VGF-Net: Visual-Geometric Fusion Learning
for Simultaneous Drone Navigation and Height Mapping

Yilin Liu
Shenzhen University

Ke Xie
Shenzhen University

Hui Huang*
Shenzhen University

Abstract

The drone navigation requires the comprehensive understanding of both visual and geometric information in the 3D world. In this paper, we present a *Visual-Geometric Fusion Network* (VGF-Net), a deep network for the fusion analysis of visual/geometric data and the construction of 2.5D height maps for simultaneous drone navigation in novel environments. Given an initial rough height map and a sequence of RGB images, our VGF-Net extracts the visual information of the scene, along with a sparse set of 3D keypoints that capture the geometric relationship between objects in the scene. Driven by the data, VGF-Net adaptively fuses visual and geometric information, forming a unified *Visual-Geometric Representation*. This representation is fed to a new *Directional Attention Model* (DAM), which helps enhance the visual-geometric object relationship and propagates the informative data to dynamically refine the height map and the corresponding keypoints. An entire end-to-end information fusion and mapping system is formed, demonstrating remarkable robustness and high accuracy on the autonomous drone navigation across complex indoor and large-scale outdoor scenes.

1. Introduction

In recent years, we have witnessed the development of autonomous robotic systems that have been broadly used in many scenarios (e.g., autonomous driving, manufacturing and surveillance). Drone belongs to the robotic system, and is well-known for its flying capacity. Navigation is extremely important to the drone fly, as it facilitates the effective exploration and recognition of the unknown environments. Yet, the navigation of drone remains a challenging task, especially for planning the pathway as short as possible to the target/destination whilst avoiding the potential collision with objects in the unexplored space. The conventional navigation heavily relies on the expertise of human, who intuitively designs the drone flyby trajectory based on the spatial layout within the visible range. The resulting

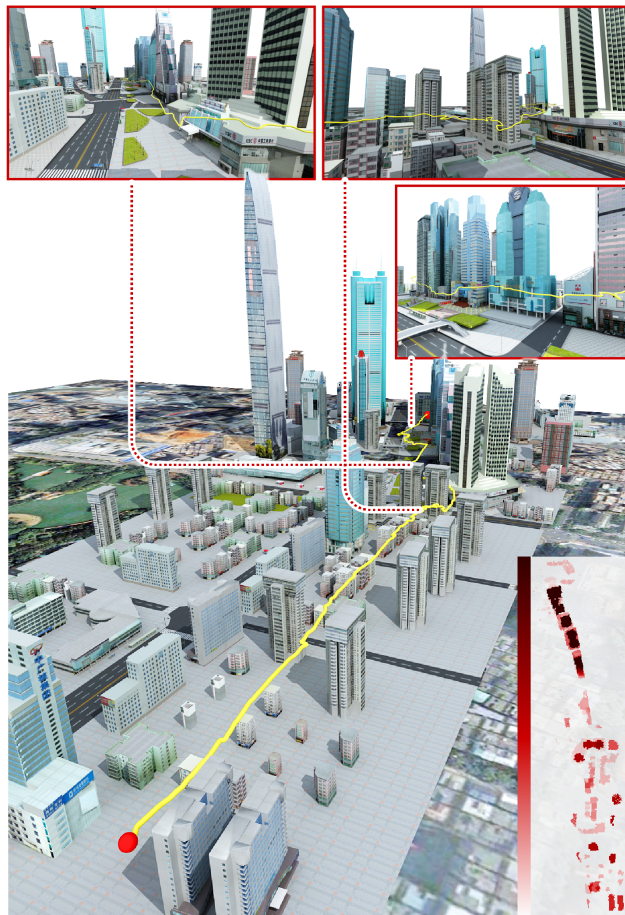*Corresponding author: Hui Huang (hhzhiyan@gmail.com)

Figure 1: We show a drone navigation trajectory (yellow curve) in 3D scene, which connects the starting and target points (red dots). During the navigation, our VGF-Net dynamically updates the 2.5D height map (see the bottom-left corner) in new places (see pictures in red rectangles), which is used to timely update the navigation trajectory.

navigation system lacks of the globe knowledge of scenes, leading to unsatisfactory or even failed path planning.

To better leverage the global information of 3D environment, researches on drone navigation have focused on collecting and memorizing the environmental informa-

tion during the navigating process. Typically, the existing works [11, 2, 3] employ the mapping techniques to construct 2D/3D maps with respect to the vacant/occupied space. The mapping result contains rich geometric relationship between objects, which helps to navigate. There have also been navigation approaches based on visual information [6, 10, 2], saving the computational overhead to construct maps. Nonetheless, these works purely condition the accuracy of navigation on either geometric or visual information.

In this paper, we utilize 2.5D height map for autonomous drone navigation. There are growing computer applications that use height map to represent the boundaries of objects (e.g., buildings or furniture). Nonetheless, there is nothing guaranteed for the quality of given height maps, as the mapping process likely involves incomplete or out-of-date information. Here, we advocate the importance of fusing geometric and visual information for a more robust construction of the height map. The new trend of researches [22, 5] on the 3D object/scene understanding has also demonstrated that the geometric relationship between objects and visual appearance of scenes are closely correlated. We thus propose a *Visual-Geometric Fusion Network* (VGF-Net) to dynamically update the height map during drone navigation by utilizing the timely captured new images (see Figure 1).

More specifically, as illustrated in Figure 2, the network takes an initial rough height map together with a sequence of RGB images as input. We use convolutional layers to compute the visual and geometric information to renew the height map. Next, we apply the simultaneous localization and mapping (SLAM) [18] module to extract a sparse set of 3D keypoints from the image sequence. These keypoints are used along with the renewed height map to construct a novel *Visual-Geometric Representation*, which is passed to a *Directional Attention Model*. This attention model exchanges visual and geometric information among objects in the scene, providing quite useful object relationship for simultaneous refinement of the height map and the corresponding keypoints, leading to the successful path planning [13] at each navigation moment. Compared to dense point clouds that require time-consuming depth estimation [4] and costly processing, the sparse keypoints we use are fast to compute yet effective in terms of capturing useful geometric information without much redundancy. As the drone flies over more and more places, our network can achieve and fuse more and more the visual and geometric information to largely increase the precision of height map and consequently the reliability of autonomous navigation.

We intensively train and evaluate our method on a benchmark of seven large-scale urban scenes and six complex indoor scenes for height map construction and drone navigation. The experimental results and comparative statistics clearly demonstrate the effectiveness and the robustness of our proposed VGF-Net.

2. Related Work

There have been an array of researches on the navigation system that allows robots to smartly explore the real world. Below, we will mainly survey on the drone navigation and environment mapping, as they are highly relevant to our work in the sense that their navigation systems are driven by the critical environment data.

2.1. Drone Navigation

The modern drone systems are generally equipped with various sensors (e.g., RGB-D camera, radar and GPS), which help the hardware devices to achieve accurate perception of the real world. Typically, the data captured by sensors is used for mapping (i.e., the construction of map), providing comprehensive information for planning the moving path of drone. During the navigation process, the traditional methods [11, 20] compute the trajectory of drone based on the pre-scribed maps. However, the construction of a precise map is generally expensive and time-consuming. Thus, the recent works [6, 10, 2] simplify the construction of map to facility more commercially-cheap navigation.

