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

CS231N Final Project - Milestone 2:
Technical approach

Zihan(Monica) Geng, Eyas Taifour, Taarush Grover

{ zihangen, etaifour, taarush } @stanford.edu

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

Overview on a page

Project refresh

- **Problem**: Feed Forward Network for 3D reconstruction (VGGT) suffers to reconstruct point cloud in textureless, B&W environments (i.e. construction sites/industrial env.)
- **Hypothesis**: Fusing IMU data enhances the quality of the reconstructed point cloud.
- **Dataset**: 12 scenes taken at 3 sites.
  Scenes are collected from a mobile rig:
  - 5 cams (different orientations, 40 Hz)
  - IMU (400 Hz)
  - LiDAR (10 Hz)
- **Evaluation Metric**: Chamfer distance
- **Challenges**: B&W, textureless images
  Lens distortion
  Human presence
  Different device frequencies
  Ground Truth not provided

M2 High level

- **Segmentation and masking of humans**
- **GT**: Pose estimate relative to previous frame. Interpolated pose transformation through point cloud registration
- **Baseline *:** Point cloud from COLMAP and VINS-Mono
- **Root cause analysis**: inaccurate camera pose estimates
- **Proposed Architecture**: Train E2E Transformer E/D that generates new camera tokens from frozen VGGT, cross attending to encoded IMU embeddings +1D Convolution that describes the motion change in an . Use newly learnt camera poses to project z-maps into global point cloud
- **Encoder**: QKV: IMU data preceding each image + a pre-integration token (learnt by 1D CNN over IMU seq)
- **Decoder:** Q: VGGT Camera tokens, KV: encoder tokens
- **Out**: Pose Estimate (Quaternion + Translation vector)
- **Loss**: multi-task (Geodesic rotation + L2 translation) weighted by the homoscedastic uncertainty of each task.

** see appendix for more details.*

Data Loader overview

x

Shots

6 DOF IMU = 400Hz

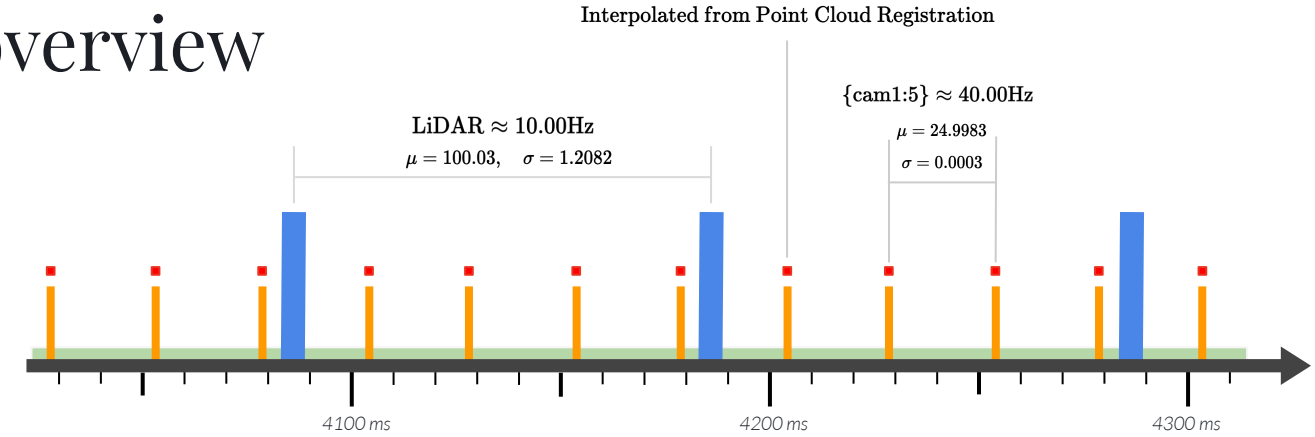
$\mu = 2.5039 \quad \sigma = 0.0048$

LiDAR sweeps

y

Not in the dataset.

Interpolated base on point cloud registration *(see next slide for more info)*



Selecting data for pre-processing:

(a) The dataset is indexed by LiDAR samples: $ts(L_{idx})$

(b) An integer n (hyperparameter) defines the number of preceding frames to include in the sample. It is a hyperparameter.

