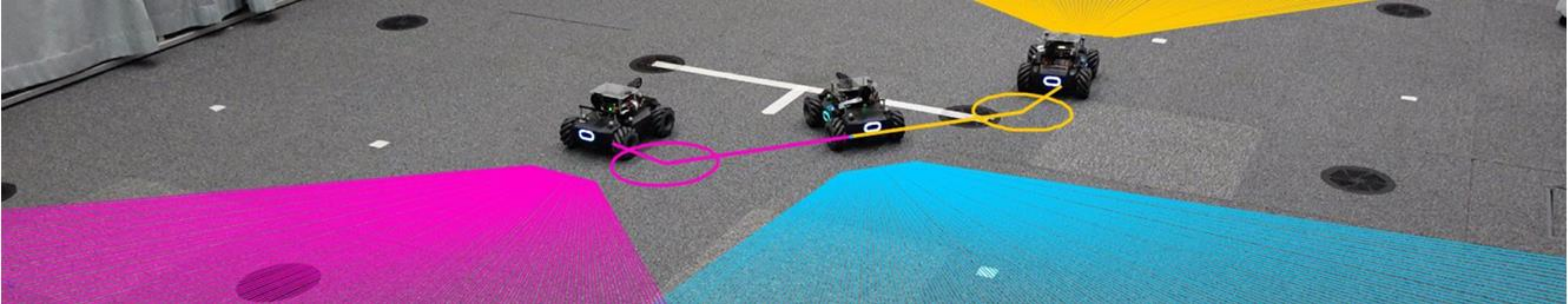


CoViS-Net: A Cooperative Visual Spatial Foundation Model for Multi-Robot Applications

Jan Blumenkamp, Steven Morad, Jennifer Gielis and Amanda Prorok
Department of Computer Science and Technology
University of Cambridge, United Kingdom
{jb2270, sm2558, jag233, asp45}@cst.cam.ac.uk

Presented by Dalil Merad

TABLE OF CONTENTS

ABOUT COLLABORATIVE RELATIVE LOCALIZATION

- Global localization:
  - Position of robot in global fixed reference frame (map, building, etc)

- Relative localization:
  - Between robots

- Collaborative localization
  - In a multi-robot setting, there one robot's belief could be useful for another robot. Raises new challenges in representation of beliefs and communication between robots

MOTIVATION

Multi-robot object manipulation

Wildfire monitoring

search and rescue

MOTIVATION

GNSS, Lidar and UWB limited by constraints like:

1. indoor operation,
2. unreliability around reflective surfaces, bright environments

RGB monocular cameras offer:

1. Low cost, low energy
2. Data rich
3. Aligned with our vision-centric human world

PROBLEM FORMULATION

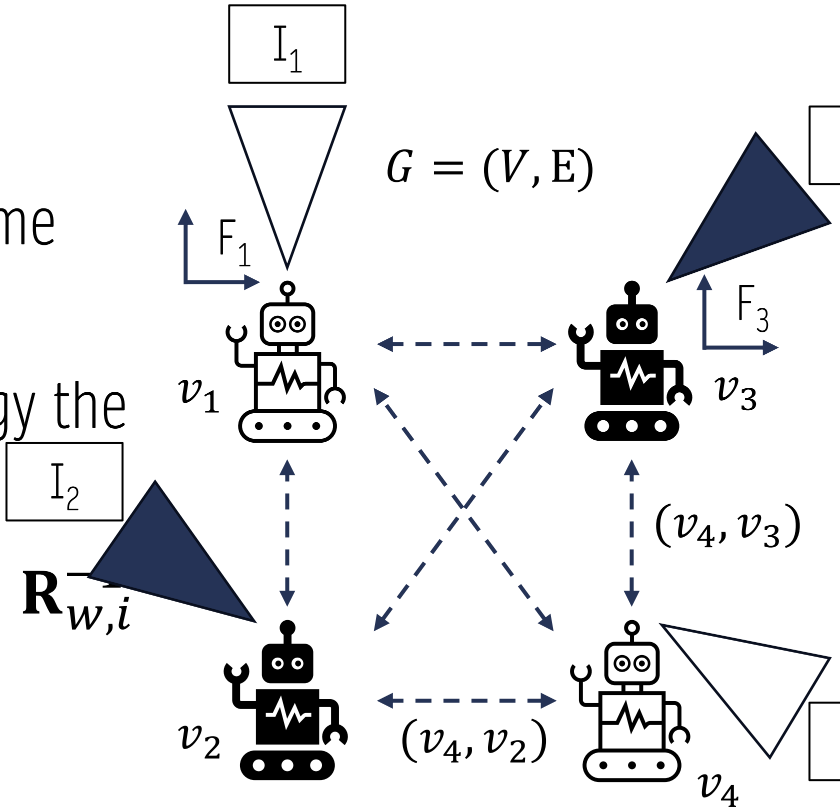
Consider a multi-robot system represented by a set of nodes $\boldsymbol{V}$

Each node $v_i \in V$ has position $\boldsymbol{p_{w,i}}$ & orient. $\boldsymbol{R_{w,i}}$ in world frame $\boldsymbol{F_w}$

The set of edges $\boldsymbol{E} \subseteq V \times V$ represents communication topology the graph is defined as $G = (V, \mathrm{E})$

We are looking for $\mathbf{p}_{i,ij} = \mathbf{R}_{w,i}^{-1} \cdot \left(\mathbf{p}_{w,j} - \mathbf{p}_{w,i}\right)$ and $\mathbf{R}_{i,ij} = \mathbf{R}_{w,i}^{-1} \cdot \mathbf{R}_{w,j}$

For each edge at each node.

