LODM: Large-scale Online Dense Mapping for UAV

Jianxin Huang*, Laijian Li* , Xiangrui Zhao, Xiaolei Lang, Deye Zhu, Yong Liu,

***Abstract*— This paper proposes a method for online large-scale dense mapping. The UAV is within a range of 150-250 meters, combining GPS and visual odometry to estimate the scaled pose and sparse points. In order to use the depth of sparse points for depth map, we propose Sparse Confidence Cascade View-Aggregation MVSNet (SCCVA-MVSNet), which projects the depth-converged points in the sliding window on keyframes to obtain a sparse depth map. The photometric error constructs sparse confidence. The coarse depth and confidence through normalized convolution use the images of all keyframes, coarse depth, and confidence as the input of CVA-MVSNet to extract features and construct 3D cost volumes with adaptive view aggregation to balance the different stereo baselines between the keyframes. Our proposed network utilizes sparse features point information, the output of the network better maintains the consistency of the scale. Our experiments show that MVSNet using sparse feature point information outperforms image-only MVSNet, and our online reconstruction results are comparable to offline reconstruction methods. To benefit the research community, we open-source our code at `https://github.com/hjxwhy/LODM.git`**

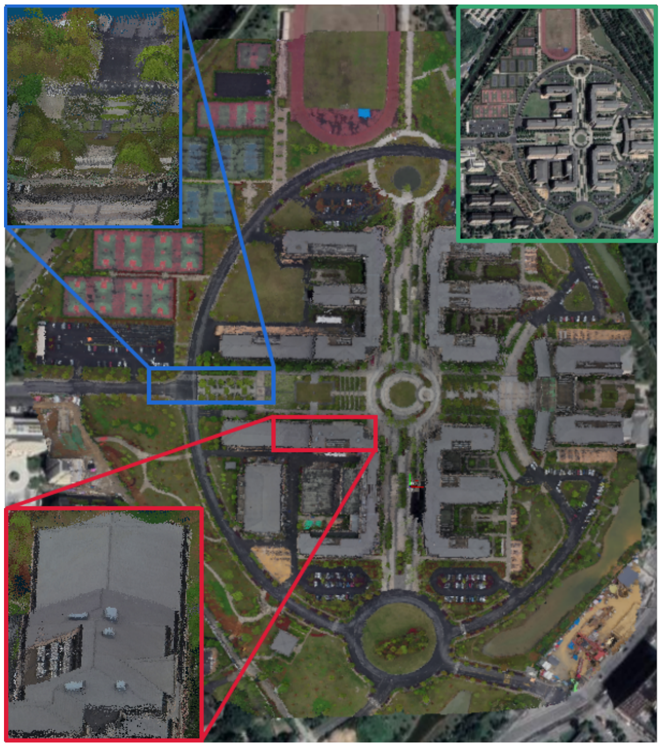

Fig. 1: LODM is a monocular online dense reconstruction method for large-scale outdoor fields. Our system can accurately reconstruct some ground details, such as the tilt angle of the roof and trees on the ground. The green box in the upper right corner is the satellite map, our method coincides with the satellite map

I. INTRODUCTION

Camera-based 3D online dense reconstruction is a challenging problem in the field of computer vision. At present, there are many methods based on different kinds of cameras, most of which are based on a depth camera or binocular camera. However, there are relatively few methods based on a monocular camera. We can easily get monocular cameras in our daily lives compared to depth cameras and binocular cameras, but using monocular reconstruction is the most difficult. The main reason is that the monocular camera can not directly obtain the depth information of the environment, so the algorithm needs to estimate the scene through the image sequence. In addition, it is impossible to obtain the scale information only by a monocular camera.

Some previous works are mostly oriented towards indoor scenes including methods based on traditional algorithms [1] and methods based on convolutional neural networks [2–8]. [8] is most closes to ours, but it is up-to-scale indoor 3D reconstruction. In addition, there are also methods for dense reconstruction of large outdoor scenes [9–12]. This paper proposes a monocular online dense reconstruction method for large-scale outdoor fields by combining multi-view stereo (MVS) and simultaneous localization and mapping (SLAM) techniques. We recover the scale by incorporating GPS information and also use SLAM's sparse geometric prior information to construct a scaled 3D world. We apply the proposed method for dense reconstruction of high-altitude UAVs and train on virtual and real scene datasets, demonstrating the robust adaptability of our method in complex scenes. Fig. 1 shows the reconstruction effect of our algorithm, in this large scene we can reconstruct online and can have good details.

To summarize, the main contributions of this paper are as follows:

- We propose a full outdoor large-scale dense reconstruction system that can be used for dense online reconstruction of high-altitude UAVs with scale.
- We leverage the sparse depth information and sparse confidence information provided by the sparse SLAM system to give the MVS coarse depth and depth confidence, achieving a better result than image only.
- Our system can run on real scene and simulation scene data and achieve results equal to the offline 3D reconstruction.

The authors are with the Institute of Cyber-Systems and Control, Zhejiang University, Hangzhou, China. (Yong Liu is the corresponding author, email: yongliu@iipc.zju.edu.cn)

* These authors contributed equally to this work.

II. RELATED WORK

3D dense reconstruction has been extensively studied, including optimizing 3D reconstruction of pure images through optimization methods and deep learning for end-to-end reconstruction. Combining deep learning and SLAM systems to simultaneously estimate pose and 3D reconstruction is also an important research direction.

