Dense Multi-Fisheye-Camera VIO for HILTI SLAM Challenge 2022

Wei Zhang[1*] and Sen Wang[2]

I. Introduction

This report summaries our approach and results using the provided datasets of the HILTI SLAM Challenge 2022. Our approach takes images of front stereo, left and right cameras, and IMU measurements as inputs to estimate the states of the sensor platform. We extended the open-sourced DROID-SLAM framework [1] and carried out three key improvements as follows:

- Explicit usage of fisheye camera model instead of pinhole model assumption to avoid the information loss due to fisheye-to-pinhole undistortion or stereo rectification.
- Integration of IMU measurements with a successful IMU initialization and after that the joint bundle adjustment (BA) of IMU pre-integration factors and visual reprojection factors.
- Support for multi-camera inputs with an adaptive implementation taking into account the respective extrinsic and intrinsic parameters of each camera.

II. System Overview

Our approach is based on DROID-SLAM [1], a versatile dense visual SLAM framework using the robust optical-flow estimates predicted by a pre-trained neural network as pixel correspondence references to optimize the reprojection errors. To better incorporate the fisheye images while not losing the wide-view information, we add the fisheye camera model to consider the distortion when projective transform the pixels between image planes.

The provided datasets is collected in very challenging environments. When applying the basic framework directly in monocular or stereo mode, it can quickly fail to track the camera, since the front cameras are often just towards to a corner or a texture-less wall. Leveraging the other camera views and integrating the IMU measurements becomes necessary to tackle this issue. More specifically, the images of left and right cameras are used and fed into the pipeline. The up view camera is left out as this view brings little useful information. For each keyframe, not only the depth map of front camera but also the depth maps of left and right cameras are estimated. With that, the reprojection errors are computed for all three views. When there is no enough features in one view, the features of other views can guide the optimization to converge.

For IMU initialization, we add the scale as an additional optimization variable, since the evaluations on the sequences 04-06 based on the given groundtruth show a scale correction would be beneficial. Theoretically this is not required with a known stereo baseline. Thus more investigation needs to be conducted in the future to verify whether the camera calibration or our stereo matching method needs to be further improved.

Furthermore, we answer the following questions as posted on the challenge website.

- Q: Filter or optimization-based?
- A: Our approach is optimization-based with a sliding optimization window of 25 keyframes at maximum. The local BA graph consists of visual reprojection factors and IMU preintegration factors. The linearized BA system is solved using the Pytorch library.

- Q: Is the method causal?
- A: Yes. Upon each new keyframe, the local window optimization is carried out, which takes only the present and past information.

- Q: Is bundle adjustment (BA) used? What type of BA?
- A: Yes. Our approach uses BA to minimize the reprojectoin errors and IMU pre-integration factor errors jointly. We construct a local BA system from the current sliding window and update the camera poses and depth maps of the keyframes inside the window.

- Q: Is loop closing used?
- A: No. The system only optimized a sliding window of maximally 25 keyframes at temporal order. Nevertheless, we expect an improvement using loop closure and global bundle adjustment. However due to time limit, this is left out for our future works.

III. Results

Our experiments are carried out on an Intel i9 platform with a TITAN RTX GPU. The processing times for final submission are measured and listed in Table 1. To achieve a better score,

[1]Wei Zhang is with the Institute for Photogrammetry, University of Stuttgart, Germany (e-mail: wei.zhang@ifp.uni-stuttgart.de).
[2]Sen Wang is with CAMP, Technical University of Munich, Germany (e-mail: sen.wang@tum.de)
*Corresponding author

we select more time-consuming but more robust parameter set. The same parameter set is used for all sequences except the sequence "Exp21 outside building", for which a stricter keyframe selection threshold is chosen, which we found empirically more suitable for outdoor scenes. Furthermore, due to the time pressure and for faster prototyping, we choose to develop the methods in Python at the moment. We expect a time boost when proper operations are converted into CUDA kernel.

TABLE I. PROCESSING TIMES OF ALL SEQUENCES

Sequence	***Time[s]***
Exp01 construction ground level	1151
Exp02 construction multilevel	3738
Exp03 stairs	2613
Exp07 long corridor	707
Exp09 cupola	2597
Exp11 lower gallery	763
Exp15 attic to upper gallery	1567
Exp21 outside building	626
Exp_04 construction upper level 1	746
Exp_05 construction upper level 2	669
Exp_06 construction upper level 3	1410

Figure 1-11 present the estimated trajectories and maps produced by our system. The map composes of the estimated depth maps of the keyframes. Note that our system has no loop closing and global optimization. Thus the drift error is accumulated inevitably. For example, it can be observed on the stairs of sequence "Exp03 Stairs".