(c) The 6DOF IMU data that covers the range $1 : n + 1$

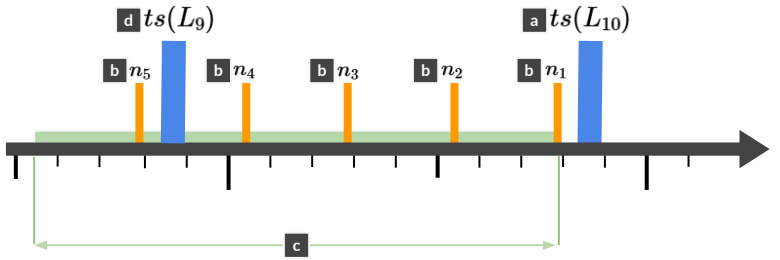
(d) Any other LiDAR sweeps within the range

The data is then pre-processed

(see Appendix for pre-processing technique)

Example data selected

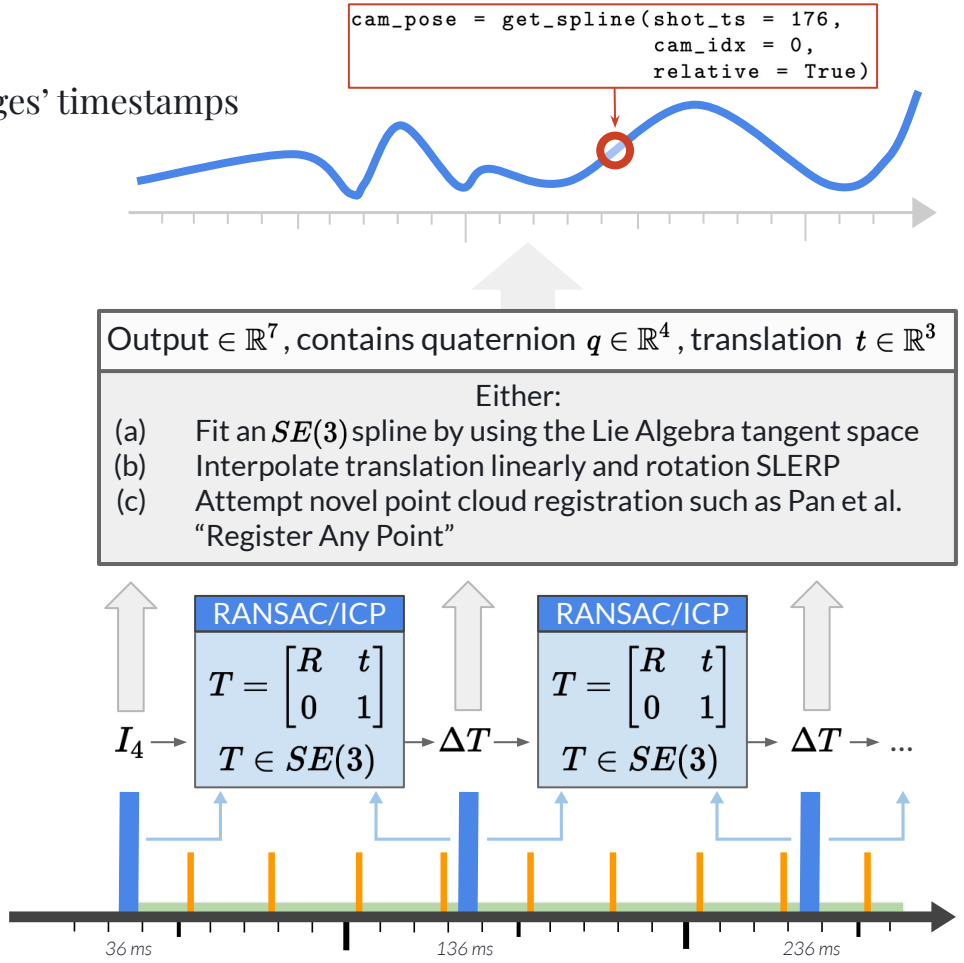
```
dataset = HiltiDataset(undistort=True, n=5)
dataset[10]
```



Ground Truth

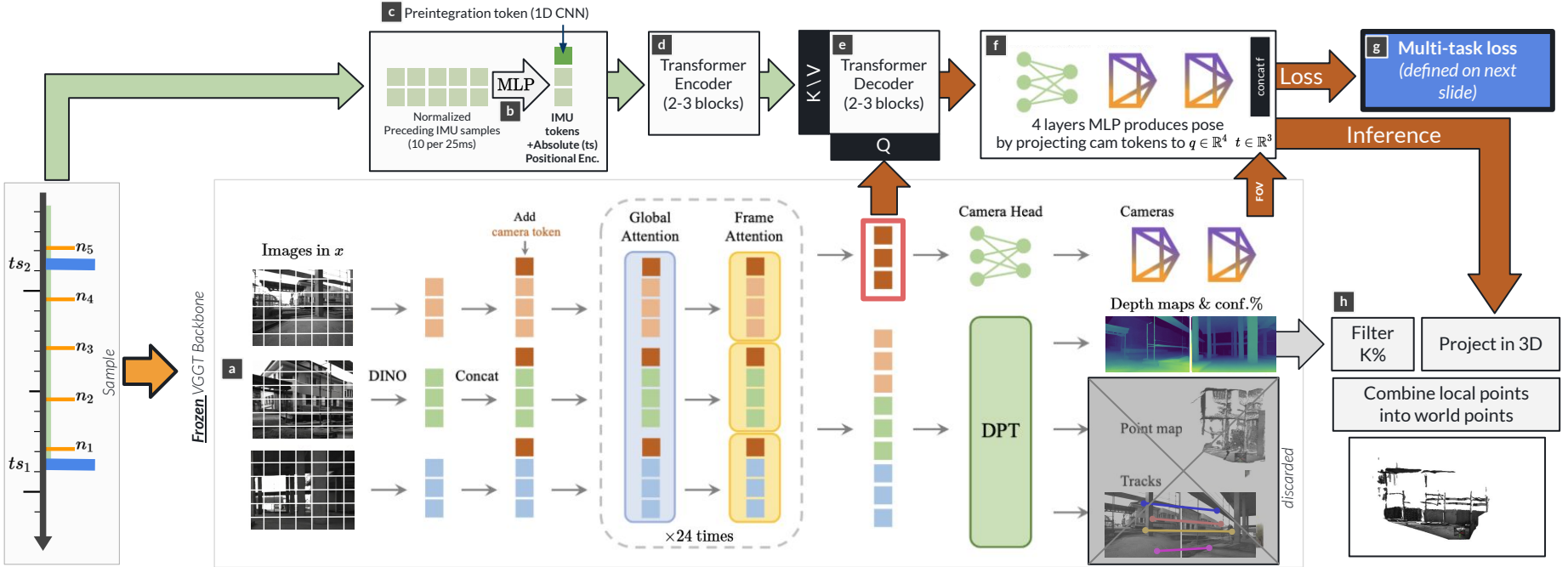
Challenge: We need ground truths (point clouds) at images' timestamps
But the cameras and LiDAR have different frequencies.