A. Monocular Vision SLAM

Localization and Mapping using monocular cameras require step-by-step construction of the environment through inter-frame disparity, which is challenging, especially in large-scale scenes. Davison et al. [13] proposed the first monocular real-time visual SLAM system, which is the first successful application of the SLAM method to a monocular camera. The first visual slam system that separates the front and rear ends is PTAM proposed by Klein in 2007 [14]. After that, many feature point-based methods have been proposed, which perform SLAM work through feature point extraction and association, including the now widely used ORB-SLAM [15–17]. Unlike the SLAM based on the feature point method, the SLAM system based on the direct method directly optimizes the photometric error. Engel et al. [18] proposed the first large-scale direct method SLAM method. After that, Engel proposed a pure vision framework DSO [19], which optimizes the camera position and 3D point inverse depth in real-time in a sliding window. Gao et al. [20] added loop closure detection to DSO, which enabled DSO to use the bag-of-words model to detect loop through a new point selection strategy.

Some work related to deep learning has also been proposed in recent years. CNN-SLAM combines depth estimation with the SLAM system. The SLAM system tracks the pose, and the dense depth map of the keyframe uses the pose to perform Gaussian filtering and fuses the depth maps of adjacent frames for dense map construction. CodeSLAM, Deepfactor, CodeVIO and CodeMapping [4–7] are a series of dense reconstruction methods based on variational auto-encoding. The image is encoded into low-dimensional latent variables, and the low-dimensional latent variables are used as optimization variables. The depth map is optimized through the jacobian of the latent variables with the depth map while optimizing the pose.

B. Learning-Based Deep Completion

Monocular depth estimation is the task of supplementing image depth values from a single RGB image and sparse depth points. With the rapid development of deep learning, many deep completion methods use deep learning. Convolutional Neural Network (CNN) replaces manual features, which is more superior in results. Uhrig et al. [21] proposed Sparse-CNN in 2007 and proposed the concept of sparse invariant convolution, and a series of methods follow this idea. Ma et al. [22] proposed an encoder-decoder structure network for depth completion, where the sparse depth input to the network can be derived from a low-resolution depth sensor or a visual SLAM system and proved the positive guiding effect of sparse depth points on depth prediction. After that, Ma [23] proposed a self-supervised method and a more accurate network. Zhang [24] does not directly use convolutional neural networks for depth estimation but predicts surface normals to densify the depth map. Chen et al. [25] proposed the method of extrapolating the sparse depth map and then refining the depth map using the network. For the UAV scene, Teixeira et al. [26] proposed a depth completion and uncertainty estimation method, which can better cope with the challenges of prominent viewpoints and depth changes of aerial platforms.

C. 3D Reconstruction

Traditional methods take images and pose as input and reconstruct 3D environments or objects through iterative optimization. Since the traditional method needs to optimize the objective function iteratively, it generally takes time. Schöps et al. [11] perform temporal, plane-sweep-based depth estimation using the poses and images obtained from a mobile tracking device for outdoor 3D reconstruction. DTAM [1] is a system for real-time camera tracking and reconstruction that relies not on feature extraction but dense, every pixel method.

Nowadays, the 3D cost volumes-based method is a popular research method. DeepMVS [27] proposed one of the first cost-volume-based approaches, which takes an arbitrary number of posed images as input and predicts high-quality disparity maps by using a set of plane-sweep volumes. MVSNet [28] proposes to build the 3D cost volume based on a 2D deep feature map via the differentiable homography warping and then use the 3D convolution to regress the initial depth map, which is then refined with the reference image to generate the final output. Yi et al. [29] introduce two novel self-adaptive view aggregations to weight pixel-wise view aggregation and voxel-wise view aggregation for different view images. To overcome the time-consuming and memory consumption of MVS, Gu et al. [30] propose a cascade cost volume which is built upon a feature pyramid encoding geometry and context at gradually finer scales and narrowing the depth (or disparity) range of each stage by the prediction from the previous stage. The closest to our work is TANDEM [8], who combined Gu et al. [30] and Yi et al. [29] into the SLAM system and proposed CVA-MVSNet to achieve better results in indoor environments. Inspired by TANDEM, we combine visual SLAM and GPS to estimate scaled poses in outdoor scenes. We use sparse depth points to provide CVA-MVSNet [8] with a coarse depth and an uncertainty map through the confidence propagates network proposed by [31] to make a large-scale 3D reconstruction at scale.

III. METHODOLOGY

The large-scale mapping method proposed by us includes three-part:visual-GPS odometer, confidence-based depth estimation with CVA-MVSNet, dense mapping, Fig.2 shows an overview of the system. The visual odometry front end uses the image and the depth map output by the network to

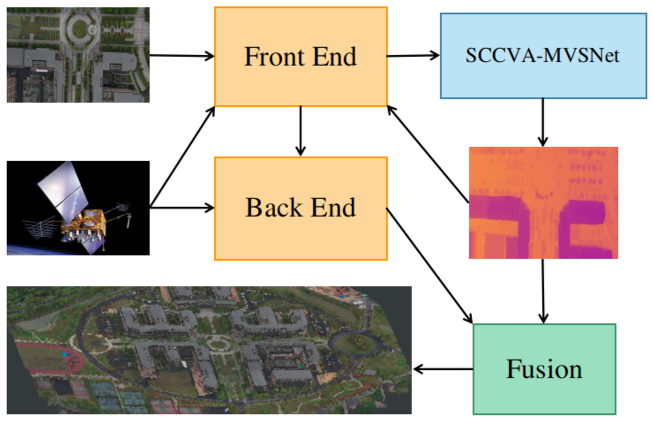

Fig. 2: The front-end uses images and GPS to generate a scaled initial pose and also receives the depth map from the network; SCCVA-MVSNet uses the information from the front-end sliding window to recover the depth of the latest frame; The back-end uses GPS to optimize the front-end Pose further; The Fusion module uses the Pose from the back-end and the depth map output by the network to fuse the final point cloud map.

output the pose and utilizes GPS to recover the scale. The SCCVA-MVSNET network utilizes frames and sparse points in the front-end sliding window for depth recovery. The 3D reconstruction process uses the back-end optimized pose and fuses the depth map of each keyframe to generate a dense 3D point cloud map.

