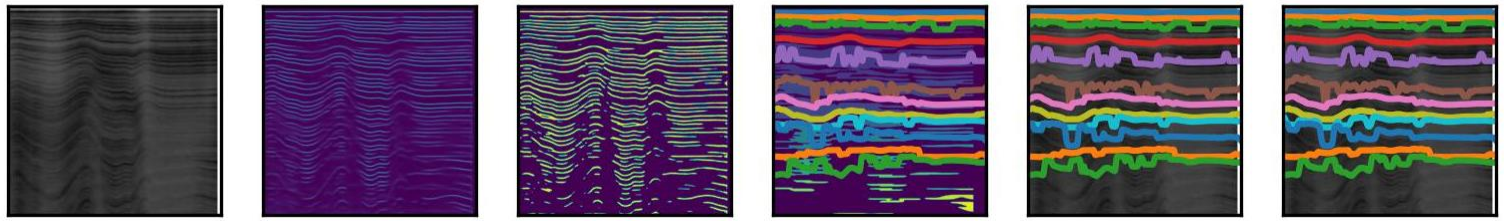


Fig. 4 sample tracking result from Antarctica 2025

5. Future work

- Expand the AI training dataset to include Antarctica data and other regions like Alaska.

- Improve the ML models with the expanded training dataset.

- Archive the training setup and steps so that the OPR toolbox community can contribute to improving the ML tracker.