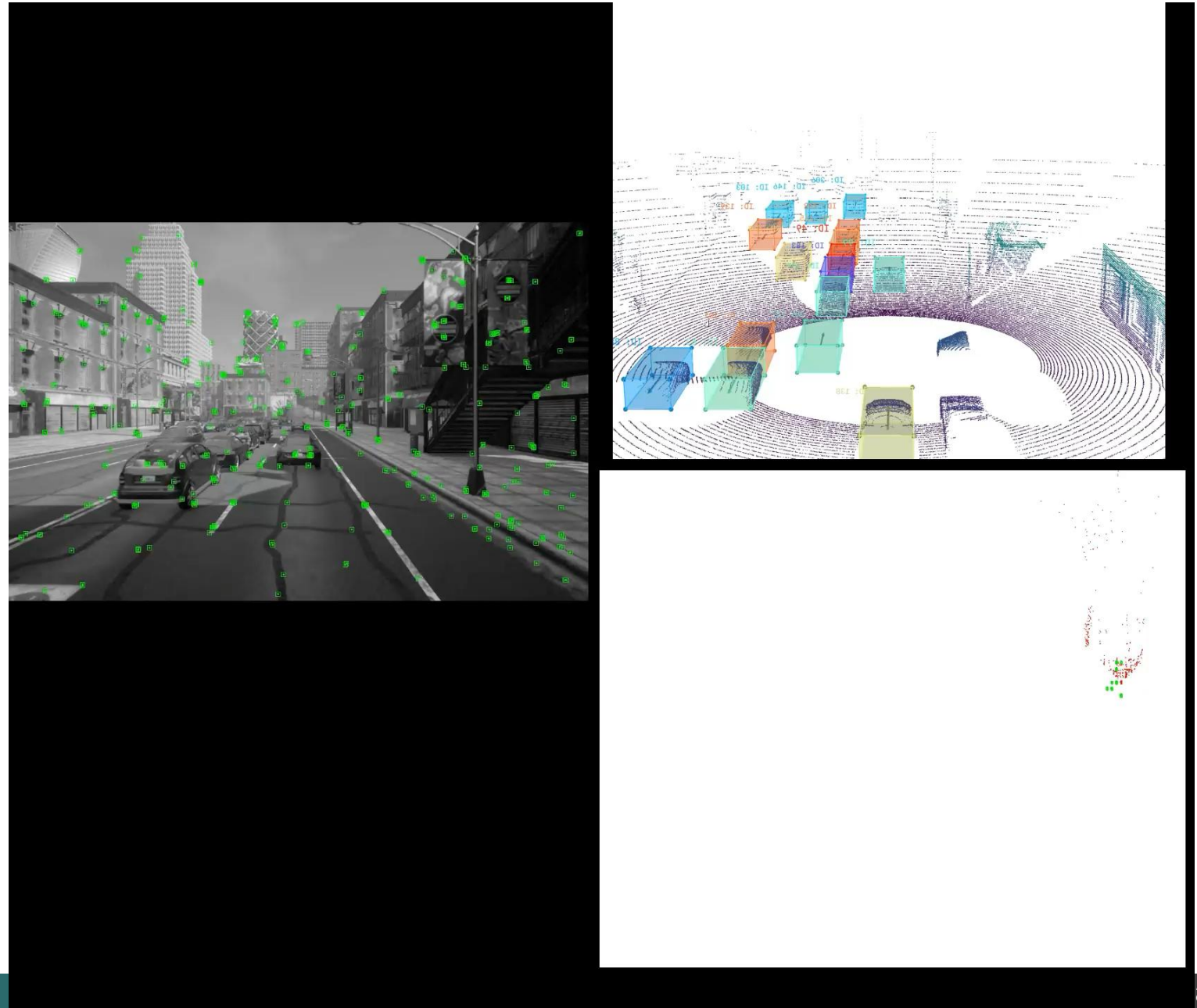
视觉SLAM从图优化到深度学习优化的发展

王永才
中国人民大学 信息学院 计算机系
ycw@ruc.edu.cn

内容提要

- 视觉里程计（Visual Odometry）问题简介
- 传统的基于特征匹配和图优化的视觉里程计
- 近两年快速发展的 深度学习视觉里程计

视觉VO、视觉SLAM问题简介



输入：连续采集的图像帧

输出：相机位姿轨迹（Locating）、重建环境的3D点云地图(Mapping)。

应用：无图自动驾驶，机器人领域

视觉SLAM主要进展

1．基于传统视觉算法提取手工特征，通过图优化进行位姿优化计算：ORB—SLAM，VINS—Mono等

2012至今
速度快、准确性好
在退化环境不鲁棒

2．基于深度学习提取特征，通过图优化进行位姿优化计算：CNN—SLAM，LIFT—SLAM，LightSLAM等

2017至今
提升特征提取的鲁棒性，但速度慢

3．直接端到端的深度学习SLAM：DeepVO，UnDeepVO等.

2017至今
效果一直不好

4．帧间图像匹配网络与位姿网络嵌套优化：TartanVO，DPVO，V2V等.

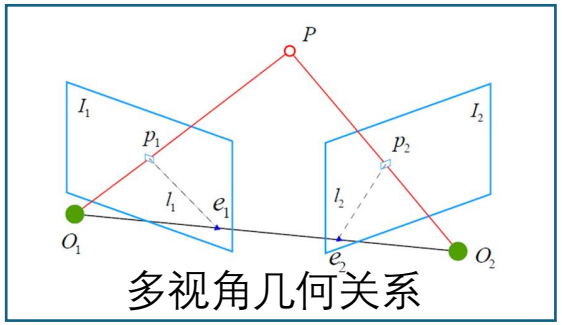
2021至今
2024年终于超越了
手工特征提取方法

传统手工特征图优化模型

传统手工特征图优化SLAM

- 相邻帧提取图像特征点，通过特征点匹配、计算相邻帧的相对位姿变换。
- 利用一个滑窗内相邻帧之间的相对位姿测量，构建一个位姿图优化问题。
- 求解位姿图优化问题，**得到最符合测量关系的所有帧的最佳位姿优化结果**。
- 代表性工作如ORB系列，VINS系列等。

特征点提取与匹配



多视角几何关系

相机相邻帧之间，以及相机到环境特征点间的距离和角度测量关系

$$\mathbf{X}^* = \arg\min_{\mathbf{X}} \sum_{(i,j)\in\mathcal{E}} \|\hat{\mathbf{Z}}_{ij} \cdot (\mathbf{X}_i^{-1}\mathbf{X}_j)\|$$