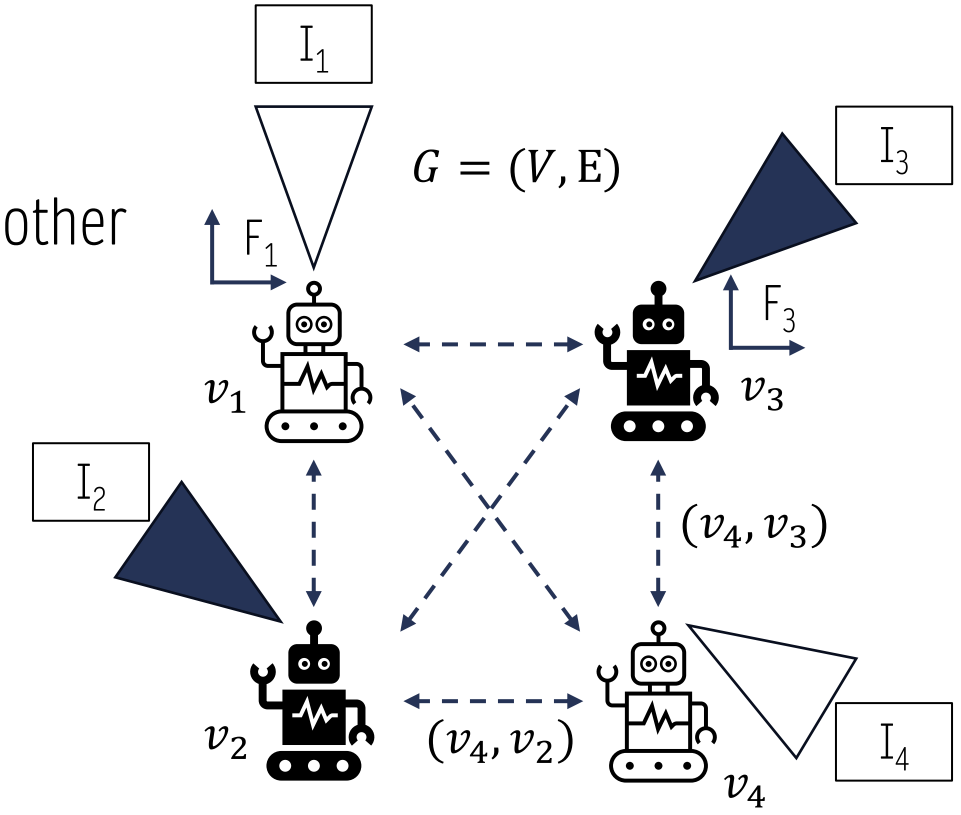


PROBLEM FORMULATION - GOALS

Goals:

(i) For each robot to predict its pose and uncertainty relative to other robots as well as corresponding embeddings using visual correspondences,

(ii) To use these embeddings for downstream tasks like local occupancy grid prediction.



RELATED WORK – RELATIVE POSE REGRESSION

Estimating the relative pose between 2 camera images 6-DoF pose from two images **without** a prebuilt map or scene-specific training:

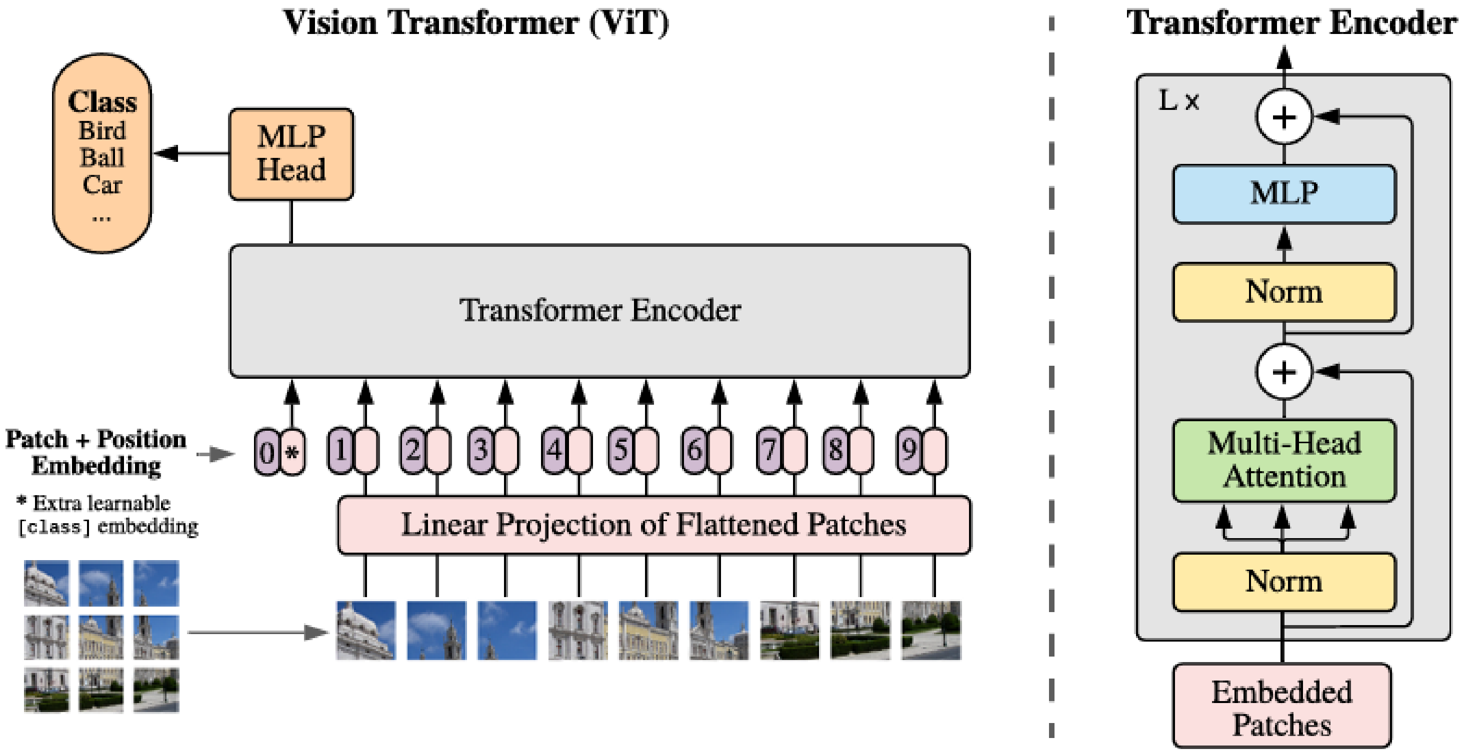
1. Traditional methods (feature based)

   Ex: SIFT/ORB → match features → estimate **E-matrix** → rotation + **scale-less** translation; **needs strong overlap**. Struggles with low texture, lighting, occlusions

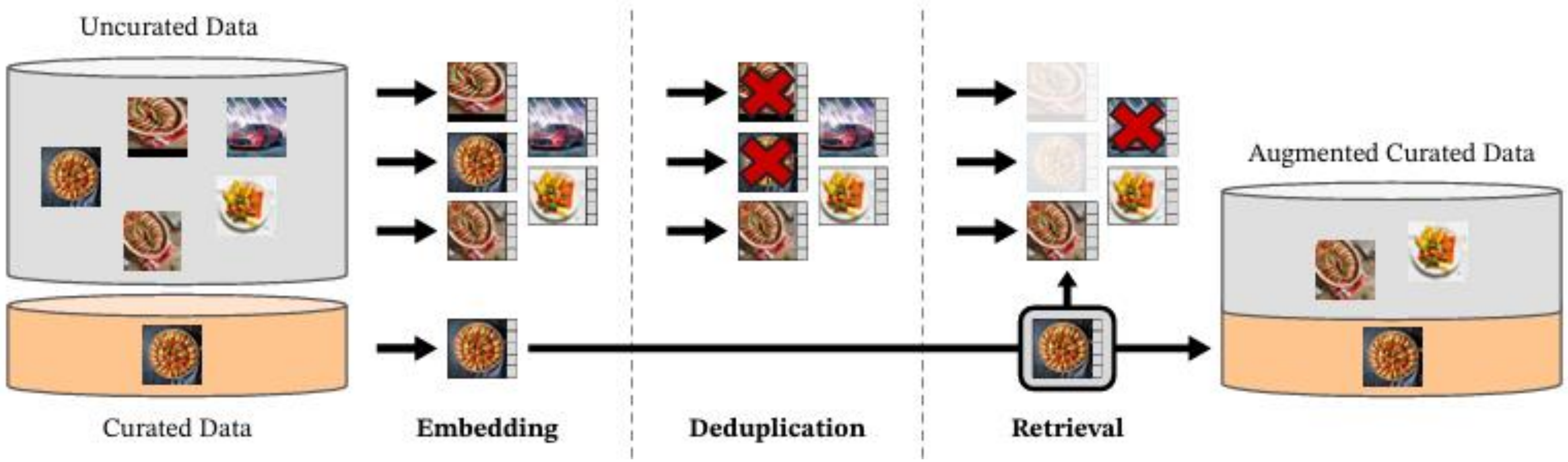
2. Learning-based approaches
   1. **Correspondence:** SuperGlue/LoFTR/LightGlue improve matches but still **require overlap**; typically output E-matrix (no scale).
   2. **Direct regression:** CNN/ViT regress pose map-free; **BUT** to get scale, need to integrate depth

