Technical University of Munich

Department of Informatics
Guided Research Project

Local and Global Mapping for Direct SLAM

Erkam Uyanik

Advisor: Nikolaus Demmel

Examiner: Prof. Daniel Cremers

Problem & Related Work

Goal

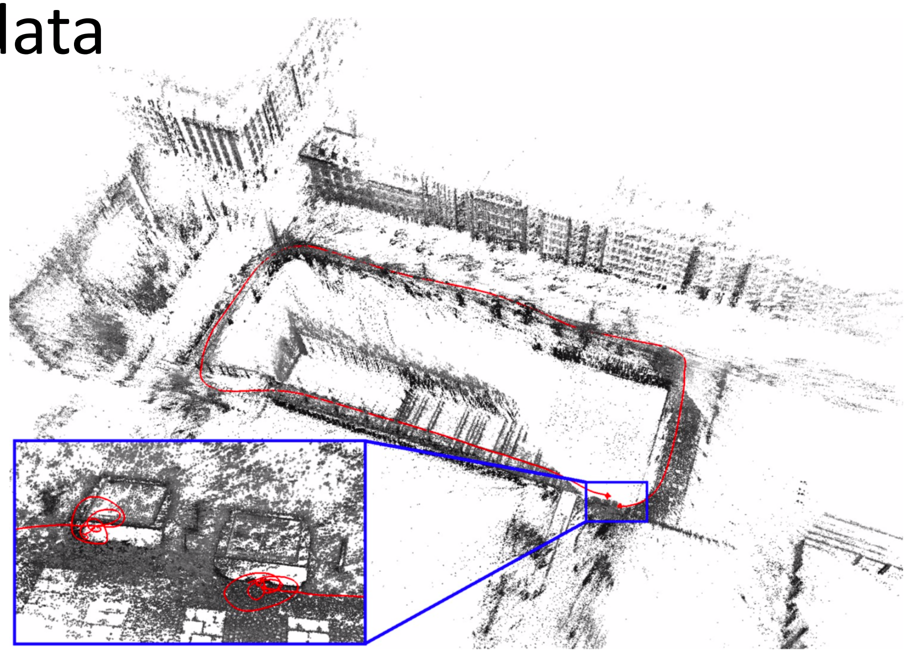
- a direct SLAM system with global mapping and loop closure

Direct Sparse Odometry (DSO) [1]

- direct VO
- advantages of direct approaches & sparse data

- uses a sliding window
- marginalizes frames and points

- accumulated drift on long trajectories



[1] J. Engel, V. Koltun, and D. Cremers, “Direct sparse odometry,” PAMI, 2017.

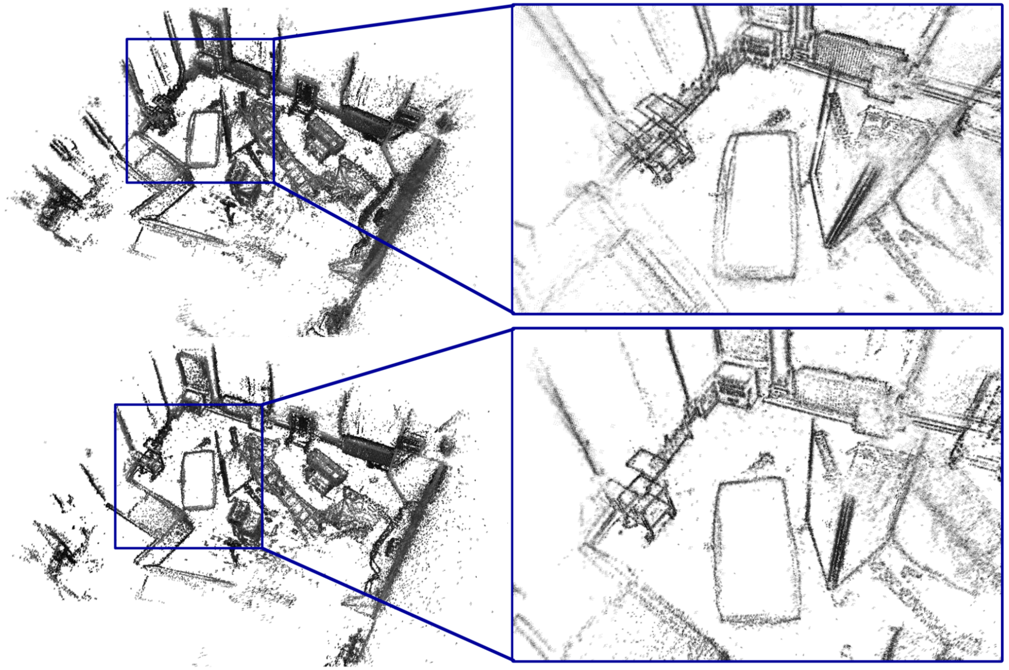
Direct Sparse Odometry with Loop Closure (LDSO) [2]