帧间相对位姿测量

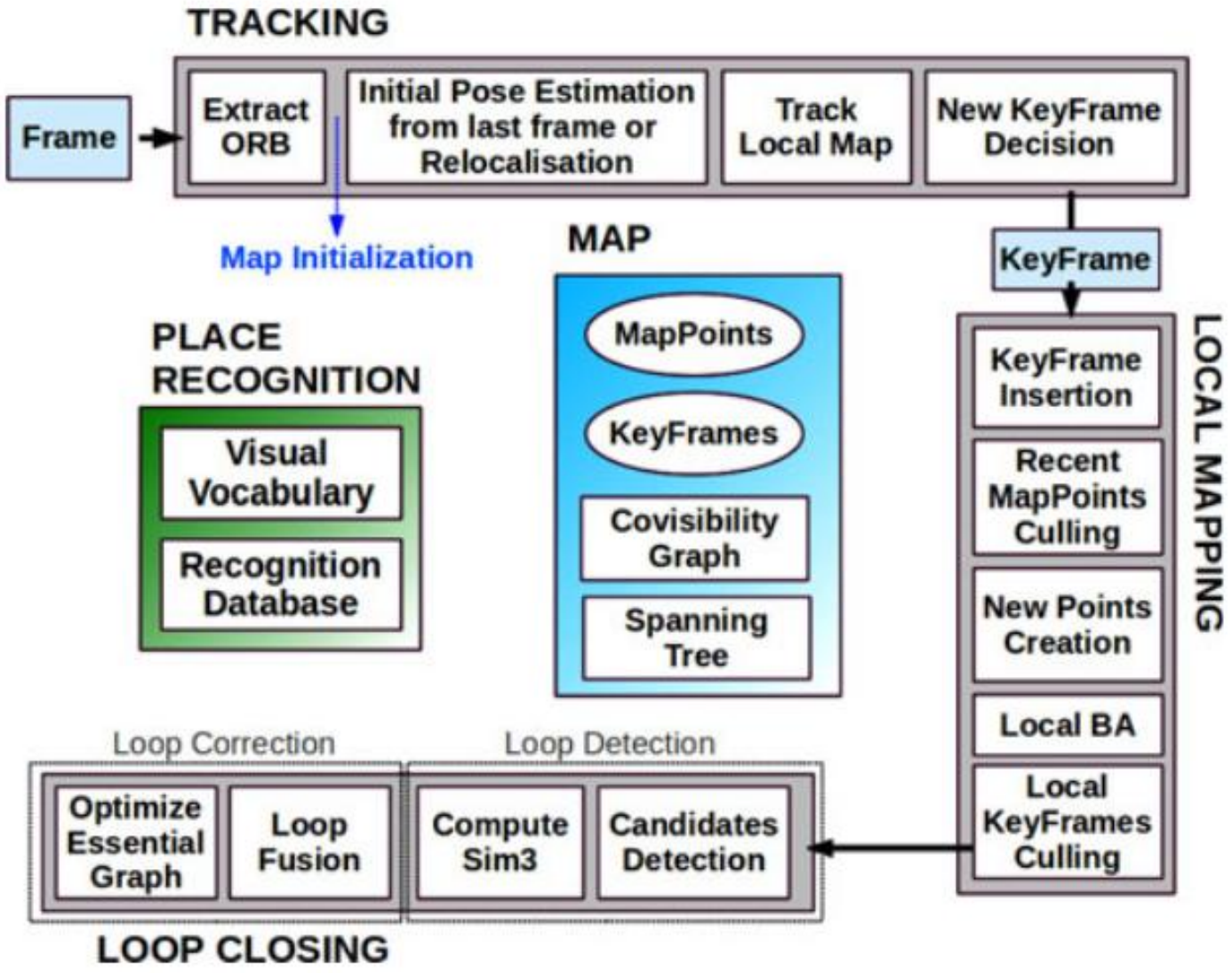
*i*帧位姿

*j*帧位姿

相邻帧之间的相对位姿测量计算 $\hat{\mathbf{Z}}_{ij}$

总体的优化目标函数

传统手工特征图优化SLAM系统



图优化SLAM代表性工作ORB–SLAM

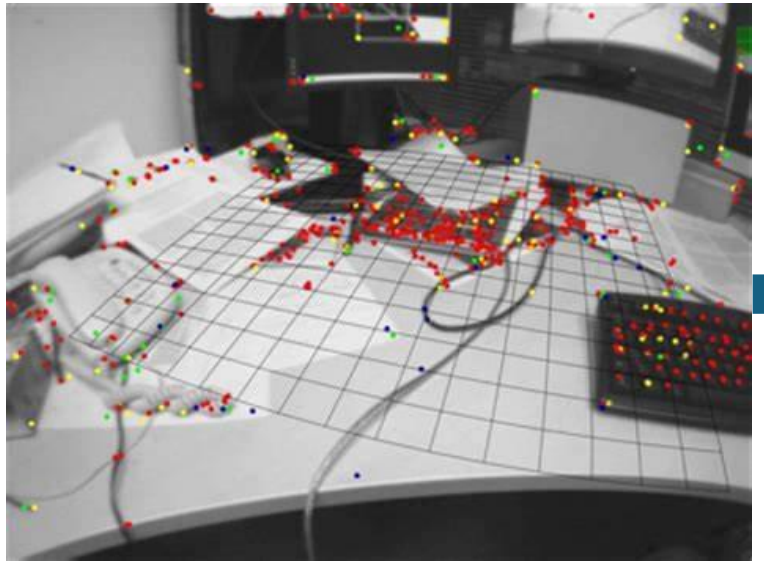
跟踪线程：提取特征点、帧间特征点匹配、相邻帧位姿变换计算

局部建图：提取关键帧、局部图优化建立局部位姿图

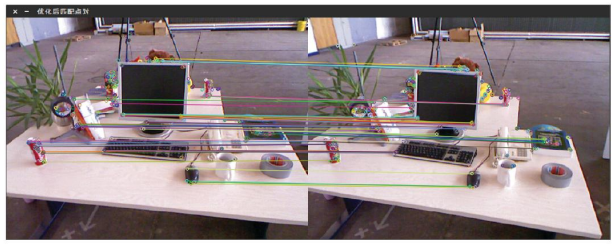
闭环检测：检测回环、进行全景地图位姿优化

位姿图优化问题

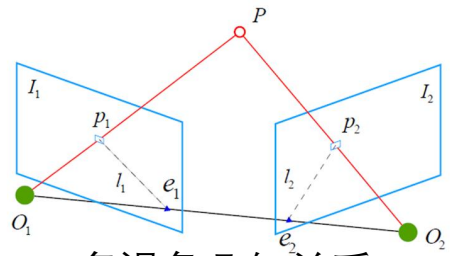
以视觉感知为例



基于多类型传感器建立Pose Graph



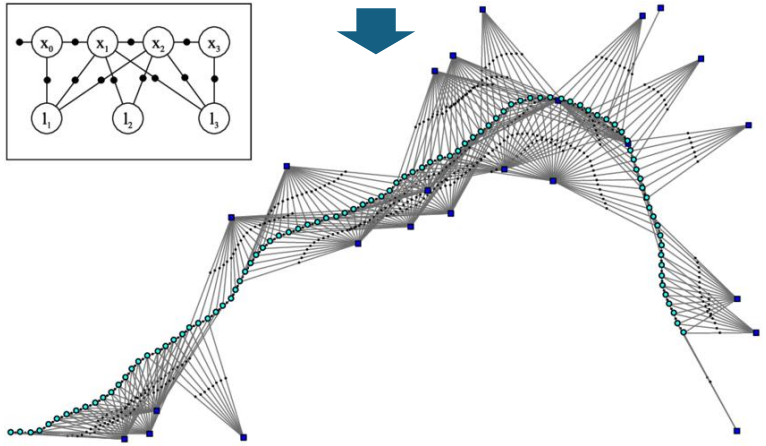
特征点提取与匹配



多视角几何关系

相机相邻帧之间，以及相机到环境特征点间的距离和角度测量关系

Pose Graph



我们近几年研究的图优化算法与工具集，(1)GPART，(2)EMI，(3)InferLoc，(4)HGO

TON2018 CCF A，TMC2020 CCF A，JSAC2018 CCF A，TON 2023 CCF A， TOSN2023 CCF B，TON2023 CCF A，TOSN2023 CCF B

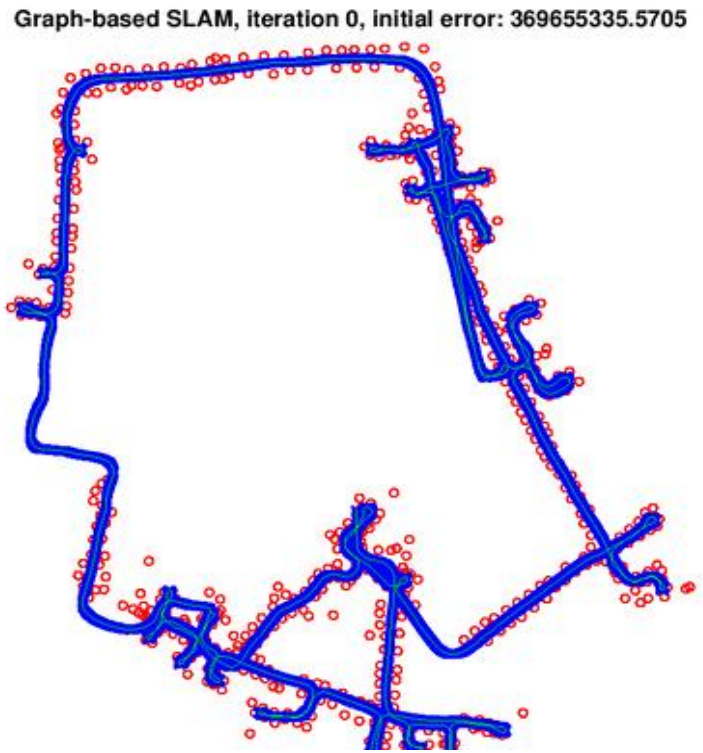
位姿图优化求解

while $\neg$converged **do**
  $\mathbf{b} \leftarrow \mathbf{0}$ $\mathbf{H} \leftarrow \mathbf{0}$
  for all $\langle \mathbf{e}_{ij}, \mathbf{\Omega}_{ij} \rangle \in \mathcal{C}$ **do**
    // Compute the Jacobians $\mathbf{A}_{ij}$ and $\mathbf{B}_{ij}$ of the error function
