



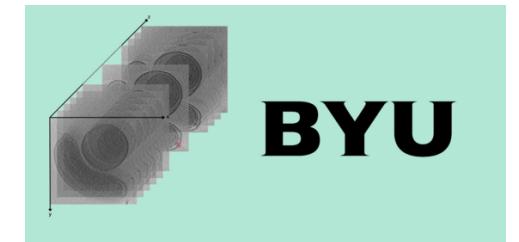
# Internship Project Report

Matthew Ward

August 4, 2025

# Introduction

- BYU Applied and Computational Math (DS & ML, April 2026)
- Biophysics Simulation Group—computer vision and competition dataset curation with 3D pictures of bacteria (cryo-ET tomograms)
- Data engineer intern with you until August 16<sup>th</sup>!



Fast, Efficient  
3D Imaging

# Overview of projects

- 1. Generate a processing performance report**
- 2. Improve registration pipeline with pose graph optimization**
- 3. Model laser spot illumination as a Gaussian from sensor data**

# Processing Performance Report

# Processing Performance Report

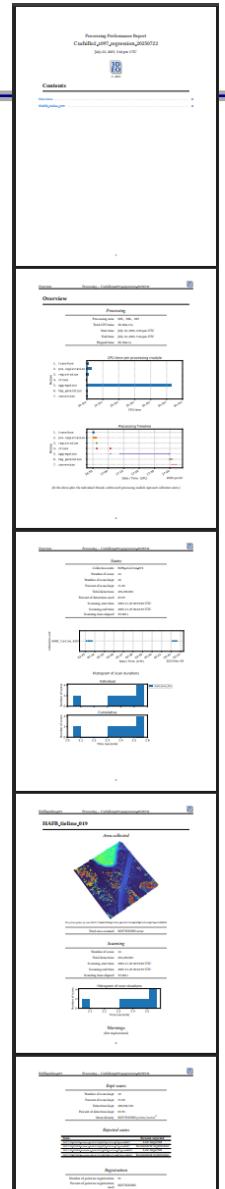
**Not to be confused with sensor performance report!**

# Processing Performance Report

**Not to be confused with sensor performance report!**

**A PDF report summarizing key insights from processing.**

**Located in <acadia-output-directory>/qc/processing\_report after processing is complete**

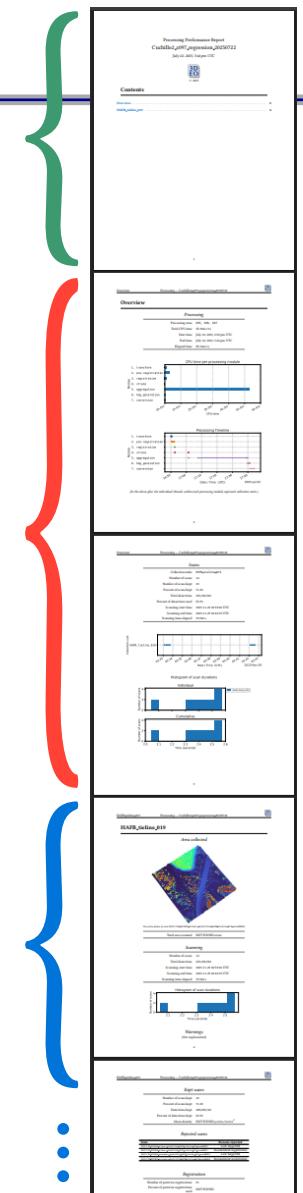


# Processing Performance Report

3D  
EO

## Report Format

- **Contents**
- **Overview** (information about all tiles together)
- **Tile 1** (information specific to this tile)
- **Tile 2**
- :
- **Tile *n***



# Processing Report Generation Process

## Processing directory

say, albert:/shares/processed/cuchillo/flightData/FlatCreek



## JSON of filepaths and statistics

to be used in report



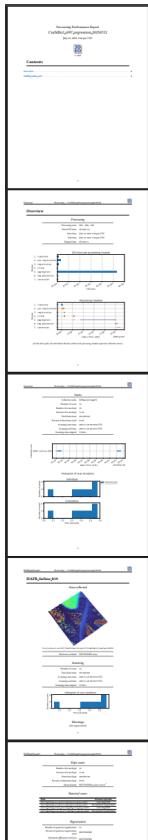
## LATEX report



## PDF report

# Example Processing Reports

## Single Tile Report



6 pages

## Mapping Report (56 tiles)



203 pages

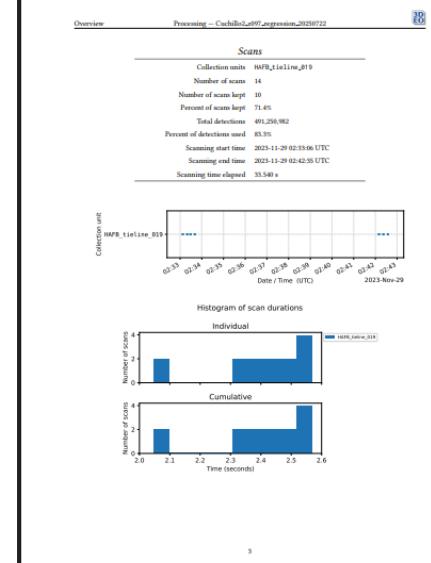
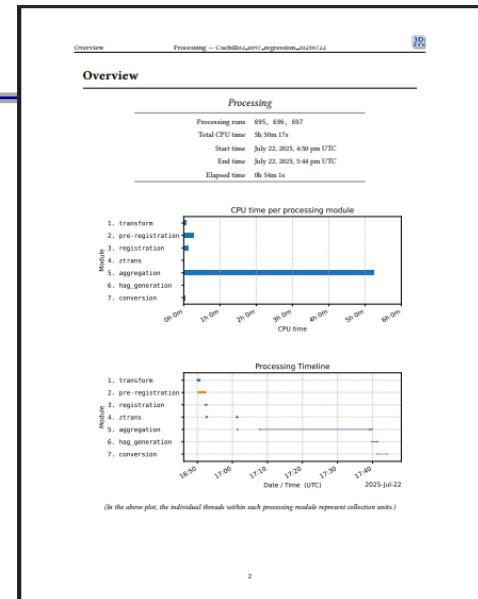
# Overview

- Processing summary**

How long processing took, CPU time, etc.

