Multi-modal Localization and Point Cloud Reconstruction in Challenging Industrial Environments

CS231N Final Project - Milestone 3
Preliminary results

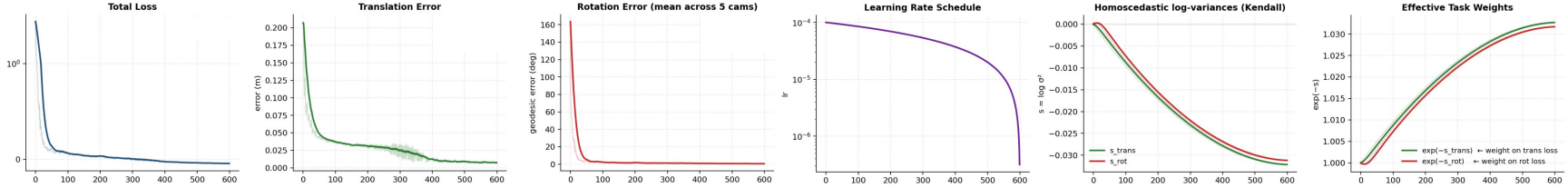
Zihan(Monica) Geng, Eyas Taifour, Taarush Grover

{ zihangen, etaifour, taarush } @stanford.edu

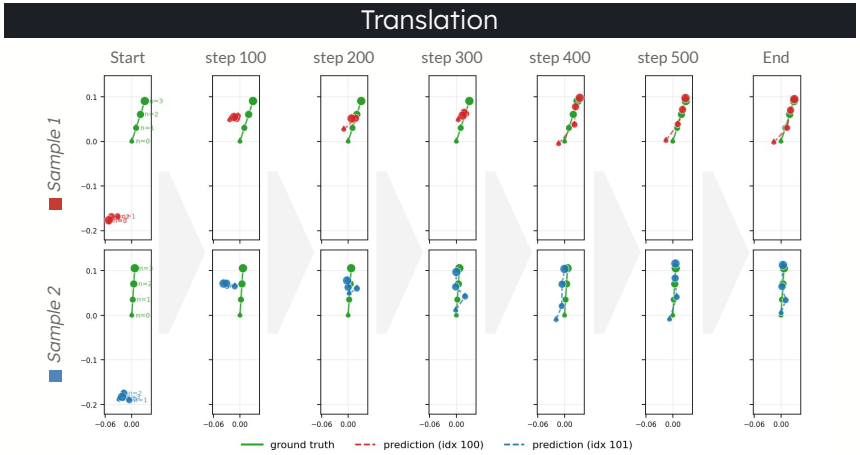
Image provided by the Hilti 2023 SLAM dataset challenge website..
Augmented with ChatGPT to extend the left wall.

