Geometry-Informed Distance Candidate Selection for Adaptive Lightweight Omnidirectional Stereo Vision with Fisheye Images

Conner Pulling[1], Je Hon Tan[2], Yaoyu Hu[1], Sebastian Scherer[1]

***Abstract*— Multi-view stereo omni-directional distance estimation usually needs to build a cost volume with many hypothetical distance candidates. The cost volume building process is often computationally heavy considering the limited resources a mobile robot has. We propose a new geometry-informed way of distance candidates selection method which enables the use of a very small number of candidates and reduces the computational cost. We demonstrate the use of the geometry-informed candidates in a set of model variants. We find that by adjusting the candidates during robot deployment, our geometry-informed distance candidates also improve a pre-trained model's accuracy if the extrinsics or the number of cameras changes. Without any re-training or fine-tuning, our models outperform models trained with evenly distributed distance candidates. All the pre-trained models are released as hardware-accelerated versions with a new dedicated large-scale dataset.**

I. INTRODUCTION

Distance perception is a key requirement in mobile robots that need to navigate and avoid obstacles. A larger field-of-view (FoV) and faster distance perception enable a robot to more effectively gather information about its surroundings, with omnidirectional sensing being most desirable. Presently, LiDAR devices are the go-to sensors for distance perception due to their accuracy and high update speed. However LiDARs are mechanically complex, and this complexity increases with an increased number of sampling points. It is technically difficult and prohibitively expensive to achieve both large FoV and high resolution with LiDARs.

Using multiple cameras as a multi-view stereo (MVS) camera set can provide high-resolution omni-directional distance perception with much lower mechanical complexity and cost. Recent research has demonstrated that using multiple cameras with large FoV lenses (e.g., fisheye lens) can achieve omni-directional distance estimation [1], [2], [3]. Compared to LiDAR devices, vision-based distance estimation typically provides larger FoV and denser measurements. However, two challenges prevent MVS-omni-directional solutions from being the go-to sensor choice: 1)they are computationally expensive and; 2) they are difficult to deploy.

The majority of the MVS-omni-directional models, both learning-based and non-learning, utilize a cost volume structure that aggregates visual features by using virtual distance candidates along a viewing direction. The model compares the features at all candidates that are present in the cost volume and picks the best weights for a linear combination of the given candidates. This cost volume approach consumes a significant amount of computing resources, which grows depending on the number of cameras and the number of distance candidates.

*This work was supported by the Defence Science and Technology Agency, Singapore.

[1]C. Pulling, Y. Hu and S. Scherer are with the Robotics Institute, Carnegie Mellon University, 5000 Forbes Avenue Pittsburgh, PA 15213-3890 USA. {cpulling, yaoyuh, basti}@andrew.cmu.edu.

[2]J. Tan is with the Defence Science and Technology Agency, 1 Depot Road, Singapore, 109679. jehontan@gmail.com.

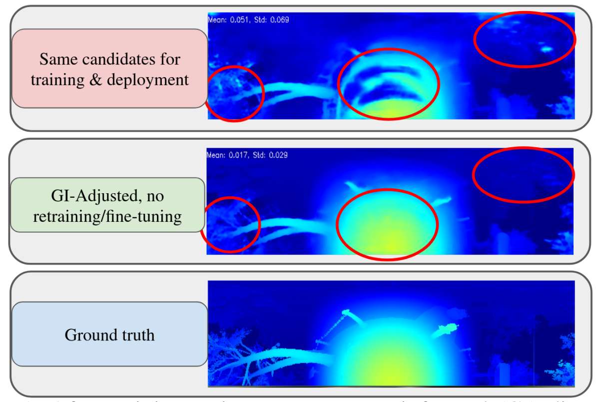

Fig. 1: After training, using our geometry-informed (GI) distance candidate distribution, the baseline distance between cameras can be changed and the model's performance can be restored without fine-tuning.

For deployment, cameras in an MVS-omni-directional system typically need to be placed such that maximum FoV can be achieved with minimum occlusions from the robot (self-occlusion). For learning-based methods, if the location or number of cameras is changed to mitigate occlusions, the method typically suffers significant performance degradation as the position of corresponding features in the camera images is changed, hence for the same distance candidates the patterns of accumulated features in the cost volume that differ greatly from training data.

