Supplementary Material: Continuous Aerial Path Planning for 3D Urban Scene Reconstruction

ANONYMOUS AUTHOR(S)
SUBMISSION ID: 144

ACM Reference Format:
Anonymous Author(s). 2021. Supplementary Material: Continuous Aerial Path Planning for 3D Urban Scene Reconstruction. 1, 1 (May 2021), 4 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

A. CONVERGENCE ANALYSIS

This section presents an empirical analysis on how the following measures vary as the algorithm progresses: (i) objective value Eq.(6); (ii) scene coverage, i.e., percentage of sampled surface points whose reconstructability (Eq.(1)) exceeds threshold h; (iii) trajectory length (Eq.(4)); and (iv) total cost for turning angles (Eq.(5)).

Fig. 1 shows the plots when running our algorithm on the dataset UK-1. From the plots, we can see that all quantities increase initially, when the algorithm starts to build up the random tree with more nodes. Later on, when the scene coverage improves to a level close to and beyond percentage G, the optimization model starts to minimize the trajectory length and total cost for turning angles, as shown by the drops near the end of the bottom two plots. Yet, for this dataset, our algorithm is able to converge in just 150 iterations.

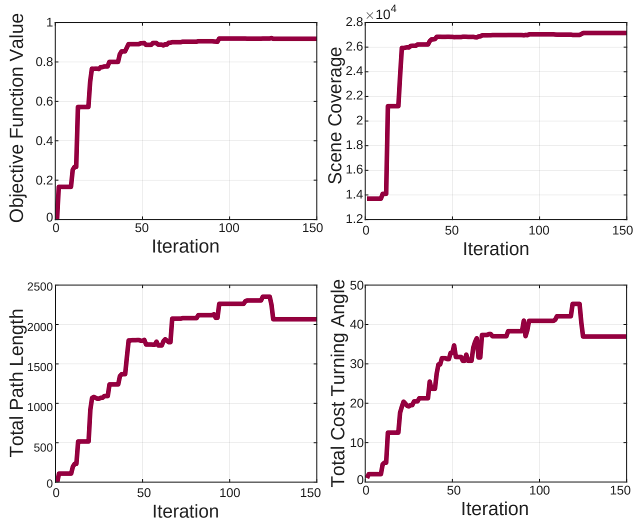


Fig. 1. Plots that show how the various measures (see above) vary as the algorithm progresses on the UK-1 dataset.

B. TRADEOFF: RECONSTRUCTION QUALITY VS. PATH LENGTH

This section presents an experiment that explores the tradeoff between the 3D scene reconstruction quality and the length of the aerial path. To explore their relationship, we make use of threshold h (see Algorithm 2), which is the minimum reconstructability value required for capturing each surface point on the scene proxy. Generally, a larger h requires more view coverage on surface points, thus demanding for a longer aerial path and longer flight time for capturing more details. Hence, we can adjust h to tradeoff between the scene reconstruction quality and path length in this experiment.

In more detail, we perform this experiment on the benchmark dataset NY-1 using a five-camera drone and plan aerial paths with different values of h. Table 1 reports the results, showing that longer aerial paths do not necessarily help to increase the accuracy of the 3D reconstruction, as revealed by the "error" measure. Yet, a larger h certainly increases the scene coverage and leads to a more complete 3D reconstruction, as revealed by the the "completeness" measure. Importantly, it is worth to note that longer aerial paths may not necessarily capture more useful information, since the images used for the 3D reconstruction are post-selected.

Table 1. Exploring the tradeoff between 3D reconstruction quality and path length. The results reveal that longer aerial paths generally improves the completeness but it may not improve the reconstruction accuracy.

Threshold	Num. Images	Length [m]	Error [m] 90%	Error [m] 95%	Completeness [%] 0.020m	Completeness [%] 0.050m	Completeness [%] 0.075m
0.05	176	814	0.029	0.162	36.37	42.48	46.67
0.10	284	1,779	0.033	0.351	36.79	43.89	47.48
0.15	428	1,870	0.021	0.087	40.92	47.61	51.97
0.20	768	2,378	0.028	0.062	41.64	46.76	51.84
0.25	1,057	3,368	0.020	0.079	41.28	47.33	52.37

C. OBLIQUE PHOTOGRAPHY PATHS ON BENCHMARKS

To supplement Fig. 11 in the main paper, we show the paths of oblique photography for the three synthetic benchmark datasets, NY-1, UK-1, and Bridge-1, in Fig. 2 of this supplementary material.

D. LARGE VIEWS ON BENCHMARK RESULTS

Due to space limit in the main paper, we present Figs. 3, 4, and 5 in this supplementary material to show large views of the 3D reconstruction results on the three benchmark datasets.

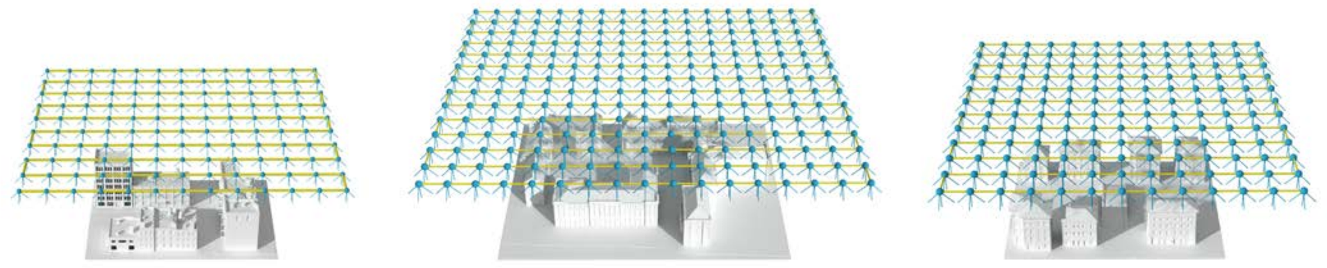


Fig. 2. The aerial paths of oblique photography for benchmarks NY-1, UK-1 and Bridge-1 from left to right.

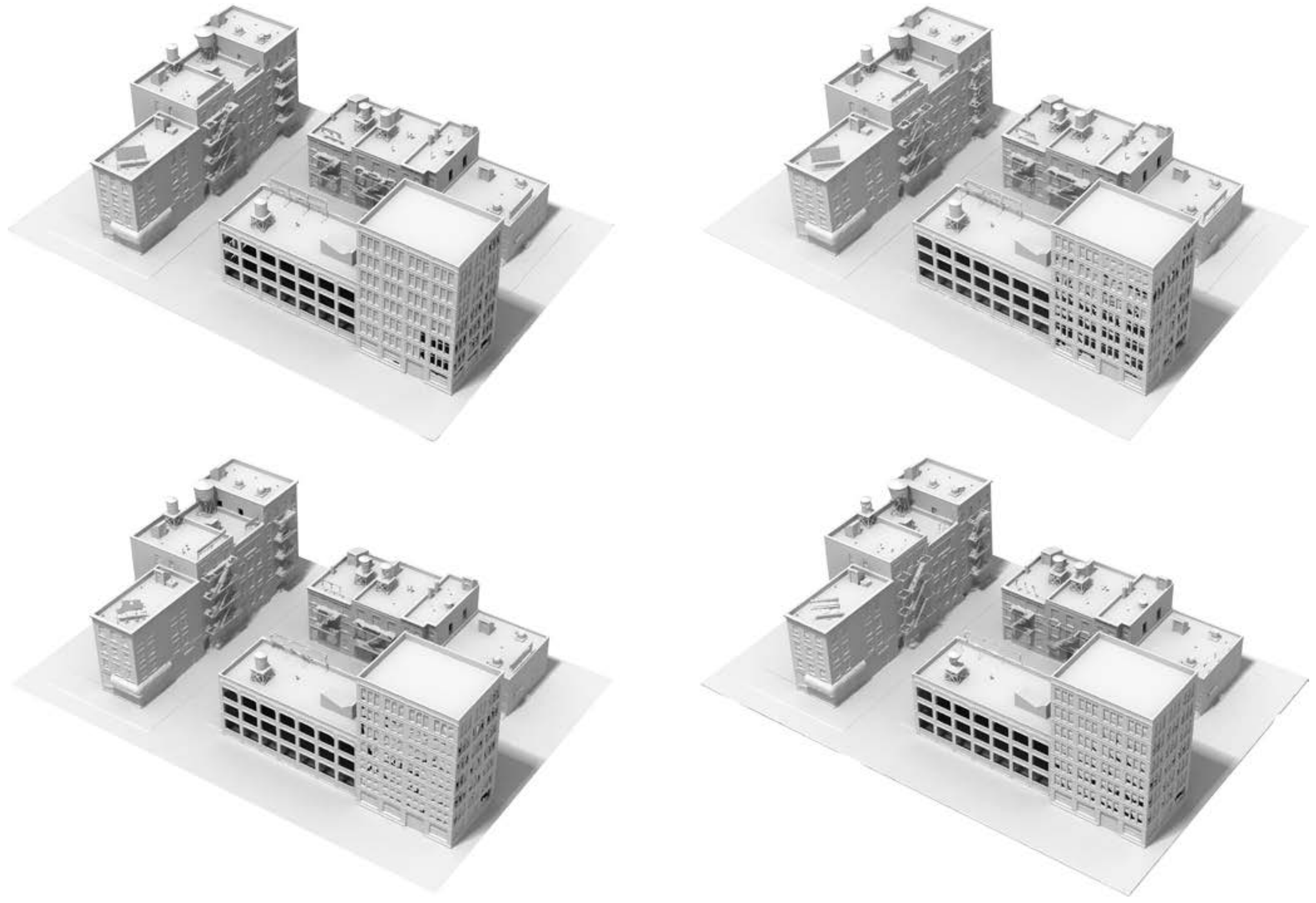


Fig. 3. Large views of the 3D reconstructions on NY-1 by our method (top-left), [Zhou et al. 2020] (top-right), [Smith et al. 2018] (bottom-left), and oblique photography (bottom-right).

REFERENCES

Neil Smith, Nils Moehrle, Michael Goesele, and Wolfgang Heidrich. 2018. Aerial Path Planning for Urban Scene Reconstruction: A Continuous Optimization Method and Benchmark. *ACM Trans. on Graphics (Proc. of SIGGRAPH Asia)* 37, 6 (2018), 183:1–183:15.

Xiaohui Zhou, Ke Xie, Kai Huang, Yilin Liu, Yang Zhou, Minglun Gong, and Hui Huang. 2020. Offsite Aerial Path Planning for Efficient Urban Scene Reconstruction. *ACM Trans. on Graphics (Proc. of SIGGRAPH Asia)* 39, 6 (2020), 192:1–192:16.