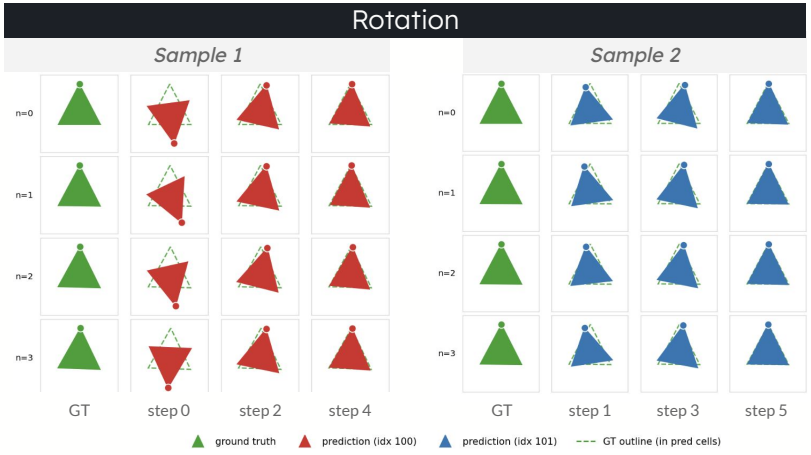
Overfitting: loss analysis

Multi-task loss (translation:L2, Rotation: Geodesic), assuming homoscedasticity



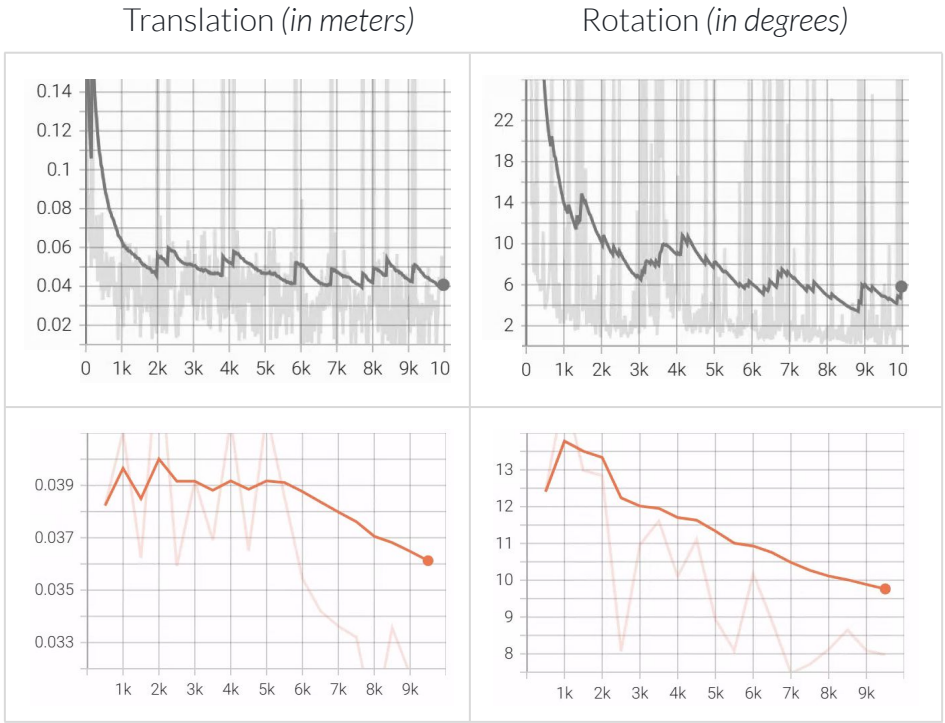
We overfit by training on 2 samples for 600 (global) steps. **Above:** Both translation and rotation errors converge to near-zero values, but at a different rates. The homoscedastic uncertainty weights do not diverge and both tasks are weighted equivalently throughout training, suggesting little impact on this small dataset. **Bottom left:** Each sample is a trajectory composed of 4 key points. We plot the trajectories of 2 Ground truths(green) and our learnt predictions (red, blue) every 100 steps on the XY plane. We observe convergence of translation starting at step 400. **Bottom Right:** Rotation converges in as little as 6 steps due to the similarity of Ground Truths. We visualize the relative rotation at each keypoint n.