The advances on deep learning have significantly improved the robustness of visual navigation, leading to the emergency of many navigation systems that do not rely on the given maps. Kim et al. [12] and Padhy et al. [19] use the classification neural network to predict the direction (e.g., right, left or straight) of moving drone. Furthermore, Loquercio et al. [15] and Mirowski et al. [17] use neural networks to compute the angle of flying and the risk of collision, which provide more detailed information to control the drone flyby. Note that the above methods learn the actions of drone from the human annotations. The latest works employ deep reinforcement learning [21, 26, 24] to optimize the network, enabling more flexible solutions for autonomous drone navigation in novel environments.

Our approach utilizes a rough 2.5D height map to increase the success rate of navigation in different complex scenes, which may have various spatial layouts of objects. Compared to the existing methods that conduct the mapping before navigation, we allow for real-time intelligent update of the height map during navigation, largely alleviating negative impacts of problematic mapping results.

2.2. Mapping Technique

The mapping technique is fundamental in the drone navigation. The techniques of 2D mapping have been widely used in the navigation task. Henriques et al. [11] and Savinov et al. [20] use 2D layout map to store useful information, which is learned by neural networks from the image
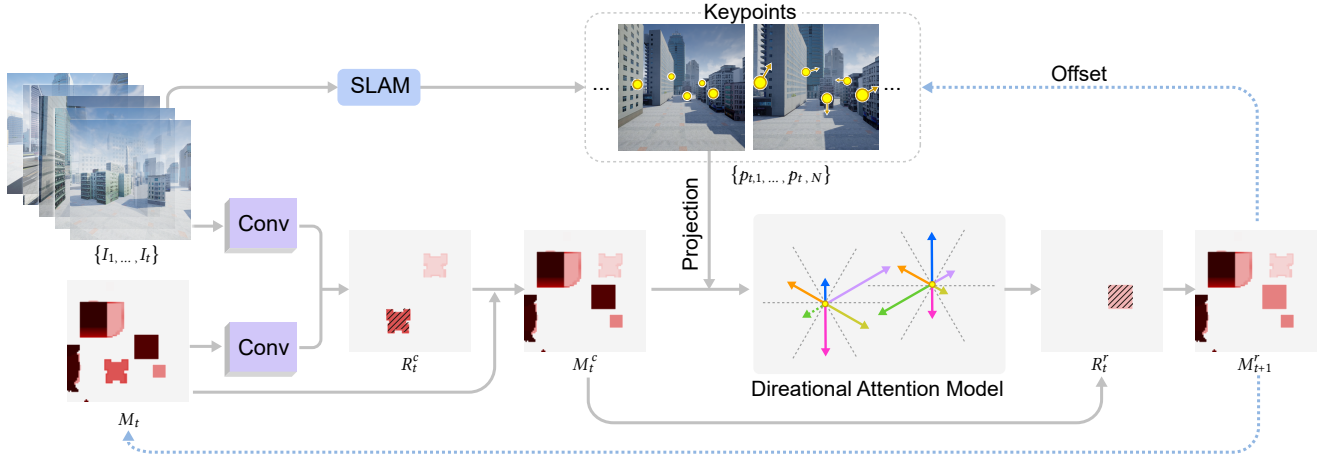
Figure 2: Overview of VGF-Net. At the tth moment, the network uses convolutional layers to learn visual and geometric representations from the RGB image I_t and 2.5D height map M_t (produced at the $(t-1)^{th}$ moment). The representations are combined to compute the residual update map R_t^c, which is added to the 2.5D height map to form a renewed height map M_t^c. Based on the new height map and the 3D keypoints $\{p_{t,1}, ..., p_{t,N}\}$ (produced by SLAM), we construct the VG representation for each keypoint (yellow dot), which is used by DAM to select useful information to refine object boundaries and 3D keypoints at the next moment. Note that the refined height map M_{t+1}^r is used for path planning, which is omitted for a simple illustration.

data of 3D scenes. Chen et al. [6] use the 2D topological map, which can be constructed using the coarse spatial layout of objects, to navigate the robot in an indoor scene. Different from the methods that consider the 2D map of an entire scene, Gupta et al. [10] unify the mapping and 2D path planning to rapidly adjust the navigation with respect to the surrounding local environment. Bansal et al. [2] utilize sparse waypoints to represent the map, which can be used to generate a smooth pathway to the target object or destination.

Compared to 2D mapping, 3D mapping provides much richer spatial information for the navigation system. Wang et al. [25] use visual odometry to capture the geometric relationship between 3D points, which is important to reconstruct the 3D scene. Engel et al. [8, 7] integrate the tracking of keypoints into the mapping process, harnessing temporal information to produce a more consistent mapping of the global environment. Besides 3D points, depth information is also important to 3D mapping. During the mapping process, Tateno et al. [23] and Ma et al. [16] use neural networks to estimate the depth map of a single image, for a faster construction of the 3D map. However, the fidelity of depth estimation is bounded by the scale of training data. To enhance, Kuznietsov et al. [14], Godard et al. [9] and Bian et al. [3] train the depth estimation network in semi-supervised/unsupervised manner, where the consistence in-between images are learned.

Nowadays, a vast of real-world 3D models and applications emerge, such as Google earth, and so there is abundant

data of height maps available for the training of drone navigation system. Nonetheless, the accuracy and timeliness of such data is impossible to be guaranteed, thus hard to be directly used in practice. We deeply exploit the visual-geometric information fusion representation to effectively and dynamically update the given height map during navigation, yielding a significant increase of the success rate of the autonomous drone navigation in various novel scenes.

3. Overview

The core idea behind our approach is to fuse the visual and geometric information for the construction of height map. This is done by our *Visual-Geometric Fusion Network* (VGF-Net) to compute the visual-geometric representation with respect to the visual and geometric consistence between the 3D keypoints and object boundaries characterized in the height map. VGF-Net uses the fused representation to refine the keypoints and height map at each moment during drone navigation. Below, we outline the architecture of VGF-Net.

As illustrated in Figure 2, at the tth moment ($t \geq 0$), the network takes the RGB image I_t and the associated height map M_t as input. The image I_t is fed to convolutional layers to compute the visual representation V_t. The height map M_t is also input to the convolutional layers for the geometric representation G_t. The visual and geometric representations are fused to compute the residual update map R_t^c that updates the height map to M_t^c, providing more consistent information for the subsequent steps.