A. *Visual-GPS Odometry*

We extend the LDSO [20], use GPS to give the actual scale to the DSO as our front-end, and add GPS constraints when optimizing the pose map at the loop closing thread as our back-end. In addition, we give the depth map output by the network to the latest keyframe in the sliding window and assign the depth value to the point just selected in this frame. In the back-end, we will use the pose constraints obtained in the front-end to constrain the pose between the two frames, and we will keep the loop closure method of LDSO. However, we do not use loop closure detection in the following experiments in section IV. We optimize the pose in the pose graph, and the paired errors are expressed as:

$$e_{ij} = \hat{T}_j^{-1}\hat{T}_i T_{ij} \tag{1}$$

Where $\hat{(\cdot)}$ is the estimated value of the variable, and T_{ij} represents the pose constraint of frame J to frame I, which can get from the front end or loop closing. In addition, we also integrate GPS in the back end and add GPS constraints to the pose to constrain the position. The error term can be expressed as:

$$e_i = |\hat{t_i} - t_{gi}| \tag{2}$$

Where t_i is the estimated value of the position of the i frame, and t_{gi} is the value of the GPS to the local coordinate system. Finally, we use G2O [32] to build the optimization graph and perform graph optimization. The overall optimization graph is shown in the Fig. 4.

B. *SCCVA-MVSNet*

Our overall network structure is shown in Fig. 3. we take the information of keyframes in the window as the input of the network. However, instead of directly inputting the image and the corresponding pose, we use the depth and confidence maps of the sparse feature points of the image together as the input to the network. The input of our network can be expressed as $\{I_i, \mathcal{S}_i, \mathcal{C}_i, \mathbf{T}_i\}, i = 1, 2, \ldots, N$, where N is the number of keyframes in the sliding window of the Vision-GPS odometry front end, and I_i represents An image of size (H, W), $\mathcal{S}_i, \mathcal{C}_i$ are the sparse feature depth map and parameterized confidence map of the image I_i, respectively, $\mathbf{T_i}$ represents the pose of each image information, which can be obtained directly from the front end of SLAM. Vision-GPS odometry can provide scaled sparse feature points, and we will move all feature points $\{P_1, P_2, \cdots, P_M\}$ in the sliding window to all keyframes are projected to get the sparse depth map S_i of each keyframe in the sliding window.

We use the photometric error of the sparse point as the confidence of the sparse input point to get the confidence map $\mathcal{C}_i$ of the keyframe,more specifically, we will calculate the confidence map of each frame with the following formula [4]:

$$c_{ik} = \frac{a}{a + e_{ik}} \tag{3}$$

Where e_{ik} is the photometric error of a specific point P_k in the sliding window projected to the i frame, if the host frame of P_k is frame i, then we will take P_k project to other keyframes and take minimum photometric. a is an average photometric error, and it maps the confidence to $[0, 1]$ by the Eq. 3 we can calculate the depth map S_i with mutual projection while the confidence map C_i, and the point with more significant photometric error will have little confidence.

The first part is a Normalized Convolution Network, a combination of normalized convolutions with confidence-aware max-pooling for down-sampling and bilinear for up-sampling. The input of Normalized Convolution Network is the sparse feature depth map $\mathcal{S}_1, \mathcal{S}_2, \ldots, \mathcal{S}_N$ and parameterized confidence map $\mathcal{C}_1, \mathcal{C}_2, \ldots, \mathcal{C}_N$.The output of the network is estimated dense feature depth map $\mathcal{S}'_1, \mathcal{S}'_2, \ldots, \mathcal{S}'_N$and parameterized confidence map $\mathcal{C}'_1, \mathcal{C}'_2, \ldots, \mathcal{C}'_N$.

Confidence is propagated forward in the network through the following formula:

$$\mathcal{C}^l_{u,v} = \frac{\sum_{m,n} \mathcal{C}^{l-1}_{u+m,v+n} \Gamma\left(\mathcal{W}^l_{m,n}\right)}{\sum_{m,n} \Gamma\left(\mathcal{W}^l_{m,n}\right)} + b^l \tag{4}$$

where $\mathcal{C}^l_{u,v}$ is the confidence output of the lth layer at locations u, v depend on the weight element $\mathcal{W}^l_{m,n}$, which is the network kernel weight, and the bias b^l, both are learnable parameters. $\mathcal{C}^{l-1}$ is the output confidence from the previous