All of these rarely model uncertainty and not often real-time on robots

PRIMER - VIT

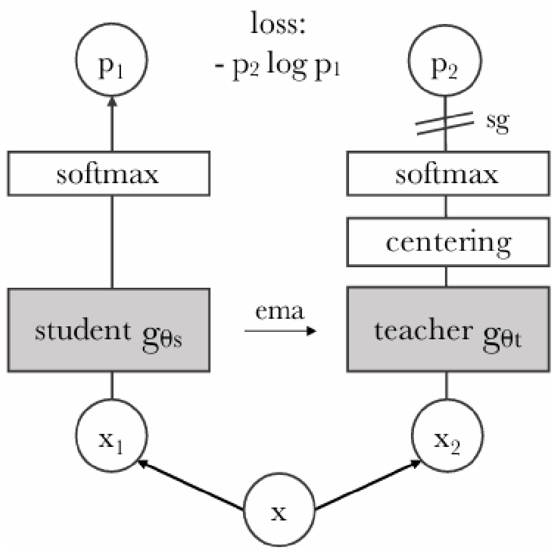

Vision Transformer (ViT)
Class
Bird
Ball
Car
...
MLP Head
Transformer Encoder
Patch + Position Embedding
* Extra learnable [class] embedding
0
*
1
2
3
4
5
6
7
8
9
Linear Projection of Flattened Patches
Transformer Encoder
L x
+
MLP
Norm
+
Multi-Head Attention
Norm
Embedded Patches

PRIMERS – DINOV2 – DATASET



1. Start from multiple curated datasets AND a 1,3B web-crawled images uncurated dateset
2. Remove images too similar (in some feature space) from the uncurated dataset 1,3B -> 744M images
3. Retrieve images from the uncurated dataset that are similar to the images in the curated dataset
4. Combine all into one dataset -> 142M images

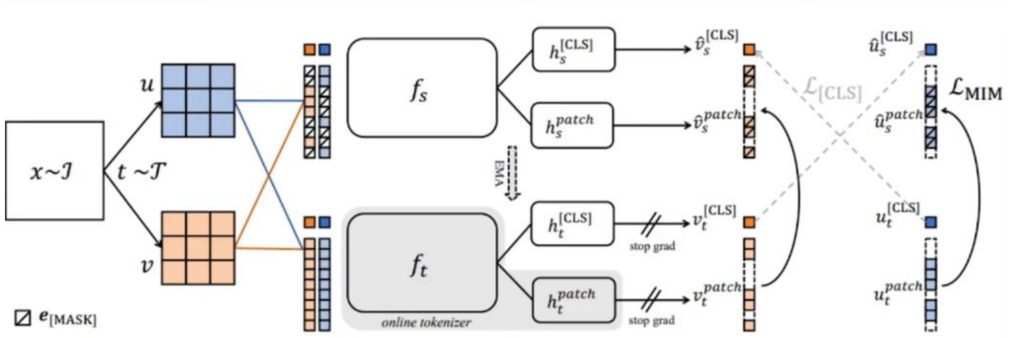
PRIMERS – DINOV2 – PRE-TRAINING



Use 2 objectives:

1. Image level objective: uses Dino loss

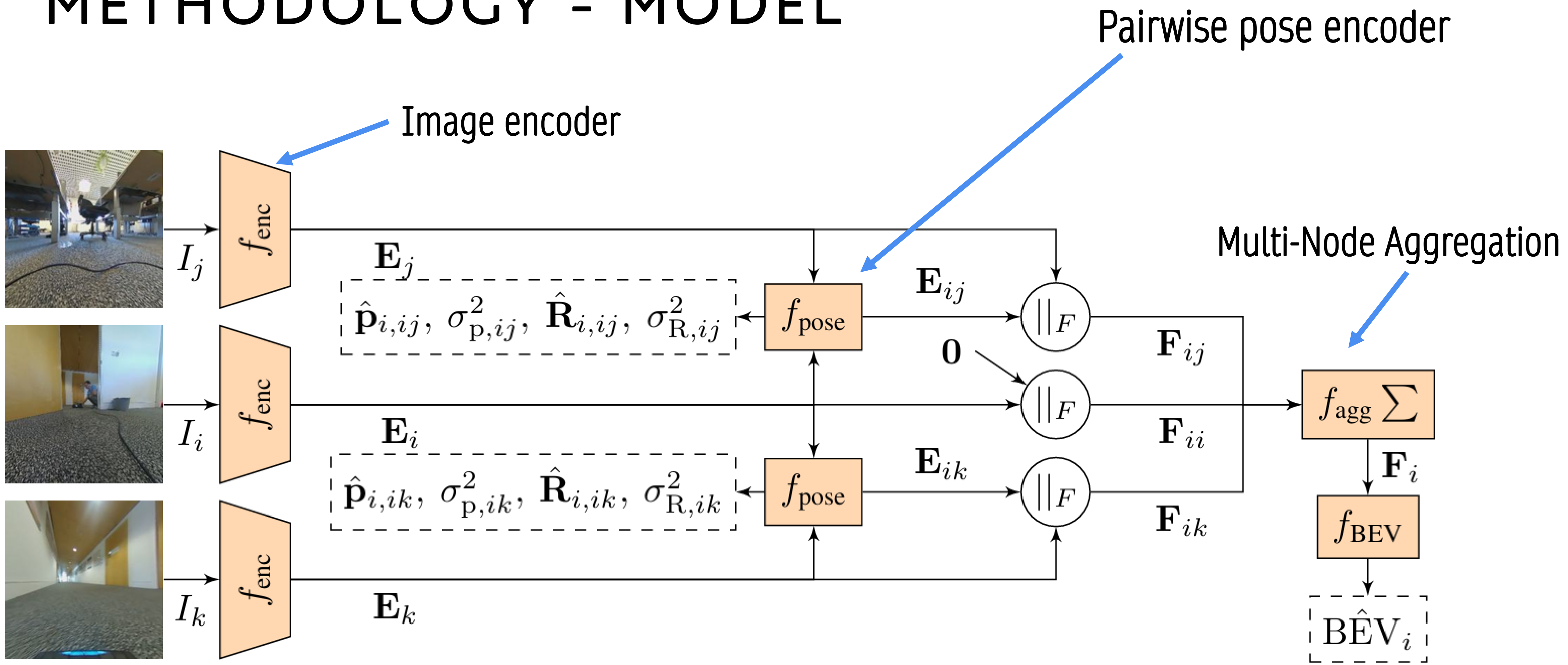
Student must learn the same representation as the teacher