Next, we use the SLAM [18] module to compute a sparse set of 3D keypoints $\{p_{t,1}, ..., p_{t,N}\}$, based on the images $\{I_1, ..., I_t\}$. We project these keypoints to the renewed height map M_t^c. For the keypoint $p_{t,i}$, we compute a set of distances $\{d_{t,i,1}, ..., d_{t,i,K}\}$, where $d_{t,i,k}$ denotes the distance from the keypoint $p_{t,i}$ to the nearest object boundary along the kth direction (see Figure 3(a)). Intuitively, the keypoint, which is extracted around the objects in the 3D scene, is also near to the boundaries of the corresponding objects in the height map. This relationship between the keypoint $p_{t,i}$ and the object can be represented by the visual and geometric information in the scene. Specifically, this is done by fusing the visual representation V_t, geometric representation G_t^c (learned from the renewed height map M_t^c) and the distances $\{d_{t,i,1}, ..., d_{t,i,K}\}$ to form a novel *Visual-Geometric* (VG) representation U_i for the keypoint $p_{t,i}$. For all keypoints, we compute a set of VG representations $\{U_{t,1}, ..., U_{t,N}\}$.

Finally, we employ a *Directional Attention Model* (DAM), which takes input as the VG representations $\{U_{t,1}, ..., U_{t,N}\}$, to learn a residual update map R_t^r to refine the height map M_t^c. The DAM produces a new height map M_{t+1}^r that respects the importance of each keypoint to the object boundaries in different directions (see Figure 3(b)). Meanwhile, we use DAM to compute a set of spatial offsets $\{\Delta p_{t+1,1}, ..., \Delta p_{t+1,N}\}$ to update the keypoints, whose locations are imperfectly estimated by the SLAM. We use the height map M_{t+1}^r for dynamic path planning [13] at the $(t+1)^{th}$ moment, and meanwhile input the image I_{t+1} and the height map M_{t+1}^r to VGF-Net at this moment for next update. As drone flies, the network achieves more accurate information and works more robustly for simultaneous drone navigation and height mapping.

4. Method

We now introduce our VGF-Net in more detail. The network extracts visual and geometric information from the RGB images, the associated 2.5D height map and 3D keypoints. In what follows, we formally define the information fusion that produces the visual-geometric representation, which is then used for the refinement of the height map and keypoints.

4.1. Residual Update Strategy

The VGF-Net refines the height map and keypoints iteratively, as the drone flies to new places and captures new images. We divide this refinement process into separate moments. At the tth moment, we feed the RGB image $I_t \in \mathbb{R}^{H \times W \times 3}$ and the height map $M_t \in \mathbb{R}^{H \times W}$ into the VGF-Net, computing the global visual representation $V_t \in \mathbb{R}^{H \times W \times C}$ and the geometric representation

$G_t \in \mathbb{R}^{H \times W \times C}$ as:

$$V_t = \mathcal{F}^v(I_t), \quad G_t = \mathcal{F}^g(M_t), \quad (1)$$

where $\mathcal{F}^v$ and $\mathcal{F}^g$ denote the two sets of convolutional layers. The H and W are the height and width of the image, respectively. Note that the value of each location on M_t represents the height of object, and we set the height of ground to be 0. We concatenate the representations V_t and G_t for computing a residual update map $R_t^c \in \mathbb{R}^{H \times W}$, which is used to update the height map M_t as:

$$M_t^c = M_t + R_t^c, \quad (2)$$

where

$$R_t^c = \mathcal{F}^c(V_t, G_t). \quad (3)$$

Here, $M_t^c \in \mathbb{R}^{H \times W}$ is a renewed height map, and $\mathcal{F}^c$ denotes a set of convolutional layers. Compared to directly computing a new height map, the residual update strategy (as formulated by Eq. (2)) adaptively reuses the information of M_t. More importantly, we learn the residual update map R_t^c from the new content captured at the tth moment. It facilitates a more focused update on the height values of regions that are unexplored before the tth moment. The height map M_t^c is fed to an extra set of convolutional layers to produce the representation G_t^c, which will be used for the construction of the visual-geometric representation.

4.2. Visual-Geometric Representation

We conduct the visual-geometric information fusion to further refine the height map. To capture the geometric relationship between objects, we use a standard SLAM [18] module to extract a sparse set of 3D keypoints $\{p_{t,1}, ..., p_{t,N}\}$ from the sequence of images $\{I_1, ..., I_t\}$. Given the keypoint $p_{t,i} \in \mathbb{R}^{1 \times 3}$ in the camera coordinate system, we project it to the 2.5D space as:

$$p'_{t,i} = p_{t,i}SR + T. \quad (4)$$

Here, $S \in \mathbb{R}^{3 \times 3}$ is decided by a pre-defined scale factor, which could be calculated at the initialization of the SLAM system or by GPS adjustment. $T \in \mathbb{R}^{1 \times 3}$ and $R \in \mathbb{R}^{3 \times 3}$ translate the origin of the 3D point set from the camera to the height map coordinate system. In the height map coordinate system, the drone is located at $(0, \frac{W}{2})$, where W represent the resolution of the height map.

Note that the first two dimensions of $p'_{t,i} \in \mathbb{R}^{1 \times 3}$ indicate the location on the height map, and the third dimension indicates the corresponding height value. The set of keypoints $\{p'_{t,1}, ..., p'_{t,N}\}$ are used for constructing the visual-geometric representations.

Next, for each keypoint $p'_{t,i}$, we compute its distances to the nearest objects in K different directions. Here, we refer
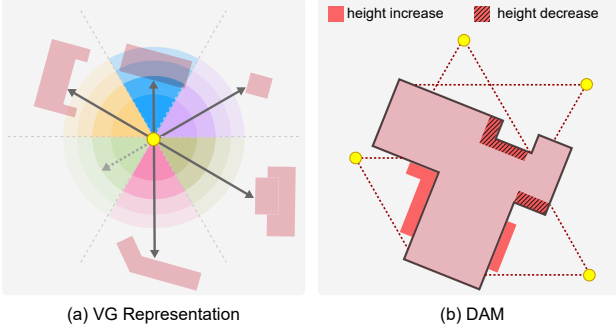
(a) VG Representation   (b) DAM

Figure 3: Illustration of fusing visual and geometric information for updating the 2.5D height map. (a) We construct the VG representation for each 3D keypoint (yellow dot) projected to the 2.5D height map. The information of VG representation is propagated to surrounding object boundaries, along different directions (indicated by different colors). The distance between the keypoint and object boundary (black arrow) determines the weight for adjusting the information propagation. The dash arrow means that there is no object along the corresponding direction. (b) Given the existing object boundary, we use DAM to select the most relevant keypoint along each direction. We use the selected keypoints to provide fused visual and geometric information, which is used for refining object boundary.