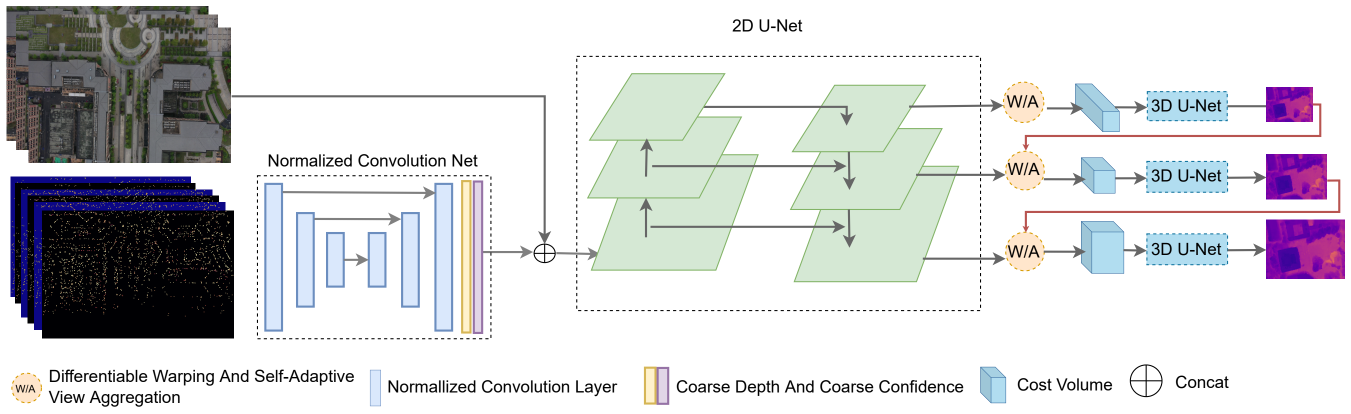
2D U-Net
Normalized Convolution Net
W/A
3D U-Net
Differentiable Warping And Self-Adaptive View Aggregation
Normallized Convolution Layer
Coarse Depth And Coarse Confidence
Cost Volume
Concat

Fig. 3: The converged feature points of all keyframes in the window are projected between all frames to form sparse depth maps(blue) and confidence maps (black) as the input of the Normalized Convolution Net. The second stage's U-Net takes concatenated coarse depth, confidence, and image to extract features for building cascaded cost volumes and hierarchically estimates the depth maps to build.

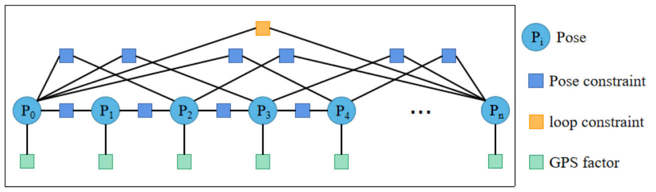

Fig. 4: As shown in the figure, the blue circle represents the pose to be optimized, the blue rectangle represents the pose constraint obtained by the front end, the orange rectangle represents the pose constraint obtained by the loop closure detection, and the green rectangle represents the GPS constraint.

layer.Γ is non-negativity enforcement function, in this paper we use SoftPlus function.

The depth is propagated forward by the following formula:

$$\mathcal{S}_{u,v}^{l}=\frac{\sum_{m,n}\mathcal{S}_{u+m,v+n}^{l-1}\mathcal{C}_{u+m,v+n}^{l-1}\Gamma\left(\mathcal{W}_{m,n}^{l}\right)}{\sum_{m,n}\mathcal{C}_{u+m,v+n}^{l-1}\Gamma\left(\mathcal{W}_{m,n}^{l}\right)+\epsilon}+b^{l} \tag{5}$$

where ϵ is a constant to prevent division by zero.Assuming that the neighbors in the depth map are locally similar, the sparse depth map is interpolated into the coarse depth map based on the confidence provided by the photometric error via Eq. 5. Points with higher confidence contribute more to the surrounding pixels, yielding a more accurate depth.When training the network, we will use Smooth L1 Loss to constrain the depth of the final output,let $\mathcal{S}_i^{gt}$ be the ground truth value,then it can be expressed as:

$$\begin{aligned} &x_{i,u,v}=\mathcal{S}_{i,u,v}^{gt}-\mathcal{S}_{i,u,v}^{\prime}\\ &smooth_{L1}(x)=\begin{cases}0.5x^2, & \text{if } |x|<1\\ |x|-0.5, & \text{if } otherwise\end{cases}\\ &L_{smooth}=\sum_{i=1}^{N}\sum_{u=0}^{W-1}\sum_{v=0}^{H-1} smooth_{L1}(x_{i,u,v})\end{aligned} \tag{6}$$

The second part is modified from CVA-MVSNet [8], CVA-MVSNet is based on the principles of multi-view stereo [33] and leverages deep neural networks [28] to estimate a depth map for the reference frame I_n. We leverage the CVA-MVSNet with a sparse depth map and confidence to directly estimate the depth map D_n of I_n . Unlike [8],we concatenate the image I_i with the coarse depth and confidence from the first stage and extract the feature map F_i^s of the N images through the 2D UNet network, where i represents $[1,N]$ images, s represents different scales.

C. Multi-View Geometric Consistency Mapping

This stage aims to integrate the depth map of the active keyframe based on the depth maps estimated. CNN-SLAM [2] fuse the previous keyframe depth and current keyframe depth by uncertainty map. [10] use the photometric and geometric consistency terms to detect outliers at a negligible computational cost robustly. Let $\mathbf{R}_{i\to j}$ and $\mathbf{t}_{i\to j}$ is the rotation and translation from source frame i to target frame $j\mathbf{K}$ is the camera intrinsic. In order to reduce the filtering time, we take five frames to detect outliers by the geometric consistency.