V1: training on a subset

1 site only, for 2 epochs

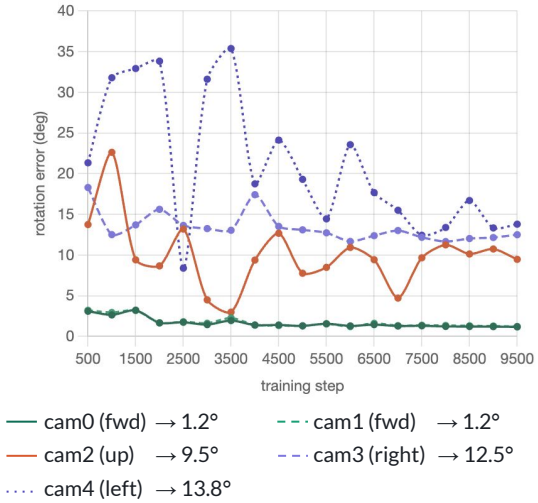


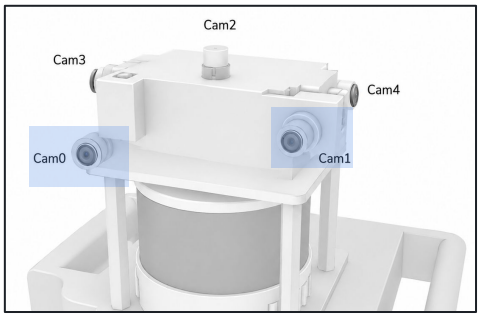
- **Translation**: Train error drops from 21 to 3.9 cm in 2K steps, where the mean validation error converges (3.6 cm). This doesn't suggest overfitting* nor underfitting**, but instead that the signal has converged to within the noise floor of the LiDAR derived GT, beyond which further improvement is not identifiable.
- **Rotation**: Validation error averaged across the 5-camera rig decreases from 12.4° to 7.9° over 10K steps (2 epochs). This seems to indicate that our IMU-augmented VGGT camera tokens learn a meaningful rotation correction from the supervision signal. These results provide preliminary support for our core hypothesis.
- **However:** The mean rotation error conceals a failure pattern across cameras that is tied to two areas (a) the rig's geometry and (b) VGGT's pre-training assumptions. (*see next slide for decomposition of the error per camera + suggested hypotheses)*

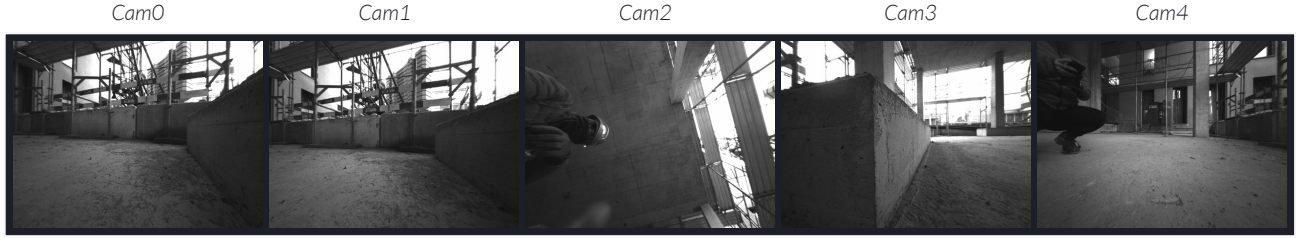
** Validation doesn't diverge | ** negligible gap*

CKPT: facebook/VGGT-1B | val_split = 0.1 | n_cam = 5 | n=4 | Single scene | IMU buckets K=40 (100ms) |IMU_seq_length =8

V1: Rotation error per camera





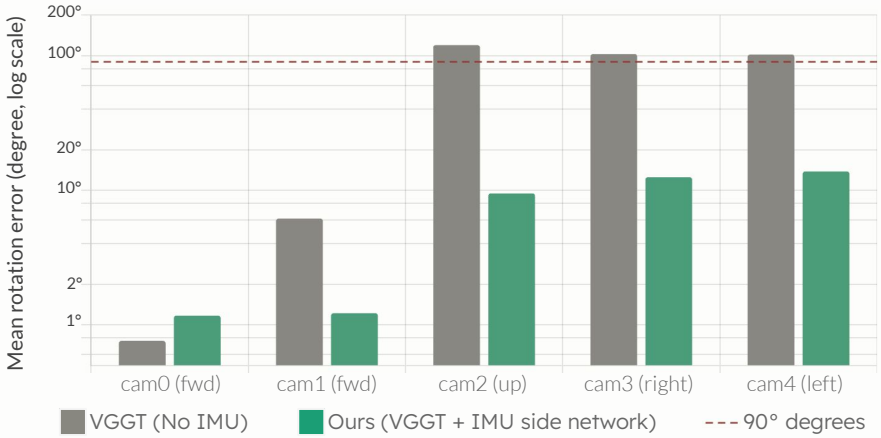


- **Cam0 & Cam1** (stereo overlap, pointing forward) converge to ~1.1°, while the remaining cameras attain an error ranging 10°-15° with high variance.
- **VGGT seems to collapse on cam 2/3/4** due to the lack of shared features. This observation invites us to reconsider the fisheye FOV of the original images (which we undistorted to align with the pinhole model VGGT adopts).
- **Our network is not learning the rig's physical constraint:** it treats each camera rotation independently. We evaluate different strategies to incorporate the rig-to-camera transforms, by either (a) extend the Transformers sequence to include this transformation token in the input, or (b) adjust the loss function, or (c) masking the output of the first camera with the expected anchor values, or finally (d) adjusting our dataset so that frames are not sequenced $t_n, t_{n+1}, t_{n+2}, \ldots$ but instead $t_n, t_{n+x}, t_{n+2x}, \ldots$ giving future $\mathbf{cam2}_{nx}$ the opportunity to track points in the present $\mathbf{cam0}_n$

Left: cropped render of the rig. Cam0 & Cam1 both point forward, their FOV overlaps.