- extends DSO to a VSLAM system with loop closure detection and pose graph optimization
  - uses typical indirect SLAM system approaches
  - detecting revisited scenes helps reduce drift

- reobservations are not used in sliding window optimization

[2] X. Gao, R. Wang, N. Demmel, and D. Cremers, “Ldso: Direct sparse odometry with loop closure,” in IROS, 2018.

Direct Sparse Mapping (DSM) [3]



- direct VSLAM
- global map
- Local Map Covisibility Window (LMCW)
- A coarse-to-fine optimization scheme
- A robust influence function & outlier management using t-distribution

[3] J. Zubizarreta, I. Aguinaga, and J. M. M. Montiel, “Direct sparse mapping,” IEEE Transactions on Robotics, 2020.

Direct Sparse Mapping (DSM) [3]

- only local PBA
- no explicit compensation of the drift for larger loops
  - keyframe reuse is limited to local mapping and smaller loops
- lacks loop closure correction

[3] J. Zubizarreta, I. Aguinaga, and J. M. M. Montiel, “Direct sparse mapping,” IEEE Transactions on Robotics, 2020.

Goal

- So far we have seen
  - DSO: a direct VO
  - LDSO: a direct SLAM with loop closure
  - DSM: a direct SLAM with global mapping
- Propose a direct SLAM system with global mapping and loop closure
  - extend DSM by introducing loop closure detection and pose graph optimization
  - Similar to how LDSO extends DSO

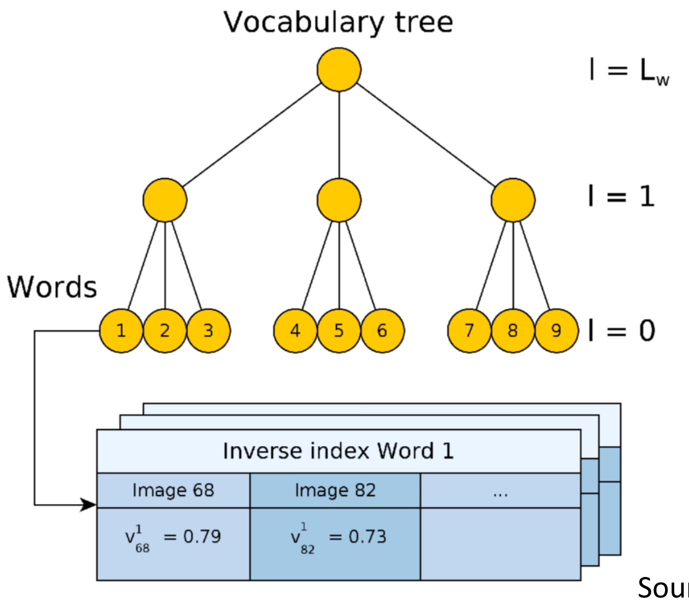
Proposed Method

Loop Closure Detection

- Visual bag of words methods are commonly used for loop detection

- We will use DBoW3 [4]
  - converts images into bag of words vectors
  - implements a database enabling queries

- For this we need repeatable feature points



Source: [5]

[4] R. Muñoz-Salinas, DBoW3, https://github.com/rmsalinas/DBoW3
[5] D. Gálvez-López and J. D. Tardos, "Bags of binary words for fast place recognition in image sequences," IEEE Transactions on Robotics, 2012.