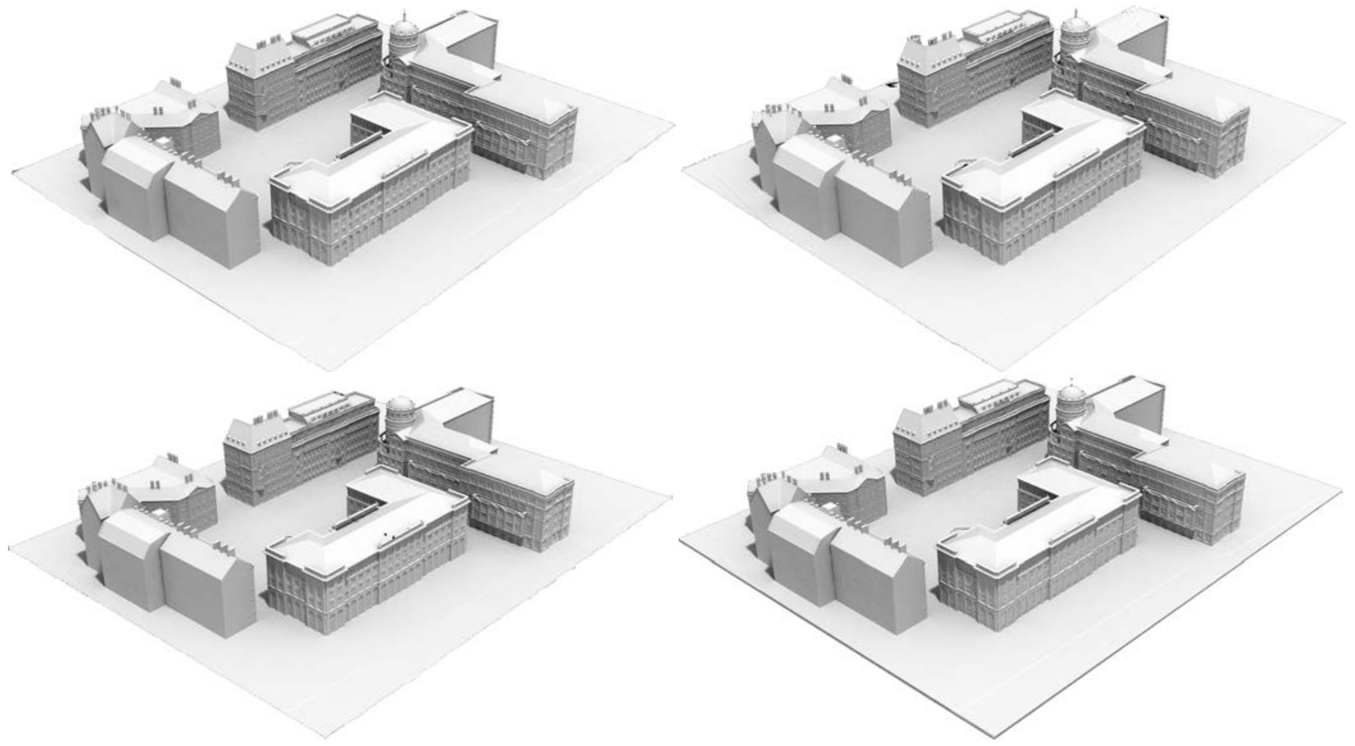


Fig. 4. Large views of the 3D reconstructions on UK-1 by our method (top-left), [Zhou et al. 2020] (top-right), [Smith et al. 2018] (bottom-left), and oblique photography (bottom-right).

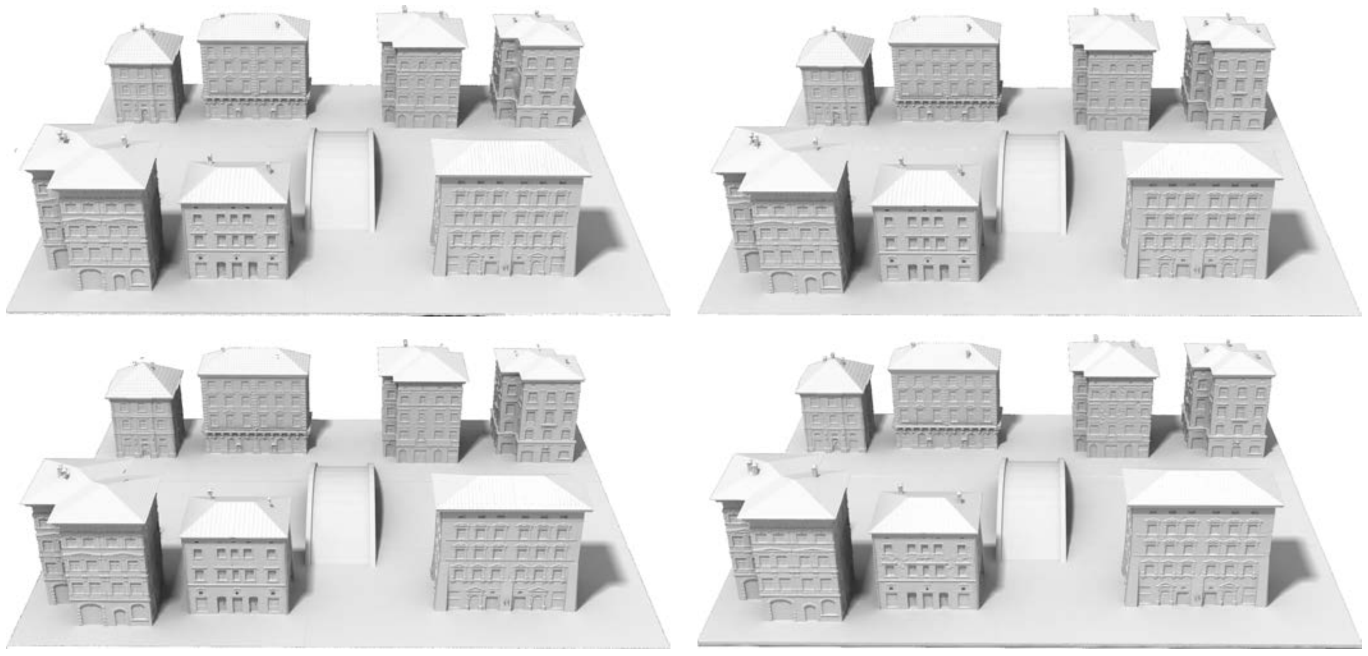


Fig. 5. Large views of the 3D reconstructions on Bridge-1 by our method (top-left), [Zhou et al. 2020] (top-right), [Smith et al. 2018] (bottom-left), and oblique photography (bottom-right).

Continuous Aerial Path Planning for 3D Urban Scene Reconstruction

ANONYMOUS AUTHOR(S)
SUBMISSION ID: 144

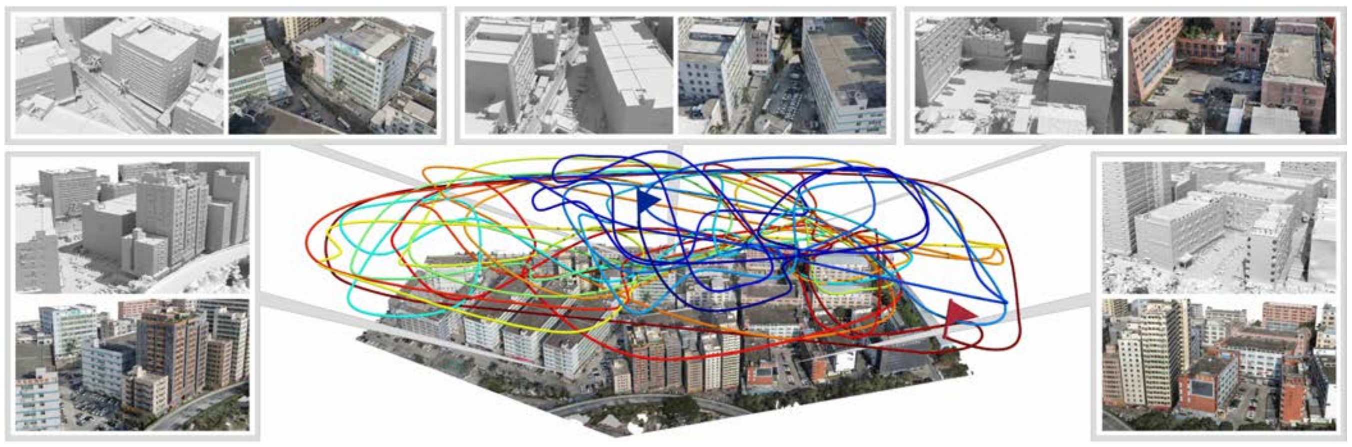


Fig. 1. A 0.1km^2 urban area (roughly 0.25km × 0.53km) is 3D-reconstructed from 1,423 images captured by a *single-camera* drone (DJI Phantom 4 RTK) flying along a *continuously-planned* path by our method: 18,491m in length and color-coded from blue to red with the two small flags indicating the start and end points. Our path planning algorithm optimizes *scene coverage, scene capture efficiency*, and *path quality*, resulting in fewer sharp turns, shorter path lengths, as well as higher 3D reconstruction quality (see the geometric and appearance details in the zoomed-in inset pairs above) at a reduced flight time.

We introduce a *path-oriented* drone-trajectory planning algorithm, which performs *continuous* image acquisition along an aerial path, aiming to optimize both the scene reconstruction quality and *path quality*. Specifically, our method takes as input a rough 3D scene proxy and produces a drone trajectory and image capturing setup, which efficiently yields a high-quality reconstruction of the 3D scene based on three optimization objectives: one to maximize the *amount of 3D scene information* that can be acquired along the entirety of the trajectory, another one to optimize the scene capturing *efficiency* by maximizing the scene information that can be acquired per unit length along the aerial path, and the last one to minimize the total *turning angles* along the aerial path, so as to reduce the number of sharp turns. Our search scheme for the optimal trajectory is formulated based on the rapidly-exploring random trees; unlike existing methods for aerial path planning, our scheme jointly samples viewpoints and plans the aerial path continuously in space to optimize the objective. We comprehensively evaluate our method not only on benchmark virtual datasets as in existing works but also on several large-scale real urban scenes. We demonstrate that the continuous paths optimized by our method can effectively reduce the onsite acquisition cost using drones, while achieving high-fidelity 3D reconstruction, compared to state-of-the-art aerial planning methods and oblique photography, a mature and popular industry solution.

CCS Concepts: • **Computing methodologies** → **Computer graphics**; **Shape modeling**; **Mesh geometry models**.

Additional Key Words and Phrases: Aerial Path Planning, Information Gain, Rapidly-Exploring Random Trees, Urban Scene Reconstruction

ACM Reference Format:
Anonymous Author(s). 2021. Continuous Aerial Path Planning for 3D Urban Scene Reconstruction. 1, 1 (May 2021), 15 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

Large-scale 3D scene acquisition at the urban city level using unmanned aerial vehicles (UAVs) or drones is becoming a well-practiced technology serving a wide variety of applications. With an increasing demand for high-fidelity captures of urban scenes with complex structures, modern techniques typically take a coarse-to-fine approach. First, a rough and conservative geometry prior, referred to as a *scene proxy*, is produced from an initial flight pass or satellite images, while the main effort for acquisition planning is devoted to the next phase for a *detailed* scene reconstruction [Hepp et al. 2018b; Roberts et al. 2017; Smith et al. 2018; Zhou et al. 2020]. Ideally, a dense view sampling is needed to reconstruct a large 3D scene with fine details. In practice, however, a drone's flight time is generally limited to 25-30 minutes by battery power, hence the main objective of drone trajectory planning is to best capture the whole 3D scene, maximize reconstruction quality while minimizing flight time.

Existing planning methods, e.g., [Hepp et al. 2018b; Roberts et al. 2017; Smith et al. 2018; Zhou et al. 2020], have focused on finding a set of "good" locations for image capture, which would serve as candidate viewpoints to compute the drone flight path in a subsequent step. The path is typically obtained using heuristic solutions, such as those developed for the traveling salesman problem (TSP), to pass through all selected viewpoints. By decoupling path construction from view optimization, which focuses on the reconstruction quality attributed to individual viewpoints, these approaches do not take advantage of the *continuity* of the path or account for *path quality*, which can significantly impact the acquisition effort needed in the field. In particular, paths obtained this way often contain many *sharp turns*, and as the drone flies across such turns, the necessary deceleration leads to a slow-down, while the ensuing acceleration consumes extra battery power. In reality, a drone is well equipped to capture images in quick successions without compromising its flying speed or image quality. Images captured this way represent a dense view sampling, which is a beneficial characteristic of the acquisition setup that has not been utilized in previous works.