V1: Baseline comparison 1

171 samples (Validation set): V1 compared against VGGT



- **Cam2/3/4 rotation errors are centered around 107°** - But VGGT is unaware of different cameras - only different images. Any rotation error is therefore expected to be normally distributed across all cameras. We have investigated this correlation, and found issues in Ground Truth representation (which are interpolated from infrequent LiDAR sweeps). We are resolving this issue as we speak.
- Our translation error seems uniform, unlike VGGT's. This was unanticipated as we have not observed varying homoscedasticity weights.

Rotation error (degrees)

	cam0	cam1	cam2	cam3	cam4
VGGT μ	0.76	6.16	119.71	102.84	102.01
VGGT σ	0.81	1.05	28.41	9.84	14.42
Ours μ	1.17	1.22	9.47	12.50	13.79
Ours σ	0.85	0.9	33.16	42.23	34.02

Translation error (meters)

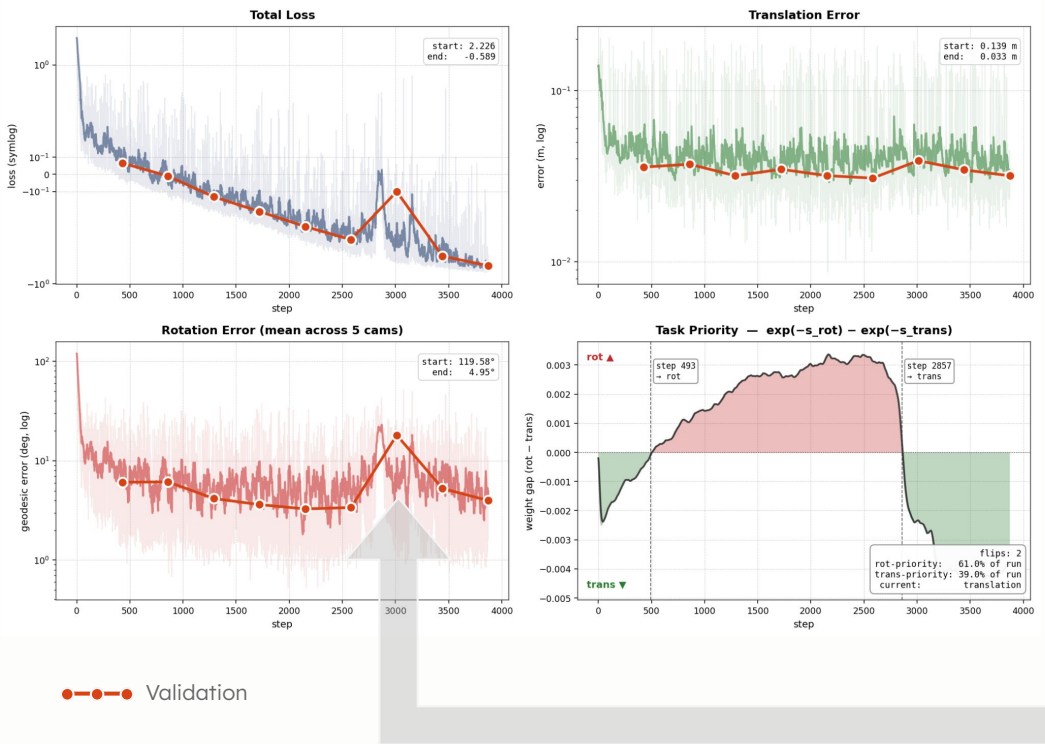
	cam0	cam1	cam2	cam3	cam4
VGGT μ	0.0221	0.1038	0.8349	0.1863	0.3955
VGGT σ	0.0495	0.0540	1.2523	0.3842	1.2370
Ours μ	0.0356	0.0283	0.0322	0.0297	0.0289
Ours σ	0.0467	0.0475	0.0474	0.0482	0.0480