To resolve the above issues related to learning-based visual omni-directional distance estimation, our insight is that we can train a model to utilize a small number of virtual distance candidates by picking distance candidates in a way that is informed by the geometry of the camera configuration. For a known set of camera extrinsics, we can select the candidates such that the positional displacement for the same feature sampled at two consecutive distance candidates are similar across all consecutive pairs of candidates. This ensures a similar pattern of feature accumulation in the cost volume, allowing the model to more effectively determine the best interpolation weights between a pair of consecutive candidates. This enables us to create models with a much lower number of candidates (16 or 8) compared to previous methods, significantly reducing computational cost. We are also able to compute such distance candidates for new camera configurations during deployment, allowing a trained model to be used and maintain its performance even if the camera extrinsics or the number of cameras is changed. In this work, our contributions are

- A geometry-informed (GI) distance candidates selection method that enables the use of fewer candidates and change of extrinsics for deployment.
- Demonstration of a relaxed version of camera layout that can generate omni-directional distance estimation with self-occlusion explicitly handled and variable translations among the cameras.

After training on our dedicated new dataset, our model can efficiently generate omni-directional distance from multiple cameras with self-occlusion explicitly handled, even if the number and position of the cameras change during physical deployment. The code, pre-trained model, and the dedicated dataset are available through the project webpage.

II. RELATED WORKS

Estimating distance from more than one camera is a common and fundamental capability of robot systems. There is a vast body of work that covers various topics, of which we will concentrate on two most closely related to ours.

A. Multi-view Omnidirectional Distance Estimation

The most relevant non-learning model is from Meuleman, *et. al.* [2] where they generate distance predictions for a reference fisheye image by selectively fusing information from other fisheye image views. A complete omni-directional distance prediction is then made by stitching multiple estimations together. They also build a cost volume to aggregate information across different distance candidates. For efficiency, the number of candidates is kept at 32. Since the model is non-learning-based, there is no training and it can be deployed on various camera layouts. This model is one of our main baseline models.

For the learning-based models, SweepNet and OmniMVS, by Won, *et. al.* [1][4][5] are the standouts among the early approaches. Like the non-learning models, SweepNet and OmniMVS will build a cost volume for a fixed number of candidates. This number is configurable but in order to achieve desired accuracy the value is set at around 100 or 200. The cost volume is consumed by the downstream part of the model, typically layers of 3D Convolutional Neural Networks (CNN), and distance values are estimated. Later, Su, *et. al.* [6] implemented a hierarchical version that makes distance predictions on different scales, where at each scale, a cost volume is built in the same way. The above models are trained with a fixed number of cameras and placement. When the camera layout changes, new training and datasets may be required.

Two recent works are closely related to our approach. One conducted by Chen, *et. al.* [7] constructs multiple cost volumes for unsupervised learning. They use feature variance to compare the cost volumes [8]. Our approach is similar with the difference being that we handle self-occlusion explicitly. The other is OmniVidar [3], which turns omni-directional distance estimation into multiple rounds of binocular stereo estimations. On a high level, the learning-based part of this approach is camera layout agnostic as long as we can cover the final omni-directional FoV by undistorting and rectifying the input fisheye images along different orientations. However, this process needs to be manually and carefully designed for every new camera layout. Our model can accommodate camera layout change through an easier process with fewer manual procedures.

B. Multi-view Stereo (MVS)

Multi-view stereo (MVS) has a longer history compared with the aforementioned multi-view omni-directional distance estimation. MVS studies are more focused on reconstructing the 3D geometry of an object or a scene, other than providing distance estimations with respect to a robot. Similar to omni-directional distance estimation, MVS studies use both non-learning [9][10][11][12][13][14] and learning-based approaches [8][15][16][17][18]. The result of an MVS method is usually a volumetric representation (e.g., voxel grid surface), point cloud, or surface mesh. Inside these learning-based models, a cost volume can be constructed following [8]. Most of the approaches use a reasonably large number of distance candidates. Some works, e.g. [19][20][21], explore multi-scale or adaptive candidates, which may use fewer candidates but need to do the computing in an iterative way, leading to additional computational overhead.

III. METHODS

A. Target Configuration

For real-world testing, we use an evaluation board with three fisheye cameras pointed in the same direction and arranged in a triangular formation as in Fig. 3 and the *training layout* in Fig. 8. This target configuration enables an aerial robot to have omni-directional vision by placing cameras safely on top of its body, e.g. Skydio 2+ Drone. Additionally, this target configuration is especially challenging due to the fact that image boundary regions from fisheye lenses are extensively used where good calibration is hard to achieve. We utilize the TartanCalib toolbox to get better calibration results with the Double Sphere camera model[22][23].