to objects as the regions that have larger height values than the ground (with height value of 0) in the height map M_t^c. As illustrated in Figure 3(a), we compute the Euclidean distance $d_{t,i,k}$ along the kth direction, from $p'_{t,i}$ to the first location, where the height value is larger than 0. We compute a set of distances $\{d_{t,i,1}, ..., d_{t,i,K}\}$ for K directions, then use V_t (see Eq. (1)), G_t^c and this distance set to form the VG representation $U_{t,i} \in \mathbb{R}^K$ as:

$$U_{t,i,k} = \mathcal{F}_k^v(W_{t,i,k}V_t) + \mathcal{F}_k^g(W_{t,i,k}G_{t,i}^c), \qquad (5)$$

where

$$W_{t,i,k}^v = \sum_{k'=1}^{K} \exp(-|d_{t,i,k} - d_{t,i,k'}|). \qquad (6)$$

Here, $G_{t,i}^c \in \mathbb{R}^C$ denotes the feature vectors located in $p'_{t,i}$ in the map G_t^c. In Eq. (5), $U_{t,i,k}$ is represented as a weighted map with the resolution equal to the geometric representation (20 × 20 by default), where $W_{t,i,k}$ plays as a weight of importance that is determined by the distance from the keypoint $p'_{t,i}$ to the nearest object boundary along the kth direction. As formulated in Eq. (5) and Eq. (6), longer distance decays the importance. Besides, we use independent set of fully connected layers (i.e., $\mathcal{F}_k^v$ and $\mathcal{F}_k^g$ in Eq. (5)) to learn important information from V_t and $G_{t,i}^c$. It allows the content, which is far from $p'_{t,i}$, to have the opportunity to make

an impact on $U_{t,i,k}$. We construct the VG representation for each keypoint in $\{p'_{t,1}, ..., p'_{t,N}\}$, while each VG representation captures the visual and geometric information around the corresponding keypoint. Based on the the VG representations, we propagate the information of the keypoints to each location on the height map, where the corresponding height value is refined. We also learn temporal information from the VG representations to refine the spatial locations of keypoints at the $(t + 1)^{th}$ moment, as detailed below.

4.3. Directional Attention Model

We use DAM to propagate the visual and geometric information, from each keypoint to each location on the height map, along different directions. More formally, for a location $p_j^h \in \mathbb{R}^{1 \times 3}$ on the height map M_t^c, we conduct the information propagation that yields a new representation $Q_{t,j} \in \mathbb{R}^{C \times K}$ as:

$$Q_{t,j} = \sum_{i=1}^{N} G_{t,j}^c U_{t,i}^\top. \qquad (7)$$

Along the second dimension of the representation $Q_{t,j}$, we perform max pooling to yield $Q'_{t,j} \in \mathbb{R}^C$ as:

$$Q'_{t,j,c} = \max(Q_{t,j,c,1}, ..., Q_{t,j,c,K}). \qquad (8)$$

As illustrated in Eq. (7), $Q_{t,j,c,k}$ summarizes the influence of all keypoints along k^{th} direction. We perform max pooling on the set $\{Q_{t,j,c,1}, ..., Q_{t,j,c,K}\}$ (see Eq. (8)), attending to the most information along a direction to form the representation $Q'_{t,j,c}$ (see Figure 3(b)). To further refine the height map, we use the representation $Q'_t \in \mathbb{R}^{H \times W \times C}$ to compute another residual update map $R_t^r \in \mathbb{R}^{H \times W}$, which is added to the height map M_t^c to form a new height map $M_{t+1}^r \in \mathbb{R}^{H \times W}$ as:

$$M_{t+1}^r = M_t^c + R_t^r, \qquad (9)$$

where

$$R_t^r = \mathcal{F}^r(V_t, Q'_t). \qquad (10)$$

Again, $\mathcal{F}^r$ denotes a set of convolutional layers. We make use of the new height map M_{t+1}^r for the path planning at the $(t + 1)^{th}$ moment.

We refine not only the 2.5D height map but also the 3D keypoints at the $(t + 1)^{th}$ moment. Assume that we use SLAM to produce a new set of keypoints $\{p_{t+1,1}, ..., p_{t+1,N}\}$. We remark that the keypoint sets at the tth and $(t + 1)^{th}$ moments are not necessary the same. To refine the new keypoint $p_{t+1,j} \in \mathbb{R}^{1 \times 3}$, we use DAM to compute the representation $\Delta p'_{t+1,j} \in \mathbb{R}^{3 \times K}$ as:

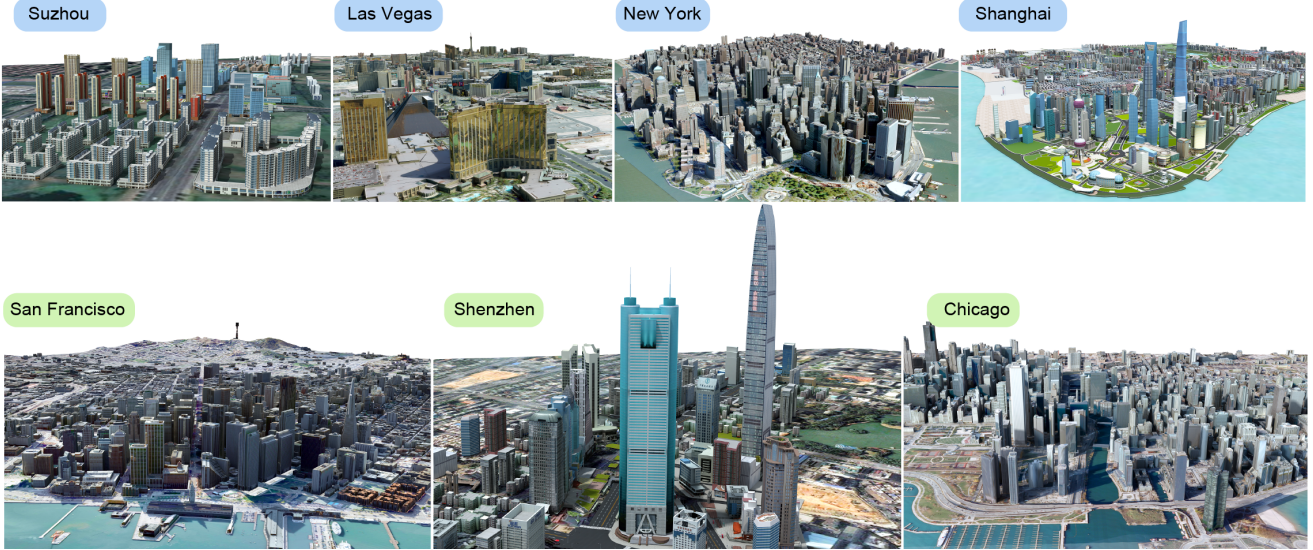$$\Delta p'_{t+1,j} = \sum_{i=1}^{N} p_{t,i} U_{t,i}^\top. \qquad (11)$$

Figure 4: Overview of our 3D urban navigation dataset, including 7 city scenes with different characteristics.

In this way, DAM distills the information of keypoints at the tth moment, which is propagated to the next moment. Again, we use max pooling to form the spatial offset $\Delta p_{t+1,j,c} \in \mathbb{R}^{1 \times 3}$ for updating keypoint $p_{t+1,j}$ as:

$$\Delta p_{t+1,j,c} = \max(\Delta p'_{t+1,j,c,1}, ..., \Delta p'_{t+1,j,c,K}). \quad (12)$$

We take the average of the updated keypoints $p_{t+1,j} + \Delta p_{t+1,j}$ and the estimated keypoints $p_{t+1,j}$ in place of the original one to construct the VG representation at the $(t + 1)^{th}$ moment.