- Scan summary**

Number of detections, how long scanning took, etc.



# Processing Summary

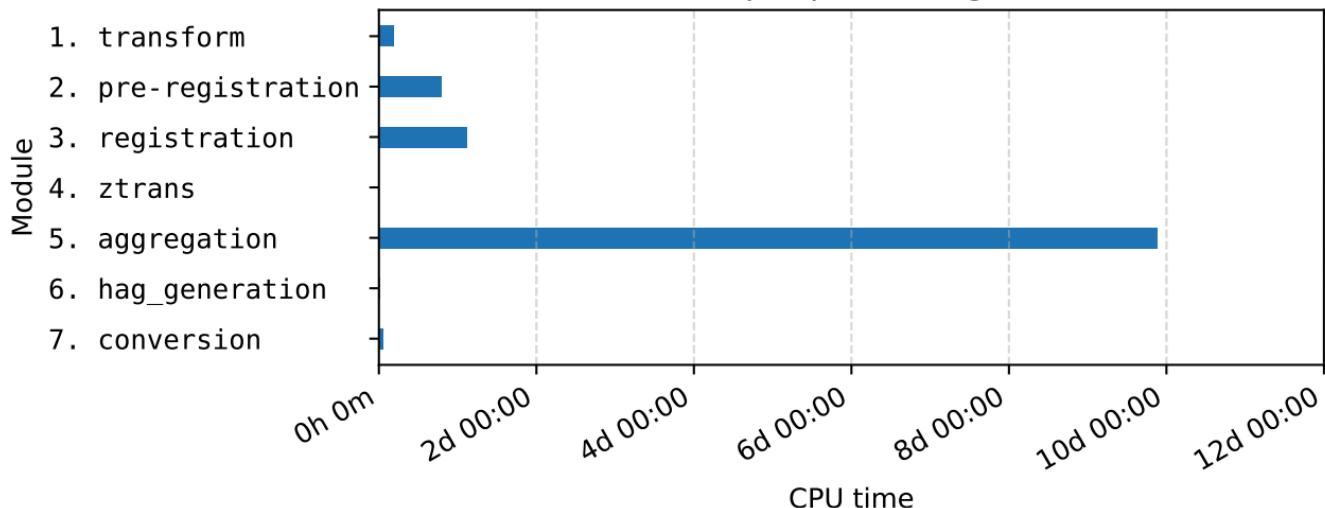
## Processing

---

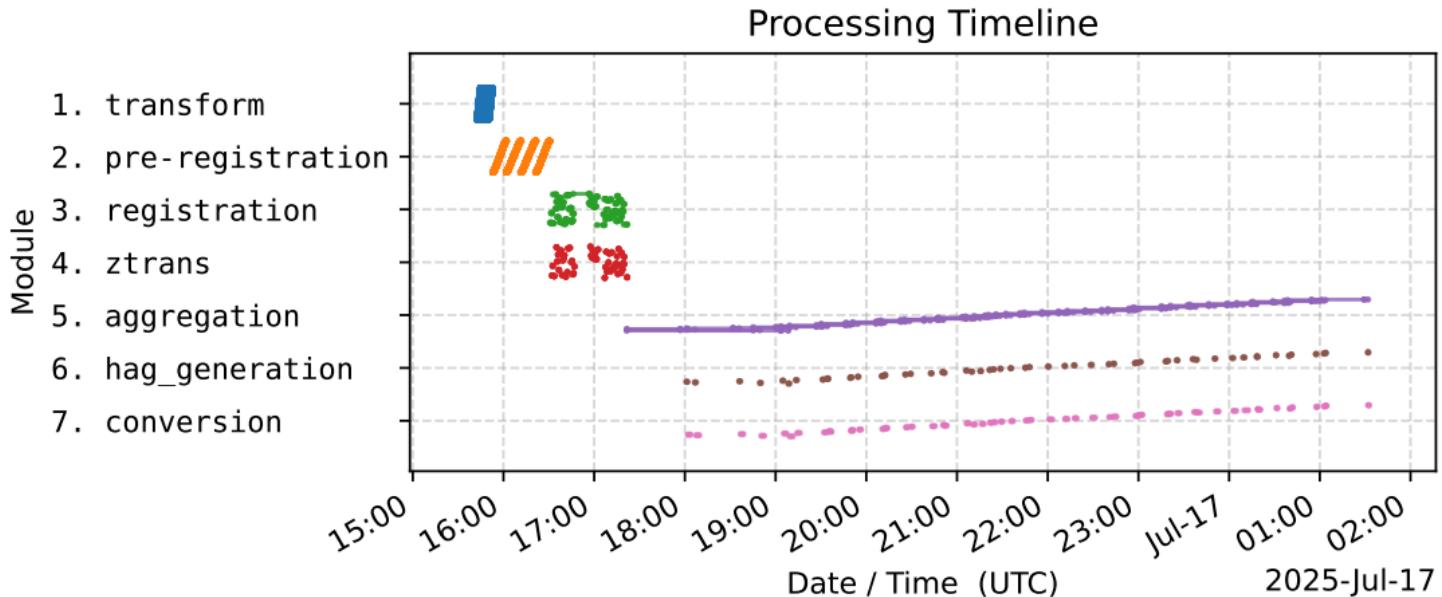
Processing runs	631
Total CPU time	12d 01:30:17
Start time	July 16, 2025, 7:42 pm UTC
End time	July 17, 2025, 5:32 am UTC
Elapsed time	9h 50m 11s

---

CPU time per processing module



# Processing Timeline



(In the above plot, the individual threads within each processing module represent collection units.)