2. Patch level objective -> iBot loss

Student must learn to predict masked tokens like the teacher

METHODOLOGY - MODEL



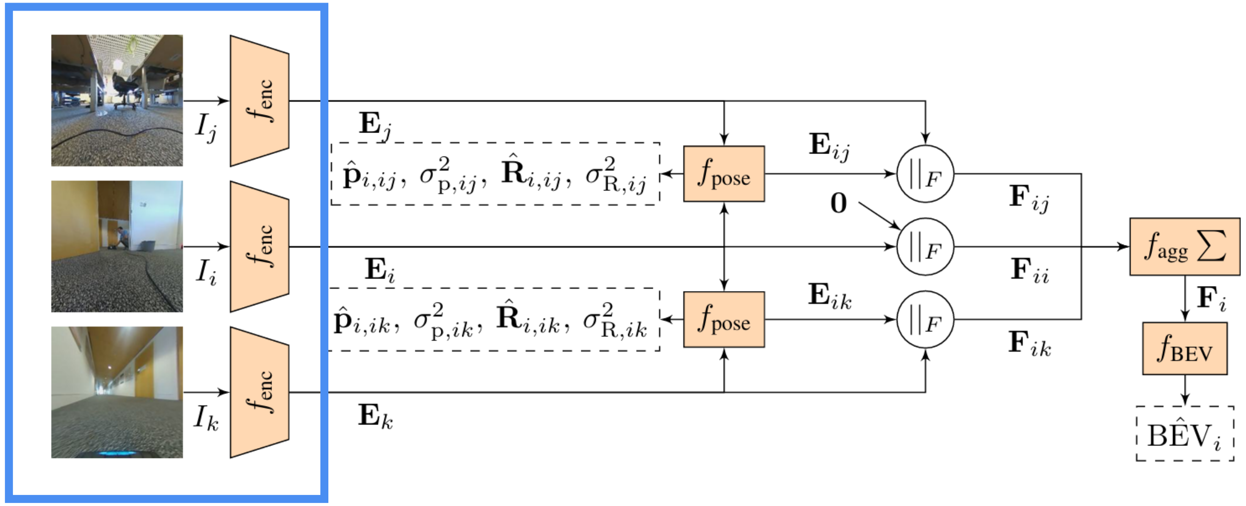
METHODOLOGY – MODEL – IMAGE ENCODER

- Smallest distilled Dinov2 available used as image encoder. Each robot produces $E_i \in \mathbb{R}^{S \times F}$ from image I_i as $I_i \rightarrow DINOv2 \rightarrow E_i$

S = Sequence length (number of patches/"tokens")

F = Feature vector size (for each "token")

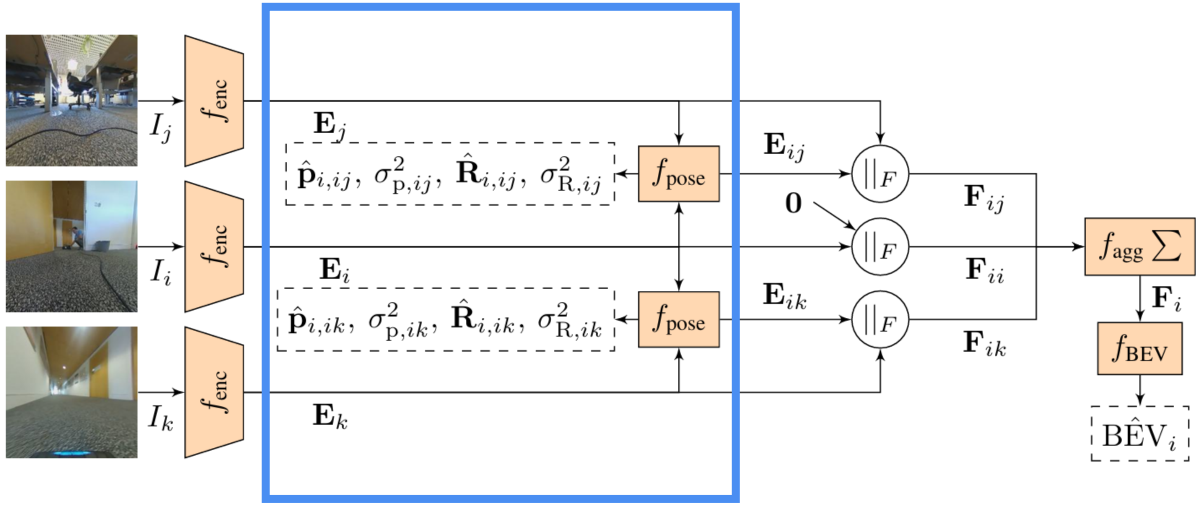
Weights are frozen



METHODOLOGY – MODEL – PAIRWISE POSE ENCODER

- Each robot broadcasts image encoding E_i to their neighbors
- Upon reception of someones E_j concatenate with self E_i along sequence dim
- Add positional embedding →pass to transformer → extract first element in sequence dim
- Pass this elem. Through 4 MLP's, get

$$\left(\hat{\mathbf{p}}_{i,ij}\,,\ \sigma^2_{\mathrm{p},ij}\,,\ \hat{\mathbf{R}}_{i,ij}\,,\ \sigma^2_{\mathrm{R},ij}\right)$$