Feature Point Selection

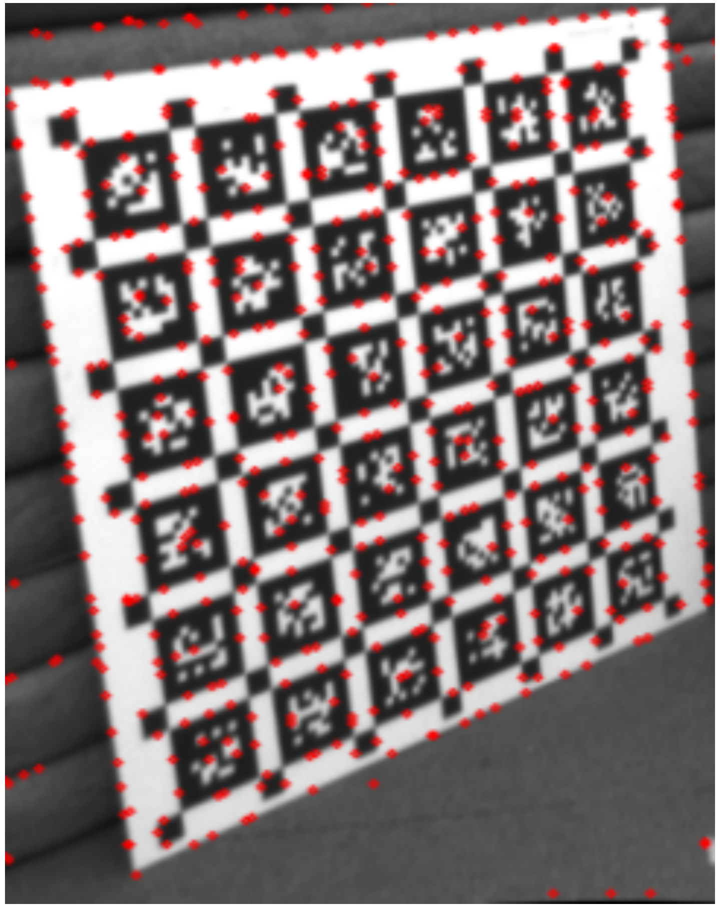
- DSM uses image gradients to select points
  - not usually repeatable, e.g. points from weeekly textured regions or edges
- We need to introduce repeatable feature points such as corners
  - Select a small percentage of points in this way