# Per Tile Sections

- **Picture of the tile**
- **Scan information**
  - Number of detections, how long scanning took, etc.
- **Rejected data**
  - Reasons for rejection
- **Registration information**
  - To evaluate success of registration



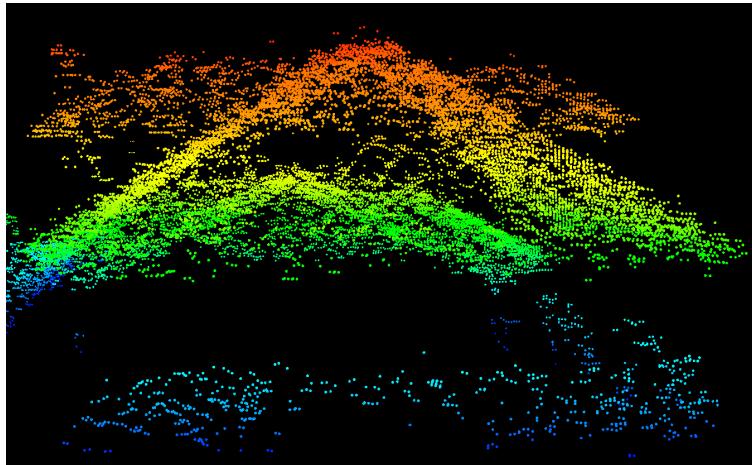
(HAFB\_tieline\_019 from Single Tile Report)

# Future Work

- Continue to refine wording and format for clarity
- Finish implementing fields
  - Scan density information
  - Report on more data rejection reasons
  - Registration success metrics
  - Warnings—for example, warn if the beam was dumping
- Get client feedback—what do they want to see in the report?
- Make  $\text{\LaTeX}$  compilation container smaller

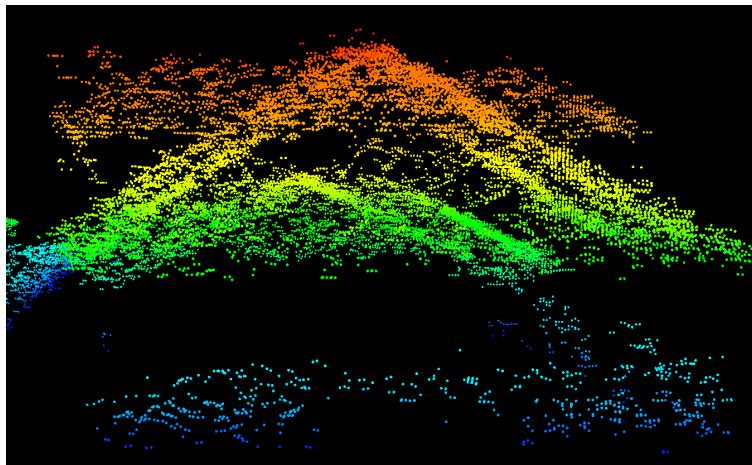
# Pose Graphs for Registration

**Good registration**



**Bad registration**

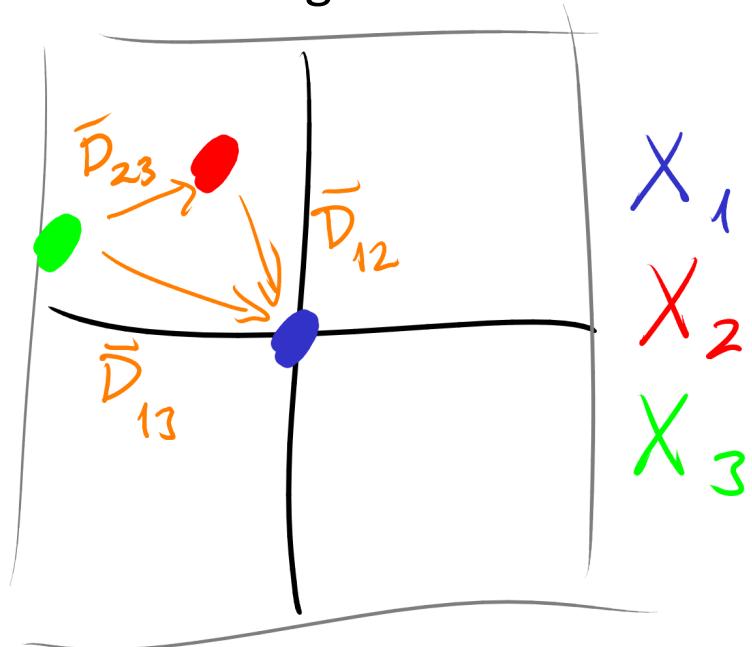
notice that the roof is doubled here



# Scan Poses

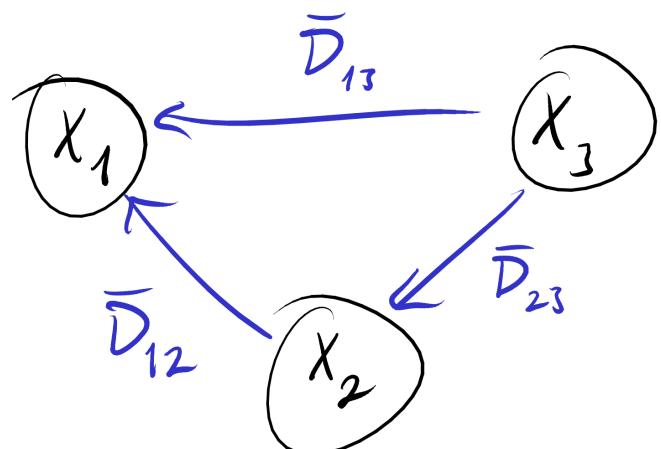


Starting Poses



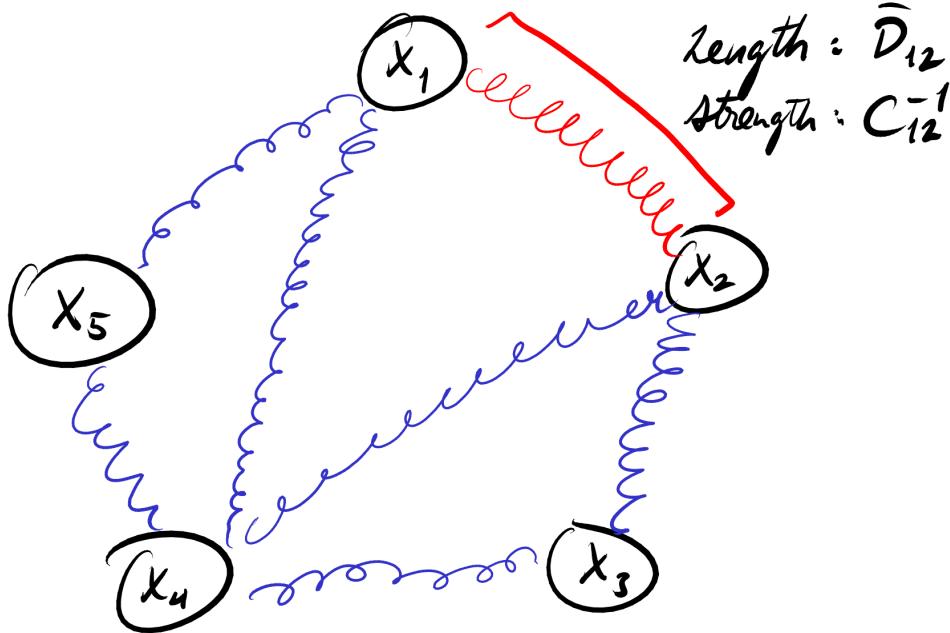
# Pose Graph

We have ways to move one scan to align with another, with some uncertainty. How can we move  $n$  scans to align with each other?