In this paper, we introduce a *path-oriented* drone-trajectory planning algorithm, which performs *continuous* image acquisition (subject to limits of the image capture rate of the drone camera) along an aerial path and aims to optimize both the scene reconstruction quality *and* path quality (see, e.g., Fig. 1). Specifically, our path planning algorithm takes as input a rough 3D scene proxy, performs path planning directly in the 3D space, and produces a drone trajectory and image capturing setup, which efficiently yields a high-quality reconstruction of the 3D scene. Our optimization objective is defined by three criteria: the first one to maximize the amount of scene information that can be acquired along the *entirety* of the trajectory, the second one to optimize the scene capturing efficiency by maximizing the scene information that can be acquired *per unit length* along the flight path, and the third one to minimize the total *turning angles* along the flight path, so as to avoid sharp turns. We formulate our search scheme for the optimal trajectory based on the *rapidly-exploring random trees (RRT)* framework [Karaman and Frazzoli 2011; Naderi et al. 2015]. Different from previous works, our scheme jointly considers viewpoint planning and path quality in a single stage, achieving a large reduction in path length and flight time, particularly for scenes that are large in scale.

As illustrated in Fig. 2, our continuous path planning approach, coupled with a dense view sampling, results in a more extensive exploration of the drone camera's flying space, compared to discrete view planning [Hepp et al. 2018b; Roberts et al. 2017; Smith et al. 2018; Zhou et al. 2020], to obtain a high-quality 3D scene reconstruction. During the reconstruction step, it is, however, not necessary to input all the captured images as they may contain redundant scene information, especially for five-camera drones. So, we select a subset of images for scene reconstruction. In the end, the reconstruction effort, as reflected by the number of input images used, is comparable to existing methods, but the path quality is improved, leading to higher acquisition efficiency; check more details in Section 6.

We perform a comprehensive evaluation of our path planning and scene reconstruction framework using both benchmark virtual datasets [Smith et al. 2018; Zhou et al. 2020] and real large-scale urban scenes. We demonstrate that the continuous paths optimized by our method can effectively reduce the onsite acquisition cost using drones, while achieving high-fidelity 3D reconstruction (Fig. 1), compared to state-of-the-art aerial planning methods and *oblique photography*, a mature and popular industry solution. In particular, our approach excels at recovering *finer-level details* of the capture scenes, owing to the dense view sampling and image-to-image correspondences as a result of the continuous scene acquisition.

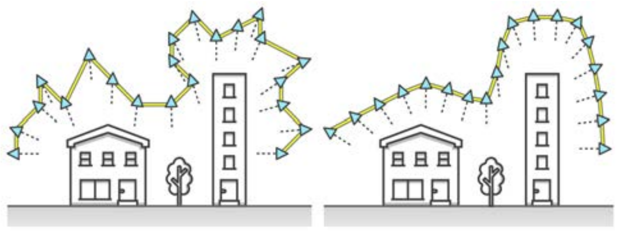


Fig. 2. An illustration contrasting a shortest path constructed from independently selected good viewpoints (left) vs. an aerial path computed using our continuous planning approach (right). Our path on the right is optimized to have fewer sharp turns while ensuring a good view coverage of the scene.

2 RELATED WORK

In this section, we cover prior works on image-based scene acquisition and reconstruction, especially those related to view selection and path planning for urban scene capture. We also describe the RRT framework from robotic path planning.

Image-based scene reconstruction and view selection. Image-based scene reconstruction [Knapitsch et al. 2017] is a fundamental problem in computer graphics, where much success has been achieved through structure from motion [Snavely et al. 2006, 2008] and multi-view stereo [Furukawa and Hernández 2015]. To ensure both efficiency and quality of the reconstruction, view selection has been one of the most intensely studied problems [Beder and Steffen 2006; Furukawa et al. 2010; Hornung et al. 2008; Mauro et al. 2014; Mendez Maldonado et al. 2016; Moreels and Perona 2007; Mostegel et al. 2016; Rumpler et al. 2011]. Mauro et al. [2014] develop a measure for image importance and use it for view selection. Hornung et al. [2008] select views by maximizing their coverage of a dynamically-updated proxy. Mendez Maldonado et al. [2016] introduce a next-best-view optimization to trade-off between coverage and accuracy. All of these works focus on designing heuristic view selection models for scene reconstruction instead of the *image acquisition* process, which involves planning the camera paths and orientations to capture the scene beforehand.

Image-based scene acquisition and path planning. Acquisition planning can play a significant role in 3D reconstruction. When capturing large-scale urban scenes using drones, the planning task becomes especially laborious and time-consuming due to scene complexity and the immense flying space to explore. To help novice users, Xie et al. [2018] propose a handy planning tool for quadrotors to easily capture compelling aerial outdoor videos. Yang et al. [2018] convert a user-drawn 2D sketch on a flat map into a feasible and desirable camera moving path in 5D. Roberts and Hanrahan [2016] design an algorithm to revise an infeasible quadrotor camera trajectory into a similar but feasible one. Balampanis et al. [2017] and Nielsen et al. [2019] employ shape decomposition to generate shorter 2D paths for energy saving. To date, many commercial software products, such as *DJI-Terra*[1], *PIX4D*[2], *3DR*[3], have become quite mature in path planning for oblique photography in 2D.

[1]https://www.dji.com/dji-terra
[2]https://www.pix4d.com
[3]https://www.3dr.com

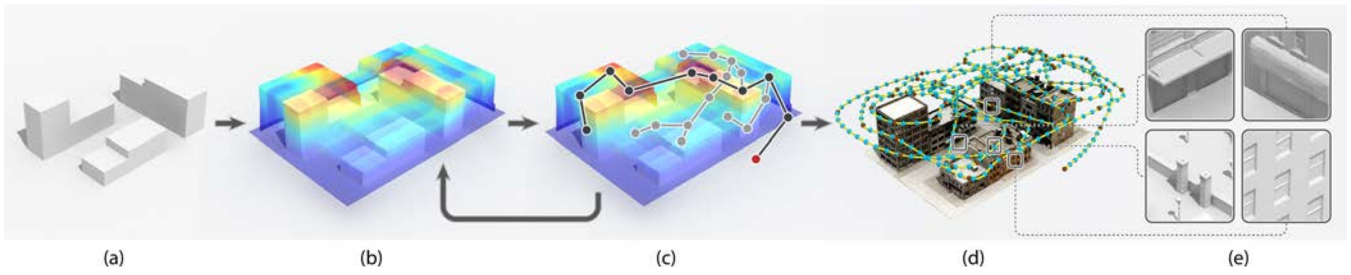


Fig. 3. Algorithm overview. Given a coarse proxy of the target scene (a), we first build a view information field (VIF) (b) to encode continuous view coverage information of the scene in the free space. Then, we start our path planning algorithm to iteratively sample viewpoints continuously in free space using the VIF and expand our random search tree (c) to explore the space, until we find a path (d) that adequately covers the scene. In (b), the red (high) to blue (low) colors indicate the amount of scene information. In (c), the red dot indicates the root of the tree and the black dots indicate the current best branch. In (d), the brown dots are waypoints and the blue dots are viewpoints in-between. In (e), some challenging regions of the final reconstructed scene are shown in blown-up views.

Without any prior scene information, earlier works [Maver and Bajcsy 1993; Whaite and Ferrie 1997] focus on planning efficient paths that *maximize scene coverage*. Heng et al. [2015] develop an efficient scheme for simultaneous visual exploration and coverage over an unknown environment. Huang et al. [2018b] plan the next best view by solving a multi-view stereo problem using an iterative method. Hepp et al. [2018a] apply a convolution neural network to score the usefulness of future viewpoints. Huang et al. [2018a] teach a drone to capture human actions cinematically by extracting skeleton points and devising a real-time planning strategy, respecting the drone's physical constraints. Kuang et al. [2020] perform an autonomous urban scene path planning by utilizing the top view to initialize a path and then refining the path iteratively via SLAM. Schmid et al. [2020] employ real-time rapid-exploring random trees [Naderi et al. 2015] to obtain a maximal coverage of the scene with minimal travel cost during exploration.

Unlike these works, our path planning solution targets efficient urban scene capture and high-quality 3D reconstruction with *fine details.* To date, most works that share a similar goal require two drone flights, where a rough scene proxy was obtained either by the first flight or reconstructed from satellite images [Zhou et al. 2020]. Given the proxy, Schmid et al. [2012] obtain a rich set of candidate views by creating a front-parallel camera for each triangle of the proxy. Roberts et al. [2017] introduce submodularity [Krause and Golovin 2014] to select the views and obtain the final trajectory by solving an orienteering problem. Hepp et al. [2018b] maximize the information gains at selected views while limiting the travel distance. Smith et al. [2018] design a heuristic system by considering the overlap between images and the coverage of the target scene, while Koch et al. [2019] accounts for semantic information. Most recently, Zhou et al. [2020] introduce redundancy minimization into the view optimization to achieve comparable scene reconstructions with much fewer images. Common to all of these works is the decoupling of view selection and path computation to connect selected views thereafter. In contrast, our method *jointly* optimizes the view sampling and path planning, considering both the scene reconstruction and path quality, along the *entire* flying path.

Rapidly-exploring Random Trees (RRT). The main idea of the RRT framework [LaValle 1998] is to iteratively add a random point in the free movement space to expand a search tree until a feasible path is found from the given start point to the target end point. Karaman et al. [2011] introduce an optimized version of RRT, coined the RRT*, where the key is a "rewiring" operator, which dynamically reconnects nodes in the tree in each RRT iteration, so as to minimize the path distance from the root; hence, a shorter path to travel to the target can be obtained. Later, Naderi et al. [2015] develop a real-time RRT* algorithm for planning paths in dynamic environments with moving obstacles. On top of these, there have been many other RRT variants [Bircher et al. 2016; Gammell et al. 2014; Schmid et al. 2020; Selin et al. 2019; Witting et al. 2018], e.g., to enhance the point sampling, to better explore the unmapped space, etc. Among these works, a recent one by Schmid et al. [2020] adopts RT-RRT* [Naderi et al. 2015] for scene exploration. However, unlike our work, their goal is to explore every part of the scene instead of capturing the scene with a drone for high-quality 3D reconstruction. Also, their method did not consider prior information of the scene for optimizing the trajectory path during the scene exploration.

In our work, we present a new application of RRT* for planning scene-capturing aerial paths for fine-detail 3D urban reconstruction. Compared to prior works, our problem has a different formulation and optimization objective, as we have to maximize scene coverage and achieve a fine 3D reconstruction while considering the aerial acquisition cost. We shall elaborate on all these in Section 5.

3 OVERVIEW

In this work, we aim to plan an efficient aerial path for a flying drone to continuously capture images of an urban scene for 3D reconstruction, given only a rough proxy geometry that approximates the target scene. Our approach is very different from the previous methods [Hepp et al. 2018b; Roberts et al. 2017; Smith et al. 2018; Zhou et al. 2020], which separate view selection and path planning in different steps. To simplify the presentation of our method, let's first assume that our drone has multiple cameras or a 360-camera system, so it does not need to rotate itself for capturing different directions at a viewpoint. In this setting, we do not need to consider the camera orientation and a viewpoint can be simply represented as a 3D location. As most commercial drones are still with a single-camera system, we will describe how our method can be adopted for single-camera drones at the end of Section 5.

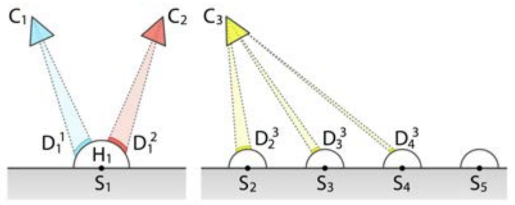


Fig. 4. The reconstructability model estimates the view coverage of surface points $\{s_i\}$ on scene proxy by viewpoints C_1, C_2, and C_3; H_i is the hemisphere at s_i; and D_i^j is a disk on H_i that indicates the coverage of s_i by C_j; its size depends on the distance from s_i to C_j and C_j's elevation angle.