B. Model Overview

Similar to [1], our model builds a cost volume from spherically-sweeping learned features and then regularizes this cost volume to achieve a probability distribution of the true distance for each pixel. First, the model takes in three fisheye images during training. Feature maps are extracted from the images with a shared 2D-convolution feature extractor. Next, spherical sweeping is employed using a set of distance candidates to warp the other fisheye images into the reference image frame at the candidate distance. To aggregate all of the views into C channels, differing

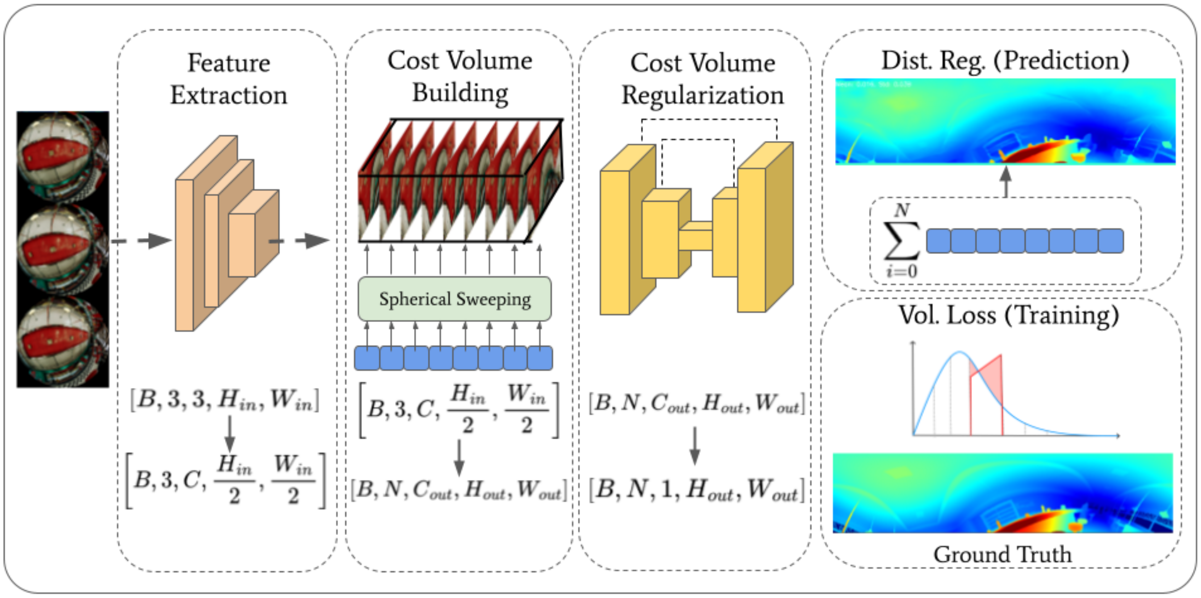

Fig. 2: Model Overview. The model takes three fisheye images as input during training and performs learned feature extraction with a shared feature extractor, builds a cost volume with spherical sweeping, and regularizes the distance with a 3D U-Net [1].

from prior works in omni-directional vision with fisheye images, one of our model variants (introduced in Section IV-A) uses feature variance to build the cost volume, similar to [8]. By using feature variance as opposed to concatenating the feature vectors together for each pixel for each warped image, the channel dimension is reduced by a factor of N (number of images). Additionally, because the variance between a set of vectors results in a same-length vector no matter how many vectors there are from the input images, the model can explicitly exclude self-occluded pixels while maintaining the required length of the C dimension. We utilize a 3D U-Net typed regularizer to process the cost volume into a probability distribution. The probability for each candidate is used in a weighted sum to regress the distance for each pixel.

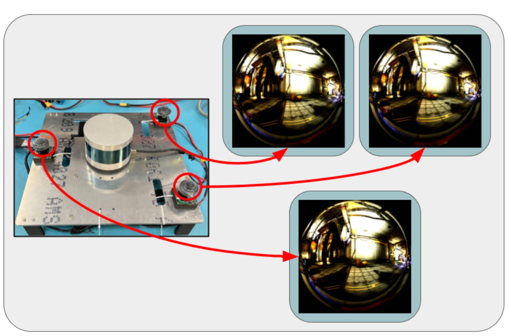

Fig. 3: Camera Configuration for the evaluation board. Three fisheye cameras are mounted pointing upwards in a triangular formation. A LiDAR, unused for this study, introduces self-occlusions.

C. Distance candidate selection

Previous work on distance perception commonly used distance candidates spaced evenly in the inverse distance space (hereinafter named *EV*). In the case of plane-sweeping[8], EV candidates have the property that moving an object between consecutive candidates results in a constant pixel displacement of the corresponding features in feature space.