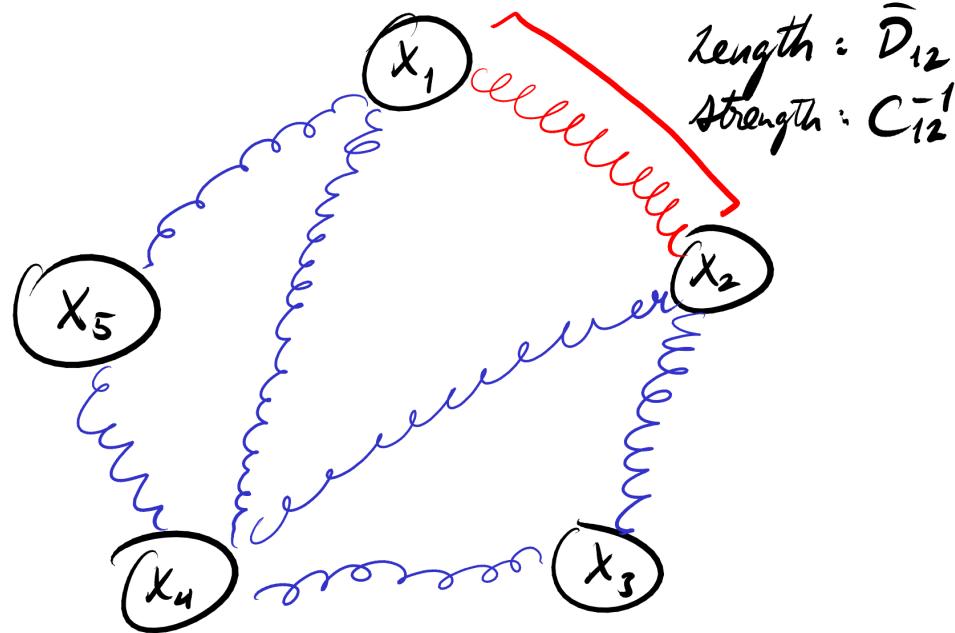


In practice,  $\bar{D}_{13} \neq \bar{D}_{12}\bar{D}_{23}$ .

# Pose Graph as Springs

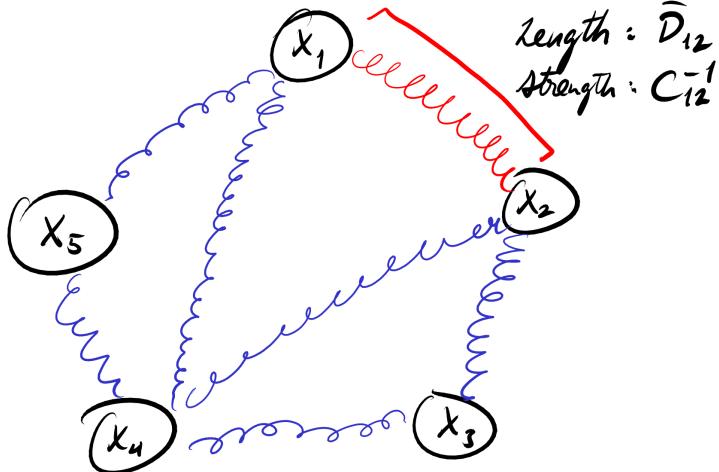


# Pose Graph as Springs



$$\underset{\{X_1, \dots, X_n\}}{\text{minimize}} \quad \sum_{i,j} \left( \bar{D}_{ij} - (X_i - X_j) \right)^T C_{ij}^{-1} \left( \bar{D}_{ij} - (X_i - X_j) \right)$$

# Pose Graph as Springs



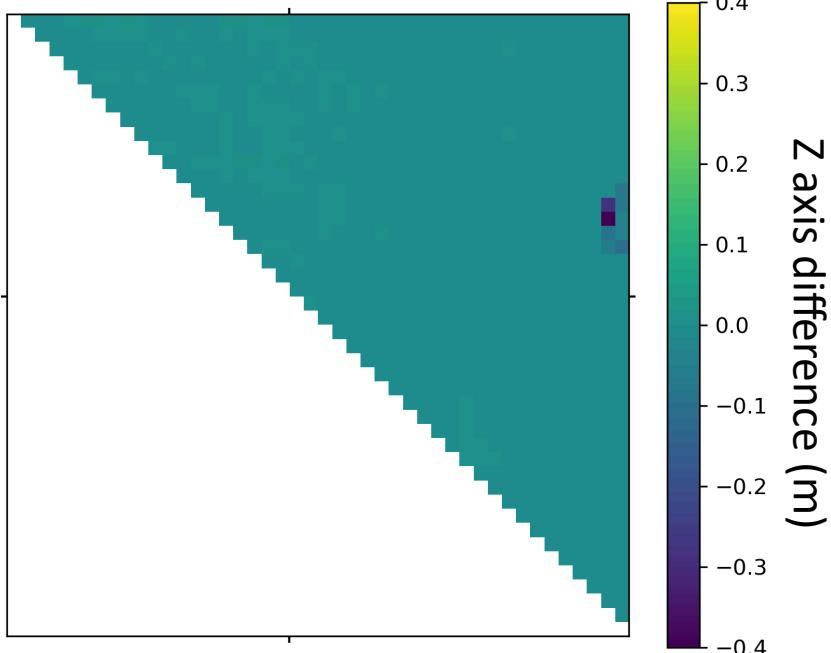
## Ways to Set Spring Elasticities (covariances)