The general algorithmic framework is presented in Fig. 3. First, as illustrated in Fig. 3(a), our input is a proxy that coarsely approximates the target scene. There are multiple ways to create this proxy, e.g., reconstructed from images captured by a crude drone fly, computed from satellite images [Zhou et al. 2020], or provided by commercial map suppliers. With this proxy, we can locate the free-flying space (or safe space) outside the proxy and predict how different viewpoints in the free space contribute to the scene reconstruction. So, in the second step illustrated in Fig. 3(b), we build a *view information field (VIF)* to encode the scene information covered by viewpoints in the free space. This VIF enables a fast and continuous query of view coverage information during the path planning. After that, we expand a random search tree (Fig. 3(c)) by iteratively sampling random viewpoints in the free space, connecting viewpoints (as nodes) to the tree, and optimizing node connections to maximize the path efficiency in the tree. We repeat this process until we find an aerial path (tree branch) that adequately covers the scene; see Fig. 3(d). The final reconstructed scene model is complete and well preserves visually-appealing geometry details; see e.g., Fig. 3(e). Importantly, our method jointly samples viewpoints and plans the aerial path, while considering the 3D construction quality, in a single step, unlike the state-of-the-art methods [Smith et al. 2018; Zhou et al. 2020]. So, the paths planned by our method can be smooth and efficient, covering the scene with far less sharp turns.

Technically, an aerial path consists of a sequence of *waypoints*, through which the drone will fly in the free space. In this work, our path planning method considers continuous image capture of the scene in the path, so our *viewpoints* locate not only at the waypoints but also in-between successive waypoints; see Fig. 3(d) for the brown and blue dots that illustrate *waypoints* and *viewpoints*.

4 BUILDING BLOCKS FOR PATH PLANNING

Before giving details to our path planning algorithm, we first present in this section important building blocks in the algorithm.

4.1 Scene Reconstructability

Reconstructability is a crucial heuristic model employed in many path-planning methods [Hepp et al. 2018b; Roberts et al. 2017; Smith et al. 2018; Zhou et al. 2020] for estimating the scene coverage and viewpoint correspondence. While different works may use different names for the term, e.g., viewpoint information and coverage model, the underlying formulations are essentially based on multi-view stereo. In this work, we aim for continuous image capture along the aerial path, so we do not have specific demand on viewpoint correspondence and focus reconstructability on scene coverage.

First, we follow [Roberts et al. 2017] to uniformly sample N surface points $\{s_i\}_{i=1}^N$ on the scene proxy (see Fig. 3(a)), and at each s_i, set up a hemisphere H_i that aligns with the surface normal. Then, as illustrated in Fig. 4, the coverage of the local area around s_i by viewpoint c_j is represented as disk D_i^j on H_i, such that D_i^j centers at the intersection between H_i and line c_j-s_i and its size varies with the distance between c_j and s_i and the elevation angle of the viewpoint relative to the surface. At each s_i, by computing the union of the D_i^j's for all the viewpoints that can see s_i, we can obtain the reconstructability of point s_i by the viewpoint set:

$$r(s_i) = \int_{\bigcup_j D_i^j} \omega_i(h) dh, \tag{1}$$

where $\omega_i(h)$ is a weight function of location h on hemisphere. Please refer to Appendix A for further details on D_i^j and $\omega_i(h)$.

4.2 Information Gain

Eq. (1) estimates the scene reconstructability subject to local areas in the scene. To plan aerial paths, we need to further estimate scene reconstructability subject to viewpoints in free space to evaluate how well different viewpoints contribute to the scene reconstruction. Particularly, to plan aerial paths with continuous image capture, we propose to estimate how well a new viewpoint or a new trajectory segment contributes to improving the scene reconstructability.

Mathematically, we first define c_j as a new viewpoint to be appended to an aerial path with a sequence of viewpoints $\{c_1, ..., c_{j-1}\}$. So, the information gain of c_j is the amount of new scene information that c_j brings *over* the scene information already captured by $\{c_1, ..., c_{j-1}\}$. To do so, we should estimate the scene reconstructability over the entire scene proxy instead of just at a specific surface point. Hence, we define the *information gain* of viewpoint c_j as

$$g(c_j) = \sum_i \int_{D_i^j \setminus \bigcup_{k=1}^{j-1} D_i^k} \omega_i(h) dh, \tag{2}$$

where we sum the information gained specifically by viewpoint c_j over all surface points s_i while excluding the coverage disks contributed by the previous viewpoints, i.e., D_i^1, ..., D_i^{j-1}. This formulation is mathematically equivalent to $\widetilde{f_i}$ in [Roberts et al. 2017] but using this formulation (Eq. (2)) allows us to better model and compute the view information field in Section 4.3.

In our continuous path-planning algorithm (Section 5), we need to compare how much new information that we can gain when rerouting the aerial path via different waypoints. So, we further extend Eq. (2) to estimate the information gain subject to the addition of a new waypoint, say w_k, appended to an existing path. Here, we first denote $\{c_1, c_2, ..., c_m\}$ as the viewpoints sampled along an existing path from the first waypoint w_1 all the way to the last waypoint w_{k-1} (where c_1 locates at w_1 and c_m locates at w_{k-1}) and we denote $\{c_{m+1}, c_{m+2}, ..., c_{m+n}\}$ as the viewpoints sampled along the new trajectory segment after waypoint w_{k-1} up to waypoint w_k (where c_{m+n} locates at w_k). Using these notations, we can then use Eq. (2)

ALGORITHM 1: Initializing the View Information Field.

```
input  : scene proxy and free space
output : view information field $\mathcal{F}$
Sample surface points $\{s_i\}_{i=1}^N$ on scene proxy
Sample viewpoints $\{c_j\}_{j=1}^M$ in free space
for each $c_j$ do
    for each $s_i$ do
        if $s_i$ is visible to $c_j$ then
            Set $D_i^j \longleftarrow$ compute reconstructability of $s_i$ by $c_j$
        else
            Set $D_i^j$ as a zero-radius disk
        end
    end
    $\mathcal{F}(c_j) \longleftarrow \{D_1^j, D_2^j, \ldots, D_N^j\}$
end
```

to express the information gain of the new trajectory segment from waypoint w_{k-1} to waypoint w_k:

$$g^*(w_k) = \sum_{c \in \{c_{m+1}, c_{m+2}, \ldots, c_{m+n}\}} g(c). \tag{3}$$

Essentially, Eq. (3) sums up the information gain of all the viewpoints sampled in the new trajectory segment from w_{k-1} to w_k.

4.3 View Information Field

To plan an aerial path, our algorithm needs to dynamically re-route waypoints to optimize for path quality and scene reconstructability (details in Section 5). The calculation of information gain is a core and time-consuming part in the algorithm. So, we propose to precompute the view information field (VIF) to enable *fast* and *continuous* evaluation of scene reconstructability in the free space.

To start, we discretize the free space by uniformly sampling a set of viewpoints $\{c_j\}_{j=1}^M$ in the space, and calculate coverage disk D_i^j for each pair of viewpoint c_j (in free space) and surface point s_i (on scene proxy). The set of pre-computed disks $\{D_i^j\}_{i=1}^N$ stored over the sampled viewpoints forms the VIF, which encodes the entire view coverage information of the scene; see Algorithm 1 for the procedure outline. With VIF, we can efficiently estimate the scene reconstructability continuously at any viewpoint in free space by interpolating the information at nearby viewpoints. Discretizing the entire free space will lead to a large number of viewpoints and a huge VIF, in which many viewpoints would have similar reconstructability information. So, we build an offset VIF only within a thin manifold outside the scene proxy in the 3D free space.

4.4 Travel Cost along an Aerial Path

Further, to improve the scene capturing efficiency, we analyze the travel cost of a drone: (i) cost c_t to turn the drone to align with the trajectory segment towards the next waypoint; and (ii) cost c_m to fly straightly to the next waypoint. From the experience of the drone pilots, the relation between the travel time and distance is roughly linear, whereas the relation between the travel time and turning angle is not. To incorporate these relations into path planning, we perform a field experiment to measure (i) the travel time by trying each specific turning angle 240 times and (ii) the travel time by repeating a straight-line flight 6 times for each specific distance. Also, we set the drone's flying speed to 8m/s and collect the data in this experiment on a sunny day with little wind.

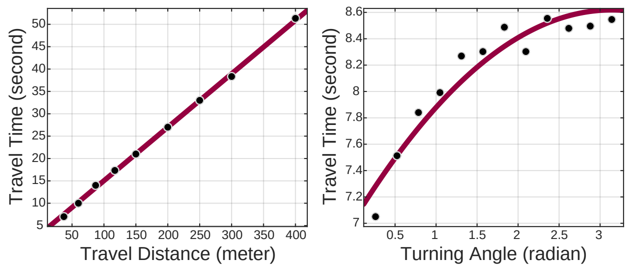


Fig. 5. Scatter plots that show travel time vs. distance (left) and travel time vs. turning angle (right) for flying a drone in an aerial path.

Fig. 5 shows the scatter plots of travel time over travel distance (left) and over turning angle (right). Observing the results, we then model the travel cost for drone movement and turning as

$$c_m(w_k) = |w_k - w_{k-1}|, \tag{4}$$
$$c_t(w_k) = 2.25 - 0.16(\theta_k - \pi)^2, \tag{5}$$

where θ_k is the turning angle at waypoint w_k. Due to the nonlinearity in turning time, we construct Eq. (5) by fitting the data samples with a 2nd-degree polynomial. Also, to measure c_t, we perform a turning of the drone after a 50m flight, so we have to subtract the time taken for the associated straight-line fly. Furthermore, different turning angles and different speeds can lead to different amount of centrifugal force acting on the drone when it turns. Hence, to maintain the drone's position precision and flight attitude, the drone has to lower its speed more for turns that are sharper.

5 AERIAL PATH PLANNING

This section gives details on our path planning algorithm. To start, we present our objective function (Section 5.1), which helps to guide the algorithm to find an aerial path that yields a high-quality scene reconstruction while reducing travel distance and saving capturing time. Then, we introduce our path planning algorithm (Section 5.2) and present strategies to accelerate it (Section 5.3). After that, we present a post-processing step for image selection (Section 5.4) and adapt our algorithm for single-camera drones (Section 5.5).

5.1 Objective Function

We formulate the following objective (which is to be maximized) to guide the path planning process:

$$E(\mathcal{T}) = E_g(\mathcal{T}) + \alpha_e E_e(\mathcal{T}) - \alpha_t E_t(\mathcal{T}), \tag{6}$$

with

$$E_g(\mathcal{T}) = \sum_{k=2}^{n} g^*(w_k), \quad E_e(\mathcal{T}) = \frac{\sum_{k=2}^{n} g^*(w_k)}{\sum_{k=2}^{n} c_m(w_k)}, \quad E_t(\mathcal{T}) = \sum_{k=2}^{n-1} c_t(w_k).$$

Here, $\{w_1, w_2, \ldots, w_n\}$ are waypoints along trajectory $\mathcal{T}$, and α_e and α_t are weights on the E_e and E_t terms, respectively.