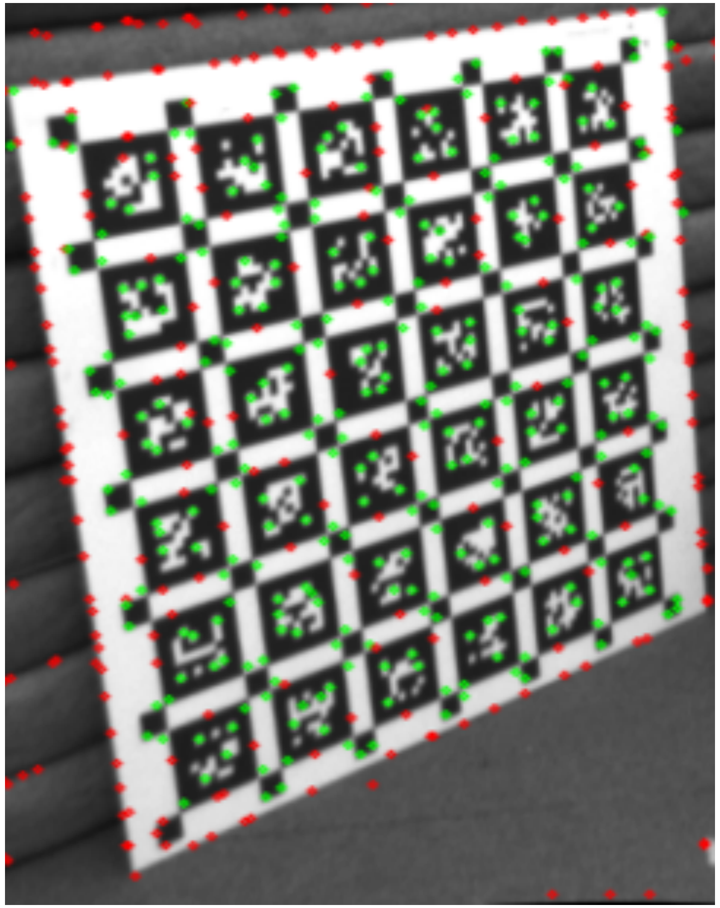
Feature Point Selection

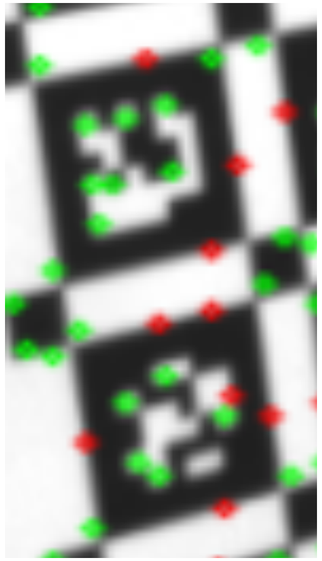
- DSM uses a grid approach to select points
  - points are homogeneously distributed
- Use the same grid approach to select feature points
  - selected corners are still usable by DSM
- Do not select non-feature points for cells if they already contain any feature points
- Use FAST corner detector [6] and ORB descriptors [7]

[6] E. Rosten and T. Drummond, "Machine learning for high-speed corner detection," in ECCV, 2006
[7] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, "Orb: An efficient alternative to sift or surf," in ECCV, 2011

Feature Point Selection







Loop Closure Detection

- Compute bag of words representation of currrent keyframe (I_{ref})
- query database for similar keyframes
- Filter active keyframes and covisible keyframes of I_{ref} from query results
- Select the candidate with the best score (I_{cand})
  - Future work: Take all candidates whose score is above an adaptive threshold into consideration

Relative Pose Estimation

- Find ORB matches using DBoW3
  - match features that correspond to the same word of vocabulary tree
  - approximate but faster than brute-force

- Estimate relative pose between I_{ref} and I_{cand} using PnP RANSAC
  - use active points and optimized candidate points as 3D points
  - apply PnP RANSAC two way to get relative scale

Relative Pose Optimization

- Find additional matches
  - project points from I_{cand} to I_{ref}
  - for each projected point, select best point among the nearby points of I_{ref}
  - mainly rely on spatial information

- Use these additional matches to optimize relative pose

Relative Pose Optimization

- $E_{LOOP} = \sum_{q_i \in Q}(|p_i - S_{rc} * q_i|_\gamma + |\Pi(p_i) - \Pi(S_{rc} * q_i)|_\gamma)$
  - S_{rc}: Sim(3) transformation from I_{cand} to I_{ref}
  - $Q = \{q_i\}$: matched 3D points from I_{cand}
  - $P = \{p_i\}$: matched 3D points from I_{ref}
  - $\Pi(\cdot)$: projection function, $|\cdot|_\gamma$: Huber norm

- Use Ceres [8] for optimization

[8] S. Agarwal, K. Mierle, and others, Ceres solver, http://ceres-solver.org

Relative Pose Optimization

- Solve the problem again with only using inliers
  - get inliers using fixed thresholds for residuals

- Check mean residual to determine whether we have a good case for loop closure

Pose Graph Optimization (PGO)

- We run PGO with all keyframes
- Each relative pose from verified loop closures and covisibility graph adds a constraint
- $E_{PGO} = \sum_{i,j} \log_{Sim(3)}(S_{ij} * S_{wj}^{-1} * S_{wi})$

- S_{wi}: Sim(3) pose of I_i
- S_{ij}: Sim(3) relative pose from I_j to I_i
- $\log_{Sim(3)}$: maps to tangent space, $\mathbb{R}^7$

Pose Graph Optimization (PGO)

- We run PGO after each new verified loop closure

- After PGO, we update frame poses and rescale depths of their points

Results

- We compare
  - DSM
  - DSM-corner: an extension of DSM with new point selection approach
  - DSM-loop: our proposed method