    $\mathbf{A}_{ij} \leftarrow \frac{\partial \mathbf{e}_{ij}(\mathbf{x})}{\partial \mathbf{x}_i}\Big|_{\mathbf{x}=\breve{\mathbf{x}}}$ $\mathbf{B}_{ij} \leftarrow \frac{\partial \mathbf{e}_{ij}(\mathbf{x})}{\partial \mathbf{x}_j}\Big|_{\mathbf{x}=\breve{\mathbf{x}}}$
    // compute the contribution of this constraint to the linear system
    $\mathbf{H}_{[ii]} \mathrel{+}= \mathbf{A}_{ij}^T \mathbf{\Omega}_{ij} \mathbf{A}_{ij}$ $\mathbf{H}_{[ij]} \mathrel{+}= \mathbf{A}_{ij}^T \mathbf{\Omega}_{ij} \mathbf{B}_{ij}$
    $\mathbf{H}_{[ji]} \mathrel{+}= \mathbf{B}_{ij}^T \mathbf{\Omega}_{ij} \mathbf{A}_{ij}$ $\mathbf{H}_{[jj]} \mathrel{+}= \mathbf{B}_{ij}^T \mathbf{\Omega}_{ij} \mathbf{B}_{ij}$
    // compute the coefficient vector
    $\mathbf{b}_{[i]} \mathrel{+}= \mathbf{A}_{ij}^T \mathbf{\Omega}_{ij} \mathbf{e}_{ij}$ $\mathbf{b}_{[j]} \mathrel{+}= \mathbf{B}_{ij}^T \mathbf{\Omega}_{ij} \mathbf{e}_{ij}$
  end for
  // keep the first node fixed
  $\mathbf{H}_{[11]} \mathrel{+}= \mathbf{I}$
  // solve the linear system using sparse Cholesky factorization
  $\mathbf{\Delta x} \leftarrow \text{solve}(\mathbf{H}\,\mathbf{\Delta x} = -\mathbf{b})$
  // update the parameters
  $\breve{\mathbf{x}} \mathrel{+}= \mathbf{\Delta x}$
end while