ALGORITHM 2: Trajectory Planning.

```
input  : $w_{start}$, the starting node/waypoint
         $\{s_1, \ldots, s_N\}$, surface points on scene proxy
         $\mathcal{F}$, precomputed view information field
         $h$, $G$ and $K$, parameters for loop termination
output : $\mathcal{T}_{best}$, trajectory for scene capture
Initialize random tree $T$ with $w_{start}$ as its root node;
while percentage of surface points whose reconstructability (Eq. (1)) is
  greater than $h$ is $\leq G$ and # iterations of this while loop is $< K$ do
    $w_{new} \longleftarrow$ Random_Sample($\mathcal{F}$,$T$);       % see ALGORITHM 3
    $\mathcal{N} \longleftarrow$ Find_Neighbors($T$,$w_{new}$);      % see ALGORITHM 4
    if $\mathcal{N}$ is empty then
        $w_p \longleftarrow$ Find the node in $T$ that is nearest to $w_{new}$ and
          reachable to $w_{new}$ within the free space;
    else
        $w_p \longleftarrow$ Find the node in $\mathcal{N}$ that maximizes Eq. (6) for
          trajectory $\{w_{start}, \ldots, w_p, w_{new}\}$;
    end
    Connect $w_{new}$ to $T$ by setting $w_p$ as $w_{new}$'s parent;
    Rewire_Tree($T$,$w_{new}$);                              % see ALGORITHM 5
    if every 50 iterations then
        Rewire_Tree($T$,$w_{start}$);
    end
end
$S \longleftarrow$ Find trajectories in $T$ with sufficient coverage on $\{s_i\}$;
$\mathcal{T}_{best} \longleftarrow$ Find the trajectory in $S$ that maximizes Eq. (6);
```

ALGORITHM 3: Random_Sample.

```
input  : $\mathcal{F}$, precomputed view information field
         $T$, the random tree to be rewired
output : $w_{new}$, a random location in free space
while true do
    $w_{new} \longleftarrow$ randomize a location in $\mathcal{F}$;
    if $w_{new}$ is reachable from any node in $T$ through the free space
      without hitting or getting close to the scene proxy then
        break;
    end
end
```

ALGORITHM 4: Find_Neighbors.

```
input  : $T$, the current random tree
         $w$, a new node to be added to $T$ or to be rewired
output : $\mathcal{N}$, a subset of nodes in $T$ that are close to $w_{new}$
$\mathcal{N} \longleftarrow$ find nodes in $T$ within a radius of 10% scene size from $w$;
for each $n \in \mathcal{N}$ do
    if $w$ is not reachable from $n$ through the free space without hitting or
      getting close to the scene proxy then
        $\mathcal{N} \longleftarrow \mathcal{N} - n$;
    end
end
```

In this formulation, the first term E_g measures the information gain of the whole trajectory, so it encourages the plan planning algorithm to find a trajectory that maximally covers the scene. The second term E_e measures the scene information acquired per unit length along the trajectory, so maximizing it encourages a higher scene capturing efficiency. With the length normalization, E_e helps to optimize trajectories in a global sense instead of greedily maximizing individual nodes; also, simply subtracting lengths would make the tree favor long sub-trees and too sensitive to α_e. Lastly, the third term E_t sums up the turning angles at all the waypoints in the trajectory; by subtracting E_t in the objective (Eq. (6)), we can reduce the amount of turnings and encourage a smoother trajectory. By this means, we can reduce the need to slow down the drone for making sharp turns at waypoints and the drone can fly mostly at a constant speed to further minimize the overall scene capture time.

Note that, although both E_g and E_e employ information gain in their formulations, they have different goals. Without E_e, we may produce a trajectory that is unnecessarily long with some inefficient trajectory segments. We will present an experiment in Section 6 to demonstrate the contributions of E_e and E_t to the results.

5.2 Path Planning Procedure

Next, we present our path planning algorithm, which is inspired by [Naderi et al. 2015] and follows the Rapidly-exploring Random Tree (RRT*) framework [Gammell et al. 2014][Karaman and Frazzoli 2011], just as [Naderi et al. 2015]. The uniqueness of our algorithm is that it plans aerial paths with continuous image capture, in which we consider the scene reconstructability in the random tree, make use of our objective to guide the tree construction, and develop various strategies to accelerate the computation. Further, compared with existing methods on planning aerial paths for scene reconstruction, our algorithm jointly samples viewpoints and plans the path, optimizing both the scene reconstruction quality and path quality.

Before going into the details, we first outline our algorithm (see Algorithm 2). Its input includes a precomputed VIF (Section 4) and a user-specified starting location, i.e., w_{start}. To begin, our algorithm initializes random tree T with w_{start} as its root node. Then, it goes into a loop that iteratively randomizes a location in the free space, regards the location as a new waypoint w_{new}, connect w_{new} as a new node to T, and restructure (or rewire) T, such that T is optimized for our objective; details of each step will be given later. In random tree T, *each node is a candidate waypoint* and *each edge is a candidate trajectory segment*, so every path from w_{start} to a leaf node is a candidate trajectory. Hence, every time we add or connect a new node w_{new} to T, we should dynamically rewire the nodes connections in T to maximize the objective value for every path in T, especially those related to w_{new}. By doing so, we can iteratively expand the random tree to explore the free space, and jointly sample viewpoints and plan the aerial path. In the end, the algorithm stops when it finds a path in T that sufficiently covers the scene or when the number of iterations exceeds a budget.

Next, we present further details in the main loop of our algorithm alongside the pseudocode outlined in Algorithm 2:

- First, we randomly sample location w_{new} in free space with the constraint that w_{new} is reachable from T through the free space (see Algorithm 3). This ensures that there is at least one collision-free trajectory segment from w_{new} to T.

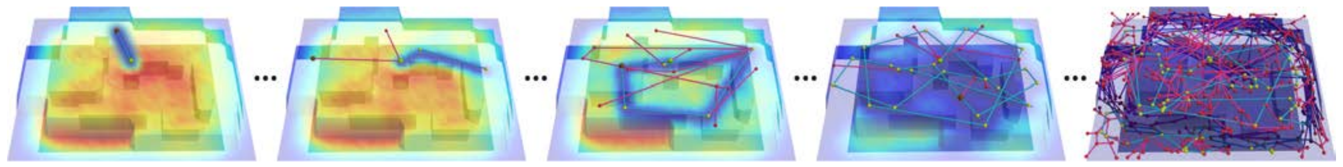


Fig. 6. A running example of our path planning algorithm. The bigger green dot near the middle marks the root node, whereas the red and yellow dots mark the internal and leaf nodes, respectively, in the random tree. Also, the yellow dots and the blue edges in-between mark the dominant (optimal) path at the moment. The colors on the scene proxy reveal the amount of scene reconstructability (red (low) to blue (high)) that remains under the dominant path.

ALGORITHM 5: Rewire_Tree.

```
input  : $T$, the random tree to be rewired
         $w$, a node in $T$ to start the rewiring
queue $Q \longleftarrow w$;
while $Q$ is not empty & within time budget do
    $w \longleftarrow$ Pop_Front($Q$);
    $\mathcal{N} \longleftarrow$ Find_Neighbors($T$,$w$) - Find_Ancestors($w$);
    for each $w_i \in \mathcal{N}$ do
        Try to connect $w_i$ to $w$ by setting $w$ as $w_i$'s parent;
        if $E(\{w_{start}, \ldots, w_i\})$ increases after the reconnection then
            Set $w$ as $w_i$'s parent;
            Push $w_i$ to $Q$;
        end
    end
end
```

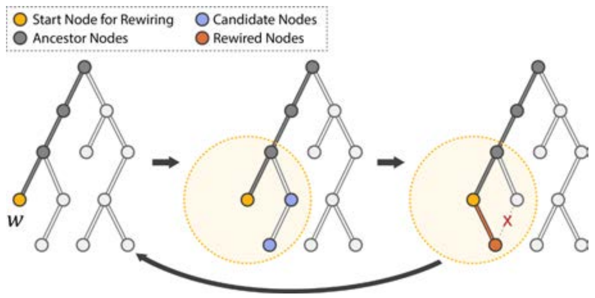


Fig. 7. Illustration of the Rewire_Tree procedure (please find details in Algorithm 5). Left: The input to Rewire_Tree is a node (say w; in yellow) in the random tree. Middle: First, Rewire_Tree finds a candidate set (in blue), which are neighbors of w, while excluding w's ancestors (in grey) to avoid creating loops in the tree. Right: For each node in the candidate set, Rewire_Tree connects the node to w as its parent, if doing so improves the objective value of the path from the root to the node. If a reconnection happens, Rewire_Tree will repeat itself on the rewired node (in orange).

- Next, Find_Neighbors (see Algorithm 4) employs a k-NN search to find nodes in T that are close to w_{new} and also reachable from w_{new}. Then, we connect w_{new} to either the nearest node in T or a neighbor node found by Find_Neighbors as w_{new}'s parent in T; see Algorithm 2 for the details.
- The current connection between w_{new} and T is unlikely optimal for our objective, so we have to rearrange (rewire) the node connections in T started from w_{new} to optimize T for our objective. See Algorithm 5 for the details and Fig. 7 for a running example. Note that, when we rewire a node, we have to exclude its ancestors (see Algorithm 5); otherwise, we could create a loop in the tree. Also, rewiring T only from w_{new} may not fully optimize the entire tree, we thus rewire T from its root every 50 iterations to trade-off between quality and speed; see Algorithm 2;

Termination. The loop terminates either when a *coverage threshold* is met *or* when a maximum loop iteration count K is reached. The latter condition follows the standard RRT* framework and guarantees that the algorithm eventually terminates. The coverage condition is defined by the percentage of sampled surface points whose reconstructability, defined in Eq. (1), exceeds a threshold h. The algorithm terminates when that percentage exceeds G, e.g., 95%, where K, G, and h are hyperparameters whose settings are provided in Section 6. Note that we do not have a theoretical proof that the coverage increases monotonically with iteration count; see an empirical analysis in supplemental material, where we show how various performance measures vary as the algorithm progresses.

Fig. 6 shows a running example of our algorithm. Usually, the random tree starts with a single dominant path. After more iterations, nodes may not be added to expand the dominant path, since doing so could reduce E_e and thus the overall objective. Hence, path planning can be seen as an iterative process that gradually refines the dominant path, which may sometimes disappear and re-appear elsewhere. Curious readers may refer to supplementary material for a convergence analysis. The elementary operations in our algorithm are node addition and tree rewiring. In general, the computational complexity is $O(mlog(m))$, where m is the number of nodes and memory complexity is $O(m)$. These complexities are independent of the scene size but planning time surely increases for larger and more complex scenes, which require more nodes to cover.

5.3 Acceleration Strategies

Algorithm 2 is computationally very heavy. We have to extensively evaluate information gain in the objective every time to decide how to connect a new node to the random tree and how to rewire the tree. Also, we have to evaluate the reconstructability of every surface point for every node in the tree. Hence, we design various strategies to accelerate the associated computations in the algorithm:

- First, we precompute the view information field (see Section 4.3), so that when we have a new waypoint w_{new}, we can use the field to quickly obtain the coverage disk D_i of w_{new} (as a viewpoint) on every surface point s_i in the scene. The same also goes to the new viewpoints sampled along the new trajectory segment that connects to w_{new}.

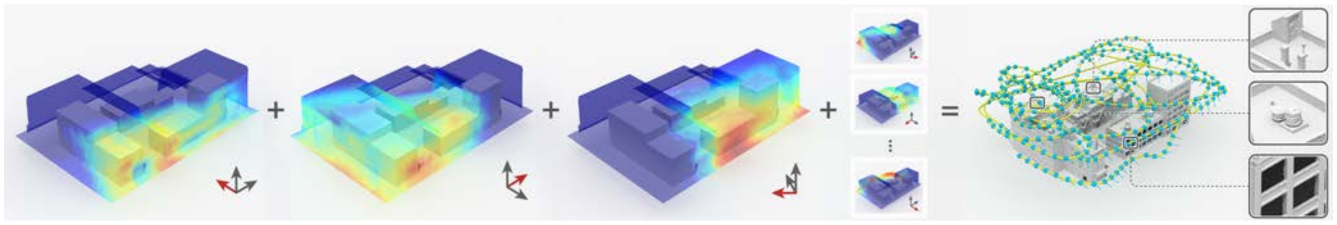


Fig. 8. To adapt our method for single-camera drones, we extend the 3D view information field to become a 5D view information field to account for the view direction ($\mathbb{S}^2$) and position ($\mathbb{R}^3$) of viewpoints in the free space (left). Note that, aerial paths planned for single-camera drones are usually longer (and more complex) than paths planned for multi-camera drones (for the same scene) since the drone has to travel longer to capture different aspects of the scene.