Improvement ratio (VGGT μ / Ours μ)

	cam0	cam1	cam2	cam3	cam4
Rot ratio	0.6x	5.0x	12.6x	8.2x	7.4x
Tra ratio	0.6x	3.7x	25.9x	6.3x	13.7x

V2: Training update

Similar to v1, but 9 epochs

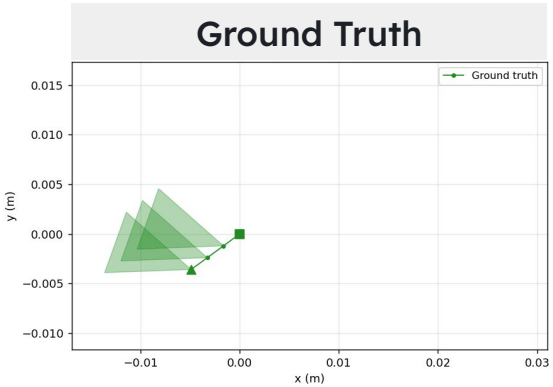


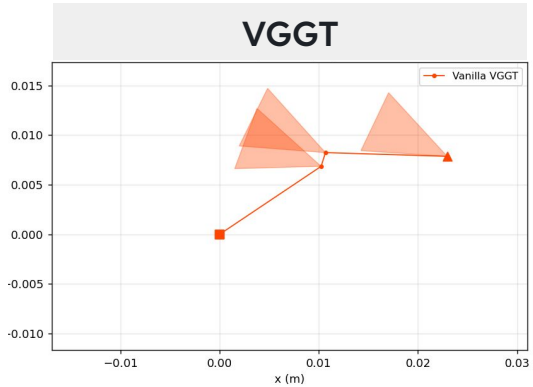
- **The total loss drops below zero**, it doesn't represent a problem for Kendall multi-task loss when the uncertainty of a task is low, and that task's loss is low.
- **The error variances remains wide** throughout training, which possibly indicates our source data is too spread to help increase the accuracy of the network. We will experiment different ways to encode IMU embeddings, different sequence lengths, and different number of Transformer Blocks (encoders/decoders) to compare results.
- We believe our network is trying to learn too much, too fast, and therefore will revisit our **learning rate and scheduling.**
- By analyzing the homoscedastic weights divergence, we see the network prioritizing rotation from epoch 1 - 7.
- We also see the **sharp increase of rotation loss between epoch 6-7,** suggesting the need to log all validation samples and further analyze them.

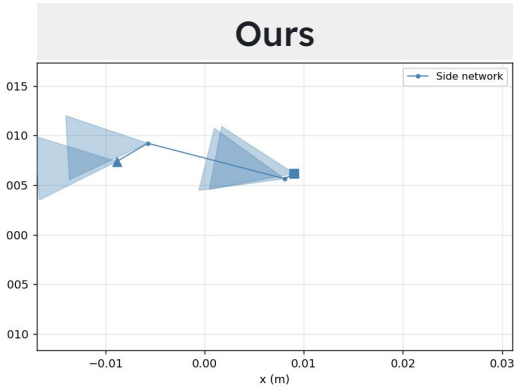
Qualitative Analysis

- We sample a trajectory from the dataset, and plot the pose on the XY (side) plane for the ground truth, VGGT, and our network. The start of a trajectory is ■, and its end is □ .
- Our Ground Truth is interpolated from infrequent LiDAR sweeps through point cloud registration, therefore the GT trajectory looks linear.
- The orientation at each keypoint is represented by the transparent isosceles triangles.

- Notice the trajectory direction in GT (it points to the 4th quadrant). VGGT predicts a trajectory in the opposite direction altogether, while ours corrects the poses' orders.
- We're actively resolving two issues prominent in the graphs:
  - The first frame is not properly initialized (not at the origin), due to the cam0 regression issue we've recently identified.
  - We produce a quaternion per keypoint. This is a bug since the orientation of the 1st camera in the first keyframe should be the basis vector.