- **<u>Solution: Point cloud registration:</u>** Due to the absence of absolute camera poses **for most*** of the frames, we use RANSAC/ICP or KissICP between LiDAR sweeps to extract the relative rotation & translation. We treat the first frame as world's origin.
- We planned to infer the trajectory spline by fitting these sparse representation in a smooth curve to model the rig's motion.
- We observed the residuals against 4 known absolute positioned and found a large cumulative drift of +4 meters, which indicates the unsuitability of an integrated trajectory model.
- We will initially resort to linear interpolation and SLERP, but will also evaluate novel methods such as Pan et al. "Register Any Point: Scaling 3D Point Cloud Registration by Flow Matching"

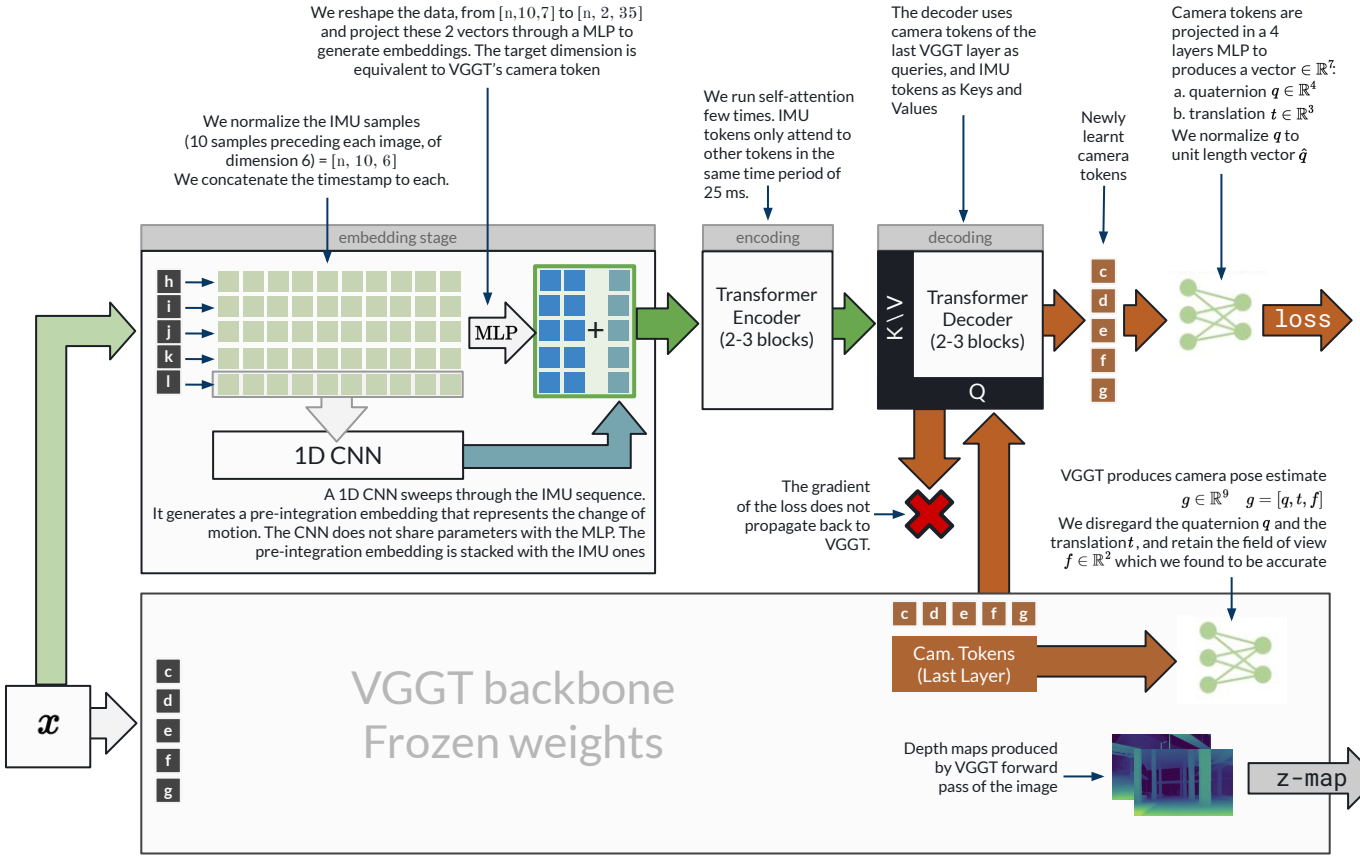


Model: Camera Tokens cross-attending to IMU

a. Freeze VGGT and forward pass the images.
b. IMU are projected into camera token dimensions through MLP
c. A preintegration token is produced by 1D CNN & stacked
d. IMU tokens are encoded in 2-3 Transformer blocks.
e. A decoder queries VGGT's camera tokens and cross attends them to the IMU representations (Keys/Values). It produces new camera tokens.
f. Camera tokens are passed through MLP to build a vector of quaternion and translation vector for each cam.
g. Multi task loss (Geodesic rotation + L2 translation) *(see next slide)*
h. We produce new world points by projecting the depth maps from the newly learnt camera poses. An $SE(3)$ matrix is produced from the quaternion and translation. We retrieve field of view from VGGT's output vector, which we use to form camera intrinsics.



Forward/Backward pass, loss



loss

The loss of the pose is the sum of the loss of the rotation and the loss of the translation.

These vectors have different units, hence we use a multi-task loss and calculate each individually.

To prevent a task loss from dominating the other in the total loss, we want to weigh each using parameters, shown naively below:

$$\mathcal{L} = \frac{1}{N}\sum_{k=1}^{N}\mathcal{L}_{\text{pose}}^{(k)}$$

$$\mathcal{L}_{\text{pose}} = \alpha\mathcal{L}_{\text{trans}} + \beta\mathcal{L}_{\text{rot}}$$

The individual loss items are Geodesic rotation and L2 loss:

$$\mathcal{L}_{\text{rot}} = \arccos\left(\left|\langle\hat{\mathbf{q}}_{\text{pred}},\ \mathbf{q}_{\text{gt}}\rangle\right|\right)$$

$$\mathcal{L}_{\text{trans}} = \left\|\mathbf{t}_{\text{pred}} - \mathbf{t}_{\text{gt}}\right\|_2$$

But finding the correct values of parameters α, β will turn our project into a long hyperparameter search exercise, and will fail to generalize.

Therefore, en lieu of the naive definition above, we rely on the approach defined by Kendall et al.

We weigh the two loss functions by considering the homoscedastic uncertainty of each task.

$$\mathcal{L}_{\text{pose}} = \mathcal{L}_{\text{trans}} \cdot e^{-s_t} + s_t + \mathcal{L}_{\text{rot}} \cdot e^{-s_r} + s_r$$

where $s_t = \log\hat{\sigma}_t^2$, $s_r = \log\hat{\sigma}_r^2$ are learnable scalars

evaluation

Finally, the VGGT produced depth-maps are reprojected in the 3D world. The complete global point cloud is assembled by taking the union of all valid projected 3D points across all cameras, omitting invalid or missing depth readings.

$$\mathcal{C} = \bigcup_{i=1}^{N}\left\{P_{\text{world}}^{(i,u,v)} \;\middle|\; \forall(u,v) \text{ where } d_i(u,v) > 0\right\}$$

We evaluate our global point cloud against the ground truth proxy by calculating Chamfer Distance:

$$CD(P,Q) = \sum_{p\in P}\min_{q\in Q}\|p-q\|_2^2 + \sum_{q\in Q}\min_{p\in P}\|q-p\|_2^2$$

Pending design decisions

Open questions

- **GT proxy:** SE(3) Cubic spline drifts. Alternative approach to consider is windows Spline, or Linear interpolation.
- **IMU embedding span:** Initially, we group each 5 IMUs into 1 embeddings, but we can explore other IMU groups (e.g. 2 IMUs per token)
- **Pre-integration token:** 1D CNN in our design, but we could try a causal Transformer decoder
- **Ablation experiment:** What is the best value of n images to use in the net? At what value of n does performance degrade?
- **Number of layers (**MLP, 1D CNN arch, Transformers Encoders/Decoders)
- **Human masking pipeline:** Grounding DINO + SAM2 most promising, but we need to refine the prompts as some masks are inaccurate.
- **Loss:** We might explore other loss types such as Huber, Log-cosh, Scale-invariant loss
- **Prior:** We do not initially incorporate prior knowledge in the model (i.e. the fact we know the relative rotation and translation of the 5 cameras on the rig), but might consider it.