4.4. Training Details

We use the L_1 loss function for training the VGF-Net as:

$$\mathcal{L}(M_t^{gt}, M_t^r) = \sum_{t=1}^{T} \sum_{j=1}^{H \times W} |M_{t,j}^{gt} - M_{t,j}^r|, \quad (13)$$

where $M_t^{gt} \mathbb{R}^{H \times W}$ is the ground-truth height map. Actually, we select 8 pairs of RGB image and height map ($T = 8$) to construct each mini-batch for the standard SGD solver. We set the height and width of each RGB image (224×224) and the height map (20×20). The overall training samples is nearly 24000 images randomly sampled in 3 scenes, while we test the model on the 24000 samples sampled on the other 3 scenes. Details about the dataset could be found in Sec. 5. We train the network for 30 epochs, and use the final snapshot of network parameters for testing. The learning rate is set to 0.001 at the first 15 epochs, and decayed to 0.0001 for a more stable optimization.

By default, the backbone of $\mathcal{F}^v$ and $\mathcal{F}^g$ is a ResNet-18, while the remained $\mathcal{F}^c$ and $\mathcal{F}^r$ is two stacked 3×3 convolutional layer with max-pooling and batch normalization.

Note that it is our contribution to learn spatial offsets of 3D keypoints, without explicitly using any ground-truth data. This is done by modeling the computation of spatial offsets as a differentiable function with respect to the VG representation. In this way, we enable the end-to-end learning of spatial offsets, where the related network parameters can be optimized by the back-propagated gradients. It significantly reduces the effort for data annotation, while allows the network training to be flexibly driven by data.

When constructing the VG representation, we set the number of directions $K = 16$ for each keypoint, and the number of keypoints $N = 50$ at each moment. We remark that these hyper-parameters are chosen based on the validation results.

5. Results and Discussion

5.1. Description of Experimental Dataset

To promote the related research on drone navigation, we newly collect a 3D urban navigation dataset. This dataset contains 7 models of different city scenes (see Figure 4).

Note that New York, Chicago, San Francisco, and Las Vegas are Google Earth models we download, which are similar to the real-world scenes with respect to the appearance but most objects inside are only buildings. We have also Shenzhen, Suzhou and Shanghai that are manually built based on the map by professional modelers, which contain rich 3D objects (e.g., buildings, trees, street lights and road signs, etc.) and other stuff (e.g., ground, sky and sea). There are various spatial configurations of objects, building styles and weather conditions in these 3D scenes. Thus, we provide challenging data for evaluating the navigation system.

Table 1: Statistics of our 3D urban navigation dataset. Note that in addition to buildings, there may also exist many other objects we must consider, such as trees, flower beds, and street lights, which highly increase the challenge for height mapping and autonomous navigation task.

scene	area (km^2)	objects (#)	model size (MB)	images (#)	texture size (MB)
New York	7.4	744	86.4	762	122
Chicago	24	1629	146	2277	227
San Francisco	55	2801	225	2865	322
Las Vegas	20	1408	108	1756	190
Shenzhen	3	1126	50.3	199	72.5
Suzhou	7	168	191	395	23.7
Shanghai	37	6850	308	2285	220

Table 2: Comparisons with different strategies of information fusion, in terms of the accuracy of height mapping (average L_1 error). We also show the accuracies (%) of predicting height values, with respect to different ranges of error ($< 3m$, $5m$ and $10m$). All strategies are evaluated on the testing (i.e., unknown and novel) scenes of San Francisco, Shenzhen and Chicago.

method	average L_1 error (m)			accuracy w.r.t. error $\in [0,3]m$ (%)		
	San Francisco	Shenzhen	Chicago	San Francisco	Shenzhen	Chicago
w/o fusion	4.57	4.57	4.49	68.95%	68.02%	70.05%
w/ fusion	2.37	2.93	3.41	85.09%	83.63%	78.44%
w/ fusion and memory	2.81	3.44	4.02	79.86%	79.20%	72.86%
w/ fusion, memory and exchange	2.35	3.04	3.80	80.54%	82.36%	74.73%
full strategy	**1.98**	**2.72**	**3.10**	**85.71%**	**86.13%**	**80.46%**
method	accuracy w.r.t. error $\in [0,5]m$ (%)			accuracy w.r.t. error $\in [0,10]m$ (%)		
	San Francisco	Shenzhen	Chicago	San Francisco	Shenzhen	Chicago
w/o fusion	75.02%	74.08%	76.86%	83.96%	83.96%	85.71%
w/ fusion	89.20%	87.39%	84.12%	93.87%	92.25%	91.18%
w/ fusion and memory	86.35%	84.56%	80.36%	93.00%	91.31%	89.51%
w/ fusion, memory and exchange	86.13%	86.43%	81.41%	93.33%	91.85%	89.94%
full strategy	**89.22%**	**88.90%**	**85.30%**	**94.10%**	**92.56%**	**91.67%**

The models are input to the render for producing sequences of RGB images. All RGB images and the associated 2.5D height maps are used to form a training set (i.e., New York, Las Vegas and Suzhou) and a testing set (i.e., San Francisco, Shenzhen, and Chicago). We provides more detailed statistics of the dataset in Table 1.

To train our VGF-Net, which takes as input a rough imperfect height map and outputs an accurate height map, we use 5 types of manipulations (i.e., translation, height increase/decrease, size dilation/contraction, creation and deletion) to disturb the object boundaries in the ground-truth height map. One time of the disturbance increases or
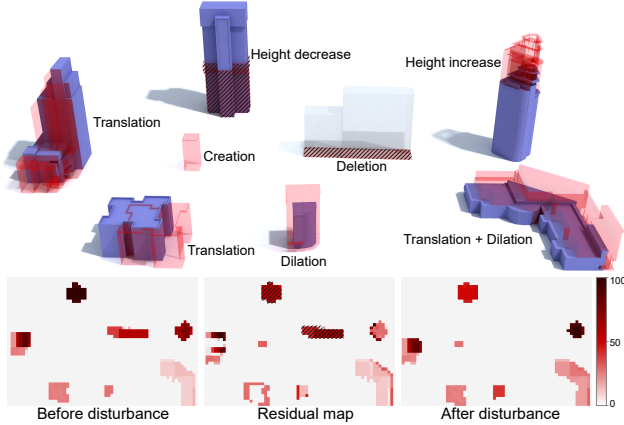
Figure 5: Illustration of disturbance manipulations. Actually, these manipulations can be combined to yield the disturbance results (e.g., translation and dilation). The bottom row of this figure shows the difference between height maps before/after disturbance. The residual map is learned by our VGF-Net, for recovering the disturbed height map to the undistorted counterpart.

decreases height values by $10m$ in certain map locations. See Figure 5 for an illustration of our manipulations.

5.2. Different Strategies of Information Fusion

The residual update, VG representation and DAM are critical components of VFG-Net, defining the strategy of information fusion. Below, we conduct an internal study by removing these components, and examine the effect on the accuracy of height mapping (see Table 2).

First, we report the performance using visual information only for height mapping, disabling any visual and geometric fusion. Here, the visual information is learned from RGB images (see the entries "w/o fusion" in Table 2). But visual information is insufficient for reconstructing height maps, which requires the modeling of geometric relationship between objects, yielding lower performances compared to other methods using geometric information.