Figure 1. Seqence 'Exp01 Construction Ground Level' with the difficulty 'Easy'. Top and side view with start and end marked with red rectangle.

Figure 2. Seqence 'Exp02 Construction Multilevel' with the difficulty 'Medium'. Top and side view with start and end marked with red rectangle.

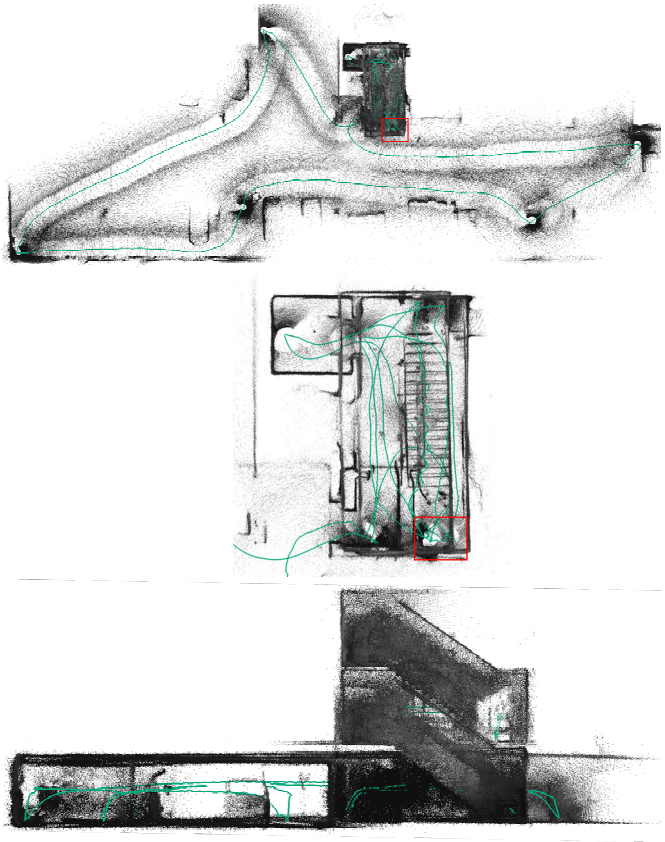

Figure 3. Seqence 'Exp03 Stairs' with the difficulty 'Hard'. Top (unzoomed and zoomed to stairs) and side view with start and end marked with red rectangle.

Figure 4. Seqence 'Exp07 Long Corridor' with the difficulty 'Medium'. Top view with start and end marked with red rectangle.