Status progression. Yellow: WIP, Green: Finalized

M1 M2

A. **Model & Data validation (ROS to PNG, IMU data, etc) -** Confirmed we can run the models/ have access to data

B. **Exploratory Data Analysis** (Validate image distortion/IMU integration/Frequency balance/Fill in GT gaps if needed (Colmap)

C. Augment GT:

In addition to the provided LiDAR Point clouds, we will also attempt at building a global PC with SfM (Colmap)
And a VIO pipeline (VINS-MONO)
LiDAR runs at a slower refresh rate than cameras (10 Hz vs 40 Hz). We will explore methods to fill the gaps (if deemed required). Alternatively, we will consider using a batch of 4 sequential images as input + their IMU data to produce 1 LiDAR point cloud

D. Find root cause of the issue + Exploration

Vanilla VGG ablation: Quantitative & Qualitative evaluation of VGGT on Hilti data (images only) - Characterize the failure of VGGT on existing data, comparing point cloud output to LiDAR GT and calculating Chamfer distance. Explore Attention maps (Hilti vs. VGGT sample dataset) to determine possible source of the construction failure
Perform confidence sweep - (Drop 3D points < X% to find the confidence threshold that results in highest Chamfer distance scores)
Explore the effect of Multi-view inference
Explore the features generated by Dinov2- If deemed the source of the 3R problem, consider augmenting them using external sources such as CLIP, FiLM, or fusing IMU data (exploring early/ late fusion, or other techniques discussed in video understanding).
Explore the effect of camera tokens / Papers such as Fisheye3R enhanced VGGT by modifying the camera tokens only. We will explore if such a strategy is applicable to our problem statement

E. Evaluate strategies to resolve the issue

Evaluate early/late fusions (IMU)
Evaluate attaching probes to VGGT (to keep the backbone frozen) vs. (least favourably) re pre-training.

References (1/2)

- Pan, Y., Sun, T., Zhu, L., Nunes, L., Armeni, I., Behley, J., & Stachniss, C. (2025). Register any point: Scaling 3D point cloud registration by flow matching. doi:10.48550/arXiv.2512.01850
- Kendall, A., Gal, Y., & Cipolla, R. (2017). Multi-task learning using uncertainty to weigh losses for scene geometry and semantics. doi:10.48550/arXiv.1705.07115
- Ashish Devadas Nair, Julien Kindle, Plamen Levchev, & Davide Scaramuzza. (2024). Hilti SLAM Challenge 2023: Benchmarking Single + Multi-session SLAM across Sensor Constellations in Construction.
- René Ranftl, Alexey Bochkovskiy, & Vladlen Koltun. (2021). Vision Transformers for Dense Prediction.
- Vizzo, I., Guadagnino, T., Mersch, B., Wiesmann, L., Behley, J., & Stachniss, C. (2022). KISS-ICP: In defense of point-to-point ICP -- simple, accurate, and robust registration if done the right way. doi:10.48550/arXiv.2209.15397
- Jianyuan Wang, Minghao Chen, Nikita Karaev, Andrea Vedaldi, Christian Rupprecht, & David Novotny. (2025). VGGT: Visual Geometry Grounded Transformer.
- Tong Qin, Peiliang Li, & Shaojie Shen (2017). VINS-Mono: A Robust and Versatile Monocular Visual-Inertial State Estimator.
- Schönberger, J., & Frahm, J.M. (2016). Structure-from-Motion Revisited. In *Conference on Computer Vision and Pattern Recognition (CVPR).*

References (2/2)

- Fischler, M. A., & Bolles, R. C. (1981). Random sample consensus. *Communications of the ACM*, *24*(6), 381–395. doi:10.1145/358669.358692
- Besl, P. J., & McKay, N. D. (1992). A method for registration of 3-D shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *14*(2), 239–256. doi:10.1109/34.121791
- Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, Mahmoud Assran, Nicolas Ballas, Wojciech Galuba, Russell Howes, Po-Yao Huang, Shang-Wen Li, Ishan Misra, Michael Rabbat, Vasu Sharma, Gabriel Synnaeve, Hu Xu, Hervé Jegou, Julien Mairal, Patrick Labatut, Armand Joulin, & Piotr Bojanowski. (2023). DINOv2: Learning Robust Visual Features without Supervision.
- Khanam, R., & Hussain, M. (2024). YOLOv11: An overview of the key architectural enhancements. doi:10.48550/arXiv.2410.17725
- Ravi, N., Gabeur, V., Hu, Y.-T., Hu, R., Ryali, C., Ma, T., … Feichtenhofer, C. (2024). SAM 2: Segment Anything in Images and Videos. doi:10.48550/arXiv.2408.00714
- Ren, T., Liu, S., Zeng, A., Lin, J., Li, K., Cao, H., … Zhang, L. (2024). Grounded SAM: Assembling open-world models for diverse visual tasks. doi:10.48550/arXiv.2401.14159

Appendix

The appendix contains slides we've moved to the back of the presentation, to make the milestone check-in most effective.