- Constant elasticity
- Model elasticity using pairwise features  
 $(i, j) \rightarrow C_{ij}^{-1}$

$$\underset{\{X_1, \dots, X_n\}}{\text{minimize}} \quad \sum_{i,j} \left( \bar{D}_{ij} - (X_i - X_j) \right)^T C_{ij}^{-1} \left( \bar{D}_{ij} - (X_i - X_j) \right)$$

# Rooting out Bad Pairwise Registrations

Comparing pairwise and global registrations



- After optimization, some springs (pairwise registrations) are stretched
- Lots of redundancy in the graph reveals poor pairwise registrations
- Weight those springs less in optimization step or remove them entirely

# Lu-Milios Implementation

- Implemented registration optimizer using the closed-form least squares pose graph solution described in Lu and Milios' 1997 paper

# Lu-Milios Implementation

- Implemented registration optimizer using the closed-form least squares pose graph solution described in Lu and Milios' 1997 paper
- Parameters:
  - Expected translational error of 1 meter, 2 mrad  
(constant covariance of  $\text{diag}(1, 1, 1, 0.002^2, 0.002^2, 0.002^2)$ )
  - Prune edges (pairwise registrations) whose optimized Mahalanobis distance has z-score higher than 1.5 among all edges

# Lu-Milios Implementation

- Implemented registration optimizer using the closed-form least squares pose graph solution described in Lu and Milios' 1997 paper
- Parameters:
  - Expected translational error of 1 meter, 2 mrad  
(constant covariance of  $\text{diag}(1, 1, 1, 0.002^2, 0.002^2, 0.002^2)$ )
  - Prune edges (pairwise registrations) whose optimized Mahalanobis distance has z-score higher than 1.5 among all edges
- Note: Improper to perform linear least squares on Euler angles, but works alright since angles are very small (no more than a few mrad)

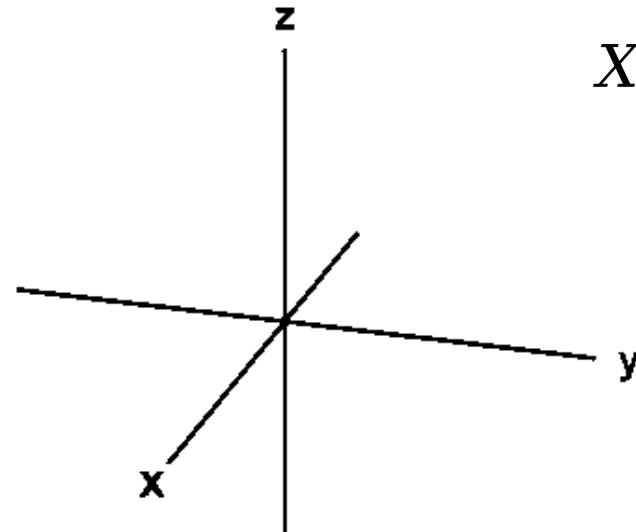
# Results from Lu-Milius Implementation

# Ongoing and Future Work

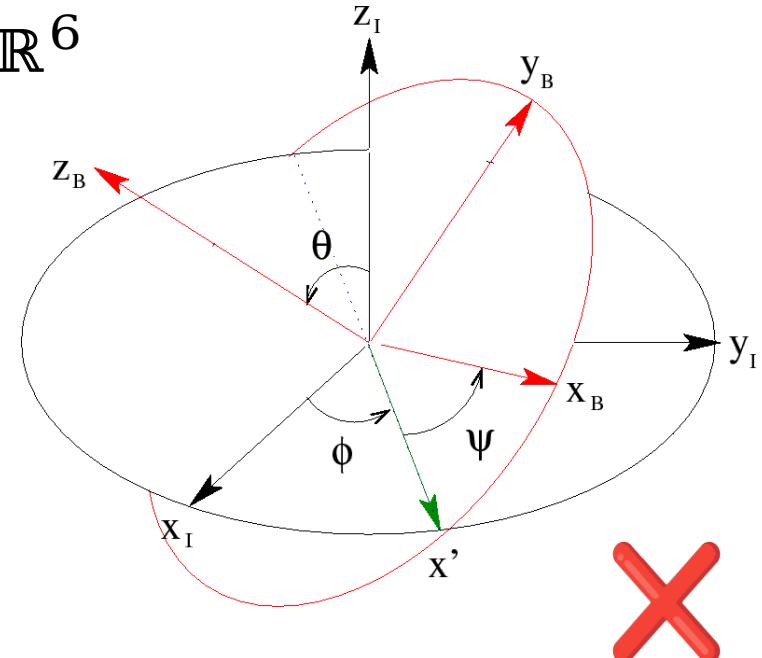
$$X_i = \begin{pmatrix} x_i \\ y_i \\ z_i \\ \theta_i \\ \phi_i \\ \psi_i \end{pmatrix} \in \mathbb{R}^6$$

$$\underset{\{X_1, \dots, X_n\}}{\text{minimize}} \quad \sum_{i,j} \left( \bar{D}_{ij} - (X_i - X_j) \right)^T C_{ij}^{-1} \left( \bar{D}_{ij} - (X_i - X_j) \right)$$

# Ongoing and Future Work



$$X_i = \begin{pmatrix} x_i \\ y_i \\ z_i \\ \theta_i \\ \phi_i \\ \psi_i \end{pmatrix} \in \mathbb{R}^6$$