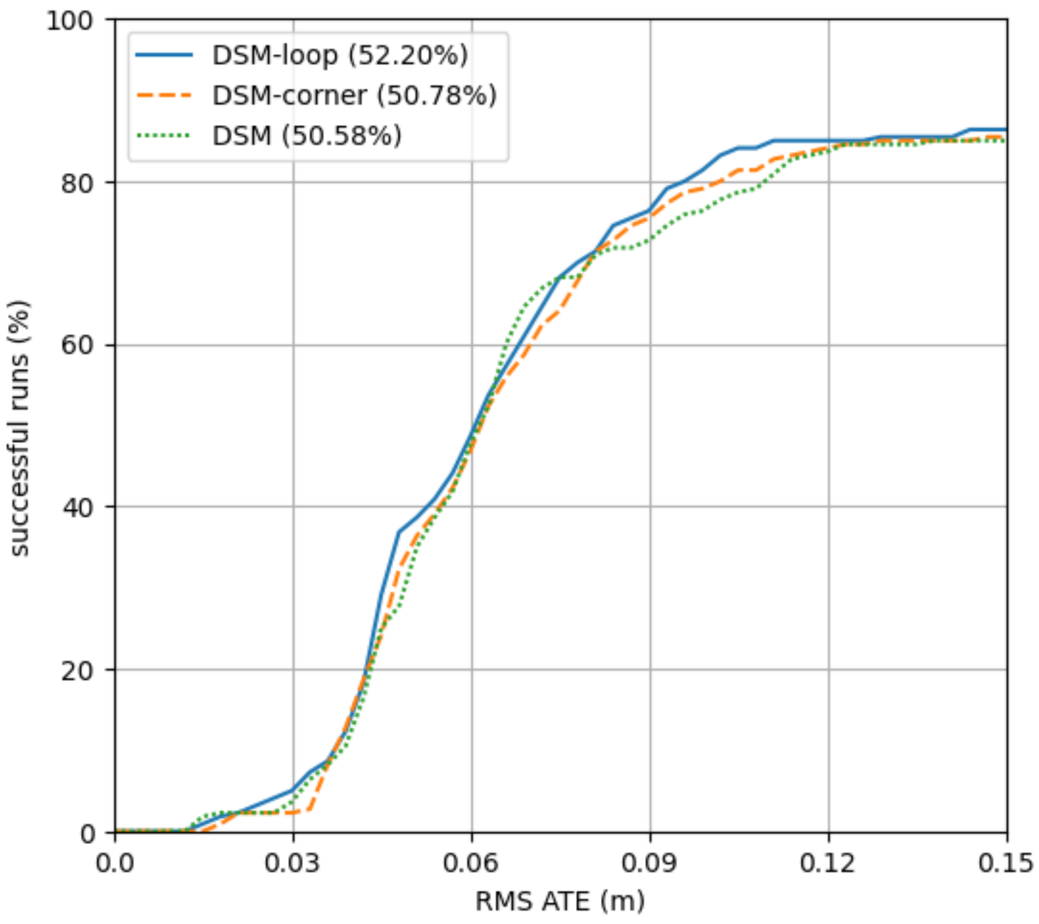
- EuRoC MAV dataset [9]
- 11 sequences in 3 indoor environments
- 220 runs for each method
  - 5 x {forwards, backwards} x {left, right}

[9] M. Burri, et al., “The euroc micro aerial vehicle datasets”, The International Journal of Robotics Research, 2016.