$$\mathbf{p}_j=\pi_j\left(\mathbf{R}_{i\to j}\phi\left(\mathbf{p}_i,d_i,\mathbf{K}_i\right)+\mathbf{t}_{i\to j},\mathbf{K}\right) \tag{7}$$

where $\phi(\mathbf{p}_i,d_i,\mathbf{K})=\mathbf{P}_i$ is the unprojection of a pixel in homogeneous coordinates $\mathbf{p}_i$ to a 3D point $\mathbf{P}_i$ for a given estimated depth d_i. Denote the projection of a 3D point back onto the image plane as $\pi_j(\mathbf{P}_j,\mathbf{K})=\mathbf{p}_j$. We can get the depth d_j on the frame j according to p_j,then we do a backprojection:

$$\begin{aligned} \mathbf{p}_i^{\prime}&=\pi_i\left(\mathbf{R}_{j\to i}\phi\left(\mathbf{p}_j,d_j,\mathbf{K}_j\right)+\mathbf{t}_{j\to i},\mathbf{K}\right)\\ d_i^{\prime}&=\left[\mathbf{R}_{j\to i}\phi\left(\mathbf{p}_j,d_j,\mathbf{K}_j\right)+\mathbf{t}_{j\to i}\right]_z\end{aligned} \tag{8}$$

where $[\cdot]_z$ represents the value of the z-axis. We compute the geometric consistency between two views as the forward

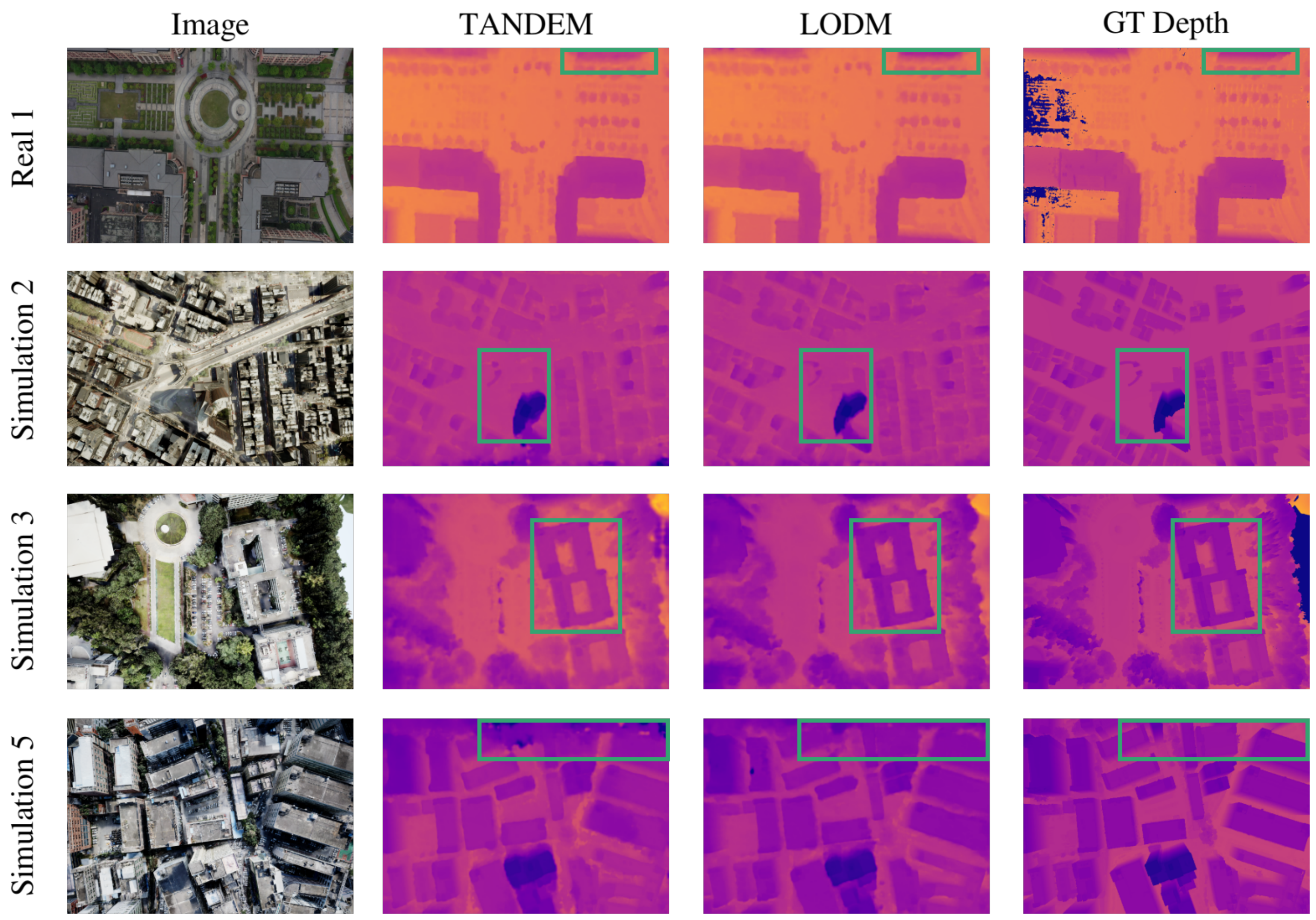

Fig. 5: Depth comparison for TANDEM and ours on real and simulated scenes, all the depth visualization maps are normalized with 250m. LODM keeps the scale well, and the boundaries of objects are more clearer, e.g., the building in the second row or LODM keeps the scale well in the third row.

reprojection error $\|\mathbf{p}_i - \mathbf{p}'_i\|^2$ and the depth error $\|d_i - d'_i\|$. Our fusion finds clusters of consistent pixels in multi frames that satisfy reprojection error $\|\mathbf{p}_i - \mathbf{p}'_i\|^2 < \psi$ and depth error $\|d_i - d'_i\| < \epsilon$ constrain. We fuse the cluster's elements if it has at least three elements. The fused point has median location overall cluster elements. The median location is used to avoid artifacts when averaging over multiple neighboring pixels at large depth discontinuities [10].