In the case of sphere-sweeping[2][5], EV candidates generally do not result in constant feature displacement due to the non-linearity of spherical sweeping. However, for small camera baselines, they provide a close approximation, as

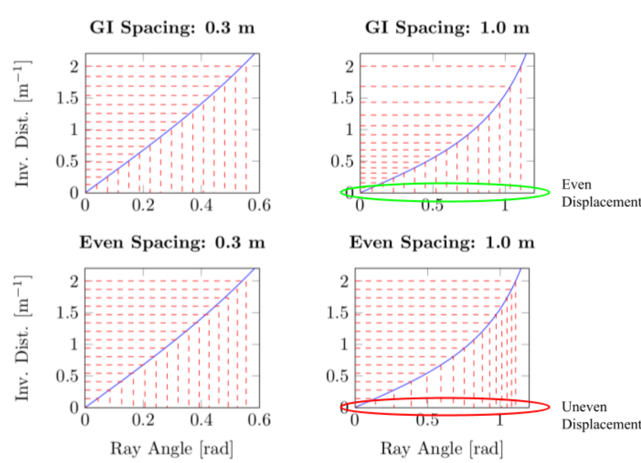

Fig. 4: GI and EV candidates for different camera spacings. EV candidates approximate constant feature displacement steps for small spacings (baselines), but result in highly uneven steps in large spacings. GI candidates generate constant displacement steps as a function of camera spacing.

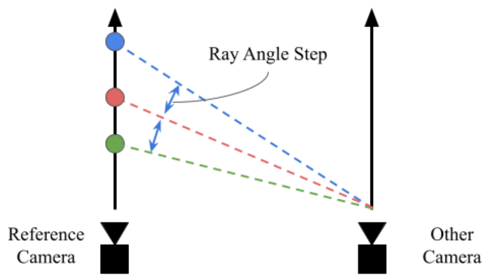

Fig. 5: Sphere-sweeping geometry. We pick distance candidates that result in constant steps in ray angle corresponding to constant displacements in the projected feature.

shown in Fig. 4. As previous work on sphere-sweeping has focused on small baseline configurations and large numbers of candidates[1][4][5], the use of EV candidates caused negligible impact on performance.

For better efficiency, we propose to use a small number of geometry-informed (GI) candidates computed for specific camera extrinsics and ensure similar displacement for each step between distance candidates. As feature position in the projected image is proportional to the feature ray angle, GI candidates are obtained by developing distance as a function of ray angle and sampling it with evenly spaced ray angle steps (see Fig. 5). Later in the experiment section, we show that the use of GI candidates improves distance prediction accuracy in the cases of large camera spacing and low candidate count.

D. Volume Loss

As seen in previous work [1][3][6], the main loss function of choice for the omnidirectional stereo vision supervised learning problem has been L1 loss on the final distance map. However, there is a rich amount of information in the cost volume itself before aggregation. Before linear combination but after softmaxing, the cost volume represents a probability distribution of which distance candidate is the most likely to be the true distance. In actuality, this probability distribution should look like the interpolation between the two closest distance candidates to the true distance value. Therefore, because the ground truth probability distribution is known and the softmax'd cost volume represents a predicted probability distribution, a soft cross-entropy loss function can be used as a more informative loss function [24]. Combined with the GI distribution described in the previous section, using the volumetric soft cross-entropy loss leads to accuracy gains.

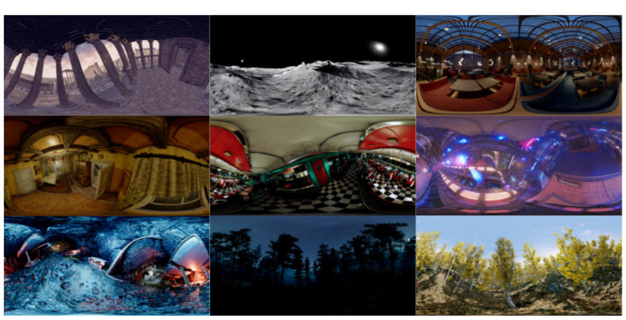

Fig. 6: Our omni-directional stereo vision with fisheye images dataset consists of about 95K samples from over 60 Unreal Engine 4 high-fidelity simulation environments, manifested in various scene styles.

E. Dataset

We create a new dataset consisting of about 95K samples collected from over 60 Unreal Engine 4 simulation environments used in the collection efforts of TartanAir [25]. This dataset is over 10x larger than any currently available dataset for omni-directional stereo vision with fisheye images [1] and will be released for download. The camera layout is the *training layout* in Fig. 8. Each sample consists of three RGB-dense distance pairs in fisheye format. There are a large variety of outside, urban, indoor, and natural environments as shown in Fig. 6.

IV. EXPERIMENTAL PROCEDURE & RESULTS

A. Model Variants and Camera Layouts

We propose that geometry-informed (GI) distance candidates can directly improve distance estimation. GI candidates can be adapted to most of the MVS-omni-directional vision models where a fixed number of candidates are applied. In this work, we use a set of model variants to show that GI candidates can work well with small candidate numbers and changes in camera layout.