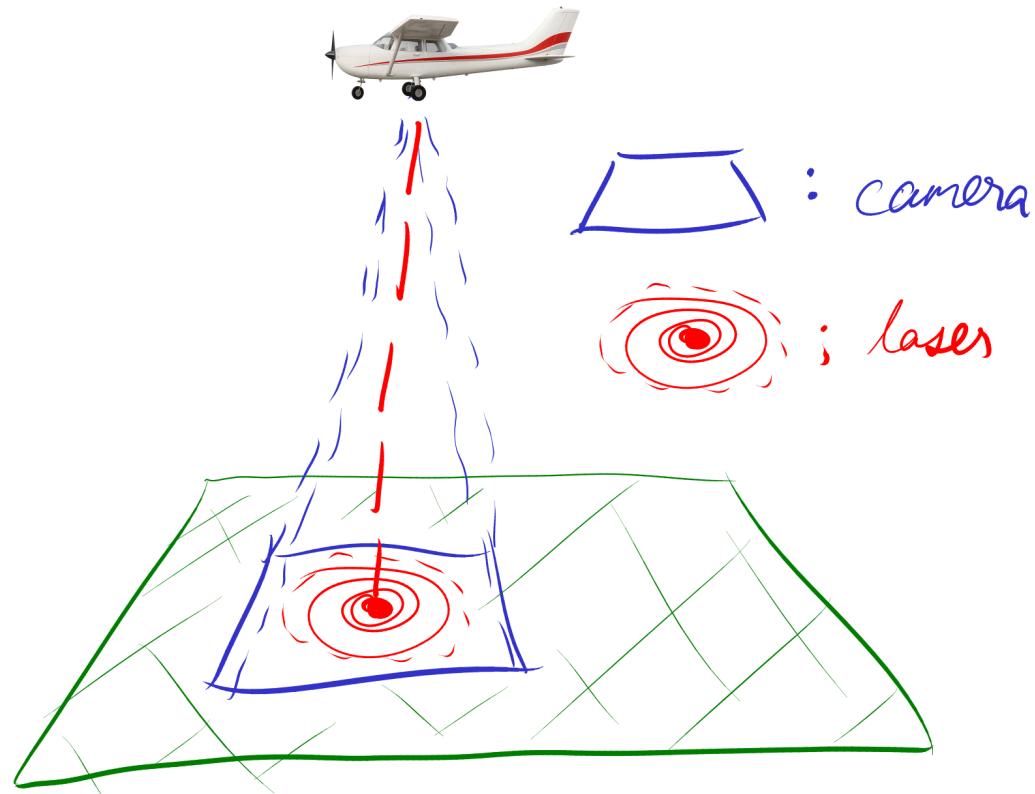
$$\underset{\{X_1, \dots, X_n\}}{\text{minimize}} \quad \sum_{i,j} \left( \bar{D}_{ij} - (X_i - X_j) \right)^T C_{ij}^{-1} \left( \bar{D}_{ij} - (X_i - X_j) \right)$$

# Ongoing and Future Work

- Use non-linear least squares optimization to better handle orientation
- [ lie algebra graph results ]
- Be smarter in choosing pairwise registration covariance (uncertainty)
  - Automatically determine covariance using properties of the sensor or the data
  - Use pairwise registration features to determine covariance