Next, we examine the efficiency of residual update strategy. At each moment, the residual update allows VGF-Net to reuse the mapping result produced earlier. This strategy, where the useful visual and geometric contents can be effectively distilled and memorized at all moments, improves the reliability of height mapping. Thus, by removing the residual update (see the entries "w/ fusion" in Table 2) from VGF-Net (see the entries "full strategy"), we degrade the performance of height mapping.

We further study the effect of VG representation on the performance. The VG representation can be regarded as an information linkage. It contains fused visual and geometric information, which is exchanged among objects. Without the VG representation, we use independent sets of convolutional layers to extract the visual and geometric representations from the image and height map, respectively. The representations are simply concatenated for computing the residual update map (see the entries "w/ fusion and memory" in Table 2). This manner successfully disconnects the communication between objects and leads to performance drops on almost all scenes, compared to our full strategy of information fusion.

We find that the performance of using memory of height values lags behind the second method without using memory (see the entries "w/ fusion" in Table 2). We explain that the information fusion with memory easily accumulates errors in the height map over time. Thus, it is critical to compute the VG representation based on the memorized information, enabling the information exchange between objects (see the entries "w/ fusion, memory and exchange"). Such exchange process provides richer object relationship to effectively address the error accumulation problem, significantly assisting height mapping at each moment.

Finally, we investigate the importance of DAM (see the entries "w/ fusion, memory and exchange" in Table 2). We solely remove DAM from the full model, by directly using VG representations to compute the residual update map and spatial offsets for refining the height map and keypoints. Compared to this fusion strategy, our full strategy with DAM provides a more effective way to adjust the impact of each keypoint along different directions. Therefore, our method achieves the best results on all testing scenes.

5.3. Sensitivity to the Quality of Height Map

As demonstrated in the above experiment, it is important to the iterative information fusion for achieving a more global understanding of 3D scene to perfect the height map estimation. During the iterative procedure, the problematic height values may be memorized to make a negative impact on the production of height map at future moment. In this experiment, we investigate the sensitivity of different approaches to the quality of height maps, by controlling the percentage of height values that are dissimilar to the ground-truth height maps. Again, we produce dissimilar height maps by using disturbance manipulations to change the object boundaries.

At each moment, the disturbed height map is input to the trained model to compute the new height map, which is compared to the ground-truth height map for calculating the average L_1 error. In Figure 6, we compare the average L_1 errors produced by 4 different information fusion strategies (i.e., see the entries "w/ fusion", "w/ fusion and memory", "w/ fusion, memory and exchange" and "full strategies" in Table 2), which learn geometric information from height maps. As we can see, heavier disturbances generally lead to the degradation of all strategies.
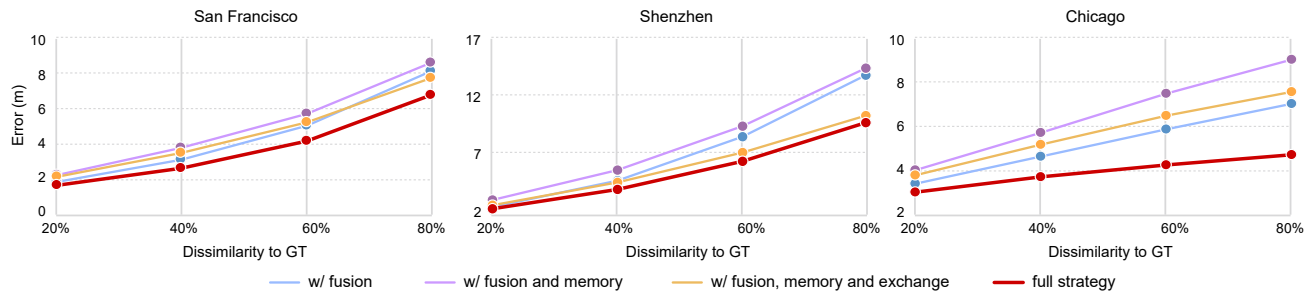
Figure 6: We disturb the 2.5D height maps, which are used to examine the robustness of different information fusion approaches. We evaluate different approaches on the testing sets of San Francisco, Shenzhen and Chicago. All results are reported in terms of L_1 errors.
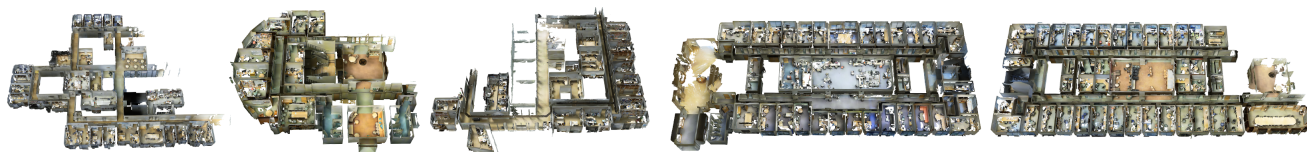

Figure 7: The five indoor training scenes selected from the S3DIS dataset [1].
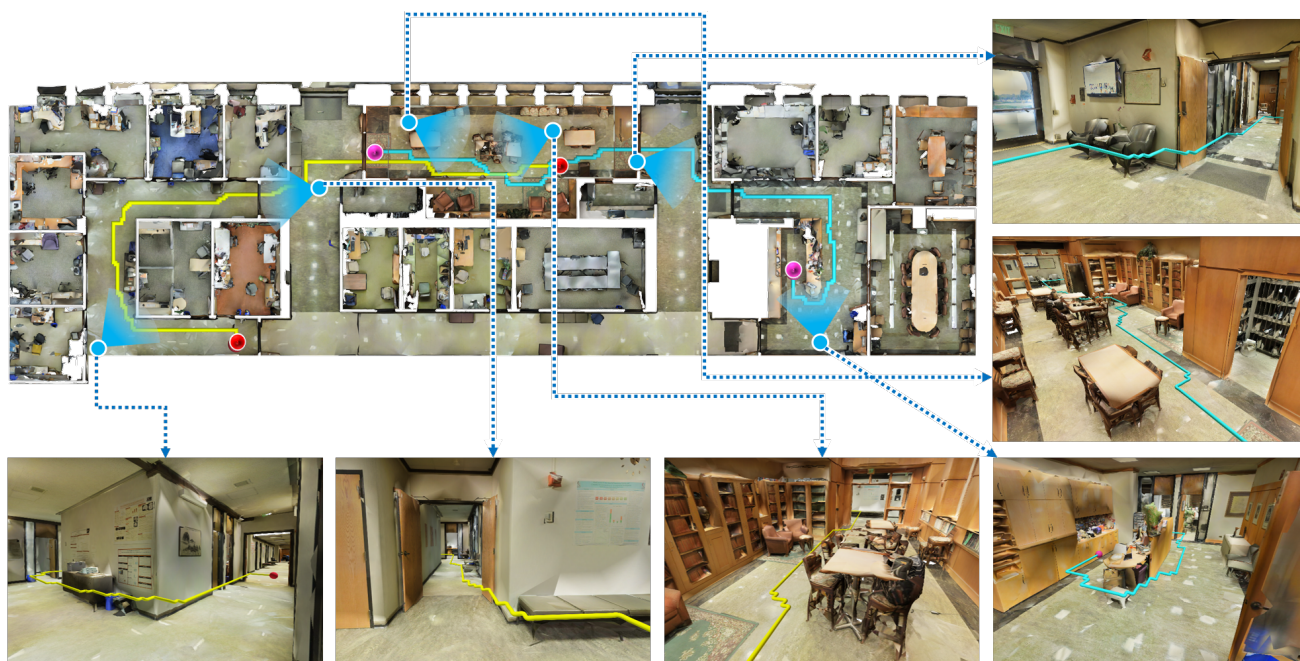

Figure 8: The successful navigation trajectories produced by VGF-Net in a complicate indoor testing scene from the S3DIS dataset [1].