1. **Data Preparation:** we explain the pre-processing techniques we will apply to the data
2. **Baselines**: We define the approach to calculate baselines using two techniques (SfM and Visual Inertial Odometry)
3. **Alternative Architectures:** During our team discussions, we have prepared two alternative architectures that we are only listing here to demonstrate our work. We do not intend on implementing these architectures, unless any future experiment takes us in that direction.
   - Architecture 3 is similar to our core architecture, except for the decoder size and structure.
     - In our core architecture, our first decoder block will query the last VGGT camera token layer.
     - In the architecture 3, each of the decoder blocks will query the respective VGGT camera token layer
   - In architecture 2, we hypothesize the problem lies in the confidence scores generated. We envision a network that re-weighs the scores based IMU sequences. Early analysis demonstrates little deviation for depth scores calculated locally for each camera.

Dataset preparation

Images

- Undistort to pinhole model
- Gray2RGB: dup.
- Normalize using ImageNet stats
- Crop Height, Width // 14
- Human masking

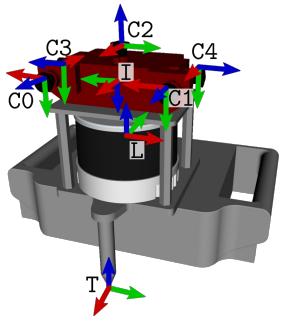
Source

YOLO +SAM2: Partial mask

Grounding DINO + SAM2: Better results

IMU

- Temporal alignment (indexing IMU sequence according to image #)
- Bias estimation and subtraction from periods of stationarity
- Gravity alignment using the IMU to body transform provided in the dataset (link)
- Coordinate frame alignment (IMU and cameras have different coordinate frames)



LiDAR

- Run LiDAR SLAM as described earlier in the Ground Truth slide
- Interpolate
  - Translation: Linear
  - Rotation: SLERP
- Apply LiDAR to camera transformation (see diagram below)

Baseline 1: VGGT vs SfM

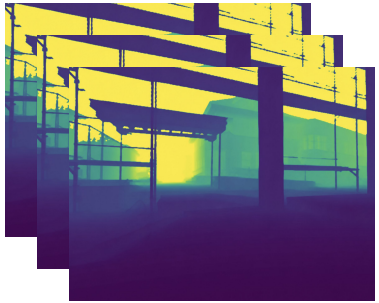
Building intuition of the capabilities of VGGT on our dataset. The baseline will allow us to understand if the failure in VGGT is at the depth map (per image), or the camera pose.

x (distorted)