IV. EXPERIMENTS

In order to achieve online dense graphs, we put the front-end visual odometer and back-end GPS optimization on different CPU threads, and the DNN inference runs on the GPU. We train our network in PyTorch [34] and perform inference in C++ using TorchScript. The network is trained on four NVIDIA RTX 3090 GPU for 50 epochs, the learning rate is 0.004, linear decay from 0.004 to 0.004/100, image size is 640 x 480, the batch size is 2, and the tuple is seven keyframes in SLAM.

We did not find a dataset with GPS above 150m suitable for our scene, so we used our drone to collect datasets of the real scene at 150-200m altitude. We use COLMAP [10] to reconstruct the depth value of each image and use the resulting depth map as the ground-truth depth for training and evaluation. In order to make the reconstruction results of COLMAP more accurate, we input the pose estimated by SLAM into COLMAP, so that the reconstruction results are more accurate and faster. We also collected simulation images from UrbanScene3D. We use the data collected from the real scene and the data from the simulated environment for training.

A. *Trajectory evaluation*

We evaluate Visual-GPS Odometry for pose estimation against other state-of-the-art monocular SLAM methods on simulation datasets. we evaluate against DSO [19], ORB-SLAM2 [16], TANDEM [8]. TANDEM is closest to ours, and we re-train the model in the aerial data. Note that for a fair comparison, we turn off global optimization and relocalization for ORB-SLAM2 and loop closure detection for ours. We trained tandem's model on our data, but we found that the depth map output by the network had obvious scale drift when running TANDEM, which may be the reason for its unsatisfactory results. Our method is scaled, which

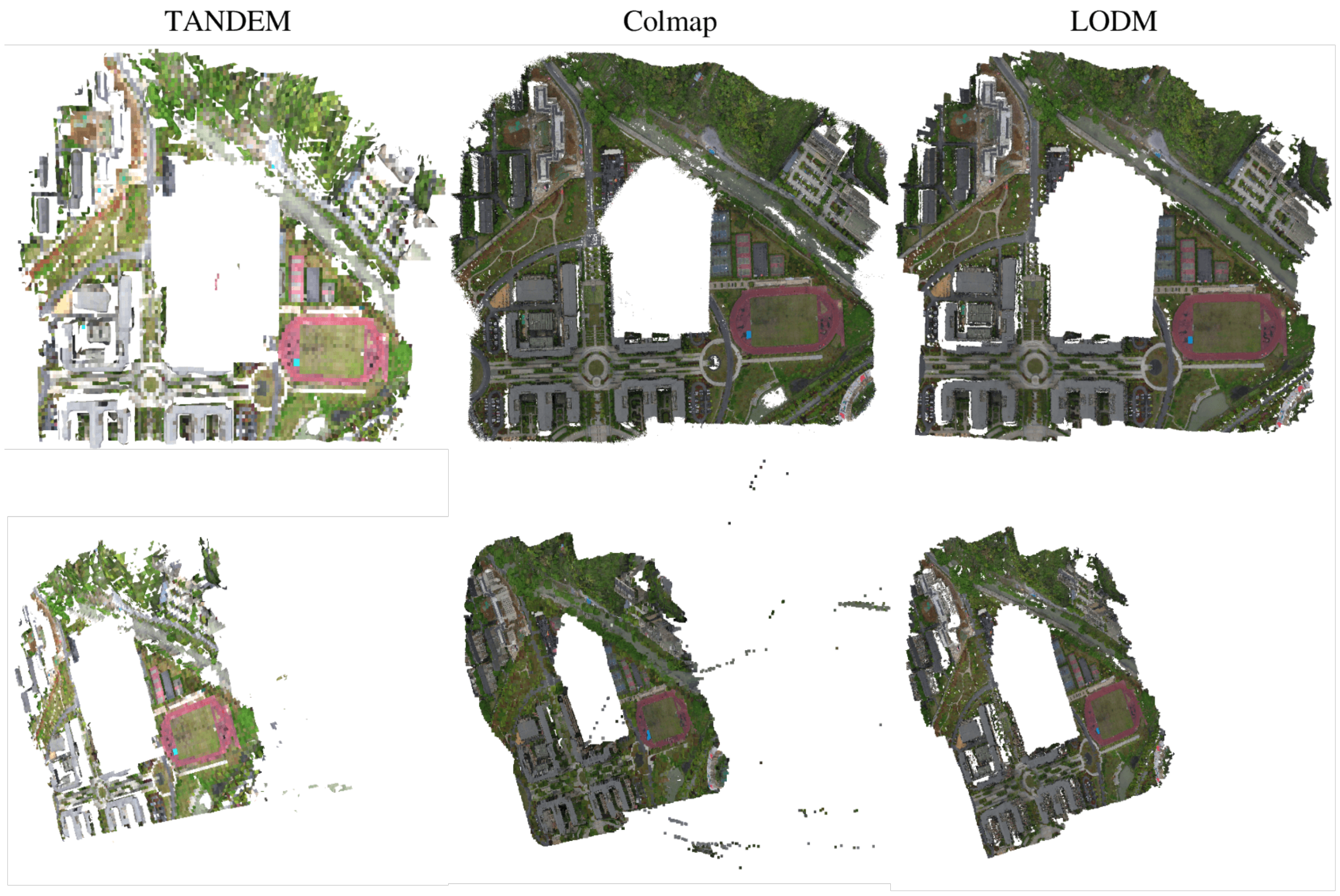

Fig. 6: The first column is the reconstruction of TANDEM; the second column is the map obtained by using COLMAP offline processing; the third column is the map obtained by our online processing. Our overall effect is the same as COLMAP, and we have little noise.