METHODOLOGY – MODEL – MULTI-NODE AGGREGATION

Inputs: E_i ,$\{E_j\}$,edge embeddings $\{E_{ij}\}$;learnable S_{agg}

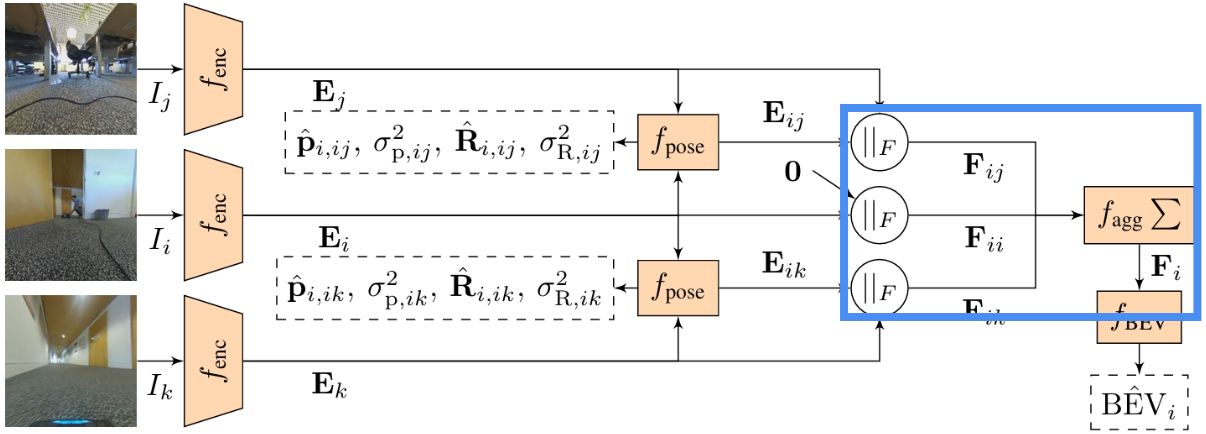
Per-edge fusion: $X_{ij} = (E_i \Vert_S E_j) + S_{agg}$ concat $f_{aggpos}(E_{ij})$along **features**.

Self-loop: add F_{ii}via $f_{aggpos}(\mathbf{0})$to handle zero-neighbor cases

Aggregate: 5-block transformer per edge → **sum over neighbors** → 1-block transformer → **take token[0]** = F_i

BEV head: 7× (conv + upsample) from $F_i$to $\widehat{\text{BEV}}_i$.

All blocks: $F = 192$, **12 heads**, MLP $4F$



METHODOLOGY – TRAINING – POSE LOSSES

Positions and **uncertainty** estimate using Gaussian Negative Log Likelihood Loss

$$\mathcal{L}^{\text{GNLL}}(\mu, \hat{\mu}, \hat{\sigma}^2) = \frac{1}{2}\left(\log\left(\hat{\sigma}^2\right) + \frac{(\hat{\mu} - \mu)^2}{\hat{\sigma}^2}\right)$$

Rotations and **uncertainty** →chordal dist. between quaternions → GNLL loss

$$d_{\text{quat}}(\hat{\mathbf{q}}, \mathbf{q}) = \min\left(\|\mathbf{q} - \hat{\mathbf{q}}\|_2, \|\mathbf{q} + \hat{\mathbf{q}}\|_2\right)$$

$$\mathcal{L}^2_{\text{chord}}(\hat{\mathbf{q}}, \mathbf{q}) = 2d^2_{\text{quat}}(\hat{\mathbf{q}}, \mathbf{q})\left(4 - d^2_{\text{quat}}(\hat{\mathbf{q}}, \mathbf{q})\right)$$

$$\mathcal{L}^{\text{GNLL}}_{\text{chord}}(\mathbf{q}, \hat{\mathbf{q}}, \hat{\sigma}^2) = \frac{1}{2}\left(\log\left(\hat{\sigma}^2\right) + \frac{\mathcal{L}^2_{\text{chord}}(\hat{\mathbf{q}}, \mathbf{q})}{\hat{\sigma}^2}\right).$$

METHODOLOGY – TRAINING – OVERALL LOSS

Poses Loss: combination of position loss and rotation loss:

$$\mathcal{L}_{i,ij,\mathrm{Pose}} = (1-\beta)\,\mathcal{L}^{\mathrm{GNLL}}(\mathbf{p}_{i,ij}, \hat{\mathbf{p}}_{i,ij}, \sigma^2_{\mathrm{p},ij}) + \beta\,\mathcal{L}^{\mathrm{GNLL}}_{\mathrm{chord}}\left(\mathbf{R}_{i,ij}, \hat{\mathbf{R}}_{i,ij}, \sigma^2_{\mathrm{R},ij}\right)$$

BEV Map: Mix of Dice Loss and Binary Cross Entropy

$$\mathcal{L}_{i,\mathrm{BEV}} = \alpha \cdot \mathcal{L}_{\mathrm{Dice}}\left(\mathrm{BEV}_i, \hat{\mathrm{BEV}}_i\right) + (1-\alpha)\cdot\mathcal{L}_{\mathrm{BCE}}\left(\mathrm{BEV}_i, \hat{\mathrm{BEV}}_i\right)$$

Overall: Combination of BEV and pose for every edge of every node

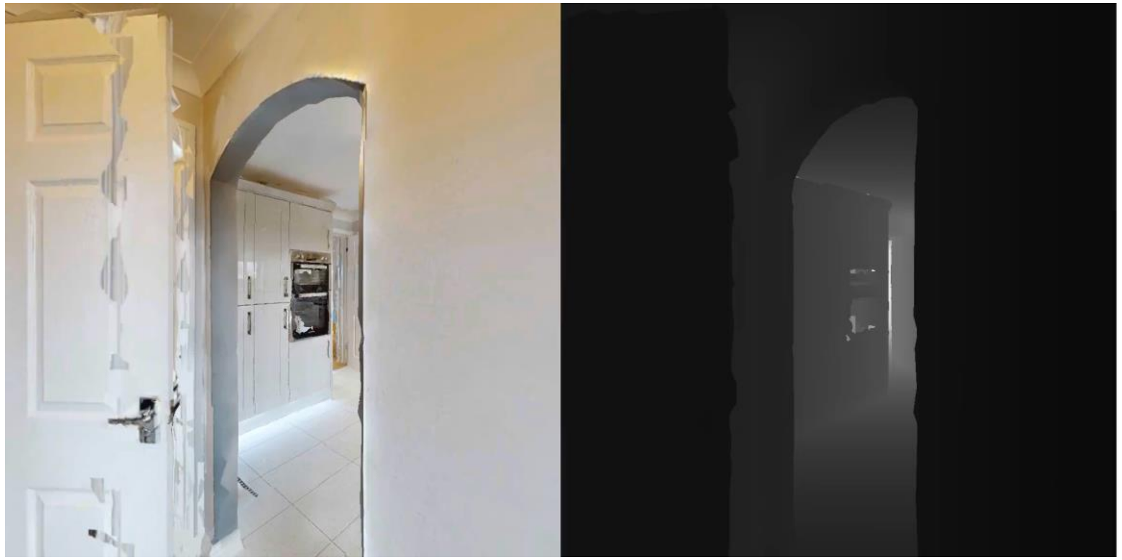
$$\mathcal{L} = \sum_{v_i \in \mathcal{V}} \left(\mathcal{L}_{i,\mathrm{BEV}} + \sum_{v_j \in \mathcal{N}(v_i)} \mathcal{L}_{i,ij,\mathrm{Pose}} \right)$$