Images $\in \mathbb{R}^{B,(n\times k),C,H,W}$

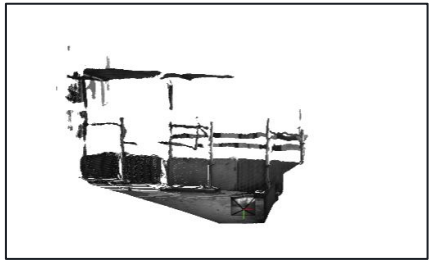
VGGT

Analyze per-pixel confidence, choose top $K\%$



COLMAP
$\forall\, n \in N$

SfM Baseline:
(1) Chamfer Distance
(2) ATE, RPE
RANSAC/ICP
Filter top $K\%$

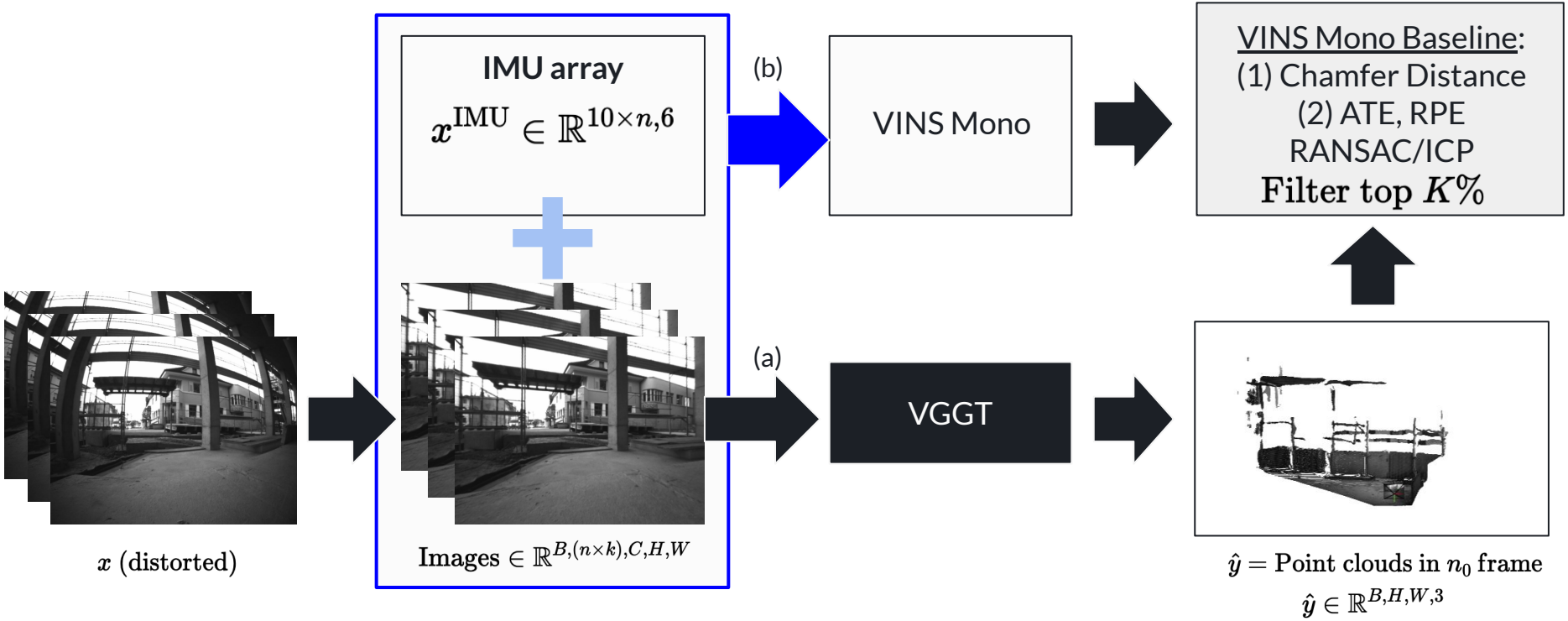


$\hat{y}$ = Point clouds in n_0 frame
$\hat{y} \in \mathbb{R}^{B,H,W,3}$

To find a conservative representation of the baseline, we will look for the combination of ts, n, K that minimize the loss function Chamfer Distance.

Baseline 2 vs VINS-Mono

Visual Inertial Odometry baseline



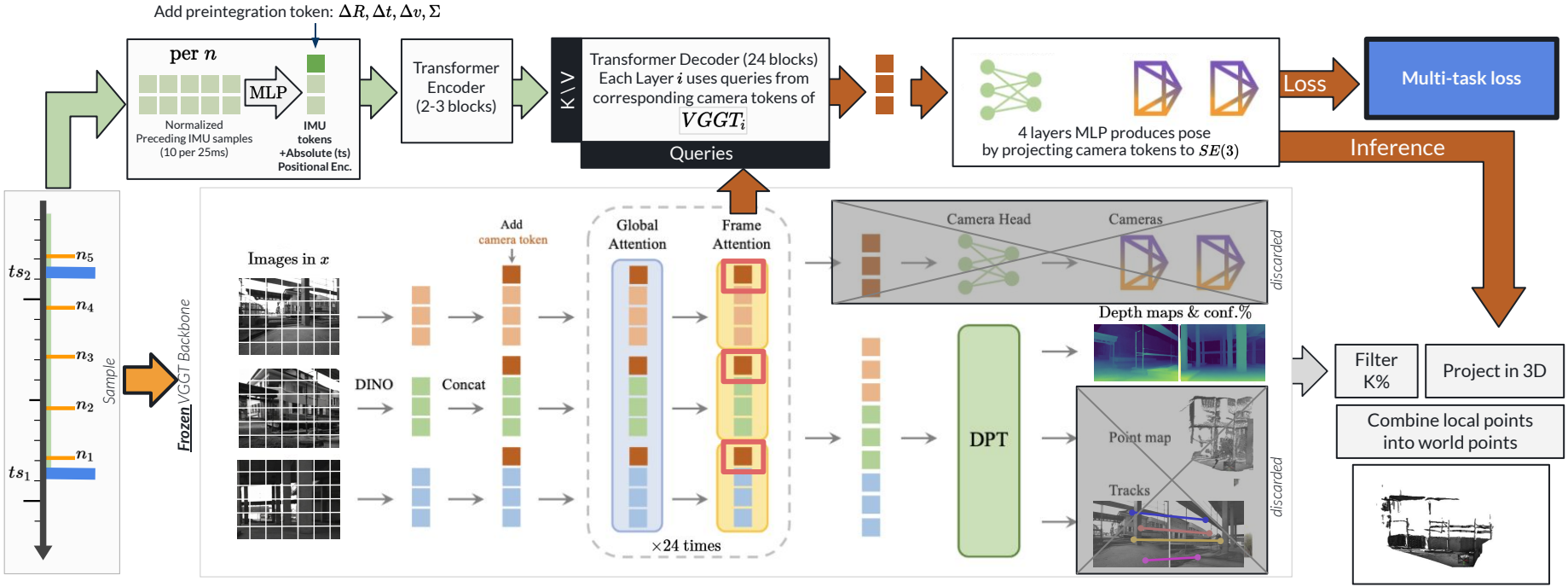
(a) Only images are fed to VGGT. We use the same hyperparameters found in the previous baseline.