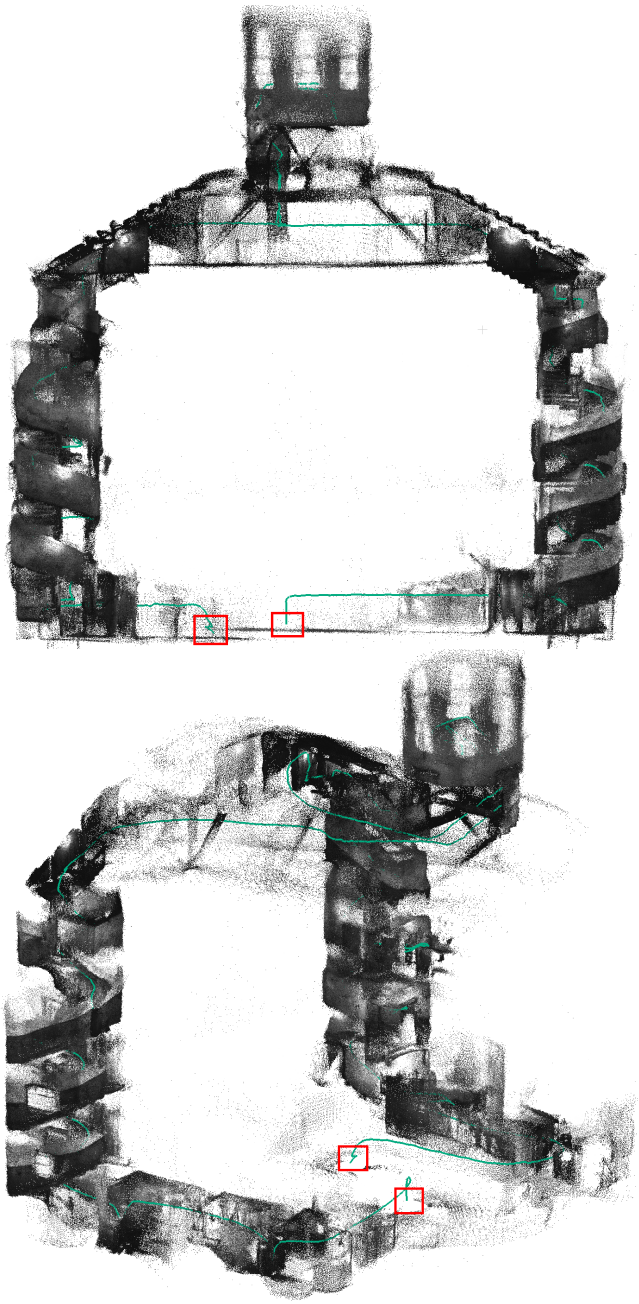

Figure 5. Seqence ‘Exp09 Cupola’ with the difficulty ‘Hard’. Side and oblique view with start and end marked with red rectangle.

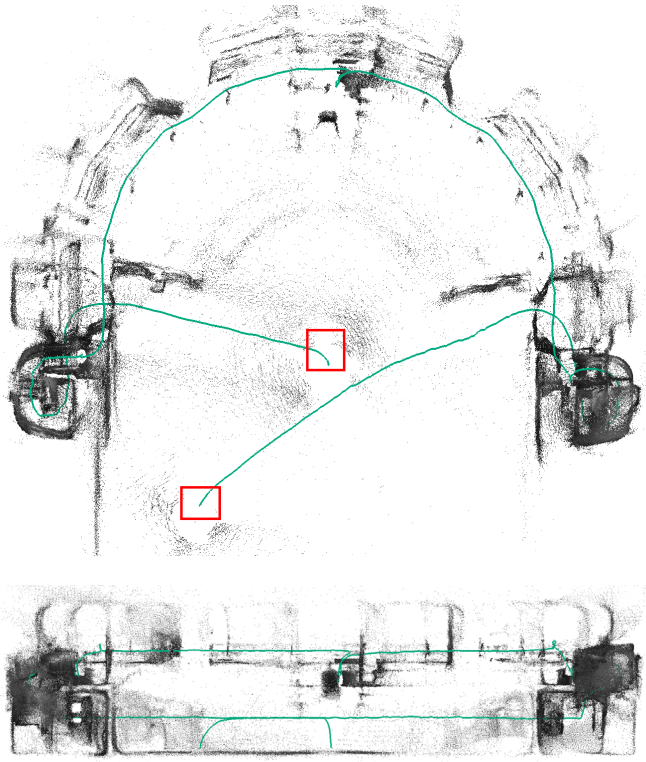

Figure 6. Seqence ‘Exp11 Lower Gallery’ with the difficulty ‘Medium’. Top and side view with start and end marked with red rectangle.