For a baseline comparison, we build a model for the 3-camera layout shown in Fig. 2 using a similar structure as the OmniMVS model[1]. We then have two simple variants from OmniMVS, based on EV and GI candidates. We are targeting models with fewer candidates to have better efficiency. We designate model names E16 and E8 for baseline models with only 16 and 8 candidates, while G16 and G8 for the GI ones. Using the same naming, let G16V be the model trained with the volume loss function. Finally, we also apply the variance cost volume [8] to G16V and get G16VV. One detail about G16VV is that when calculating the cost volume, we explicitly handle the self-occlusion from the robot. This is done by additionally showing the model a binary mask for every input fisheye image. Such a mask marks non-occluded pixels as valid pixels. When building the cost volume, a variance value is calculated by only considering visual features from the non-masked regions. G16VV is smaller than other variants as a result of using feature variance for building the cost volume. Besides E16 and E8, we also make compare with the RTSS model [2].

All models are trained on the dataset in Section. III-E. The distance range is fixed at 0.5-100m during training. For comparison purposes, all models are trained with the same fixed learning rate (0.0001) and batch size (16). We reserve some simulation environments from training and collect ground truth data for evaluation.

Several camera layouts are used in the following experiments. As shown in Fig. 8, all models are trained using the *training layout*. This three-camera plenary setup is the

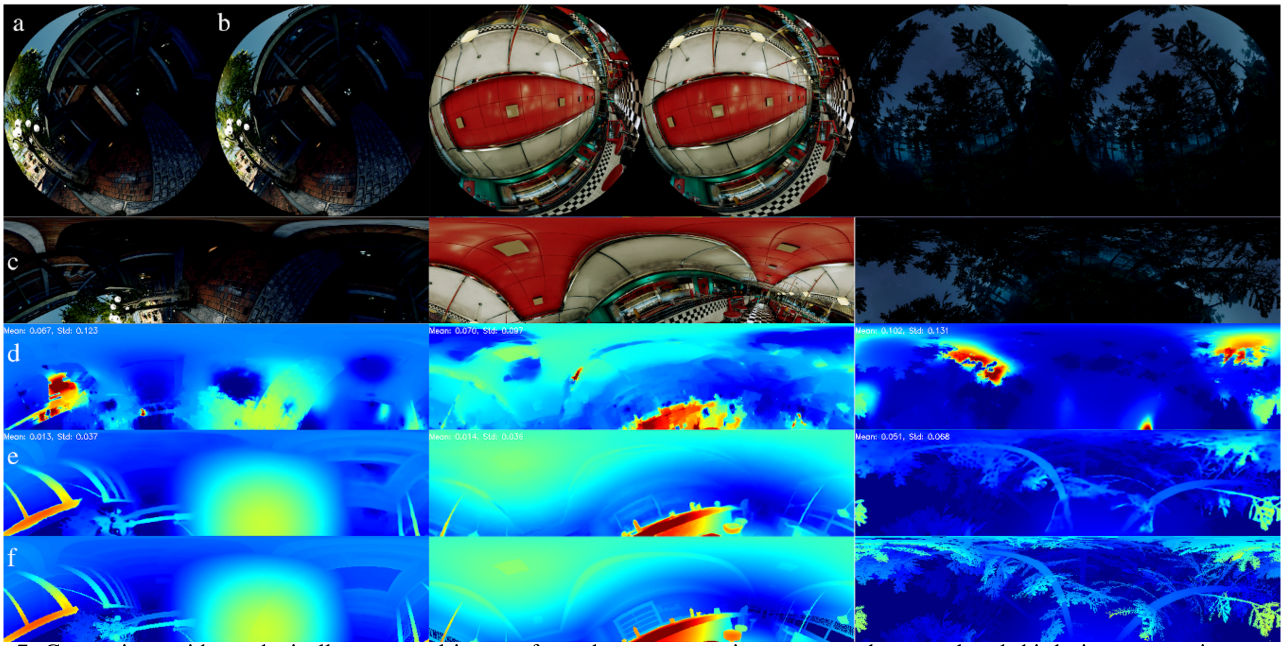

Fig. 7: Comparison with synthetically-generated images from the unseen environments. a, b: second and third views. c: equirectangular projected reference (first) view. d, e: outputs of RTSS [2] and G16VV (ours). f: ground truth distance aligned with the reference view. In scenes with low light, high-frequency features such as patterns and trees, and thin objects, G16VV is more accurate and it can resolve fine details.

minimum to cover the semi-sphere FoV on top of the plane. This setup also ensures that the robot body at the middle of the cameras will not block the view of more than two cameras, making stereo distance estimation possible for all FoV directions. A location on the plane is picked as the reference and the omni-directional distance image is generated w.r.t this reference location. We test the models on different layouts representing the change of spacing, number of cameras, and reference location.