- Second, to efficiently store and compute the coverage disks, we discretize hemisphere H_i (see Section 4.1) as N_h points evenly on H_i and represent a coverage disk as an N_h-bit vector; each bit marks if the associated point on H_i is inside or outside D_i^j. In our implementation, N_h is 256, so a coverage disk consumes only 16 bytes. Also, this representation allows us to implement the disk union and disk subtraction in Eqs. (1) & (2) efficiently as bit-wise operations.
- Third, a crucial element in the computation of reconstructability and information gain is the union of coverage disks that accumulates along the trajectory from the root of the tree; see Eqs. (1) and (2). Here, we propose to store $\cup_j D_i^j$ per surface point per node in tree as an N_h-bit vector. This can speed up various computations, e.g., when we rewire a node to a new parent, we can update its $\cup_j D_i^j$ based on the $\cup_j D_i^j$ of its parent, rather than recomputing its $\cup_j D_i^j$ all the way from the root. Also, this strategy enables us to readily compute the reconstructability of any s_i at any node.

5.4 Post-Selection of Scene Images

With continuous image capture along with the aerial path, we typically will obtain a dense image set, which provides scene information more than necessary for reconstructing the scene. As suggested in [Hepp et al. 2018b; Seitz et al. 2006], using excessive images may even degrade the reconstruction quality. Therefore, we propose to select an image subset before the scene reconstruction.

Denoting $\mathcal{I}$ as the dense image set captured by aerial path $\mathcal{T}_{best}$, we aim to select a subset of $\mathcal{I}$ with minimal reduction in scene reconstructability. To do so, we initialize the final image set $\mathcal{I}'$ as an empty set and iteratively pick images one at a time from $\mathcal{I}$ to put into $\mathcal{I}'$. In each iteration, we pick the image that remained in $\mathcal{I}$ that can bring the largest information gain based on Eq. (2). Note that when we calculate Eq. (2), we compute the information gain relative to the viewpoints of the selected images in $\mathcal{I}'$. We repeat this process until the reconstructability of G% surface points is greater than threshold h (same as Algorithm 2). Note that post selection should not be used for culling the path before the flight, since it will destroy the smoothness of the aerial path. Also, we include post selection mainly for five-camera drones, which unavoidably generate a huge amount of redundant scene information.

5.5 Adaptation for Single-Camera Drones

So far, we assume that the drone has multiple fixed cameras, so it can capture a panoramic view of the scene at any location in an aerial path. However, most conventional portable drones contain only a single camera stabilized by a 3-axis (pitch, roll, yaw) mechanical gimbal, so earlier works in aerial path planning typically study under such a setting, i.e., optimizing 5D viewpoints. To enhance the practical usage and facilitate a fair comparison with the existing techniques, we can adapt our scheme to support single-camera path planning with the following modifications in the algorithm:

- First, we can regard viewpoints in free space as 5D vectors, each with a view vector (in $\mathbb{S}^2$) and a viewpoint position (in $\mathbb{R}^3$) in the free space (left). Then, we can extend the view information field (VIF) from 3D to 5D by discretizing the view space $\mathbb{S}^2$ with a set of sample points, each with a 3D VIF; see Fig. 8 for an illustrative example.
- Second, we need to assign a view vector to each node (waypoint) in the tree to represent the drone's camera direction. Then, we can obtain the view vector along an edge (trajectory segment) in the tree by interpolation.
- Third, while the view vector at the root of the tree is given by the user, we have to determine the view vector at each node of the tree. In Algorithm 2, whenever we create a new edge in the tree (i.e., when we create a new waypoint and connect it to the tree, or when we rewire a waypoint to connect to a new parent), we have to determine the view vector at the waypoint. We do so by finding the view vector that maximizes the information gain of the waypoint over its parent, i.e., Eq. (3). Yet, we need to constrain the view vector not to deviate too much from its parent's view vector (unless the two waypoints are far apart); otherwise, the drone's camera may rotate too fast and produce blurry images.

6 RESULTS AND EVALUATION

In this section, we show quantitative and qualitative results and compare our work to state-of-the-art methods, on both synthetic benchmark datasets and several large-scale real scenes.

Parameters. In our objective function, we empirically set $\alpha_e = 50$ and $\alpha_t = 20$, for capture with a five-camera drone, and $\alpha_e = 8$ and $\alpha_t = 10$, for capture with a single-camera drone. Also, we limit the maximum number of iterations (K) in Algorithm 2 to 400 and set the time budget (maximum number of rewiring) in Algorithm 5 to 20. Depending on the size of the scene, we sample $N = 800$, 1200, or 1600 surface points on the scene proxy, and build a view information field with $M = 800$, 1000, or 2000 viewpoints (3D) and 17 view directions (2D). In Algorithm 2, we set $h = 30$ and $G = 95$ for the benchmark datasets, which are smaller in size, and $h = 15$ and $G = 85$ for real scenes, which are larger in size and have more complex geometry, to balance the trajectory length and reconstruction quality. Based on the feedback from our pilot and the experiments, these settings are sufficient for scenes of size 0.1km^2.

Datasets. We tested and evaluated the different methods on two kinds of datasets. First, following the state of the art [Zhou et al. 2020], we employed the following three benchmarks: NY-1, UK-1 [Smith et al. 2018], and Bridge-1 [Zhou et al. 2020], with the associated scene proxies (2.5D) kindly provided by Zhou et al. [2020]. These benchmark datasets are synthetic and provide ground-truth information to facilitate quantitative evaluation. At the same time, the covered scenes are small in size and contain much less geometric details than real scenes. Hence, for a real test, we also run the path-planning methods on several large-scale real scenes.

Implementation details. Our method was implemented in C++ and run on a computer with a 2.4GHz Intel Xeon E5-2680 CPU, 64GB RAM, and NVIDIA Quadro M5000. To capture real scenes, we experimented with two drone systems: DJI M300RTK with PSDK 102S, which is a five-camera drone with a focal length of 35mm, and DJI Phantom 4 RTK, which is a single-camera drone with a focal length ranged from 8.8 to 24mm. Lastly, we use the commercial software RealityCapture to reconstruct the 3D mesh of the scene from the captured images. For a fair evaluation, we use the default setting for this software to reconstruct scenes in all of our experiments.

Evaluation metrics. To quantitatively evaluate the reconstruction quality using the benchmarks, we used the following two metrics provided by [Smith et al. 2018]:

- *Error* measures how close the reconstructed mesh is when compared with the ground truth provided in the benchmark. For each point p_i in the reconstructed mesh, we find its minimum distance d_i to the points in the ground truth. Then, we determine the 90% error (and the 95% error), which is the smallest distance (in meters) in all d_i, such that 90% (and 95%) of distances $\{d_i\}$ are shorter than it.
- *Completeness* measures the coverage of the ground truth by the reconstructed mesh. Its formulation is opposite to error. For each point q_j in the ground truth, we find its minimum distance d_j to the points in the reconstructed mesh. Given distance d (in meters), we determine the percentage of points in the ground truth whose d_j is smaller than d.

For the error metric, a smaller (distance) value indicates a better result, whereas, for the completeness metric, a larger (percentage) value indicates a better result. Besides, we evaluate the quality of the planned paths in terms of trajectory length, number of waypoints, flight time spent on movement (estimated by Eq. (4)), and flight time spent on turning (estimated by Eq. (5)).

Table 1. Self evaluation #1. We explore how the terms in our objective function (Eq. (6)) affect path quality, measured by path length, number of waypoints, and estimated time spent on movement and turning.

Objective	Length ↓	#Waypoints ↓	Movement ↓	Turning ↓
only E_g	3,274 m	117	614 sec.	215 sec.
only E_g and E_e	2,298 m	117	455 sec.	168 sec.
only E_g and E_t	2,700 m	89	488 sec.	151 sec.
full Eq. (6)	**2,151 m**	**84**	**384 sec.**	**115 sec.**

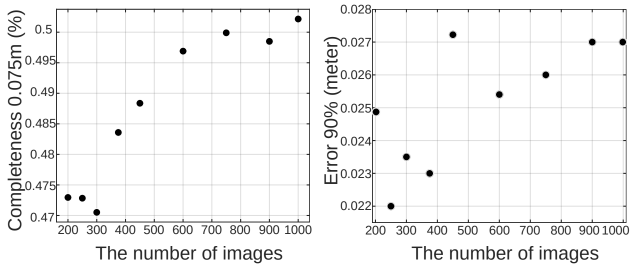


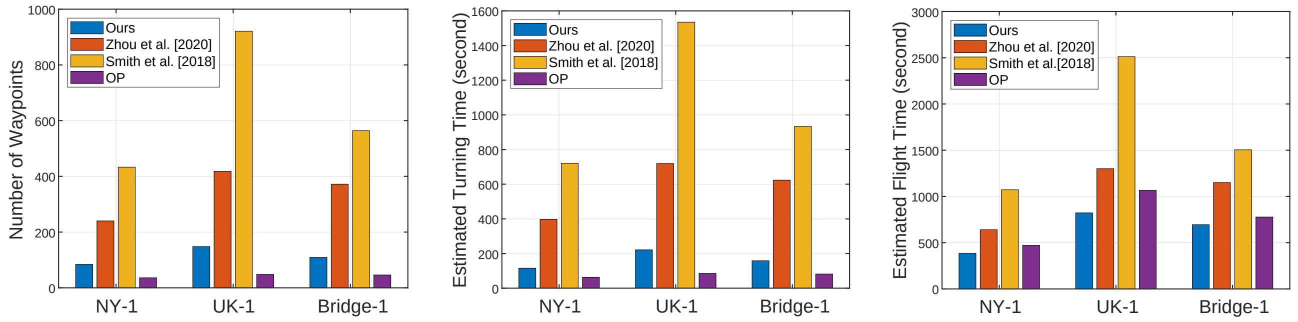
Fig. 9. Scene reconstruction quality (in terms of completeness & error) vs. the number of images employed in the 3D reconstruction.

6.1 Self Evaluation

Path quality vs. terms in objective function. First, to demonstrate how the E_e and E_t terms in our objective (Eq. (6)) contribute to improve the quality of the planned paths, we conduct an experiment by using our method to plan aerial paths with different forms of objective functions: (i) only E_g; (ii) only E_g and E_e; (iii) only E_g and E_t; and (iv) full objective as in Eq. (6). Here, we employ the NY-1 dataset and compute the planned paths for a single-camera drone, meaning that the planned paths are likely more complex.

From the results reported in Table 1, we can first see that using only E_g leads to the creation of an unnecessarily long aerial path, since E_g simply maximizes the scene information that one can acquire. If we include E_e, we can improve the scene capture efficiency by shortening the path, while ensuring the optimized path sufficiently covers the scene. Next, comparing the first vs. third rows, or second vs. last rows in Table 1, we can see that the E_t term helps to reduce the amount of time spent on turning the drone (estimated based on Eq. (5) from our field experiments), as well as other path quantities. Hence, putting all three terms together leads to the best quality path, i.e., shorter path length, fewer waypoints, less sharp turns, and shorter movement time, as reported in Table 1.

Reconstruction quality vs. number of images. Using more images generally means having more scene information in the 3D reconstruction. However, having too many images could degrade the reconstruction quality, as suggested in [Hepp et al. 2018b; Seitz et al. 2006]. In this experiment, we evaluate how the number of images selected in the post-selection step affects the reconstruction quality in terms of completeness and error. Here, we use the NY-1 dataset and plan aerial paths using a different number of images. Fig. 9 reports the results. For completeness, it is straightforward to see that more images clearly leads to higher scene coverage, so completeness increases strictly with the number of images. For error, we can see



(a) Number of waypoints (b) Estimated extra time for turning (c) Estimated overall flight time

Fig. 10. Comparing the quality of the paths produced by our method, [Zhou et al. 2020], and [Smith et al. 2018] on NY-1, UK-1, and Bridge-1. Our paths are of higher quality with smaller #waypoints, as well as shorter turning and overall flight times. We also show numbers for oblique photography (OP), but for reference only, since it requires a five-camera drone to capture more images and make fewer turns, while others use only a single-camera drone.