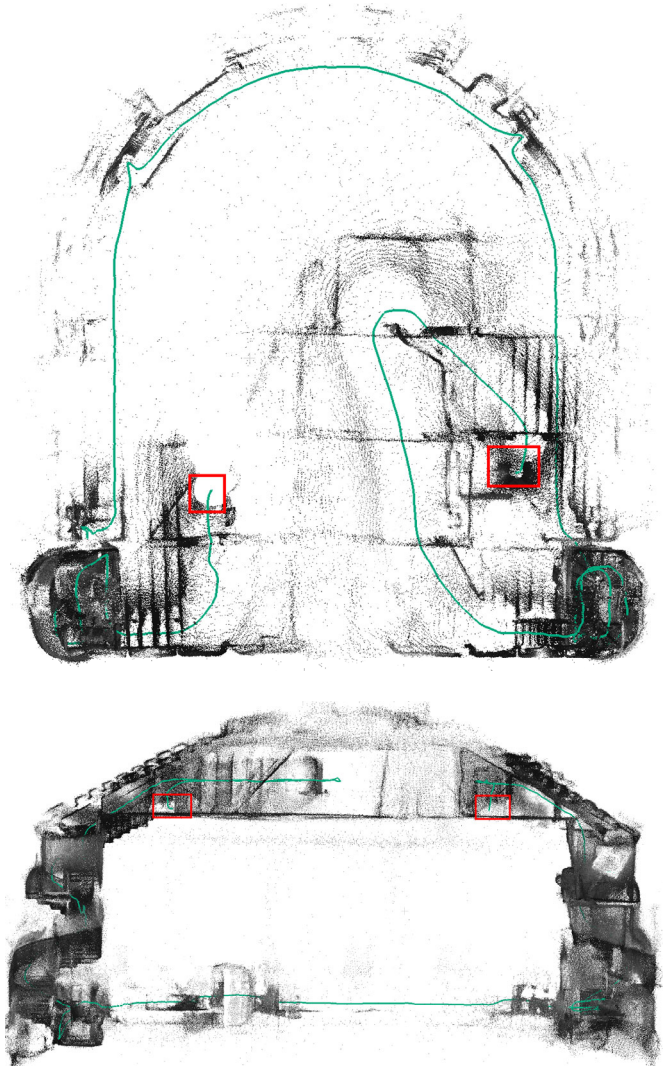

Figure 7. Seqence ‘Exp15 Attic to Upper Gallery’ with the difficulty ‘Hard’. Top and side view with start and end marked with red rectangle.

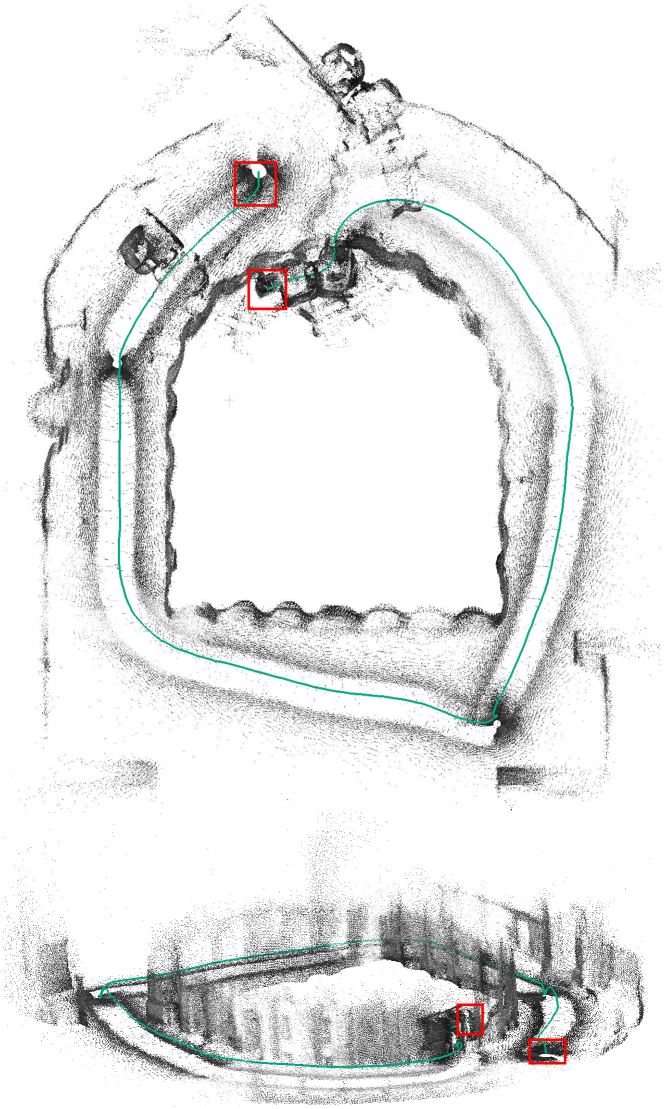

Figure 8. Seqence ‘Exp21 Outside Building’ with the difficulty ‘Easy’. Top and oplique view with start and end marked with red rectangle.

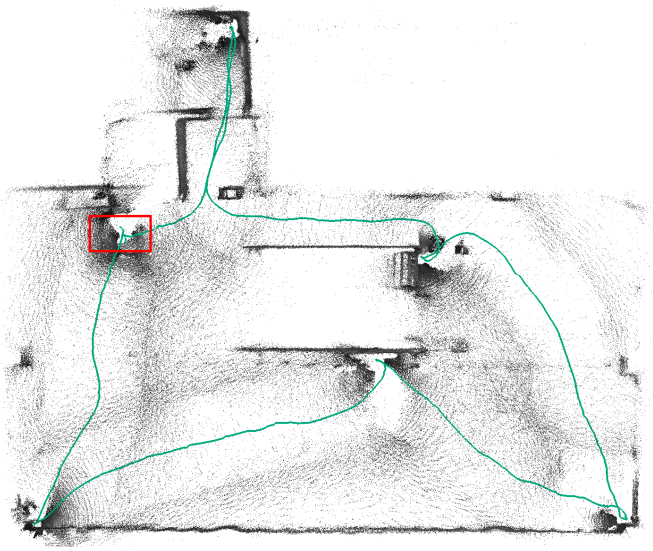

Figure 9. Seqence ‘Exp_04 Construction Upper Level 1’ with the difficulty ‘Easy’. Top view with start and end marked with red rectangle.