B. Evaluation with the Same Camera Layout

We first collected over 1000 samples using *testing layout 1* in Fig. 8. Model predictions are compared with ground truth omni-directional distance images. We use simple metrics including mean absolute error (MAE), mean root square error (RMSE), and the Structural Similarity Index (SSIM) as in [2]. All metrics are computed using the inverse distance (ranging from 0.01 to 2). We use a single NVIDIA V100 GPU for measuring the execution time and GPU memory usage. Table I shows that with 16 or 8 candidates, a model can have very competitive efficiency and GPU consumption compared to the real-time baseline model (RTSS) [2]. This

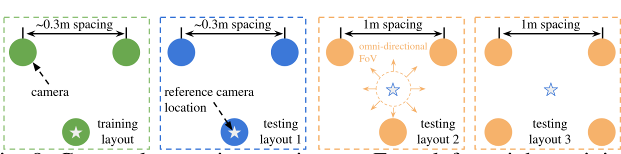

Fig. 8: Camera layouts in experiments. From left to right: training, testing 1 (same as training), testing 2 (larger spacing, new reference location), testing 3 (larger spacing, new reference location, and more cameras).

TABLE I: Comparison using the same camera layout

model	candidates		metrics			time	GPU (MB)	
	type	num	MAE	RMSE	SSIM	(ms)	start	peak
RTSS[2]	EV	32	0.053	0.101	0.776	144	330	4240
E8	EV	8	0.013	0.032	0.862	65	790	1030
G8	GI	8	0.012	0.029	0.867			
E16	EV	16	0.011	0.028	0.876	111	790	1230
G16	GI	16	0.010	0.028	0.877			
G16V	GI	16	0.013	0.029	0.861			
G16VV	GI	16	0.012	0.028	0.872	114	800	1090

EV: evenly distributed candidates. *GI*: geometry-informed.

table also shows that when using very few candidates, such as E8 and G8, the geometry-informed one tends to be better.

C. Evaluation with Different Camera Layout

To show the GI candidates' ability to handle a camera layout that is different from the training setup, we collect over 100 samples from the evaluation environments with larger distances among the cameras, as illustrated in *testing layout 2* Fig. 8. For this test, we only use the variants that have 16 candidates. In the tests, we apply a trained model twice, one with the candidates it was trained on, and the other with the dedicated new candidates that are calculated concerning the deployed camera layout (denoted as *new* in the following table). Table II shows that the GI candidates can boost the performance of a trained model when deployed on a camera layout that has longer displacement than the training data. We also observe from Table II that model G16V, which is trained with our volume loss, tends to have better SSIM values.

Using the G16VV model, since it builds the cost volume with feature variance across all views, we can demonstrate that using the GI candidates, our model can also handle the change of camera number. A separate set of over 100 samples

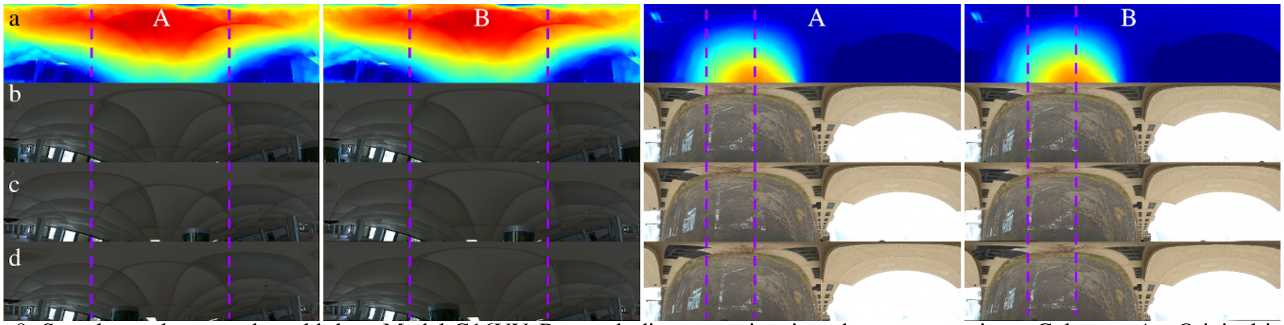

Fig. 9: Sample results on real-world data. Model G16VV. Row a-d: distance estimation, three camera views. Columns: A - Original input image, B - Input image warped using the predicted distance (Raw a). Purple lines: vertical guidelines. If the distance prediction is good, then the pixels on the purple line across Column B should align among Row b-d. Our model trained only on synthetic data evaluated on real images. The model can be optimized with NVIDIA TensorRT for better inference speed on real robots.

is collected from evaluation environments with four cameras laid out as *testing layout 3* in Fig. 8. We show a sample result in Fig. 10. The quantitative results are also listed in Table II. G16VV gains better performance from only changing the candidate values without any new training. On the speed side, from 3 cameras to 4 cameras, the processing time of G16VV adds only 6ms while the RTSS model experiences a 35ms time increase. On the GPU memory side, since the RTSS model precomputes the best view pairs for every output pixel, its memory does not change. For G16VV, we observe an increase of about 60M Bytes.