只求一阶导数，近似牛顿法

用cholesky分解快速求逆

位姿图优化的求解方法：高斯牛顿法



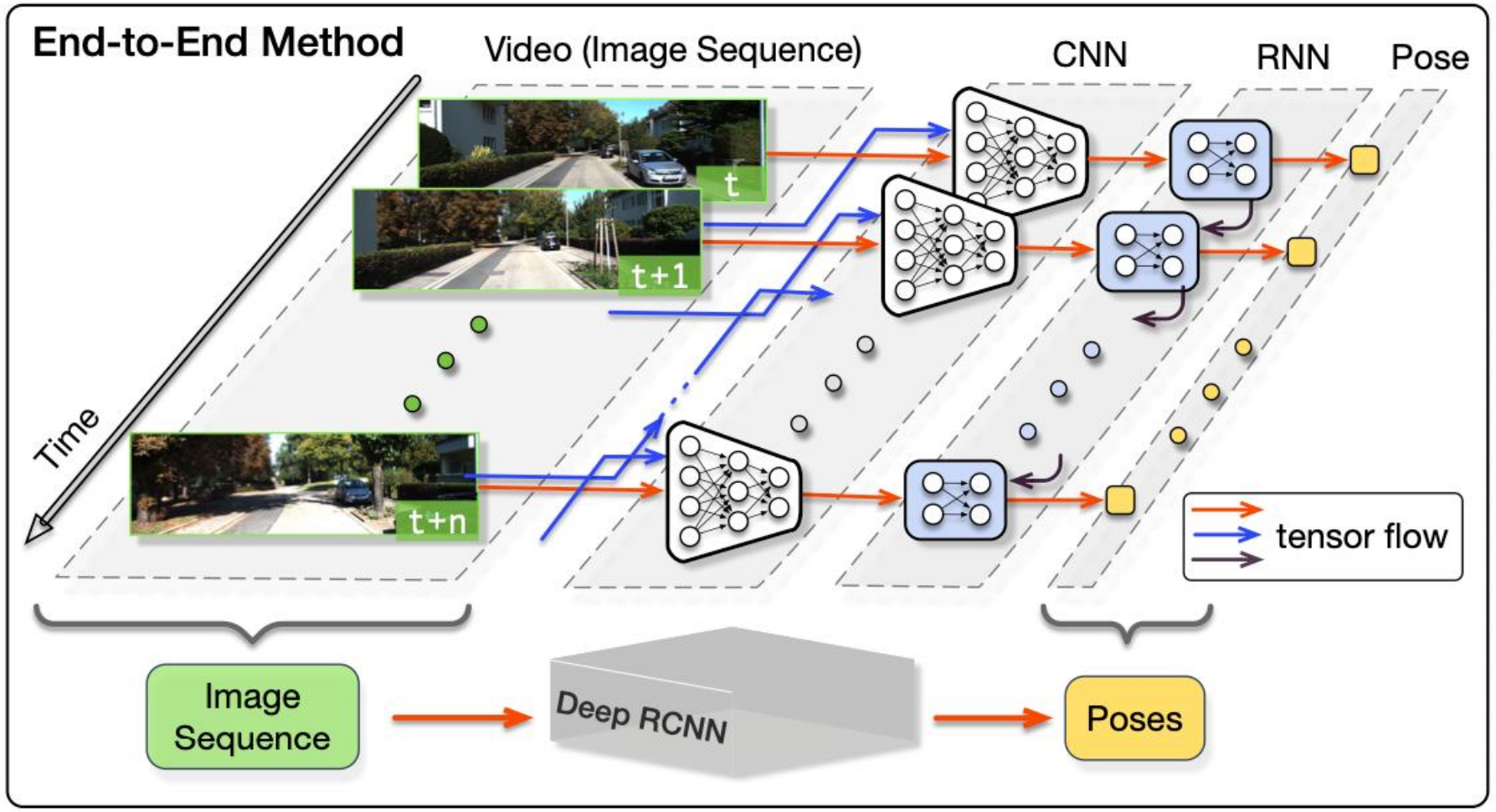
高斯牛顿求解图优化的算法

基于图优化SLAM的代表性方法的性能

	Absolute KeyFrame Trajectory RMSE (cm)			
	ORB-SLAM	PTAM	LSD-SLAM	RGBD- SLAM
fr1_xyz	**0.90**	1.15	9.00	1.34 (1.34)
fr2_xyz	0.30	**0.20**	2.15	2.61 (1.42)
fr1_floor	**2.99**	X	38.07	3.51 (3.51)
fr1_desk	**1.69**	X	10.65	2.58 (2.52)
fr2_360 _kidnap	3.81	**2.63**	X	393.3 (100.5)
fr2_desk	**0.88**	X	4.57	9.50 (3.94)
fr3_long _office	**3.45**	X	38.53	–
fr3_nstr_tex_far	ambiguity detected	4.92 / 34.74	18.31	–
fr3_nstr_ tex_near	**1.39**	2.74	7.54	–
fr3_str_tex_far	**0.77**	0.93	7.95	–
fr3_str_ tex_near	1.58	**1.04**	X	–
fr2_desk_person	**0.63**	X	31.73	6.97 (2.00)
fr3_sit_xyz	**0.79**	0.83	7.73	–
fr3_sit_halfsph	**1.34**	X	5.87	–
fr3_walk_xyz	**1.24**	X	12.44	–
fr3_walk_halfsph	**1.74**	X	X	–

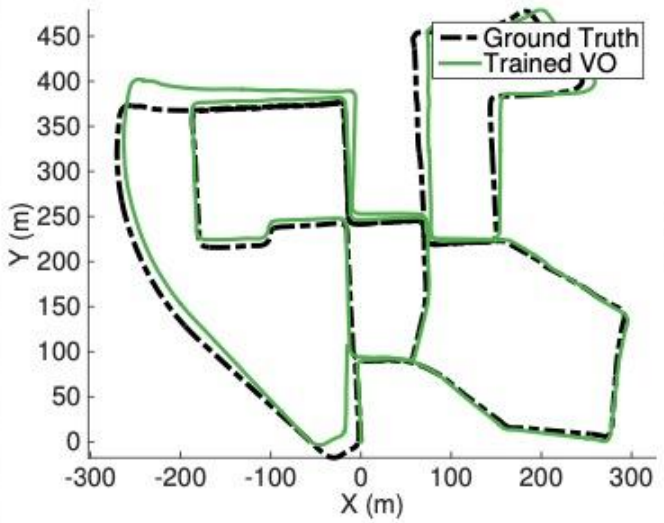
深度学习视觉SLAM

- DeepVO (ICRA2017) 第一个提出了基于深度递归卷积神经网络的端到端视觉里程计方法。

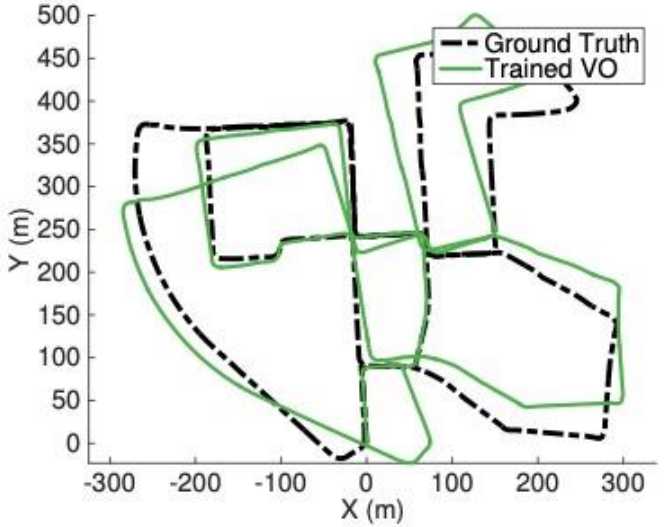


- 但是深度神经网络直接回归位姿的方法，效果并不好，远差于图优化SLAM方法

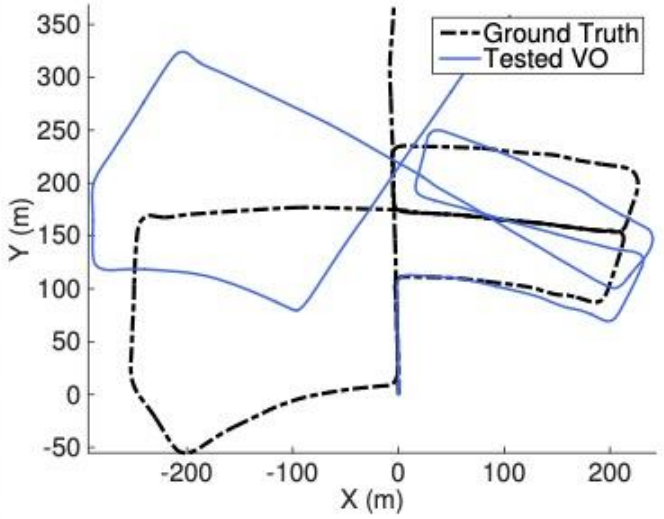
- 而且后续的各种尝试性能都没有超越传统图优化方法，直到2021年。



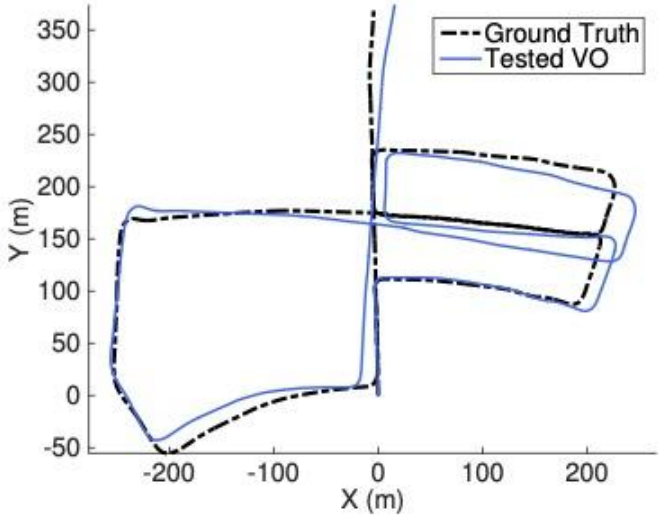
(c) Trained VO: Overfitting.



(d) Trained VO: Good Fit.

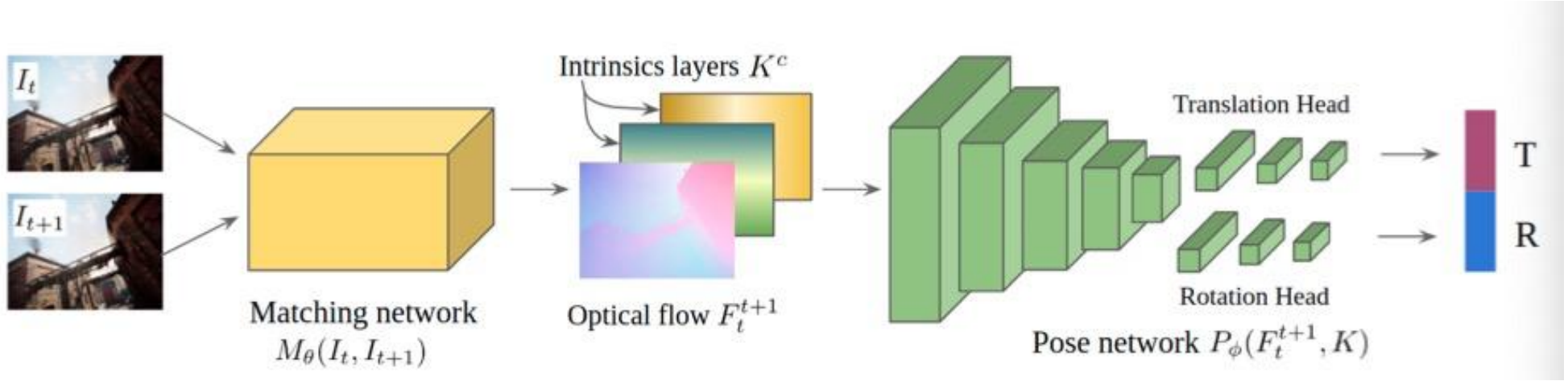


(e) Tested VO: Overfitting.