The strategy "w/ fusion and memory" performs the worst among all approaches, showing very high sensitivity to the quality of height maps. This result further evidences our finding in Sec. 5.2, where we have shown the unreliability of the method with memory of height information but without information exchange. Compared to other methods, our full strategy yields better results. Especially, given a very high percentage (80%) of incorrect height values, our full strategy outperforms other methods by remarkable margins. These results clearly demonstrate the robustness of our strategy.

Table 3: We compare VGF-Net with/without using depth to other methods. All methods are evaluated on the outdoor sets (i.e., San Francisco, Shenzhen and Chicago) and the indoor set (i.e., S3DIS). Results are reported in terms of the success rates of navigation.

outdoor test	w/ depth		w/o depth
	ground-truth depth	estimated depth [3]	VGF-Net
San Francisco	100%	27%	**85%**
Shenzhen	100%	34%	**83%**
Chicago	100%	19%	**82%**

indoor test	w/ depth			w/o depth		
	LSTM [10]	CMP [10]	VGF-Net	LSTM [10]	CMP [10]	VGF-Net
S3DIS	71.8%	78.3%	**92%**	53%	62.5%	**76%**

5.4. Comparison on the Navigation Task

The quality of 2.5D height maps, which are estimated by the height mapping, largely determines the accuracy of drone navigation. In this experiment, we compare our VGF-Net to different mapping approaches. All methods are divided into two groups. In the first group, the approaches apply depth information for height mapping. Note that the depth information can be achieved by scanner [10], or estimated by deep network based on the RGB images [3]. The second group consists of approaches that only use RGB images to reconstruct the height map. In addition to an initial height map that can be easily obtained from various resources, our VGF-Net only requires image inputs, but can also accept depth information if available without changing any scheme architecture. We set the height of flight to be $10 \sim 30m$ for drone, evaluating the success rate of 3D navigation on our outdoor dataset. Overheight (e.g., $100m$) always leads to successful navigation, making the evaluation meaningless. On the indoor dataset [1] (see also Figure 7 and Figure 8), we report the success rate of 2D drone navigation, by fixing the height of flight to $0.5m$. All results can be found in Table 3.

Obviously, using accurate depth information can yield a perfect success rate of navigation (see the entry "ground-truth depth"). Here, the depth data is directly computed from the synthesized 3D urban scenes, without involving any noise. However, due to the limitation of hardware device, it is difficult for the scanner to really capture the accurate depth data of outdoor scenes. A simple alternative is to use deep network to estimate the depth based on the RGB image (see the entry "estimated depth"). Depth estimation often produces erroneous depth values for the height mapping, even with the most advanced method [3], thus severely
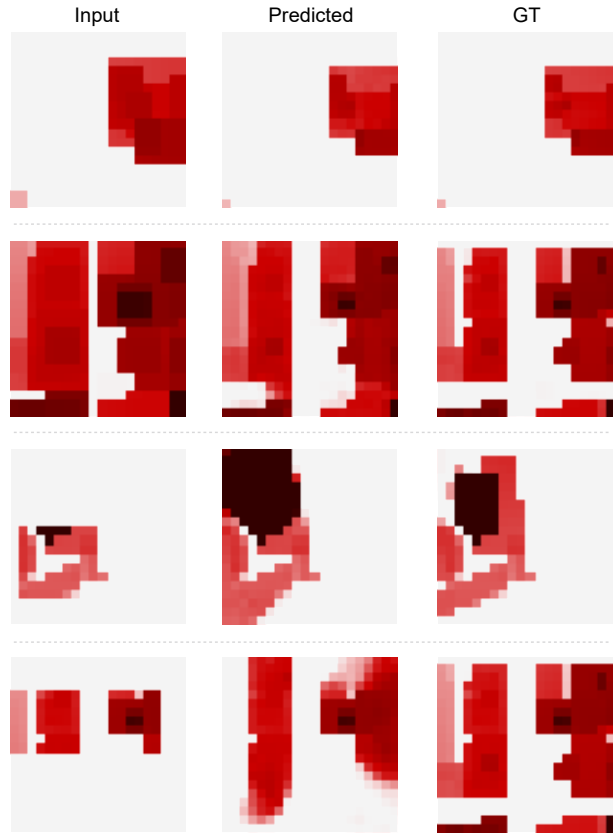

Figure 9: Examples of height mapping. All the height maps are selected from the outdoor dataset. Here, we compare the height maps with noise (in the first column), predicted height maps (in the second column) and ground-truth height maps (in the last column).

misleading the navigation process. Similar to depth information, the sparse 3D keypoints used in our approach also provide valuable geometry information of objects. More importantly, our VGF-Net uses visual cues to assist the learning of geometric representations. Therefore, our approach without using depth produces better results than that of using depth estimated by state-of-the-art techniques. We have shown an example of trajectory for 3D drone navigation in Figure 1. We also show examples of height mapping in Figure 9, where the height map with redundant boundary (see the first two rows of Figure 9) or missing boundary (see the last two rows of Figure 9) is input to the VGF-Net. Even given the input height maps with much noise, our network still precisely recovers the height information.

Depth data of indoor scenes (see Figure 7) can be more easily achieved. With the available depth information, we can trivially input the RGB image along with the associated depth to the VGF-Net, producing the height map. We compare VGF-Net to the recent approach [10] (see the entries "LSTM" and "CMP") that produces state-of-the-art indoor navigation accuracies. Our method achieves a better result under the same condition of training and testing. Without depth, our approach still leads to the best result among all image based methods. It demonstrates the generality and ability of our approach, in terms of stably learning useful information from different data sources. In Figure 8, we show more navigation trajectories planned by our approach in an indoor testing scene.

6. Conclusions and Future Work

The latest progress on drone navigation is largely driven by the active sensing and selecting the useful visual and geometric information of surrounding 3D scenes. In this paper, we have presented VGF-Net, where we fuse visual and geometric information for simultaneous drone navigation and height mapping. Our network distills the fused information, which is learned from the RGB image sequences and an initial rough height map, constructing a novel VG representation to better capture object/scene relation information. Based on the VG representation, we propose DAM to establish information exchange among objects and select essential object relationship in a data-driven fashion. By using residual update strategy, DAM progressively refines the object boundaries in the 2.5D height map and the extracted 3D keypoints, showing its generality to various complicate outdoor/indoor scenes. The mapping module runs at nearly 0.2sec on a mobile GPU, which could be further optimized by compression and pruning in an embedded system.