TABLE II: Comparison using new camera layouts

new layout	model	candidates		metrics		
		train	eval	MAE	RMSE	SSIM
3 cam 1m apart	RTSS[2]	-	EV	0.124	0.217	0.697
	E16	EV	EV	0.033	0.053	0.768
		EV	new	0.018	0.037	0.829
	G16	GI	GI	0.030	0.050	0.786
		GI	new	0.020	0.029	0.823
	G16V	GI	GI	0.029	0.056	0.823
		GI	new	0.015	0.034	0.868
4 cam 1m apart	RTSS[2]	-	EV	0.090	0.147	0.637
	G16VV	GI	GI	0.024	0.041	0.817
		GI	new	0.016	0.033	0.860

Candidates type: *EV* - evenly distributed, *GI* - geometry informed, *new* - GI for the 1m spacing. All models are trained with a camera spacing of about 0.3m and tested with 1m.

D. Deployment & Hardware Acceleration

We deploy the model on small-drone-compatible compute devices, namely the Nvidia Xavier NX and Orin NX. To obtain better performance we utilize the Nvidia TensorRT SDK for model optimization and hardware acceleration. We demonstrate that our optimized model is capable of running in real-time at approximately 10 Hz on the Orin NX with hardware acceleration. Along with this paper, the hardware accelerated model will be released.

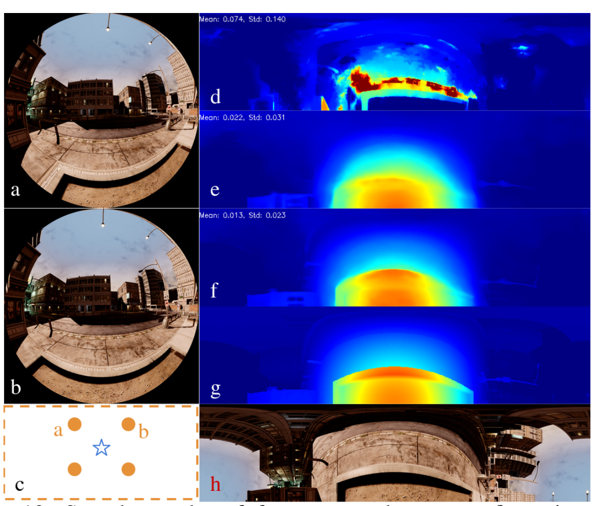

Fig. 10: Sample results of four-camera layout. a: first view. b: second view. c: camera layout. d: RTSS, e: G16VV w/ training candidates. f: G16VV w/ adjusted GI candidates calculated w.r.t. camera layout. g: ground truth distance map. h: ground truth equirectangular image view. When adjusted GI candidates are used, G16VV is more accurate and resolves more details.

V. CONCLUSION

This work introduces Geometry-Informed (GI) distance candidate selection for omni-directional vision models. GI candidate approximate constant feature displacement between distance candidates. Additionally, GI candidates give the model extra flexibility after training: camera spacings (stereo baselines) can be adjusted after training without fine-tuning while maintaining good performance. We develop a set of models with our improvements and compare them against available state-of-the-art baseline models and show accuracy, speed, and memory consumption improvements. Lastly, we release several model variants and our dataset for use by the community.

ACKNOWLEDGMENT

Special thanks to Wenshan Wang and the TartanAir team for allowing us to use their packaged simulation environments for collecting our dataset. Special thanks to Siheng Teng for helping with the hardware acceleration of our model. This work was supported and funded by the Defence Science and Technology Agency, Singapore. This work used Bridges-2 at PSC through allocation cis220039p from the Advanced Cyberinfrastructure Coordination Ecosystem: Services & Support (ACCESS) program which is supported by NSF grants #2138259, #2138286, #2138307, #2137603, and #213296.

References

[1] C. Won, J. Ryu, and J. Lim, "Omnimvs: End-to-end learning for omnidirectional stereo matching," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 8987–8996, 2019.

[2] A. Meuleman, H. Jang, D. S. Jeon, and M. H. Kim, "Real-time sphere sweeping stereo from multiview fisheye images," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 11423–11432, 2021.

[3] S. Xie, D. Wang, and Y.-H. Liu, "Omnividar: Omnidirectional depth estimation from multi-fisheye images," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 21529–21538, 2023.

[4] C. Won, J. Ryu, and J. Lim, "Sweepnet: Wide-baseline omnidirectional depth estimation," in *2019 International Conference on Robotics and Automation (ICRA)*, pp. 6073–6079, IEEE, 2019.