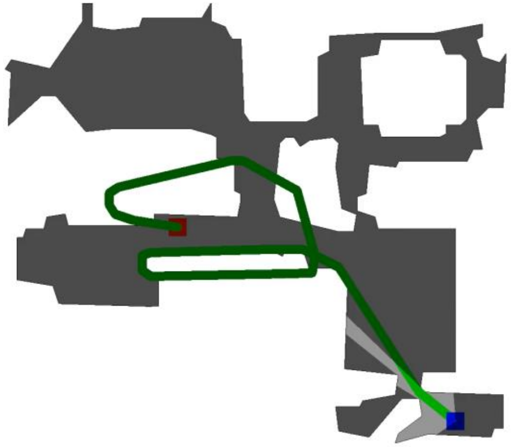
METHODOLOGY – TRAINING – DATASET

Simulated Dataset: Habitat Simulator + HM3D corpus of 800 scenes of 3D-scanned real-world multi-floor buildings

They extract from these 3,816,288 images from a calibrated camera with a 120◦ FOV

$$\mathcal{D}^{\text{Sim}}_{\text{Train}}(80\%), \qquad \mathcal{D}^{\text{Sim}}_{\text{Test}}(19\%), \qquad \mathcal{D}^{\text{Sim}}_{\text{Val}}(1\%)$$





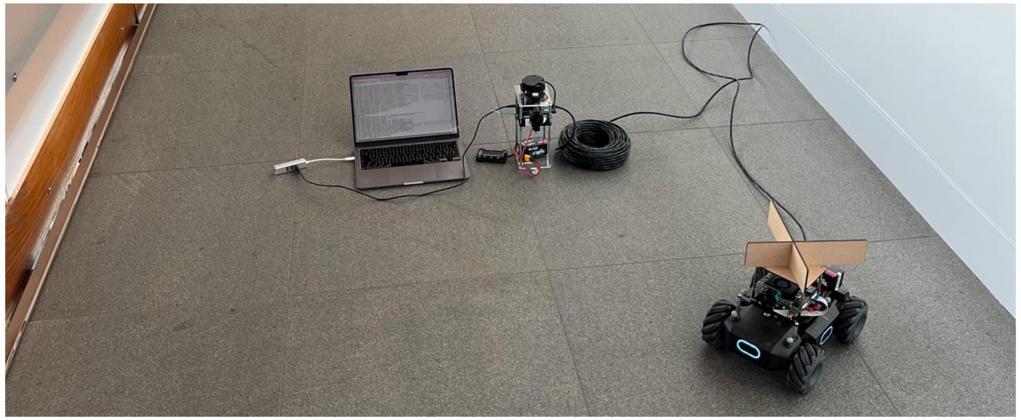
METHODOLOGY – TRAINING – DATASET

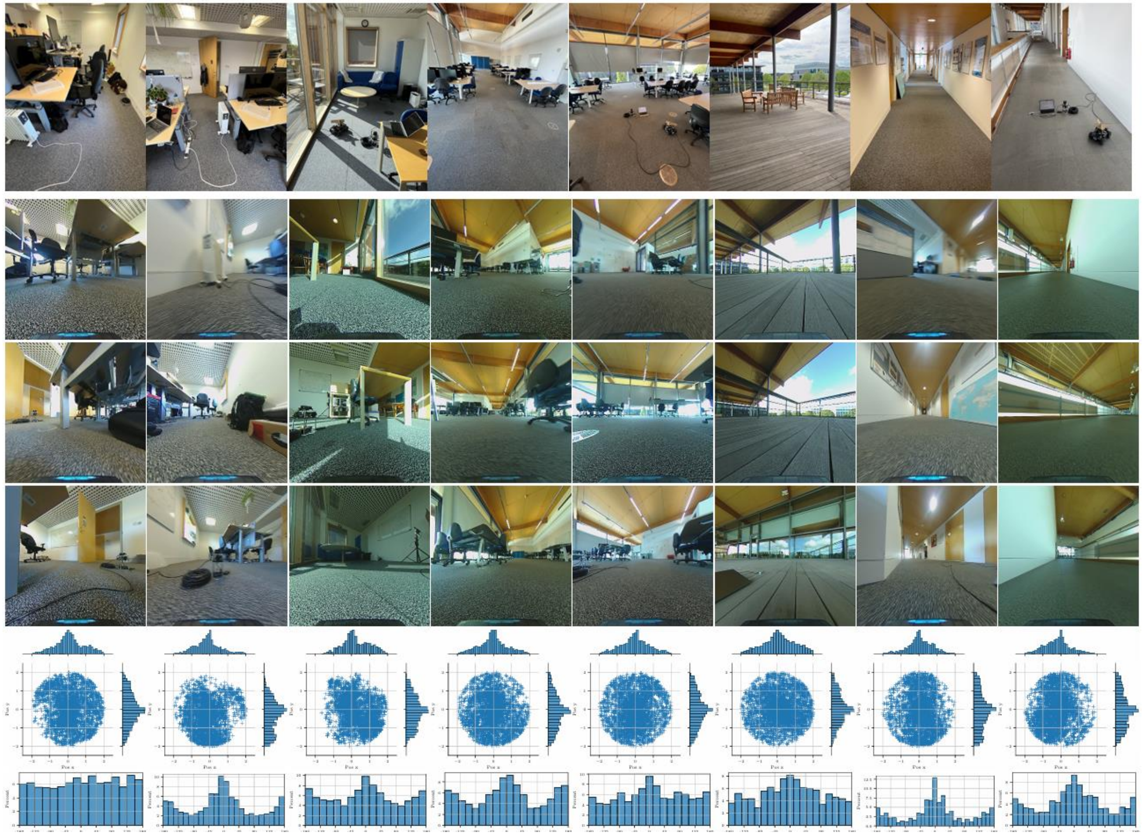
Real World Dataset: used for validation

Collected from Cambridge 4 RoboMaster and 1 Unitree go1

5692 images → 14008 sets of 3 robots

→ 84048 pose edges (32% no visual. overlap)

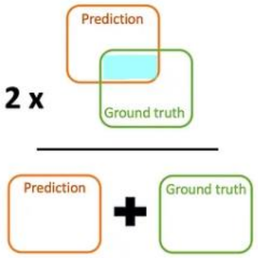




EXPERIMENTS - METRICS

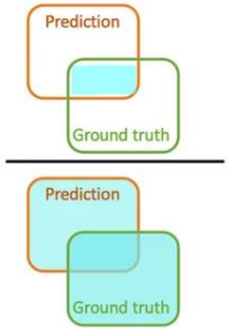
- Dice

$$\text{Dice} = \frac{2\ X\ \textit{Area of overlap}}{\textit{Total area}} =$$



- IoU

$$\text{IoU} = \frac{\textit{Area of overlap}}{\textit{Area of union}} =$$



- Euclidian distance for positions

$$D_{\text{pos}}(\mathbf{p}_{i,ij}, \hat{\mathbf{p}}_{i,ij}) = \|\mathbf{p}_{i,ij} - \hat{\mathbf{p}}_{i,ij}\|$$

- Geodesic distance for quaternions

$$D_{\text{rot}}(\mathbf{R}_{i,ij}, \hat{\mathbf{R}}_{i,ij}) = 4 \cdot \arcsin\left(\frac{1}{2} d_{\text{quat}}(\hat{\mathbf{R}}_{i,ij}, \mathbf{R}_{i,ij})\right)$$

RESULTS – ABLATION OVER SEQUENCE LENGTH AND FEATURE VECTOR SIZE

Table 1: Ablation study over the number of patches S and size of features F per patch. We report the BEV representation performance and the median error for poses on the dataset $\mathcal{D}^{\mathrm{Sim}}_{\mathrm{Test}}$ and $\mathcal{D}^{\mathrm{Real}}_{\mathrm{Test}}$.