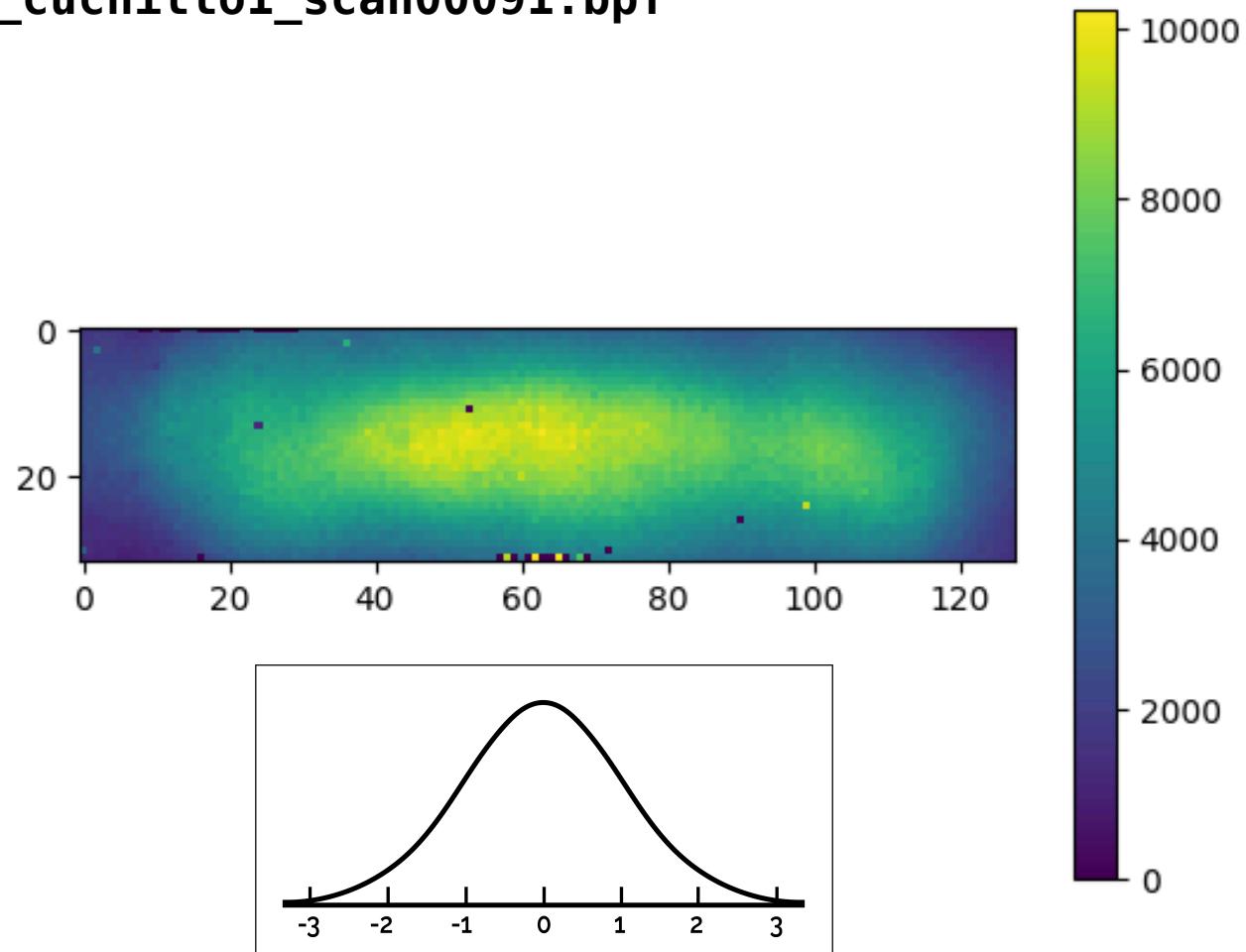
# Modeling the Laser Illumination Spot

# Laser Illumination Spot

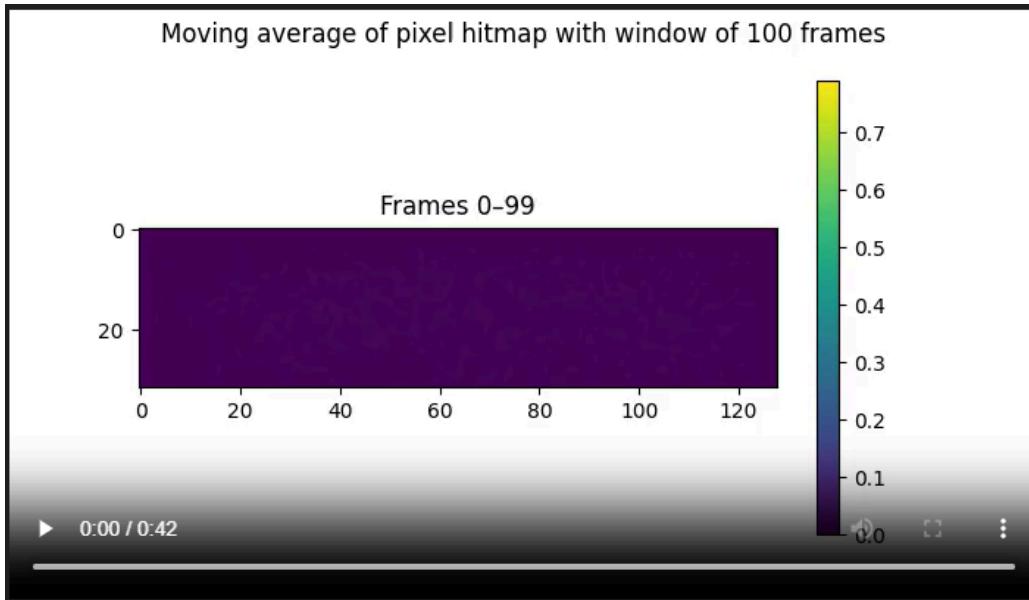


# Spot is Approximately Gaussian

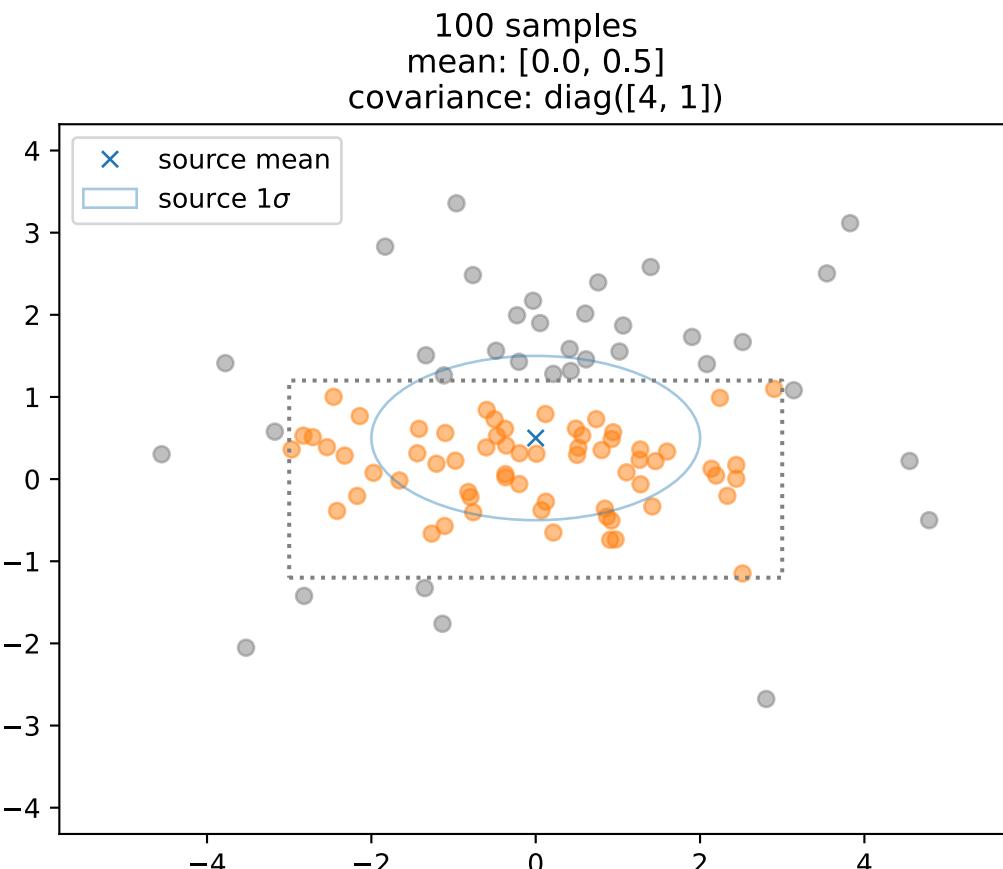
Total photon detections per pixel during  
**20230627\_095732\_cuchillo1\_scan00091.bpf**