[5] C. Won, J. Ryu, and J. Lim, "End-to-end learning for omnidirectional stereo matching with uncertainty prior," *IEEE transactions on pattern analysis and machine intelligence*, vol. 43, no. 11, pp. 3850–3862, 2020.

[6] X. Su, S. Liu, and R. Li, "Omnidirectional depth estimation with hierarchical deep network for multi-fisheye navigation systems," *IEEE Transactions on Intelligent Transportation Systems*, 2023.

[7] Z. Chen, C. Lin, N. Lang, K. Liao, and Y. Zhao, "Unsupervised omnimvs: Efficient omnidirectional depth inference via establishing pseudo-stereo supervision," *arXiv preprint arXiv:2302.09922*, 2023.

[8] Y. Yao, Z. Luo, S. Li, T. Fang, and L. Quan, "Mvsnet: Depth inference for unstructured multi-view stereo," in *Proceedings of the European conference on computer vision (ECCV)*, pp. 767–783, 2018.

[9] K. N. Kutulakos and S. M. Seitz, "A theory of shape by space carving," *International journal of computer vision*, vol. 38, pp. 199–218, 2000.

[10] S. M. Seitz and C. R. Dyer, "Photorealistic scene reconstruction by voxel coloring," *International Journal of Computer Vision*, vol. 35, pp. 151–173, 1999.

[11] Y. Furukawa and J. Ponce, "Accurate, dense, and robust multiview stereopsis," *IEEE transactions on pattern analysis and machine intelligence*, vol. 32, no. 8, pp. 1362–1376, 2009.

[12] M. Lhuillier and L. Quan, "A quasi-dense approach to surface reconstruction from uncalibrated images," *IEEE transactions on pattern analysis and machine intelligence*, vol. 27, no. 3, pp. 418–433, 2005.

[13] N. D. Campbell, G. Vogiatzis, C. Hernández, and R. Cipolla, "Using multiple hypotheses to improve depth-maps for multi-view stereo," in *Computer Vision–ECCV 2008: 10th European Conference on Computer Vision, Marseille, France, October 12-18, 2008, Proceedings, Part I 10*, pp. 766–779, Springer, 2008.

[14] Q. Xu and W. Tao, "Multi-scale geometric consistency guided multi-view stereo," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 5483–5492, 2019.

[15] A. Kar, C. Häne, and J. Malik, "Learning a multi-view stereo machine," *Advances in neural information processing systems*, vol. 30, 2017.

[16] Y. Yao, Z. Luo, S. Li, T. Shen, T. Fang, and L. Quan, "Recurrent mvsnet for high-resolution multi-view stereo depth inference," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 5525–5534, 2019.

[17] H. Yi, Z. Wei, M. Ding, R. Zhang, Y. Chen, G. Wang, and Y.-W. Tai, "Pyramid multi-view stereo net with self-adaptive view aggregation," in *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part IX 16*, pp. 766–782, Springer, 2020.

[18] R. Peng, R. Wang, Z. Wang, Y. Lai, and R. Wang, "Rethinking depth estimation for multi-view stereo: A unified representation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8645–8654, 2022.

[19] X. Gu, Z. Fan, S. Zhu, Z. Dai, F. Tan, and P. Tan, "Cascade cost volume for high-resolution multi-view stereo and stereo matching," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 2495–2504, 2020.

[20] J. Li, Z. Lu, Y. Wang, Y. Wang, and J. Xiao, "Ds-mvsnet: Unsupervised multi-view stereo via depth synthesis," in *Proceedings of the 30th ACM International Conference on Multimedia*, pp. 5593–5601, 2022.

[21] S. Gao, Z. Li, and Z. Wang, "Cost volume pyramid network with multi-strategies range searching for multi-view stereo," in *Computer Graphics International Conference*, pp. 157–169, Springer, 2022.

[22] B. P. Duisterhof, Y. Hu, S. H. Teng, M. Kaess, and S. Scherer, "Tartancalib: Iterative wide-angle lens calibration using adaptive subpixel refinement of apriltags," 2022.

[23] V. Usenko, N. Demmel, and D. Cremers, "The double sphere camera model," *CoRR*, vol. abs/1807.08957, 2018.

[24] T. Nuanes, M. Elsey, A. Sankaranarayanan, and J. Shen, "Soft cross entropy loss and bottleneck tri-cost volume for efficient stereo depth prediction," in *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pp. 2840–2848, 2021.

[25] W. Wang, D. Zhu, X. Wang, Y. Hu, Y. Qiu, C. Wang, Y. Hu, A. Kapoor, and S. Scherer, "Tartanair: A dataset to push the limits of visual slam," 2020.