Table 2. Impact of using proxies of different precisions on our path-pathing algorithm. Top: randomly perturbing the building heights in proxies. Bottom: smooth building boundaries by morphological operator "dilation".

Height Perturb	Num. Images	Length [m]	Error ↓ [m] 90%	95%	Completeness ↑ [%] 0.020m	0.050m	0.075m
0%	252	2,151	0.030	0.261	43.50	52.89	57.42
10%	252	2,250	0.037	0.807	43.87	52.43	57.28
20%	255	2,276	0.039	0.668	43.43	51.95	57.05

Building Boundary	Num. Images	Length [m]	Error ↓ [m] 90%	95%	Completeness ↑ [%] 0.020m	0.050m	0.075m
original	252	2,151	0.030	0.261	43.50	52.89	57.42
minor	252	2,290	0.038	0.728	42.11	51.48	55.40
medium	252	2,293	0.039	0.994	41.71	51.48	56.00
rough	254	2,283	0.039	0.792	42.37	51.87	56.36

Table 3. Comparing the reconstruction quality (Error) and scene converage (Completeness) of paths produced by different methods (Smith: [Smith et al. 2018]; Zhou: [Zhou et al. 2020]; and OP: oblique photography) on the synthetic datasets NY-1, UK-1, and Bridge-1. We highlight the first- and second-place performances using bold and italic fonts, respectively.

Scene	Method	Num. Images	Length [m]	Error ↓ [m] 90%	95%	Completeness ↑ [%] 0.020m	0.050m	0.075m
NY-1	Smith	433	2,808	0.053	0.792	36.01	44.74	49.47
	Zhou	248	1,816	**0.028**	0.315	*40.24*	*47.57*	*52.04*
	OP	1,297	3,264	0.044	*0.296*	36.58	44.74	49.33
	Ours	252	2,151	*0.030*	**0.261**	**43.50**	**52.89**	**57.42**
UK-1	Smith	923	7,819	**0.028**	*0.051*	**32.04**	**37.74**	40.62
	Zhou	418	4,648	**0.028**	**0.049**	*30.75*	35.96	40.29
	OP	2,056	7,844	0.057	0.103	30.52	35.11	38.25
	Ours	420	5,483	*0.034*	0.067	30.60	*36.08*	**53.12**
Bridge-1	Smith	565	4,569	**0.014**	**0.022**	49.47	55.73	59.07
	Zhou	372	4,217	*0.015*	*0.023*	**52.78**	*59.81*	*63.06*
	OP	1,232	5,565	0.029	0.043	50.57	57.66	61.36
	Ours	380	5,415	0.018	0.027	*51.64*	**60.48**	**70.41**

from the associated plot that its value drops only initially with the number of images but increases when there are too many images. Yet, the plot's vertical range shows that the errors are typically very small, only in the range of two to three centimeters.

Impact by imperfect proxies. In this work, we employ 2.5D scene proxies, in which each building is represented by a height value and a boundary given on the map. These proxies can be readily generated by the commercial map suppliers, but they may contain errors due to inaccuracies in building heights and boundaries. To evaluate the impact of imperfect proxies on our method, we perform an experiment similar to [Zhou et al. 2020]. First, we randomly perturb the building heights to different extents. On the other hand, we smooth and expand the building boundaries in image space using the morphological operator dilation to different extents. Then, we use our method to plan aerial paths with these proxies and evaluate the quality of the resulting scene reconstructions.

Table 2 summarizes the results. Overall, using lower quality scene proxies unavoidably reduces the scene reconstruction quality, especially the error term. Yet, for the completeness term, it drops very little for perturbation in building heights and only slightly for changes in building boundaries, showing that the robustness of our method on completeness in scene coverage.

Path planning time. Further, we study the running time of Algorithm 2. For NY-1, UK-1, and Bridge-1, our algorithm takes 5.3 minutes, 10.5 minutes, and 8.9 minutes, respectively, to plan the paths. Though these numbers are slightly longer than the time taken by previous works [Smith et al. 2018; Zhou et al. 2020], our algorithm plans continuous paths, while previous works do not. Also, these numbers are not significant, when comparing to the several hours needed to reconstruct the scene from the captured images.

6.2 Results and Comparisons on Synthetic Data

Next, we compare our method, on synthetic datasets (NY-1, UK-1, and Bridge-1), to several state-of-the-art methods, including [Smith et al. 2018], [Zhou et al. 2020], and oblique photography (OP), which is a commercial technology for real scene capture. For this experiment, we obtained the paths planned by [Zhou et al. 2020] from the authors. For [Smith et al. 2018], we obtained the paths planned for NY-1 and UK-1 from the project webpage and the path for Bridge-1 from the authors of [Zhou et al. 2020]. Finally, for oblique photography (OP), the path is obtained as a simple sweep pattern, alternating between left-to-right and right-to-left, to cover a designated area. Since the path is situated at a fixed height above all buildings in the area, OP requires a five-camera drone to capture more images, since the drone does not move to a lower height to capture more details. We follow the practice in [Zhou et al. 2020] and [Smith et al. 2018] to set the parameters for oblique photography: focal_length = 30*mm*; overlap_ratio = 15%; and the flying height for NY-1, UK-1, and Bridge-1 are 50*m*, 70*m*, and 70*m*, respectively.

Quantitative results. Table 3 and Fig. 10 summarize the quantitative evaluation results, where the third column in Table 3 shows the actual number of images used for 3D reconstruction in RealityCapture. In terms of reconstruction quality, we can see that overall, our method achieves comparable results in the various measures using a similar number of images as [Zhou et al. 2020], but much fewer images than [Smith et al. 2018] and oblique photography. Our results typically attain higher scores in completeness, i.e., scene coverage, reflecting its ability to better capture scene geometries including their details. The performance numbers for the "Error" measures are generally comparable, except for the "95%" numbers for the NY-1 dataset, where our method outperforms the others significantly. In all other cases, the performance discrepancies between all three methods are all within margins of few centimeters.

In terms of path quality, while the length of the paths produced by our method may not be shorter than others, the estimated flight times, both turning and movement for our paths, are clearly shorter, as reported by the plots shown in Fig. 10. Since our paths are more smooth, they could be longer than the paths produced by other methods, but yet, the flight time can be shorter, since the drone can fly at a mostly constant speed. Note that OP is served as a reference in Fig. 10, since its sweep paths are at a fixed height and make far fewer turns than others; hence, the resulting reconstructions are often more rough and less complete, as revealed in Table 3.

Visual comparison and detail recovery. We visually compare the aerial paths and the 3D scene reconstructions associated with the four methods on the three datasets in Fig. 11. Please pay special attention to our method's superior ability to recover finer-level details compared to the other methods. This is a clear and consistent trend, since the scenes were captured continuously using our method. Even when we only sample a subset of the images for reconstruction, the denser view sampling, and hence the denser image-to-image correspondences, compared to the alternatives, still lead to better detail reconstruction.

6.3 Results and Comparisons on Real Scenes

Lastly, we use drones to capture images of three large-scale real scenes. We plan aerial paths for each scene using various methods, conduct field trips to capture each scene with different planned paths, and reconstruct the scenes from each set of capture images.

Comparison with [Zhou et al. 2020]. We use our method and [Zhou et al. 2020] to plan aerial paths for two large-scale urban areas, namely ArtSci and Library. Fig. 12 shows the planned aerial plans and the resulting 3D reconstructions, whereas Table 4 reports the

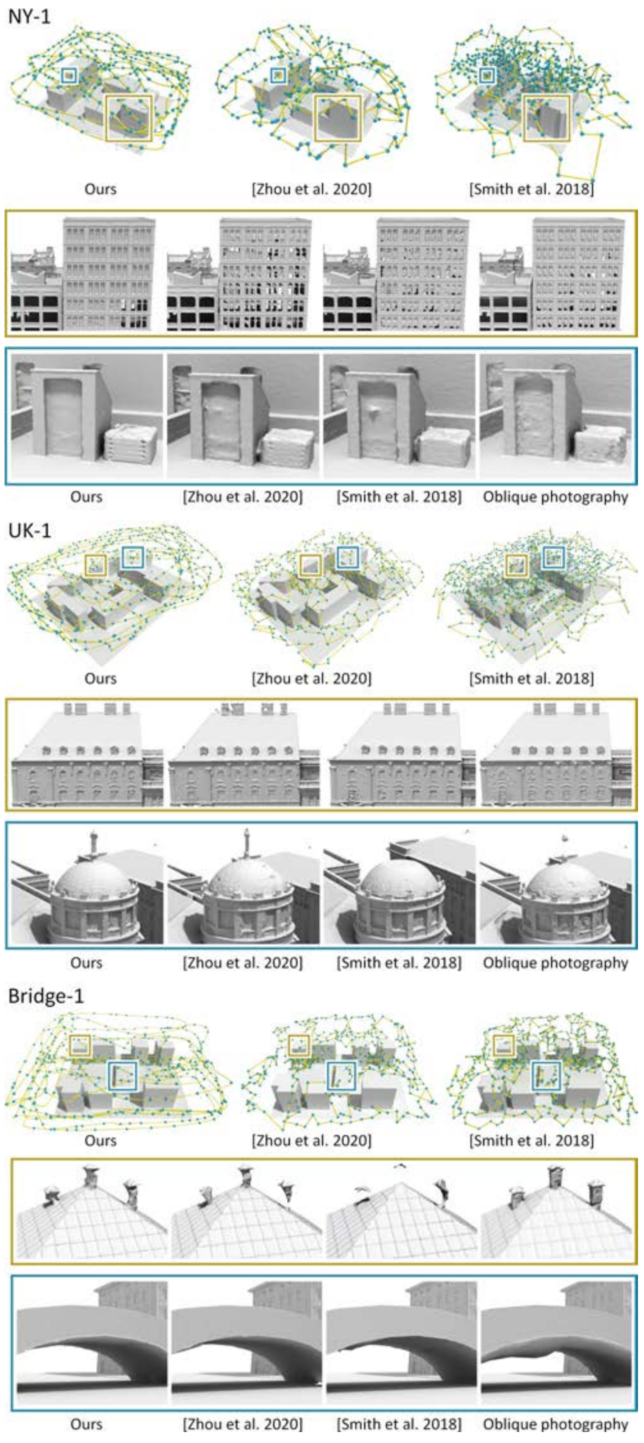


Fig. 11. Visual comparisons of the planned aerial paths and resulting 3D reconstructions (see particularly the blown-up views) produced by our method, [Zhou et al. 2020], [Smith et al. 2018], and oblique photography (OP) on the synthetic datasets: NY-1, UK-1, and Bridge-1 (top to bottom). Due to space limit, please find the paths of OP in supplementary material.