- Modified point selection procedure does not deteriorate the performance

- DSM-loop performs 3.2% better

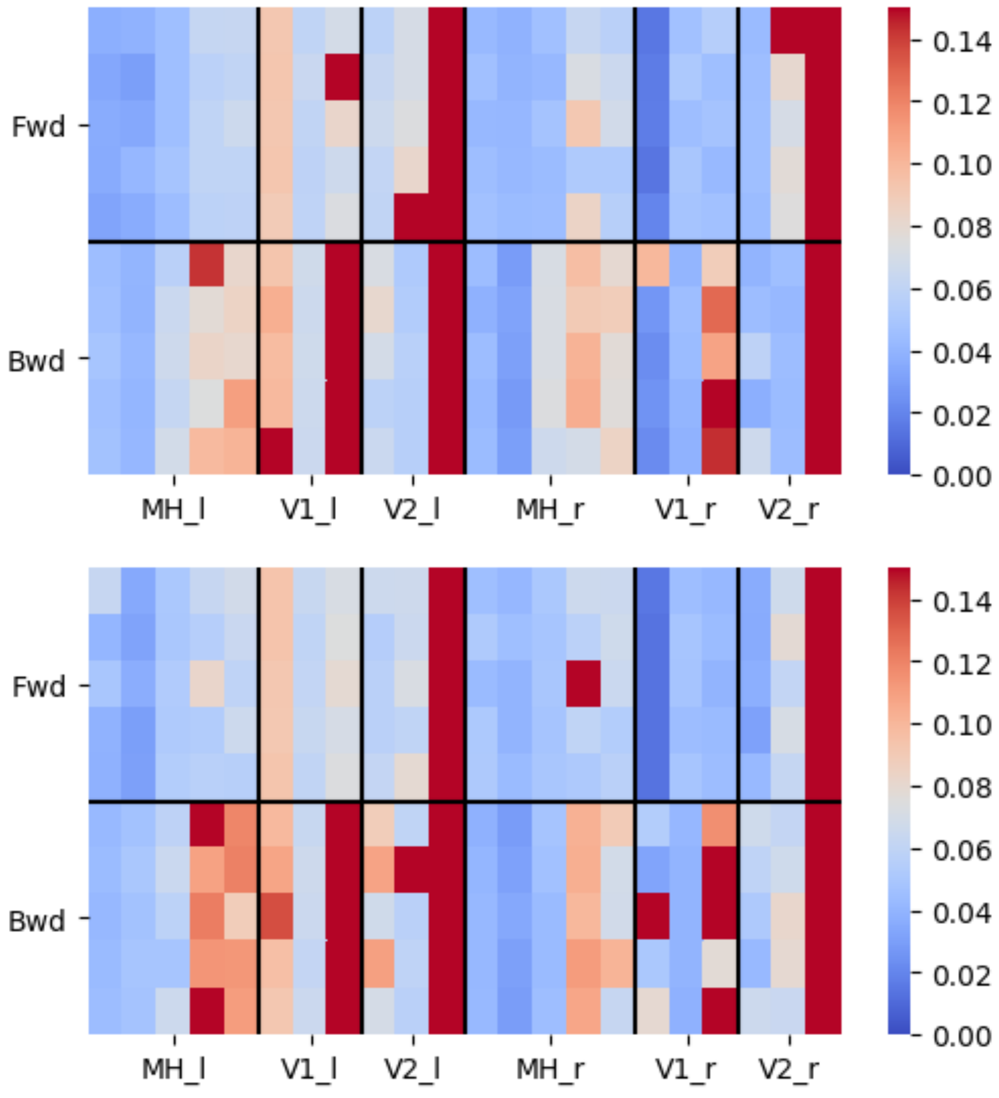
- Performance heavily depends on number of detected loops



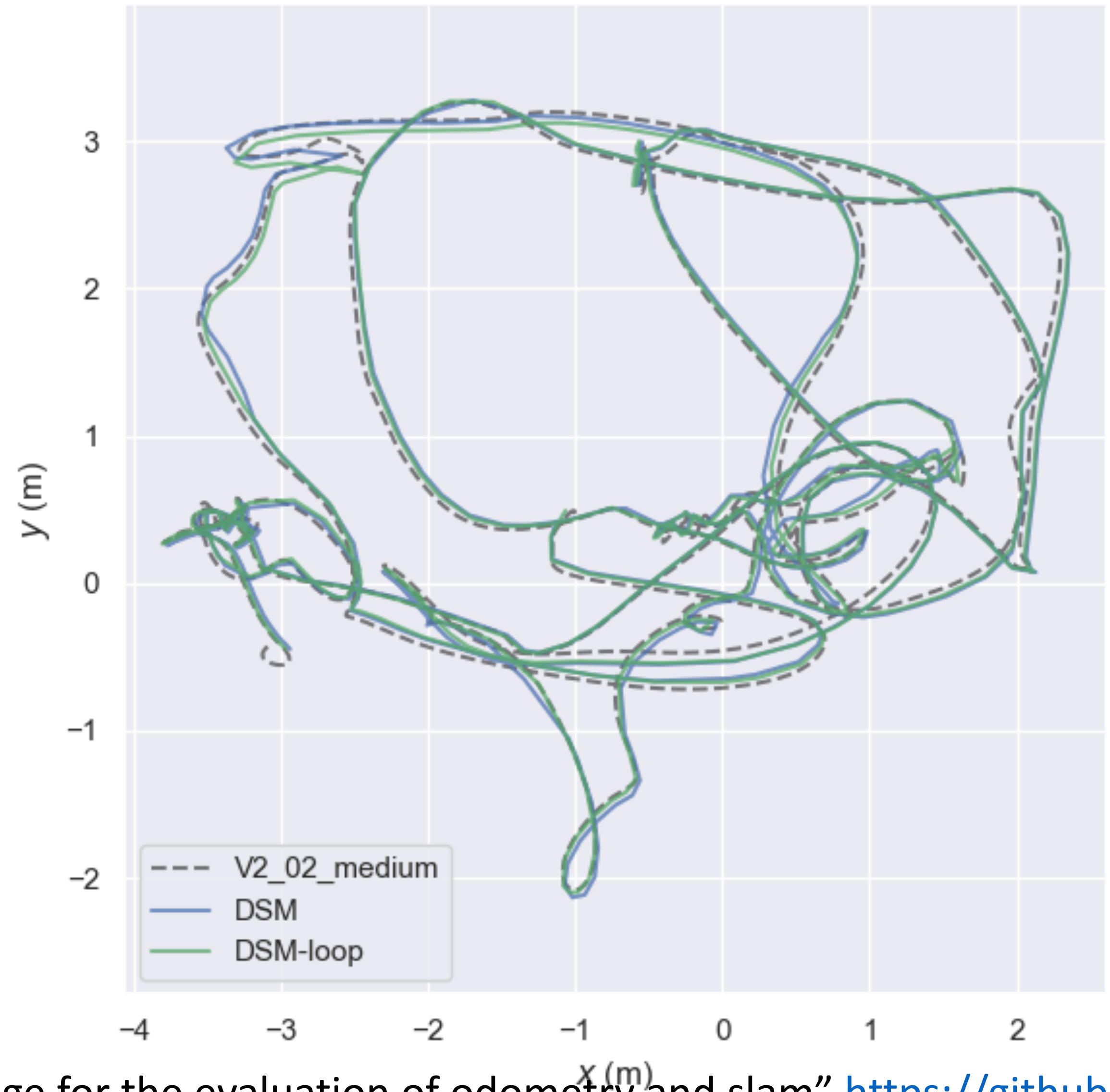
- We find >1 loops on
  - MH_01
  - V1_01
  - V2_01
  - V2_02