Model		$\mathcal{D}^{\mathrm{Sim}}_{\mathrm{Test}}$				$\mathcal{D}^{\mathrm{Real}}_{\mathrm{Test}}$					
S	F	Dice	IoU	All		Invis. Filt.		Invisible		Visible	
256	48	**69.1**	**57.1**	**36 cm**	8.3°	61 cm	**6.8°**	**97 cm**	7.9°	33 cm	5.8°
128	96	68.8	56.8	40 cm	8.4°	**55 cm**	7.7°	97 cm	**7.4°**	32 cm	**5.6°**
128	48	67.9	56.1	38 cm	**7.7°**	67 cm	9.9°	113 cm	9.6°	**29 cm**	5.7°
128	24	66.7	54.6	50 cm	9.5°	83 cm	11.2°	112 cm	9.7°	31 cm	5.7°
64	48	61.0	43.5	51 cm	9.6°	81 cm	9.4°	119 cm	10.8°	36 cm	6.3°
1	3072	47.0	1.4	144 cm	89.9°	123 cm	25.7°	122 cm	25.8°	93 cm	138.2°
1	348	47.1	1.4	84 cm	11.7°	150 cm	164.1°	135 cm	37.9°	80 cm	11.1°

EXPERIMENTS – 6D DATASET

- Train **7 models** on a new $\mathcal{D}^{\mathbf{Sim}}_{\mathbf{Train6D}}$ split with:
  - **fully randomized 6-DoF** camera poses (no roll/pitch constraint)
  - **randomized FOV**
- Vary **S** and **F** as in main ablations
- Evaluate on $\mathcal{D}^{\mathbf{Sim}}_{\mathbf{Test6D}}$ and the **same real-world set** as before to compare

RESULTS – 6D DATASET

Table 4: Ablation study over the number of patches S and size of features F per patch for models trained on the dataset $\mathcal{D}^{\text{Sim}}_{\text{Train6D}}$. We report the BEV representation performance and the median error for poses on the dataset $\mathcal{D}^{\text{Sim}}_{\text{Test6D}}$ and $\mathcal{D}^{\text{Real}}_{\text{Test}}$.

Model		$\mathcal{D}^{\text{Sim}}_{\text{Test6D}}$				$\mathcal{D}^{\text{Real}}_{\text{Test}}$					
S	F	Dice	IoU	All		Invis. Filt.		Invisible		Visible	
256	48	65.1	53.8	**39 cm**	**46.5**°	**75 cm**	**9.3**°	**112 cm**	16.0°	44 cm	7.1°
128	96	64.8	53.3	42 cm	46.6°	100 cm	13.3°	119 cm	16.8°	41 cm	6.8°
128	48	64.9	53.4	43 cm	46.6°	88 cm	13.0°	113 cm	**14.9**°	**41 cm**	**6.5**°
128	24	**66.8**	**54.1**	48 cm	47.2°	107 cm	23.7°	120 cm	23.3°	45 cm	7.0°
64	48	64.3	52.7	52 cm	47.9°	138 cm	16.0°	126 cm	16.2°	48 cm	7.1°
32	24	66.0	53.5	59 cm	48.8°	166 cm	15.4°	127 cm	19.8°	55 cm	7.5°
1	48	58.3	46.4	106 cm	58.5°	153 cm	101.6°	132 cm	66.4°	102 cm	24.3°

- **Simulation**: higher rotation error vs. main split due to more non-overlap and larger maximum pose error.
- **Real-world**: performance ≈ identical with slight degradation relative to main models.

EXPERIMENTS – BASELINE FEATURE DETECTORS

Baselines:

1. **ORB/OpenCV + brute force feature matching**
2. **LightGlue learning based feature matching**

Both estimate Essential matrix → rotation + scale-less translation.

Metric: **AUC at {20°, 45°, 90°}** on **All / Invisible / Visible** splits

RESULTS – BASELINE FEATURE DETECTORS

Table 2: We report the FPS and AUC metric at 20, 45 and 90°on two baselines.

		All			Invisible			Visible		
AUC@	FPS	20	45	90	20	45	90	20	45	90
ORB/OpenCV [80]	2.79	6.45	15.43	28.26	0.16	1.60	7.43	9.41	21.92	38.03
LightGlue [42]	1.17	16.30	27.14	37.63	0.02	0.09	0.31	23.94	39.82	55.14
Ours	**42.73**	**25.23**	**47.63**	**66.34**	**9.22**	**24.40**	**45.74**	**32.74**	**58.53**	**76.01**

- Their model beats in every aspect

EXPERIMENTS – BEV ABLATIONS

Assess the effectiveness of BEV predictions, focusing on the contribution of pose predictions to enhancing BEV accuracy,

Retrained models with three BEV setups:
(1) **Pose-free** BEV model,

(2) BEV + **predicted poses** (F=48, S=128)

(3) BEV prediction + **ground-truth poses** (upper bound).