VGF-Net eventually outputs the residual update map and spatial offsets, which are used for explicitly updating the geometric information of objects (i.e., the 2.5D height map and 3D keypoints). It should be noted that we currently use convolutional layers to learn implicit representation from the fused information, and update the visual representation. The visual content of the sequence of RGB image shows complex patterns, which together form the global object/scene relationship. However, these patterns may be neglected by the implicit representation during the learning process. Thus, in the near future, we would like to investigate a more controllable way to update the visual representation. Additionally, complex occlusion relations in the real scenarios often lead to inaccurate height mappings in the occluded areas. In the future, we would like to further utilize the uncertainty map of the environment, together with the multi-view information to improve both the accuracy and the efficiency of the mapping process.

References

[1] I. Armeni, O. Sener, A. R. Zamir, H. Jiang, I. Brilakis, M. Fischer, and S. Savarese. 3D semantic parsing of large-scale indoor spaces. In *Proc. IEEE Conf. on Computer Vision & Pattern Recognition*, pages 1534–1543, 2016. 9, 10

[2] S. Bansal, V. Tolani, S. Gupta, J. Malik, and C. Tomlin. Combining optimal control and learning for visual navigation in novel environments. In *Proc. Conf. on Robot Learning*, 2019. 2, 3

[3] J. Bian, Z. Li, N. Wang, H. Zhan, C. Shen, M.-M. Cheng, and I. Reid. Unsupervised scale-consistent depth and ego-motion learning from monocular video. In *Proc. of Advances in Neural Information Processing Systems*, pages 35–45, 2019. 2, 3, 10

[4] G. Chaurasia, S. Duchene, O. Sorkine-Hornung, and G. Drettakis. Depth synthesis and local warps for plausible image-based navigation. *ACM Trans. on Graphics*, 32(3):30:1–30:12, 2013. 2

[5] D. Chen, B. Zhou, V. Koltun, and P. Krähenbühl. Learning by cheating. In *Proc. Conf. on Robot Learning*, 2019. 2

[6] K. Chen, J. P. de Vicente, G. Sepulveda, F. Xia, A. Soto, M. Vázquez, and S. Savarese. A behavioral approach to visual navigation with graph localization networks. In *Proc. of Robotics: Science and Systems*, 2019. 2, 3

[7] J. Engel, V. Koltun, and D. Cremers. Direct sparse odometry. *IEEE Trans. Pattern Analysis & Machine Intelligence*, 40(3):611–625, 2017. 3

[8] J. Engel, T. Schöps, and D. Cremers. Lsd-slam: Large-scale direct monocular slam. In *Proc. Euro. Conf. on Computer Vision*, pages 834–849, 2014. 3

[9] C. Godard, O. Mac Aodha, and G. J. Brostow. Unsupervised monocular depth estimation with left-right consistency. In *Proc. IEEE Conf. on Computer Vision & Pattern Recognition*, pages 270–279, 2017. 3

[10] S. Gupta, J. Davidson, S. Levine, R. Sukthankar, and J. Malik. Cognitive mapping and planning for visual navigation. In *Proc. IEEE Conf. on Computer Vision & Pattern Recognition*, pages 2616–2625, 2017. 2, 3, 10, 11

[11] J. F. Henriques and A. Vedaldi. Mapnet: An allocentric spatial memory for mapping environments. In *Proc. IEEE Conf. on Computer Vision & Pattern Recognition*, pages 8476–8484, 2018. 2

[12] D. K. Kim and T. Chen. Deep neural network for real-time autonomous indoor navigation. *arXiv preprint:1511.04668*, 2015. 2

[13] S. Koenig and M. Likhachev. D* lite. In *Proc. of Association for the Advancement of Artificial Intelligence*, pages 476–483, 2002. 2, 4

[14] Y. Kuznietsov, J. Stuckler, and B. Leibe. Semi-supervised deep learning for monocular depth map prediction. In *Proc. IEEE Conf. on Computer Vision & Pattern Recognition*, pages 6647–6655, 2017. 3

[15] A. Loquercio, A. I. Maqueda, C. R. Del-Blanco, and D. Scaramuzza. Dronet: Learning to fly by driving. *IEEE Robotics and Automation Letters*, 3(2):1088–1095, 2018. 2

[16] F. Ma and S. Karaman. Sparse-to-dense: Depth prediction from sparse depth samples and a single image. In *Proc. IEEE Int. Conf. on Robotics & Automation*, pages 1–8, 2018. 3

[17] P. Mirowski, M. Grimes, M. Malinowski, K. M. Hermann, K. Anderson, D. Teplyashin, K. Simonyan, A. Zisserman, R. Hadsell, et al. Learning to navigate in cities without a map. In *Proc. of Advances in Neural Information Processing Systems*, pages 2419–2430, 2018. 2

[18] R. Mur-Artal and J. D. Tardós. Orb-slam2: An open-source slam system for monocular, stereo, and rgb-d cameras. *IEEE Trans. on Robotics*, 33(5):1255–1262, 2017. 2, 4

[19] R. P. Padhy, S. Verma, S. Ahmad, S. K. Choudhury, and P. K. Sa. Deep neural network for autonomous UAV navigation in indoor corridor environments. *Procedia Computer Science*, 133:643–650, 2018. 2

[20] N. Savinov, A. Dosovitskiy, and V. Koltun. Semi-parametric topological memory for navigation. In *Proc. Int. Conf. on Learning Representations*, 2018. 2

[21] L. Tai, G. Paolo, and M. Liu. Virtual-to-real deep reinforcement learning: Continuous control of mobile robots for mapless navigation. In *Proc. IEEE Int. Conf. on Intelligent Robots & Systems*, pages 31–36, 2017. 2

[22] M. Tatarchenko, S. R. Richter, R. Ranftl, Z. Li, V. Koltun, and T. Brox. What do single-view 3D reconstruction networks learn? In *Proc. IEEE Conf. on Computer Vision & Pattern Recognition*, pages 3405–3414, 2019. 2

[23] K. Tateno, F. Tombari, I. Laina, and N. Navab. Cnn-slam: Real-time dense monocular slam with learned depth prediction. In *Proc. IEEE Conf. on Computer Vision & Pattern Recognition*, pages 6243–6252, 2017. 3

[24] C. Wang, J. Wang, Y. Shen, and X. Zhang. Autonomous navigation of uavs in large-scale complex environments: A deep reinforcement learning approach. *IEEE Trans. on Vehicular Technology*, 68(3):2124–2136, 2019. 2

[25] R. Wang, M. Schworer, and D. Cremers. Stereo dso: Large-scale direct sparse visual odometry with stereo cameras. In *Proc. Int. Conf. on Computer Vision*, pages 3903–3911, 2017. 3

[26] Y. Zhu, R. Mottaghi, E. Kolve, J. J. Lim, A. Gupta, L. Fei-Fei, and A. Farhadi. Target-driven visual navigation in indoor scenes using deep reinforcement learning. In *Proc. IEEE Int. Conf. on Robotics & Automation*, pages 3357–3364, 2017. 2