TABLE I: Pose evaluation on Sequence dataset. All the methods are Sim(3) aligned w.r.t. the ground-truth trajectories. The mean absolute pose errors deviations over five runs are shown

Sequence	DSO	ORB-SLAM2	TANDEM	Ours
Simulation 1	1.491321	<u>0.632357</u>	1.770763	**0.305023**
Simulation 2	<u>0.912785</u>	1.053979	1.046717	**0.883941**
Simulation 3	<u>0.445929</u>	0.472736	0.448445	**0.277206**
Simulation 4	<u>0.496166</u>	-	0.510875	**0.328094**
Simulation 5	<u>0.420060</u>	0.457639	0.623198	**0.370440**

makes our method more applicable to large scenes.

We evaluated after doing Sim(3) alignment with the real world. Table I shows our results on the simulated dataset. It can be seen that there is a certain improvement in each simulation data set after using GPS through back-end optimization. Our method utilizes GPS, this comparison is not fair, but it also shows that our method can better use GPS information. In addition, our method is designed for large-scale drones, which usually have excellent satellite signals.

TABLE II: Depth evaluation on real and simulation scene.

		TANDEM	Ours
Real 1	AE(m)	1.158	**1.016**
	RMSE(m)	1.879	**1.804**
	a_1(%)	81.9	**85.4**
Simulation 2	AE(m)	6.736	**5.434**
	RMSE(m)	10.507	**8.735**
	a_1(%)	23.4	**24.7**
Simulation 5	AE(m)	7.543	**3.116**
	RMSE(m)	9.958	**4.878**
	a_1(%)	10.1	**40.5**

B. Depth map evaluation

We evaluate the accuracy of the reconstruction on the simulated and real datasets. We compare with TANDEM [8], We show the percentage of pixels for which the estimated depth falls within 1% of the groundtruth value (a_1), the Root Mean Square Error (RMSE), and Absolute Error (AE). Our method outperforms image-only methods on all metrics in table II. The *simulation 2* sequence is a CAD model, and we do not have such images in the training dataset, so the performance drops a lot. The *simulation 5* is a very crowded urban simulation environment, so the performance of both algorithms drops, but our algorithm also performs better. A depth error of 1% for an altitude of 200 meters is a very critical metric, so our algorithm is competent for dense reconstruction in high-altitude environments.

We show some depth map results in Figure 5, where the groundtruth of Real 1 sequences are obtained by COLMAP reconstruction. We can see in the figure that our method does better in detail. In particular, our method shows strong robustness on complex data, while TANDEM's network produces incorrect spurious points in complex scenarios (as in the fourth row). Figure 6 qualitatively shows the results of our map building, where TANDEM outputs a TSDF map, while both our method and COLMAP result in a point cloud map. From the overall effect, our result is closer to COLMAP, but our method does not have as severe outlier points as COLMAP. Similarly, TAMDEN also has more outliers.

V. CONCLUSIONS and Future Work

We propose LODM, a monocular outdoor large-scale in-the-spot reconstruction method, which we use for 3D online reconstruction of large-scale drone scenes. We propose the SCVA-MVSNet network, which can quickly complete sparse depth maps to obtain high-quality depth maps. We show the performance of our method in large-scale drone scenes, and our method can achieve similar results as offline methods. The 3D online reconstruction of large-scale UAV scenes is most studied in the field of surveying and mapping, usually multi-camera offline methods, some of which can achieve high accuracy. We believe that in the future our method can achieve the same accuracy as drone mapping without taking a lot of time.

References

[1] R. A. Newcombe, S. J. Lovegrove, and A. J. Davison. "DTAM: Dense tracking and mapping in real-time". In: *2011 international conference on computer vision*. IEEE. 2011, pp. 2320–2327.

[2] K. Tateno, F. Tombari, I. Laina, and N. Navab. "Cnn-slam: Real-time dense monocular slam with learned depth prediction". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, pp. 6243–6252.

[3] H. Zhou, B. Ummenhofer, and T. Brox. "Deeptam: Deep tracking and mapping". In: *Proceedings of the European conference on computer vision (ECCV)*. 2018, pp. 822–838.

[4] M. Bloesch, J. Czarnowski, R. Clark, S. Leutenegger, and A. J. Davison. "CodeSLAM—learning a compact, optimisable representation for dense visual SLAM". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2018, pp. 2560–2568.

[5] J. Czarnowski, T. Laidlow, R. Clark, and A. J. Davison. "Deepfactors: Real-time probabilistic dense monocular slam". In: *IEEE Robotics and Automation Letters* 5.2 (2020), pp. 721–728.

[6] X. Zuo, N. Merrill, W. Li, Y. Liu, M. Pollefeys, and G. Huang. "CodeVIO: Visual-inertial odometry with learned optimizable dense depth". In: *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2021, pp. 14382–14388.

[7] H. Matsuki, R. Scona, J. Czarnowski, and A. J. Davison. "CodeMapping: Real-Time Dense Mapping for Sparse SLAM using Compact Scene Representations". In: *IEEE Robotics and Automation Letters* 6.4 (2021), pp. 7105–7112.