(f) Tested VO: Good Fit.

TartanVO PLMR2021



TartanVO 提出了一个两阶段的网络架构。

- **该模型包括一个匹配网络，用于从连续的两帧 RGB 图像估计光流**
- **接着是一个姿态网络，通过光流预测相机运动。**
- **TartanVO还做了图像增强，增加模型泛化性的工作**

TartanVO效果

	Seq.	MH-04	MH-05	VR1-02	VR1-03	VR2-02	VR2-03
Geometry-based *	SVO [46]	1.36	0.51	0.47	x	0.47	x
	ORB-SLAM [3]	**0.20**	0.19	x	x	**0.07**	x
	DSO [5]	0.25	**0.11**	**0.11**	0.93	0.13	1.16
	LSD-SLAM [2]	2.13	0.85	1.11	x	x	x
Learning-based †	TartanVO (ours)	0.74	0.68	0.45	**0.64**	0.67	**1.04**

* These results are from [46]. † Other learning-based methods [36] did not report numerical results.

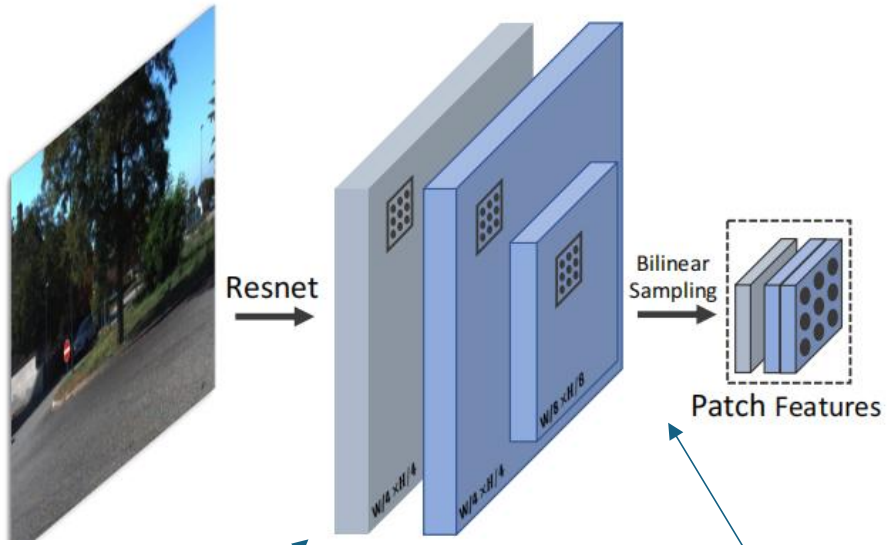
在容易的轨迹上效果不如ORB-SLAM，但能完成ORB-SLAM不能完成的轨迹

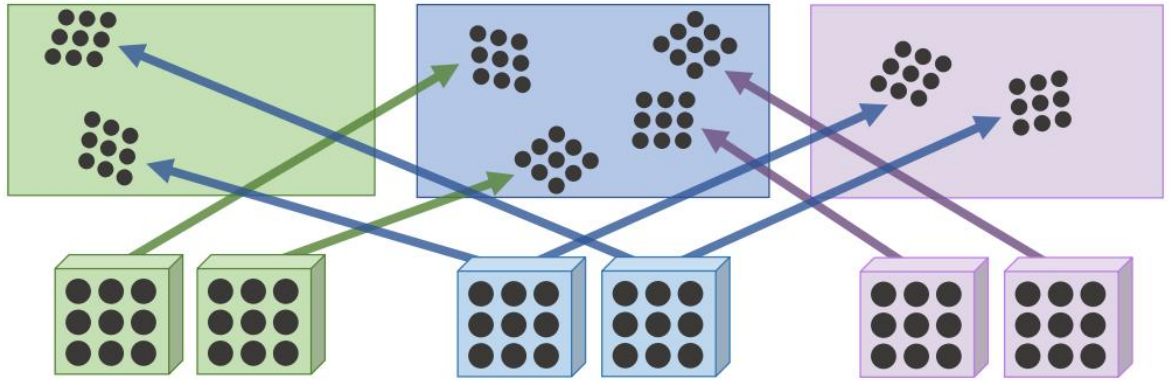
Table 4: Comparison of ATE on TartanAir dataset. These trajectories are not contained in the training set. We repeatedly run ORB-SLAM 5 times and report the best result.

Seq	MH000	MH001	MH002	MH003	MH004	MH005	MH006	MH007
ORB-SLAM [3]	**1.3**	**0.04**	2.37	2.45	x	x	21.47	2.73
TartanVO (ours)	4.88	0.26	**2**	**0.94**	**1.07**	**3.19**	**1**	**2.04**