Next steps: fixes, enhancements, more experiments

- **Dataset:**
  - Resolve camera 0 regression issues
  - Experiment different IMU sequence length
  - Skip *s* frames between *n*, *n-1*
  - Enhance code and fix issues (caching, batching)
- **Ground Truth:**
  - Explore Visual Inertial Odometry as GT proxy to create denser clouds
  - Assign timestamp to individual points in the clouds (instead of whole sweep)
- **Images:**
  - Create better refined masks and apply them to images and LiDAR point clouds
  - Apply Normalization and Data augmentation
  - Evaluate the use of distorted (fisheye) images
- **Training:**
  - Train on all the data, not just 1 site
  - Hyperparameter search
- **Objective:**
  - Investigate different loss functions
  - Consider Incorporating rig geometry as a prior
- **Evaluation**
  - Implement Chamfer Distance
  - Evaluate changing the objective to cater for depth adjustment
- **Architecture:**
  - Extend training to learn depth-map multiplier
  - Evaluate different model dimensions, number of blocks, regularization, normalization
  - Evaluate other IMU pre-integration approaches (1D CNN)
  - If needed to support experiments, augment camera tokens with external features (DINO, CLIP, FiLM, Depth Anything v2)
- **Tools:**
  - Build Visualization tools (point cloud - Viser) - Visualize translation, rotation, and least priority: Point Clouds

References

- Reizenstein, J., et al, "Common Objects in 3D: Large-Scale Learning and Evaluation of Real-life 3D Category Reconstruction," in *International Conference on Computer Vision*, 2021.
- S. Umeyama, “Least-squares estimation of transformation parameters between two point patterns,” IEEE Trans. Pattern Anal. Machine Intell., vol. 13, no. 4, pp. 376–380, Apr. 1991, doi: 10.1109/34.88573.
- S. Zheng, M. Oechsle, E. Sandström, M.-J. Rakotosaona, F. Tombari, and I. Gilitschenski, ‘Good token hunting: A hitchhiker’s guide to token selection for visual geometry transformers’, *arXiv [cs.CV]*, 22-May-2026.

Human masking removal (LiDAR GT)

Cam2: 27/29 frames flagged (93.1%)

Cam4: 26/29 frames flagged (89.7%)

LiDAR: 34,936 / 2,982,670 points removed over 50 sweeps

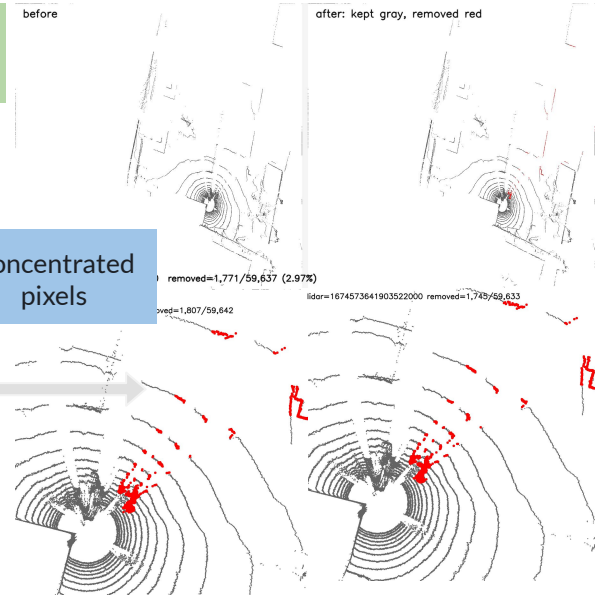
Camera Image - Person mask generator - Binary (0/1) Mask

timestamp matching (find nearest camera frame) - LiDAR projection

LiDAR 3D (x,y,z) to Camera image (u,v)

mask[v,u] > 0 = **LiDAR point**

Output kept points **without human contamination**



Concentrated pixels

Caveat:

This diagnosis is only for camera 2 and 4, with 29 image masks and first 50 LiDAR scans