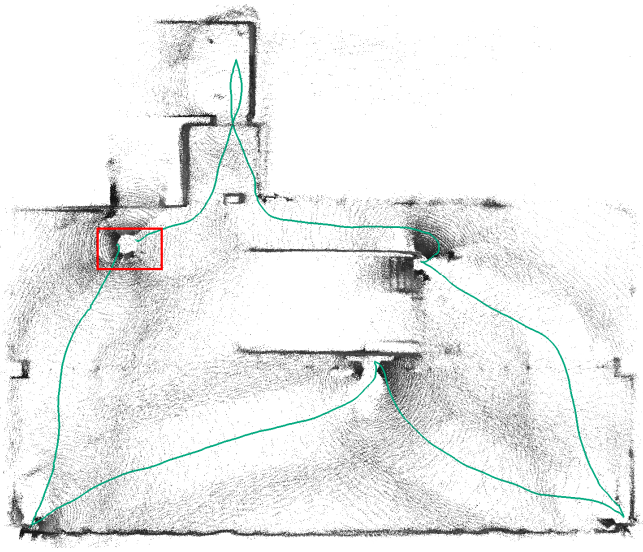

Figure 10. Seqence ‘Exp_05 Construction Upper Level 2’ with the difficulty ‘Easy’. Top view with start and end marked with red rectangle.

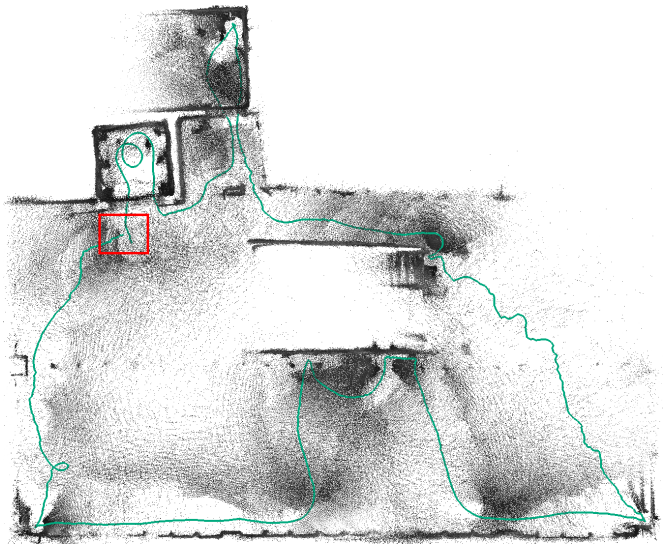

Figure 11. Seqence ‘Exp_06 Construction Upper Level 3’ with the difficulty ‘Medium’. Top view with start and end marked with red rectangle.

IV. Conclusion

In this work, we extend the open-sourced DROID-SLAM framework. The original DROID-SLAM supports monocular, stereo and RGB-D video. To utilize multiple input sources in this challenge, we extend with IMU pre-integration into the BA calculation and adaptive support for multi-cameras as additional constraints into sliding-window. With more information, we have observed improvements of more complete and accurate trajectory estimation. Meanwhile, to avoid loss of information due to the fisheye distortion, we take the original fisheye images as input and make the transform operation adaptive the respective camera parameters. Further improvement on the APE shows that the usage of complete wide-view information benefits the final result.

In the future, loop closure and global bundle adjustment can help minimize drift errors. The IMU data may also be processed with deep learning methods, which is more robust to bias and noise.

Acknowledgment

We thank the organizers of the Hilti SLAM challenge for providing the great and challenging dataset. The challenge has motivated us to work on new features to tackle practical problems. We also thank Hongjie You for offering us the compute platform to run the experiments.

References

[1] Teed, Zachary, and Jia Deng. "Droid-slam: Deep visual slam for monocular, stereo, and rgb-d cameras." Advances in Neural Information Processing Systems 34 (2021).