在难度大的轨迹上，效果好于ORB-SLAM

DPVO ： Neurips2024



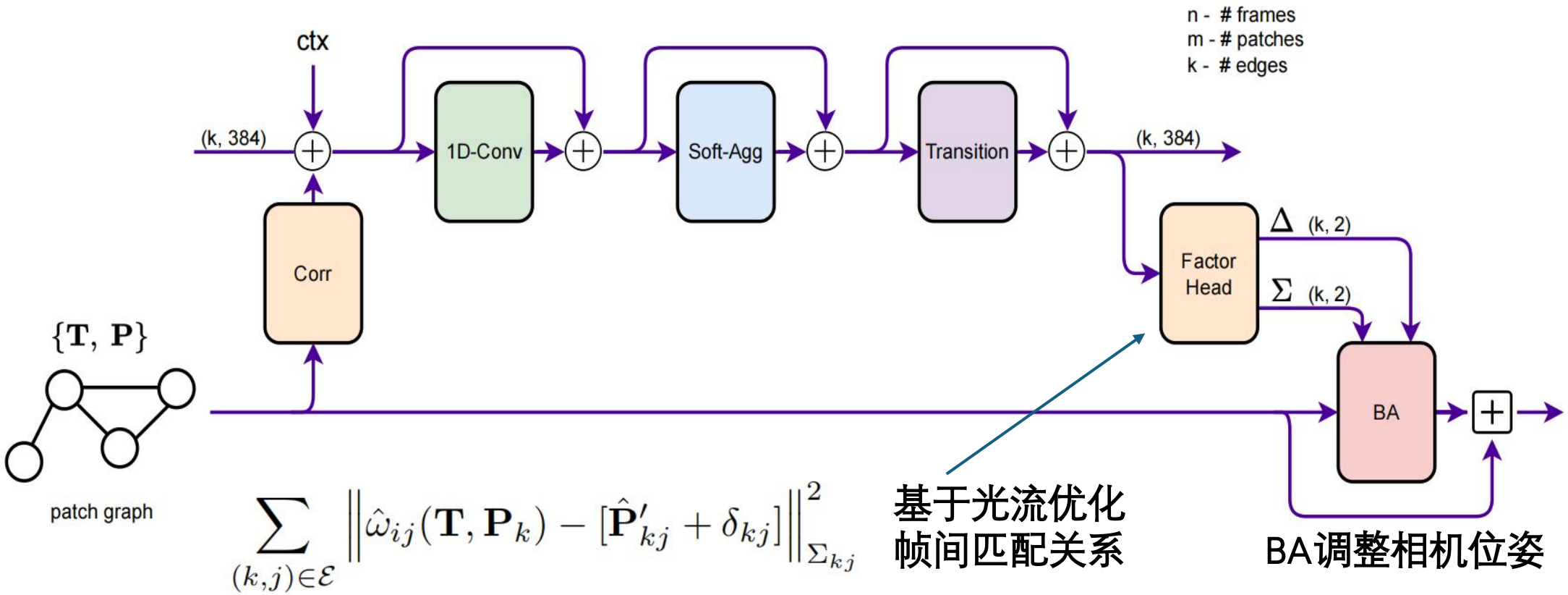


基于匹配的Patch的帧间共视关系图

DPVO： Neurips2024

DPVO的核心是一个**嵌套的深度神经网络**，这是一个循环神经网络

- 首先优化patch之间的光流匹配，建立相机位姿优化的帧间匹配约束
- 再优化每一帧相机姿态和每个图像块深度，求解相机位姿序列，完成建图



DPVO：Neurips2024

位姿监督信号:

$$\mathcal{L}_{pose} = \sum_{(i,j)\ i\neq j} \|Log_{SE(3)}[(\mathbf{G}_i^{-1}\mathbf{G}_j)^{-1}(\mathbf{T}_i^{-1}\mathbf{T}_j)]\|$$

其中**G**是位姿真值，**T**是预测的位姿序列

光流匹配监督:

$$\mathcal{L}_{\text{flow}} = \sum_{j,k} \|\mathbf{p}_{jk}^{*} - \hat{\mathbf{p}}_{jk}\|_2$$

基于光流一致性的自监督误差函数

总体监督信号:

$$\mathcal{L} = 10\mathcal{L}_{pose} + 0.1\mathcal{L}_{flow}.$$

DPVO

	360	desk	desk2	floor	plant	room	rpy	teddy	xyz	Avg
ORB-SLAM3 [27]	x	**0.017**	0.210	x	0.034	x	x	x	**0.009**	-
DSO [12]	0.173	0.567	0.916	0.080	0.121	0.379	0.058	x	0.036	-
DSO-Realtime [12]	0.172	0.718	0.728	0.068	0.167	0.767	x	x	0.031	-
DROID-VO [37]	0.161	0.028	0.099	**0.033**	**0.028**	**0.327**	**0.028**	0.169	0.013	0.098
Ours (Default)	**0.135**	0.038	**0.048**	0.040	0.036	0.394	0.034	**0.064**	0.012	**0.089**
Ours (Fast)	0.169	0.029	0.064	0.047	0.047	0.396	0.034	0.074	0.012	0.097

TUM-RGBD

	MH01	MH02	MH03	MH04	MH05	V101	V102	V103	V201	V202	V203	Avg
TartanVO [43]	0.639	0.325	0.550	1.153	1.021	0.447	0.389	0.622	0.433	0.749	1.152	0.680
SVO [15]	0.100	0.120	0.410	0.430	0.300	0.070	0.210	-	0.110	0.110	1.080	0.294
DSO [12]	**0.046**	**0.046**	0.172	3.810	**0.110**	0.089	**0.107**	0.903	**0.044**	0.132	1.152	0.601
DROID-VO [37]	0.163	0.121	0.242	0.399	0.270	0.103	0.165	0.158	0.102	0.115	**0.204**	0.186
Ours (Default)	0.087	0.055	**0.158**	**0.137**	0.114	**0.050**	0.140	**0.086**	0.057	**0.049**	0.211	**0.105**
Ours (Fast)	0.101	0.067	0.177	0.181	0.123	0.053	0.158	0.095	0.095	0.063	0.310	0.129

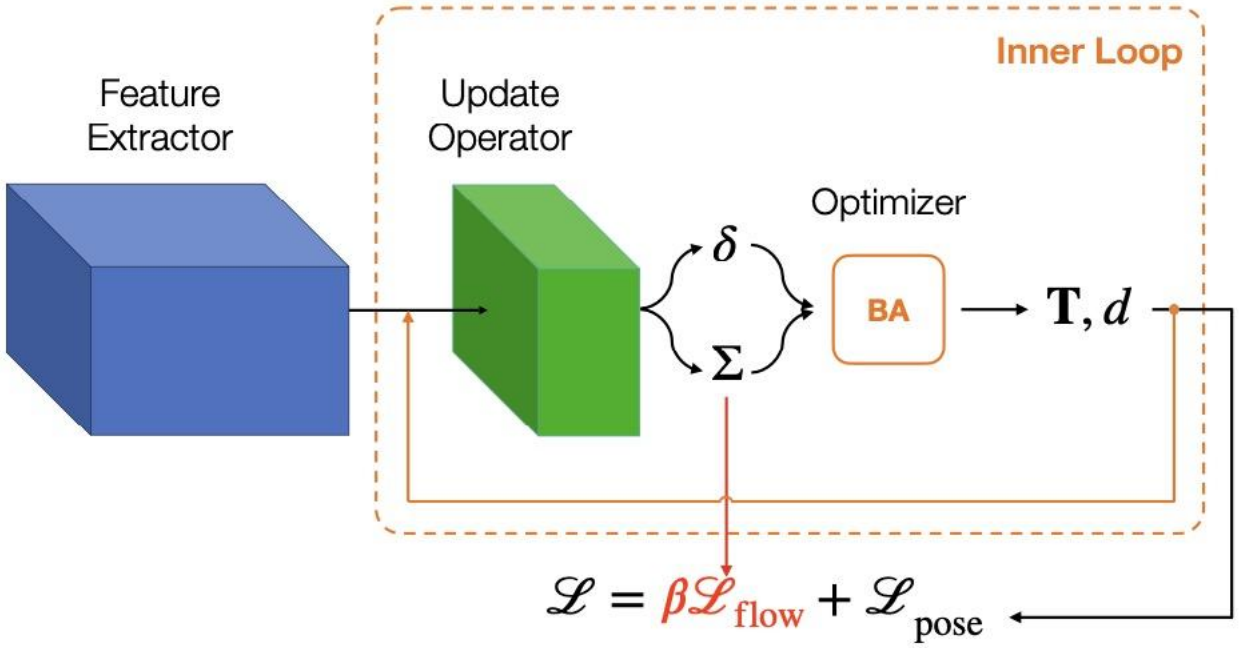
EUROC

DPVO：Neurips2024

训练细节：在一台单独的 RTX-3090 GPU 上进行总共 240k 次迭代的训练，批次大小为 1。训练过程持续了 3.5 天。

仿真数据集上训练的模型在各类实际数据集上进行测试。

From Variance to Veracity (V2V): Unbundling and Mitigating Gradient Variance in Differentiable Bundle Adjustment Layers CVPR2024



- 深入分析了嵌套式的深度优化网络收敛慢的原因
- 主要原因来自于**异常匹配数据带来的梯度方差不稳定**

1．用光流匹配输出的权重对光流loss的计算进行加权

$$\mathcal{L}_{\text{flow}} = \sum_{j,k} \|\mathbf{p}^*_{jk} - \hat{\mathbf{p}}_{jk}\|_{\Sigma^{\perp}_{jk}}$$