- For these sequences, DSM-loop achieves 8.86% better ATE than DSM
  - 0.0545 vs 0.0598

DSM-loop



DSM



(3 loops)

M. Grupp, “evo: Python package for the evaluation of odometry and slam” https://github.com/MichaelGrupp/evo, 2017.

Conclusion

- We proposed a direct SLAM system with global mapping and loop closure by extending DSM
- Increased point repeatability while retaining robustness
- Slight performance improvement with loop closure correction
  - 3.2% improvement overall
  - 8.86% improvement for sequences with detected loops

Future Work

- Fine tune parameters in loop detection and verification part
- Merge map points and keyframes after loop closure corrections
- Sparsify the map by continuously merging and removing redundant keyframes and points
- Test method on other datasets

Thank you for listening!

Sequence	DSM	DSM-corner	DSM-loop	%	# loops
MH_01_l	0.0446	0.0417	**0.0407**	2.34	**1.3**
MH_02_l	0.0406	0.0388	**0.0385**	0.90	0.3
MH_03_l	0.0558	0.0566	**0.0548**	1.78	0.5
MH_04_l	0.0821 (2)	**0.0703 (1)**	0.0777	-10.56	0.1
MH_05_l	0.0867	0.0777	**0.0768**	1.06	0.1
V1_01_l	0.0998	0.0997 (2)	**0.0950 (1)**	4.75	**3.0**
V1_02_l	**0.0632**	0.0647	0.0635	-0.43	0.7
V1_03_l	0.0741 (5)	0.0974 (3)	**0.0731 (6)**	1.38	0.4
V2_01_l	0.0742	**0.0643**	0.0655	-1.90	**4.1**
V2_02_l	0.0642 (2)	0.0647	**0.0635 (1)**	1.12	**2.3**
MH_01_r	0.0449	**0.0408**	0.0435	-6.70	**2.6**
MH_02_r	0.0363	0.0370	**0.0356**	2.02	0.0
MH_03_r	**0.0474**	0.0589	0.0581	-22.63	0.7
MH_04_r	0.0845 (1)	0.0860	**0.0827**	2.09	0.1
MH_05_r	**0.0701**	0.0738 (1)	0.0706	-0.74	0.3
V1_01_r	0.0320 (1)	0.0552	**0.0282**	11.88	**3.3**
V1_02_r	**0.0433**	0.0450	0.0448	-3.38	0.7
V1_03_r	0.0582 (3)	**0.0466 (5)**	0.0786 (1)	-34.99	0.9
V2_01_r	0.0469	0.0509	**0.0468**	0.11	**3.5**
V2_02_r	0.0696	**0.0577**	0.0581 (1)	-0.68	**2.2**