(b) The same images, accompanied by the respective IMU data points are fed to VINS-Mono

Fallback option only if main architecture fails

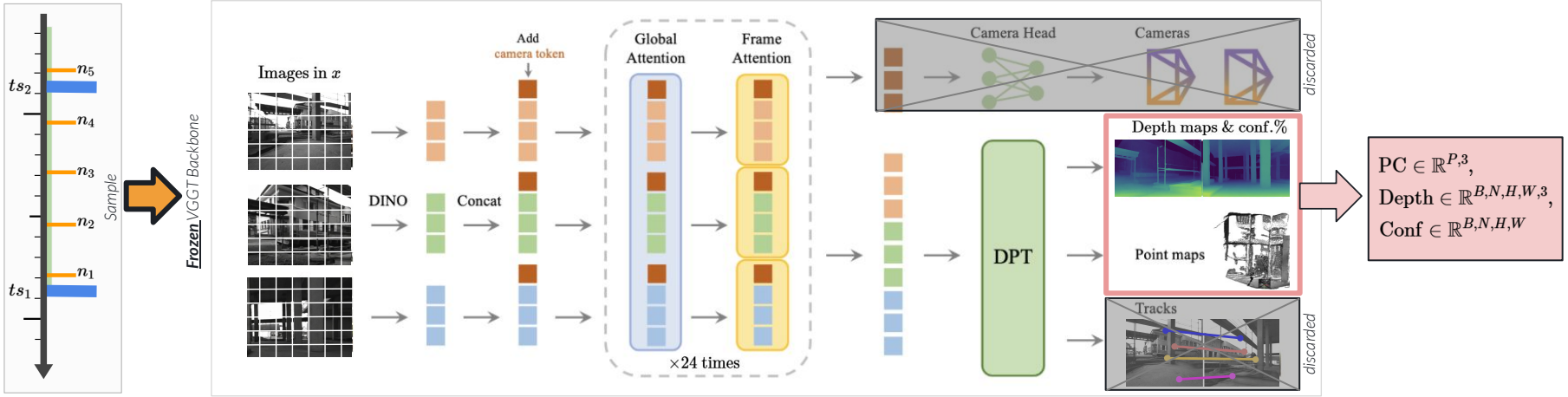
Arch #2: Camera Tokens conditioned on IMU (ladder)

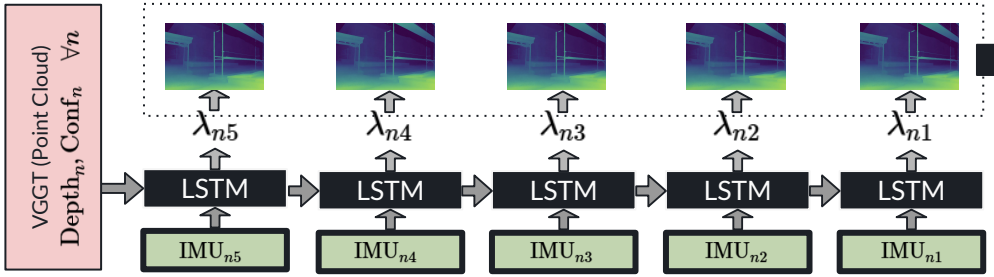
Hypothesis: The previous method shows promise, but can be pushed further to achieve better results.
Method: Freeze VGGT. Similar our core architecture, but instead of only decoding the final camera features, it conditions IMU on the camera tokens as they are trained across all the VGGT stack (24 blocks). Each decoder layer in our Network L[i] provides Keys and Values from the IMU stream, and 'reads' queries from VGGT[i]



Arch #3: IMU Adaptive Confidence Masking

Hypothesis: IMU data can determine which shots' confidence are more trustworthy for 3D reconstruction.
Method: Freeze VGGT. Produce depth maps, per-pixel confidence maps and world point clouds. Discard camera pose and point tracks. Train an LSTM on the IMU seq, conditioned on the VGGT output. Learn a scalar that adjusts the weights of confidence maps





Filter z map $\geq K\%$

Reproject points in 3D

RANSAC / ICP

LOSS: Chamfer Distance

- The LSTM is conditioned on the VGGT output
- At each timestep, the LSTM is fed the preceding 25ms of IMU data that have been embedded using an MLP
- The LSTM outputs λ at each timestep. λ Assigns a new weight to each confidence map
- The newly weighted confidence maps are used to filter the z-maps above a pre-selected threshold $\geq K\%$
- These filtered z-values are each projected in local 3D space. They are aggregated with RANSAC and ICP to form the combined point clouds, $\hat{y}$
- Finally, we calculate Chamfer Distance loss.