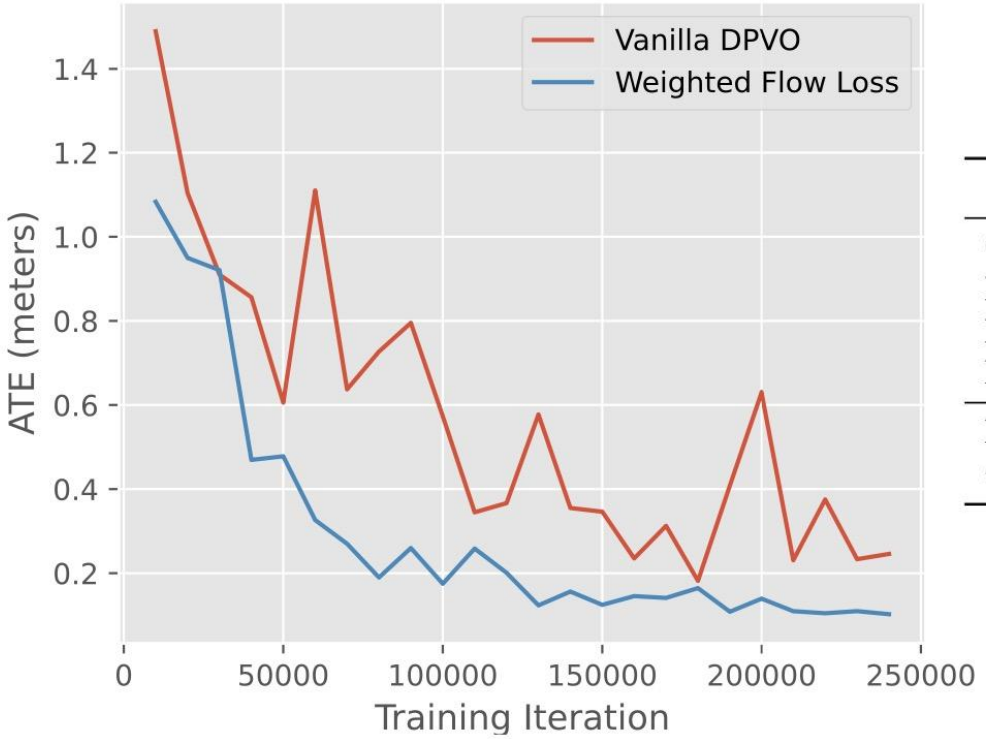
提高可信的匹配的影响力

2．自适应的loss比值函数

$$\beta = \frac{\|\nabla_\theta \mathcal{L}_{\text{pose}}\|_2}{\|\nabla_\theta \mathcal{L}_{\text{flow}}\|_2} \quad \mathcal{L} = \mathcal{L}_{\text{pose}} + \beta\mathcal{L}_{\text{flow}}$$

自动平衡两种loss的影响

From Variance to Veracity: Unbundling and Mitigating Gradient Variance in Differentiable Bundle Adjustment Layers, CVPR2024

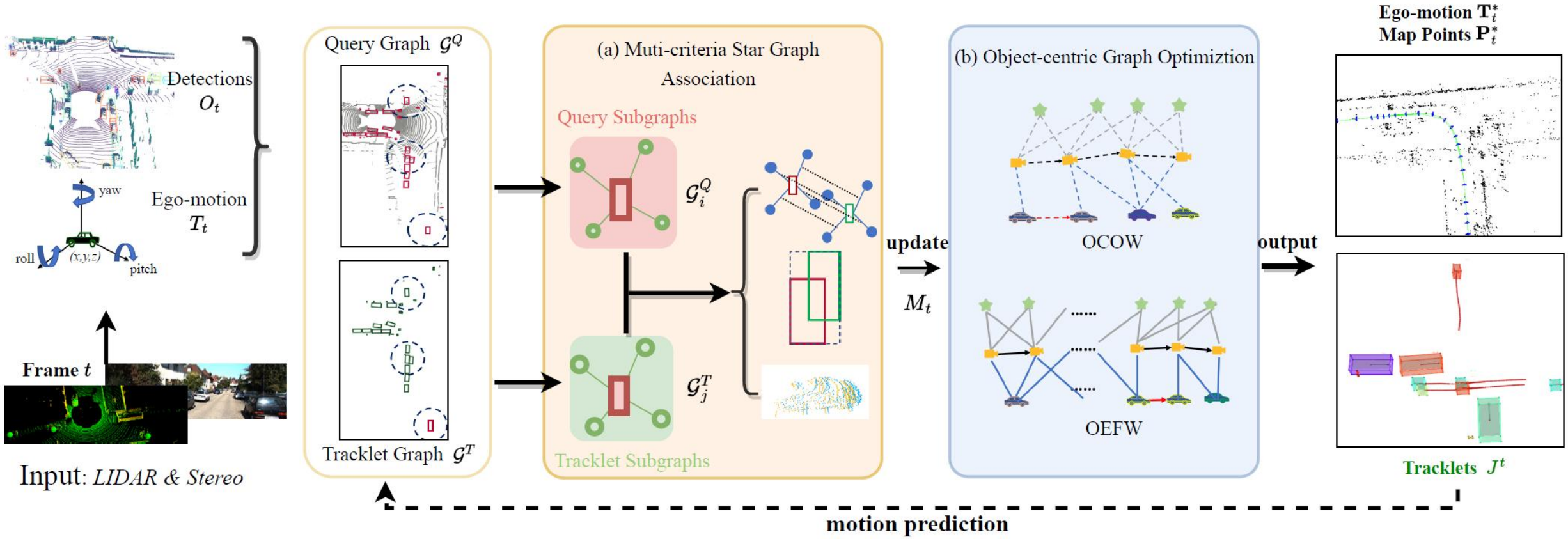


收敛速度提高3倍
一天多可训练完

	360	desk	desk2	floor	plant	room	rpy	teddy	xyz	Avg
ORB-SLAM3 [14]	x	**0.017**	0.210	x	0.034	x	x	x	**0.009**	-
DSO [24]	0.173	0.567	0.916	0.080	0.121	0.379	0.058	x	0.036	-
DSO-Realtime [24]	0.172	0.718	0.728	0.068	0.167	0.767	x	x	0.031	-
DROID-VO [54]	0.161	0.028	0.099	**0.033**	**0.028**	**0.327**	**0.028**	0.169	0.013	0.098
DPVO	**0.135**	0.038	0.048	0.040	0.036	0.394	0.034	0.064	0.012	**0.089**
Ours	0.145	0.026	**0.044**	0.064	0.031	0.434	0.045	**0.046**	0.012	0.094