[8] L. Koestler, N. Yang, N. Zeller, and D. Cremers. "TANDEM: Tracking and Dense Mapping in Real-time using Deep Multi-view Stereo". In: *Conference on Robot Learning*. PMLR. 2022, pp. 34–45.

[9] J. L. Schönberger and J.-M. Frahm. "Structure-from-Motion Revisited". In: *Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016.

[10] J. L. Schönberger, E. Zheng, M. Pollefeys, and J.-M. Frahm. "Pixelwise View Selection for Unstructured Multi-View Stereo". In: *European Conference on Computer Vision (ECCV)*. 2016.

[11] T. Schöps, T. Sattler, C. Häne, and M. Pollefeys. "Large-scale outdoor 3D reconstruction on a mobile device". In: *Computer Vision and Image Understanding* 157 (2017), pp. 151–166.

[12] F. Wimbauer, N. Yang, L. Von Stumberg, N. Zeller, and D. Cremers. "MonoRec: Semi-supervised dense reconstruction in dynamic environments from a single moving camera". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2021, pp. 6112–6122.

[13] A. J. Davison, I. D. Reid, N. D. Molton, and O. Stasse. "MonoSLAM: Real-time single camera SLAM". In: *IEEE transactions on pattern analysis and machine intelligence* 29.6 (2007), pp. 1052–1067.

[14] G. Klein and D. Murray. "Parallel tracking and mapping for small AR workspaces". In: *2007 6th IEEE and ACM international symposium on mixed and augmented reality*. IEEE. 2007, pp. 225–234.

[15] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardos. "ORB-SLAM: a versatile and accurate monocular SLAM system". In: *IEEE transactions on robotics* 31.5 (2015), pp. 1147–1163.

[16] R. Mur-Artal and J. D. Tardós. "Orb-slam2: An open-source slam system for monocular, stereo, and rgb-d cameras". In: *IEEE transactions on robotics* 33.5 (2017), pp. 1255–1262.

[17] C. Campos, R. Elvira, J. J. G. Rodríguez, J. M. Montiel, and J. D. Tardós. "Orb-slam3: An accurate open-source library for visual, visual–inertial, and multimap slam". In: *IEEE Transactions on Robotics* 37.6 (2021), pp. 1874–1890.

[18] J. Engel, T. Schöps, and D. Cremers. "LSD-SLAM: Large-scale direct monocular SLAM". In: *European conference on computer vision*. Springer. 2014, pp. 834–849.

[19] J. Engel, V. Koltun, and D. Cremers. “Direct sparse odometry”. In: *IEEE transactions on pattern analysis and machine intelligence* 40.3 (2017), pp. 611–625.

[20] X. Gao, R. Wang, N. Demmel, and D. Cremers. “LDSO: Direct sparse odometry with loop closure”. In: *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2018, pp. 2198–2204.

[21] J. Uhrig, N. Schneider, L. Schneider, U. Franke, T. Brox, and A. Geiger. “Sparsity invariant cnns”. In: *2017 international conference on 3D Vision (3DV)*. IEEE. 2017, pp. 11–20.

[22] F. Ma and S. Karaman. “Sparse-to-dense: Depth prediction from sparse depth samples and a single image”. In: *2018 IEEE international conference on robotics and automation (ICRA)*. IEEE. 2018, pp. 4796–4803.

[23] F. Ma, G. V. Cavalheiro, and S. Karaman. “Self-supervised sparse-to-dense: Self-supervised depth completion from lidar and monocular camera”. In: *2019 International Conference on Robotics and Automation (ICRA)*. IEEE. 2019, pp. 3288–3295.

[24] Y. Zhang and T. Funkhouser. “Deep depth completion of a single rgb-d image”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018, pp. 175–185.

[25] Z. Chen, V. Badrinarayanan, G. Drozdov, and A. Rabinovich. “Estimating depth from rgb and sparse sensing”. In: *Proceedings of the European Conference on Computer Vision (ECCV)*. 2018, pp. 167–182.

[26] L. Teixeira, M. R. Oswald, M. Pollefeys, and M. Chli. “Aerial single-view depth completion with image-guided uncertainty estimation”. In: *IEEE Robotics and Automation Letters* 5.2 (2020), pp. 1055–1062.

[27] P.-H. Huang, K. Matzen, J. Kopf, N. Ahuja, and J.-B. Huang. “Deepmvs: Learning multi-view stereopsis”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018, pp. 2821–2830.

[28] 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)*. 2018, pp. 767–783.

[29] 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: *European Conference on Computer Vision*. Springer. 2020, pp. 766–782.

[30] 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*. 2020, pp. 2495–2504.

[31] A. Eldesokey, M. Felsberg, and F. S. Khan. “Confidence propagation through cnns for guided sparse depth regression”. In: *IEEE transactions on pattern analysis and machine intelligence* 42.10 (2019), pp. 2423–2436.

[32] R. Kümmerle, G. Grisetti, H. Strasdat, K. Konolige, and W. Burgard. “g 2 o: A general framework for graph optimization”. In: *2011 IEEE International Conference on Robotics and Automation*. IEEE. 2011, pp. 3607–3613.

[33] Y. Furukawa and C. Hernández. “Multi-view stereo: A tutorial”. In: *Foundations and Trends® in Computer Graphics and Vision* 9.1-2 (2015), pp. 1–148.

[34] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, et al. “Pytorch: An imperative style, high-performance deep learning library”. In: *Advances in neural information processing systems* 32 (2019).