# Cannot Assume the Spot Stays Still

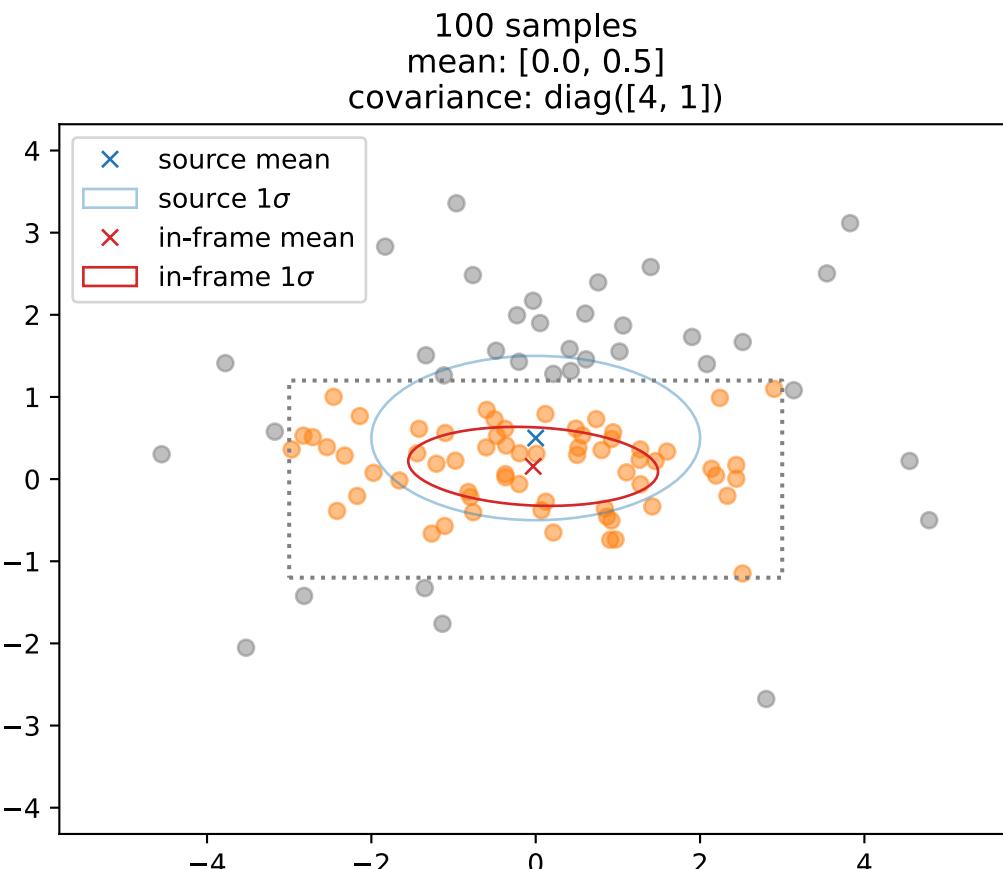


# Cannot Use Sample Mean and Covariance



- 100 random Gaussian samples representing photons, not all of which are in-frame
- Ring represents 1 standard deviation
- We only see the samples within the frame

# Cannot Use Sample Mean and Covariance



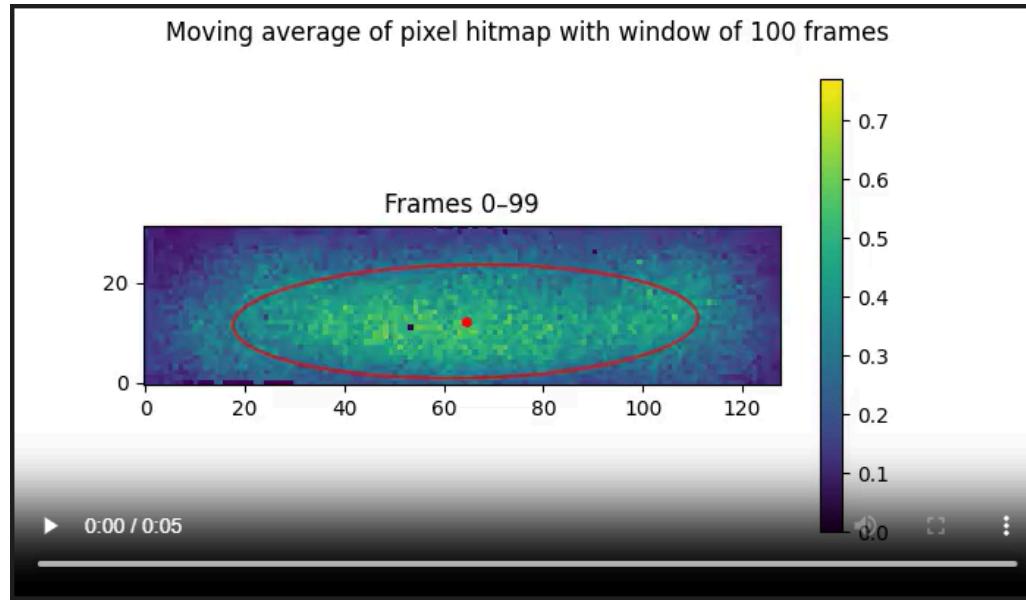
- 100 random Gaussian samples representing photons, not all of which are in-frame
- Ring represents 1 standard deviation
- We only see the samples within the frame
- Fitting Gaussian with sample mean and covariance doesn't work

# Approach

**Fit a new Gaussian to each 100-frame pixel hitmap average (no filtering) with pygmmis**

- Written by Princeton astronomers to model luminosity of galaxies from truncated camera data (like ours)
- Gaussian Mixture Models (GMMs) that can handle occluded data
- Under the hood, uses Expectation Maximization (standard for GMMs), but also generates mock samples to handle occluded regions

# Successful Fit



# Potential Applications

- Automatic quality control—non-invasive way to check alignment in-flight without a particular scan pattern
- Generate better pointwise reflectivity estimates
- Track pixels exhibiting unusual behavior in real time

# Future Work

- Continue investigating faster alternatives, using pygmmis as ground truth
  - Blur pixel heatmap to quickly track spot center without having to recalculate covariance
  - Fit to some subset of the data—random subset works surprisingly well
- Test on more scans
- Use to create an improved reflectivity estimate

# References

## Processing Performance Report

- [processing-performance \(master\) on Bitbucket](#)
- [acadia \(master\) on Bitbucket](#) Apptainers in apptainer\_creation and slurm bricks in processing\_workflow
- [python\\_3deo/fileIO/readGSOF.py](#) Used to collect certain scanning information

## Pose Graphs for Registration

- [zreg\\_ncc/python \(master\) on Bitbucket](#) see lu\_milos.py for my initial implementation, pose\_graph.py for improved Lie algebra implementation
- [Presentation I gave on pose graph registration](#)
- [Ideas related to the above presentation](#)
- [Lu and Milios paper](#) “Globally Consistent Range Scan Alignment for Environment Mapping”, April 1997. Introduces these ideas in the context of robotics
- [Borrmann et al. paper](#) “The Efficient Extension of Globally Consistent Scan Matching to 6 DoF”, June 2008. Extends concepts in the above paper to 3D. We aren’t using this paper’s new ideas, but it helped me understand what was going on better

## Modeling the Laser Illumination Spot

- [Write-up](#) Describes background and my work thus far on this
- **I have the git repo, but the code is unfinished, so I haven’t pushed to Bitbucket. Should I push?**