RESULTS – BEV ABLATIONS

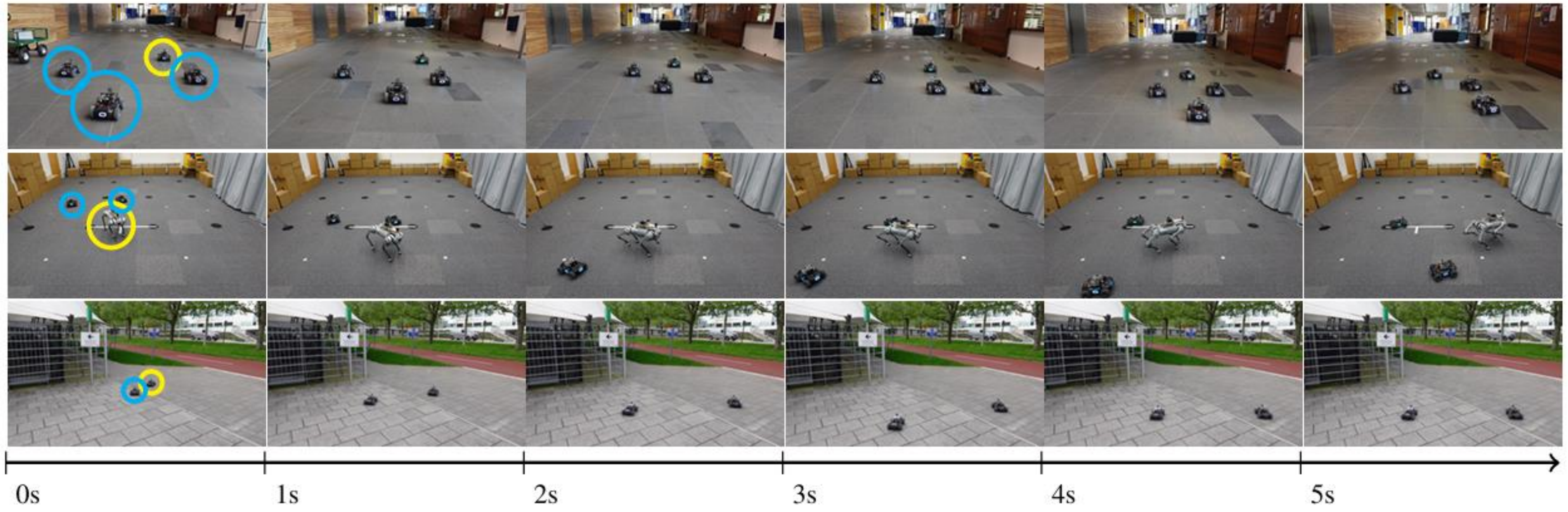
Table 3: Ablation over different modes for the BEV prediction on the simulation testset $\mathcal{D}_{\mathrm{Test}}^{\mathrm{Sim}}$.

Experiment	Dice	IoU	Median Pose Err.
None	0.628	0.495	N/A
Predicted	0.683	0.561	31 cm, 5.0°
Ground truth	0.743	0.632	0 cm, 0.0°

- **Predicted poses** → +8.75% BEV accuracy over pose-free baseline.
- **GT poses** → +18.31% over pose-free; their method sits between baseline and oracle.

EXPERIMENTS – REAL WORLD POSE CONTROL

- Deploy model (F=24, S=128) compiled with TensorRT; sub-30 ms processing.
- 15 Hz embedding exchange (~6 KiB each) over ad-hoc Wi-Fi; custom TDMA + load control to reduce losses.
- **Task**: two followers keep fixed offset to leader along reference trajectories; PD controller gates actions using predicted uncertainty (deactivates on high-uncertainty)



RESULTS – REAL WORLD POSE CONTROL

Trajectory	Robot	Mean Abs.	Median	Vel
Figure 8 dynamic	A B	38 cm, 5.6° 28 cm, 5.1°	38 cm, 4.8° 25 cm, 4.7°	0.59 m/s 0.58 m/s
Figure 8 static	A B	51 cm, 5.2° 28 cm, 5.6°	48 cm, 5.3° 26 cm, 5.4°	0.60 m/s 0.61 m/s
Rectangle dynanic	A B	41 cm, 4.4° 29 cm, 5.1°	42 cm, 4.4° 27 cm, 5.1°	0.32 m/s 0.81 m/s

Table 8: We report the mean and absolute tracking error for both leaders for all three trajectories, as well as average velocities

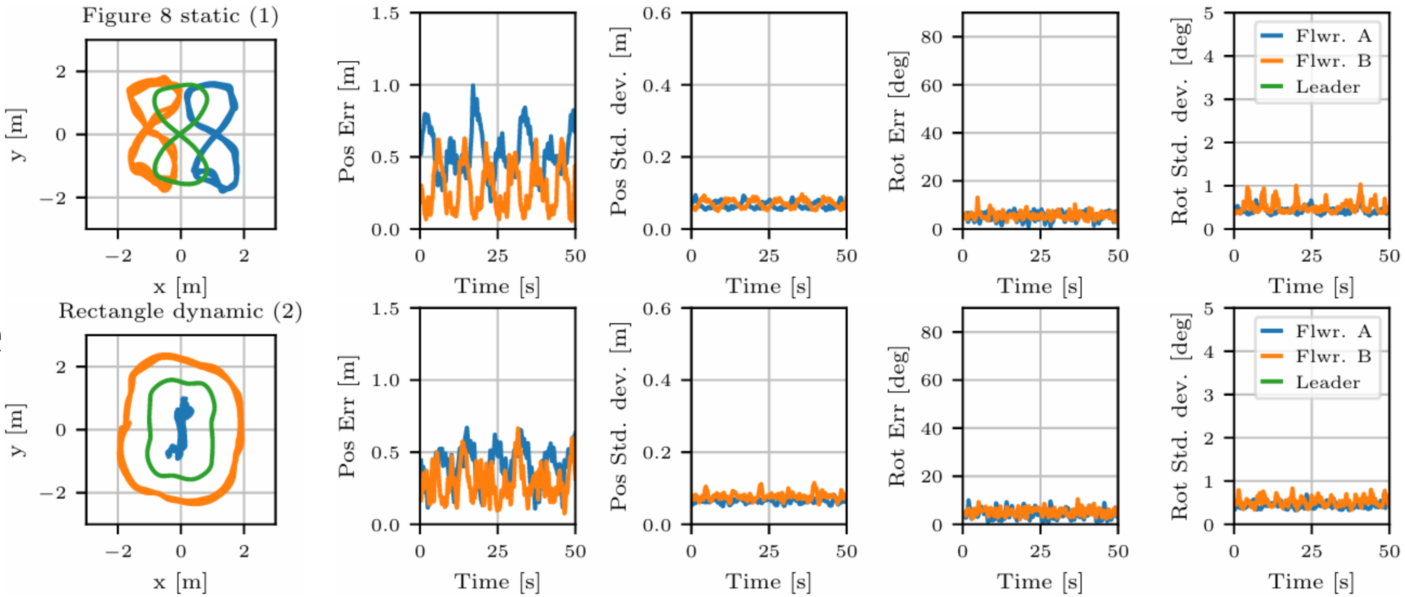


Figure 20: Tracking performance of our model and uncertainty-aware controller on two additional reference trajectories, with two follower robots (in blue and orange), positioned left and right of the leader robot (in green).

LIMITATIONS

- **Training:** Other foundation models that are trained unsupervised, theirs is trained in a supervised manner on indoor HM3D data
- **Outdoor generalization :** Works outdoors, but scale is less reliable beyond indoor domain
- **Communication:** Needs peer-to-peer networking between robots to fuse information.
- **Onboard compute**: Assumes GPU-accelerated hardware for real-time.
- **Team size:** Real-robot count limited by the custom networking stack

FURTHER WORK

- Broader pretraining / data: Move toward unsupervised/self-supervised training and add outdoor data to improve scale robustness
- Improve communication stack
- Include IMU/odometry to stabilize metric scale, espetially outdoors

QUESTIONS?

THANKS FOR LISTENING