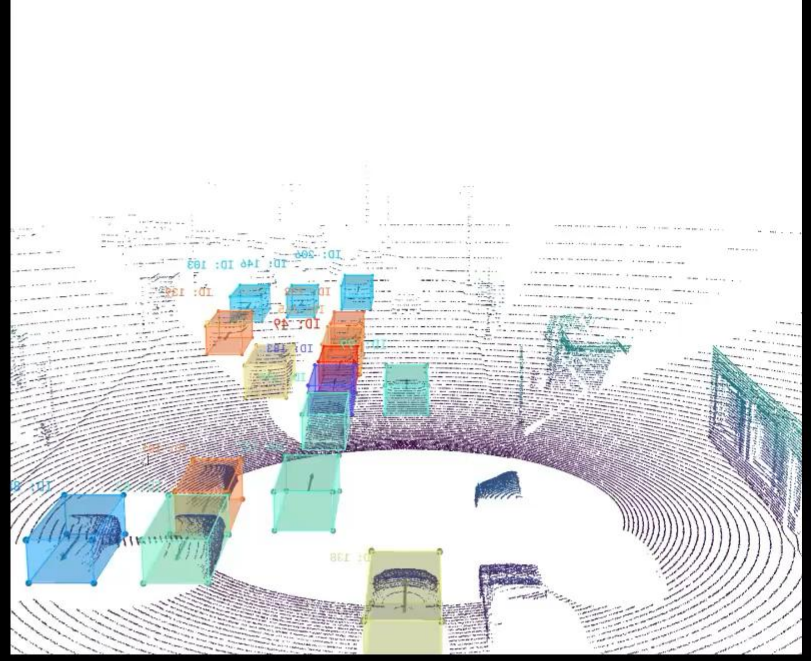
准确性同DPVO不相上下

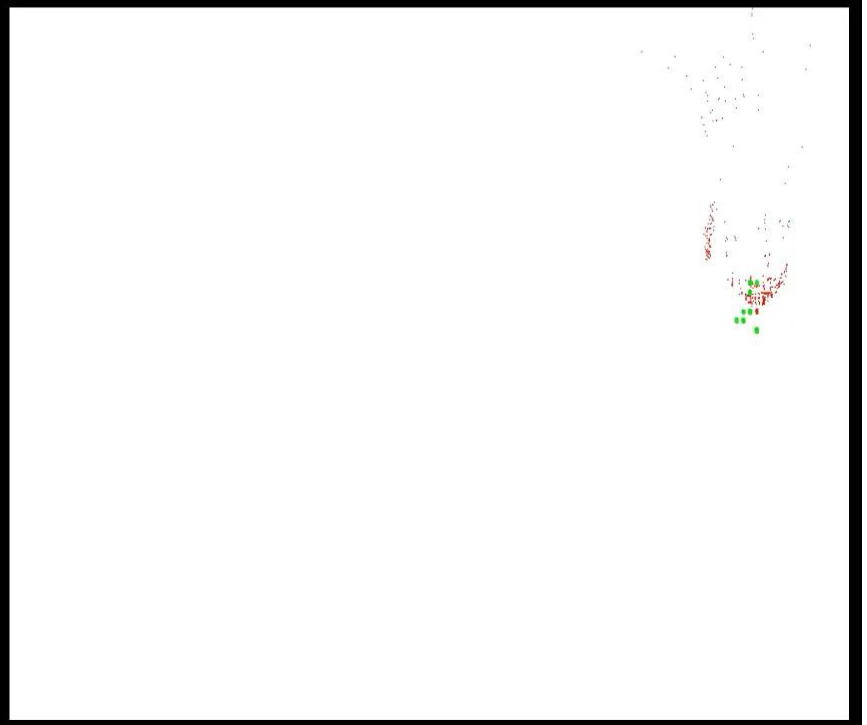
我们的工作：图优化SLAM方面，同步定位建图与目标追踪



输入　　目标检测　　当前帧查询历史目标轨迹图　　自身轨迹、建图、多目标跟踪结果

GSLAMOT: A Tracklet and Query Graph-based Simultaneous Locating, Mapping, and Multiple Object Tracking System, Shuo Wang, Yongcai Wang et al., ACM Multimedia 2024, CCF A





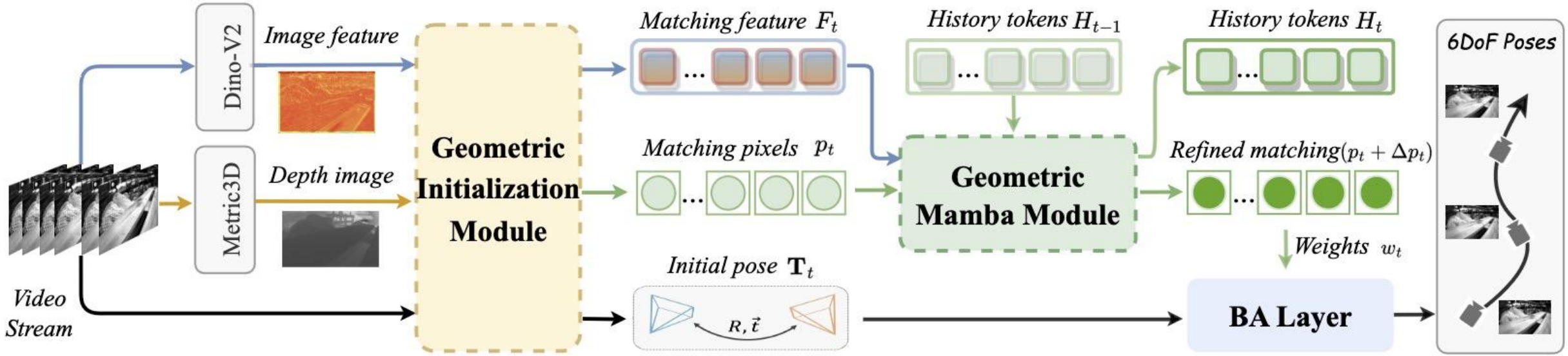
录像展示

Speed X 10

多目标追踪效果，
不同颜色代表追踪
的不同目标.

- Green box: tracklet
- Red point: local map point
- Black point: map point

我们的工作：深度学习SLAM，MambaVO



Method	AUC@1°↑	AUC@2°↑	AUC@5°↑	AUC@10°↑
DROID-VO[48]	0.123	0.167	0.294	0.53
DPVO[50]	0.299	0.485	0.738	0.925
V2V[21]	0.399	0.584	0.855	0.957
MambaVO (Ours)	**0.471**	**0.678**	**0.892**	**0.975**

在匹配方面准确性有显著提升

我们的工作：MambaVO

平均准确性

	Method	MH000	MH001	MH002	MH003	MH004	MH005	MH006	MH007	AVG
w. loop	ORBSLAM3[3]	15.44	2.92	13.51	8.18	2.59	21.91	11.7	25.88	14.38
	COLMAP[43]	12.26	13.45	13.45	20.95	24.97	16.79	7.01	7.97	12.5
	DeFlowSLAM[57]	0.63	0.06	**0.02**	**0.01**	2.8	0.2	0.31	0.45	0.56
	DROID-SLAM[48]	**0.08**	0.05	0.04	0.02	**0.01**	0.68	0.3	0.07	0.33
	DPV-SLAM[33]	0.23	0.05	0.04	0.04	0.54	0.15	0.07	0.14	0.16
	MambaVO++(Ours)	0.12	**0.04**	**0.02**	0.02	0.37	**0.14**	**0.05**	**0.05**	**0.10**
w.o. loop	TartanVO[54]	4.88	0.26	2	0.94	1.07	3.19	1	2.04	1.92
	DROID-VO[48]	0.32	0.13	0.08	0.09	1.52	0.69	0.39	0.97	0.58
	DPVO[50]	0.21	0.04	0.04	0.08	0.58	**0.17**	0.11	0.15	0.17
	V2V[21]	0.18	0.03	**0.03**	**0.02**	0.58	0.3	**0.08**	**0.05**	0.18
	MambaVO (Ours)	0.24	**0.02**	**0.03**	**0.02**	**0.46**	0.18	0.13	**0.05**	**0.14**

整体准确性上好于DPVO，V2V等现有最好方法，投稿至CVPR2025

总结

- 2024年深度学习SLAM终于从准确性上超越了图优化SLAM
- 泛化性、挑战性环境的普遍适用性是它的优势。
- 但速度上还是它的最大劣势，但似乎已经找到了一个合适的框架。
- 现有的降低嵌套优化梯度方差的手段都还很简单和初步。
- 如何继续提升深度学习SLAM还是很有挑战的问题。

谢谢，Q&A

ycw@ruc.edu.cn

18910215881

http://www.yongcaiwang.cn