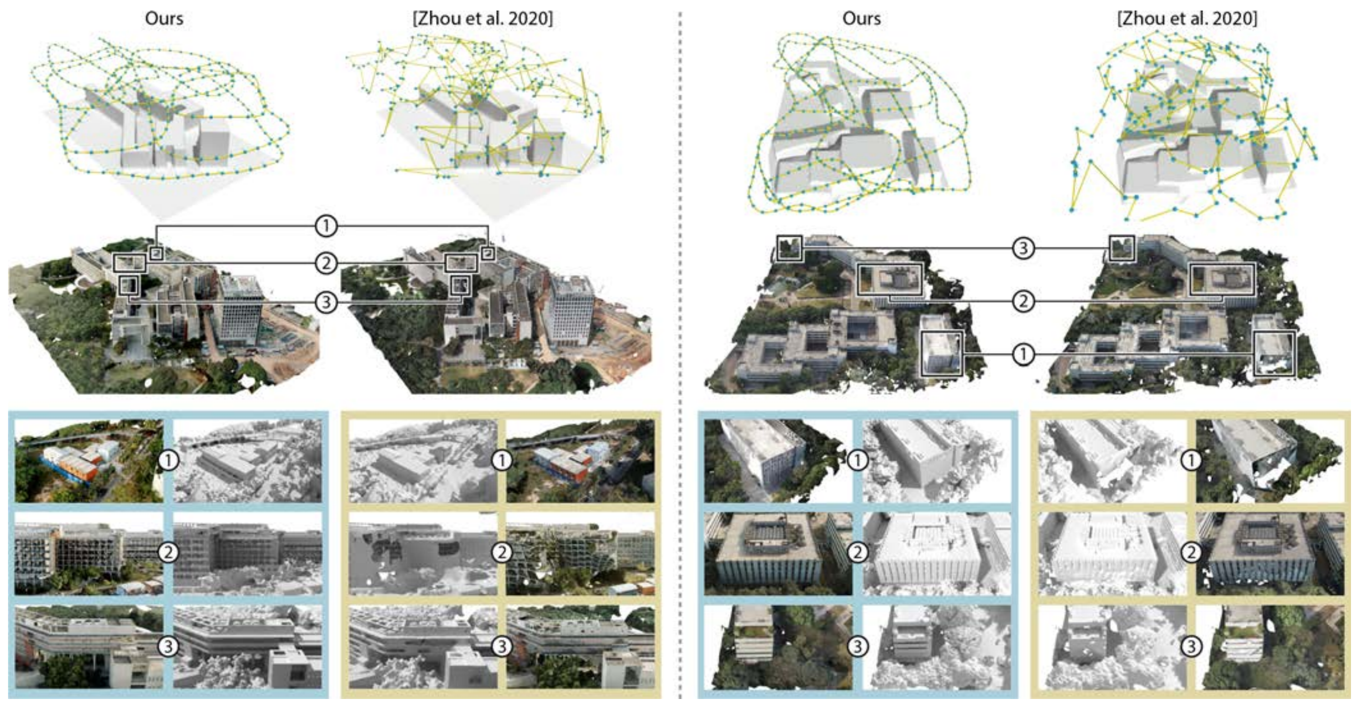


Fig. 12. Visual comparison on ArtSci (left : 76,900m^2) and Library (right : 69,471m^2). In each scene, first column shows the result of our methods, whereas second column shows the result of [Zhou et al. 2020]. Please zoom in to look at the blown-up views to compare the details in the 3D reconstructions.

Table 4. Comparing the quality of the aerial paths planned by [Zhou et al. 2020] (Zhou) and by our method on real scenes ArtSci and Library.

Scene	Method	Length ↓ [m]	Flight Time ↓ [min]	Battery ↓ Consump.%
ArtSci	[Zhou et al. 2020]	7,233	41.0	146%
	Ours	**4,137**	**13.7**	**50%**
Library	[Zhou et al. 2020]	6,531	28.1	108%
	Ours	**4,627**	**15.5**	**50%**

Table 5. Comparing the quality of the aerial paths planned by oblique photography (OP) and our method on real scenes Residency and High-rise.

Scene	Method	Length ↓ [m]	Flight Time ↓ [min]	Battery ↓ Consump.%
Residency	OP	8,269	34.0	486%
	Ours	**7,665**	**22.0**	**304%**
High-rise	OP	15,852	51.0	1,112%
	Ours	**13,916**	**42.0**	**801%**

flight details. From Table 4, we can see the superior quality of the paths planned by our method, particularly in terms of flight time and battery consumption. Also, the blown-up views shown in Fig. 12 confirms the visual quality of the 3D reconstructions produced from our captured images, both in terms of scene coverage and the recovery of finer-level geometric details.

Comparison with oblique photography (OP). In this experiment, we plan aerial paths independently using our method and oblique photography (OP) for capturing two large-scale urban areas, namely Residency (0.1km^2; Fig. 13) and High-rise (0.5km^2; Fig. 14).

Fig. 13 (top) shows the two aerial paths for capturing Residency, in which the one for OP (in green) is planned using commercial software DJI-Terra. Using each path, we captured the same urban area using the same five-camera drone and measured the time taken, path length, and battery consumption of each flight. Since the path of OP is uniform spacing, the drone may need to fly further out of the target scene area to capture lower portions of the buildings in the scene. Our planner avoids this, since drones can fly at a lower altitude and our path better fits the building distribution to reduce unnecessary flight. These features make our method more efficient.

The other scene High-rise shown in Fig. 14 features a very large-scale urban area, in which the buildings have large variation in height. Since OP needs to set a fixed flying height well above all buildings, the drone cannot capture sufficient details in this scene, especially for the lower-height building parts. The paths planned by our method do not suffer from such problems.

Table 5 reports the overall quantitative comparison results, showing that the aerial paths planned by our method clearly have shorter lengths, and more importantly, shorter flight time, for both large-scale real scenes. Also, the resulting battery consumption by our paths is comparatively lower. Fig. 13 (bottom) and Fig. 14 (bottom) further show 3D scene reconstructions produced from images captured by different aerial paths. From the blown-up views, we can see

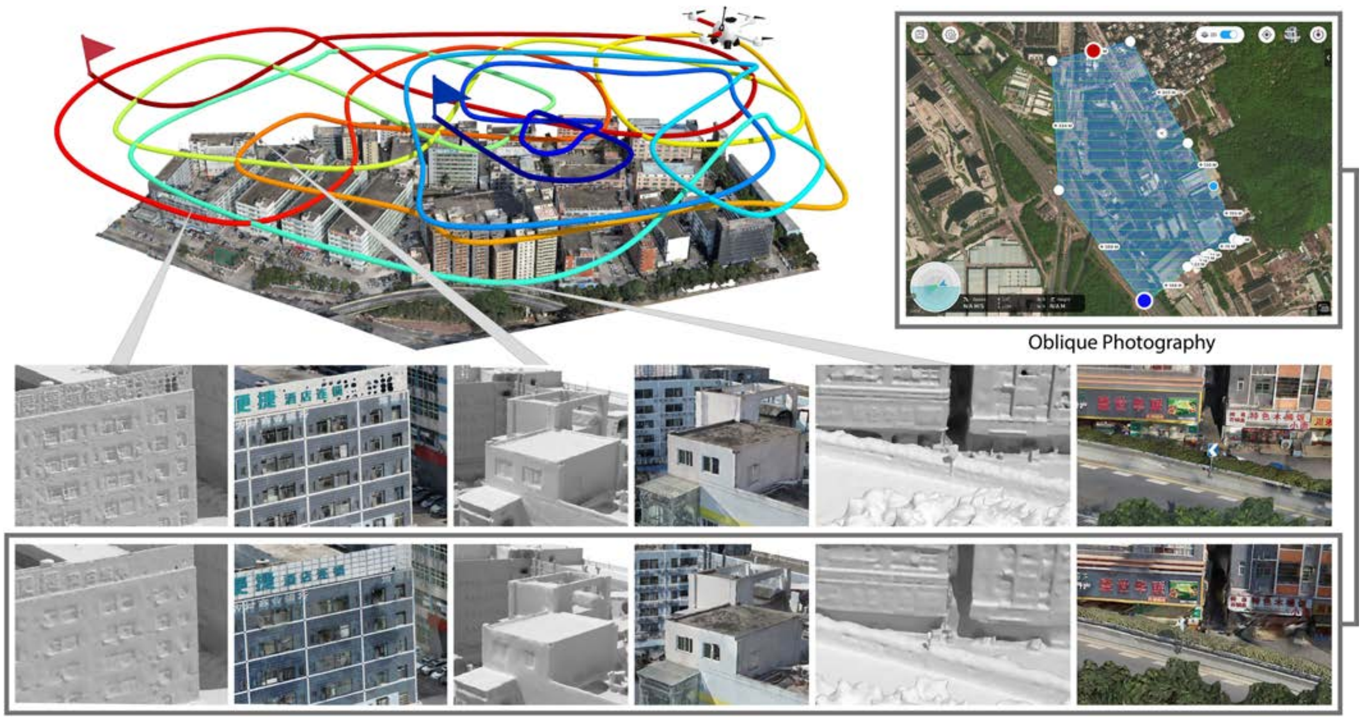


Fig. 13. The scene Residency is of $0.1km^2$. Top left: our planned aerial path (from blue to red). Top right: aerial path (in green, from blue dot to red dot) by oblique photography (OP). Bottom: "detail" comparison on this scene; 3D reconstruction results of our method (first row) and OP (second row).

that images captured by our path lead to higher 3D reconstruction quality, especially in finer-level details.

As we can only show static images on paper, please watch our supplemental video for 3D views of our results.

7 CONCLUSION, DISCUSSION, AND FUTURE WORK

We developed the first continuous aerial path planning framework for high-quality 3D urban scene capture and reconstruction. We take advantage of the drone's ability to capture images in quick successions along its flying path, which enables a dense view sampling of the scene to improve reconstruction quality. Also, our path optimization jointly considers the path quality to avoid sharp turns to better utilize the drone's limited battery power and flying time. We demonstrate the superiority of method over state-of-the-art alternatives, in terms of path quality and reconstruction quality, especially for the recovery of geometric details, on both synthetic benchmark datasets and large-scale real-world scans.

A key assumption made by our current approach is the availability of a rough scene proxy. Extending our method to accommodate unknown 3D environments can be challenging but is a worthy pursuit. On the technical front, our setup of the view information field is rather time-consuming and our objective function enforces the tree search to go into high depth, both contributing to a relatively long path planning time. Moreover, our current greedy scheme for selecting the final set of images for scene reconstruction is still quite simplistic and elementary. We expect that a more sophisticated approach would lead to further improvement in the trade-off between scene reconstruction quality and image usage.

In addition to addressing the above limitations, we would also like to explore *learning-based* approaches to continuous path planning. Indeed, there have been recent attempts at using reinforcement learning to tackle view planning problems [Kaba et al. 2017] and for obtaining the next best views in the context of scan completion [Peralta et al. 2020]. Network training may be conducted over procedurally generated urban scene data [Smelik et al. 2014]. We expect that the predictive power of a trained network can be more apt at path planning over unknown urban environments.

REFERENCES

Fotios Balampanis, Ivan Maza, and Anibal Ollero. 2017. Spiral-like coverage path planning for multiple heterogeneous UAS operating in coastal regions. In *Proc. of Int. Conf. on Unmanned Aircraft Systems*. 617–624.

Christian Beder and Richard Steffen. 2006. Determining an Initial Image Pair for Fixing the Scale of a 3D Reconstruction from an Image Sequence. In *Pattern Recognition*. 657–666.

Andreas Bircher, Mina Kamel, Kostas Alexis, Helen Oleynikova, and Roland Siegwart. 2016. Receding horizon" next-best-view" planner for 3d exploration. In *Proc. of IEEE Int. Conf. on Robotics & Automation*. 1462–1468.

Yasutaka Furukawa, Brian Curless, Steven Seitz, and Richard Szeliski. 2010. Towards Internet-scale Multi-view Stereo. *Proc. of IEEE Conf. on Computer Vision & Pattern Recognition* 2010, 1434–1441.

Yasutaka Furukawa and Carlos Hernández. 2015. Multi-View Stereo: A Tutorial. *Foundations and Trends in Computer Graphics and Vision* 9, 1âĂŞ2 (2015), 1âĂŞ148.

J. D. Gammell, S. S. Srinivasa, and T. D. Barfoot. 2014. Informed RRT*: Optimal sampling-based path planning focused via direct sampling of an admissible ellipsoidal heuristic. In *Proc. of IEEE Int. Conf. on Intelligent Robots & Systems*. 2997–3004.

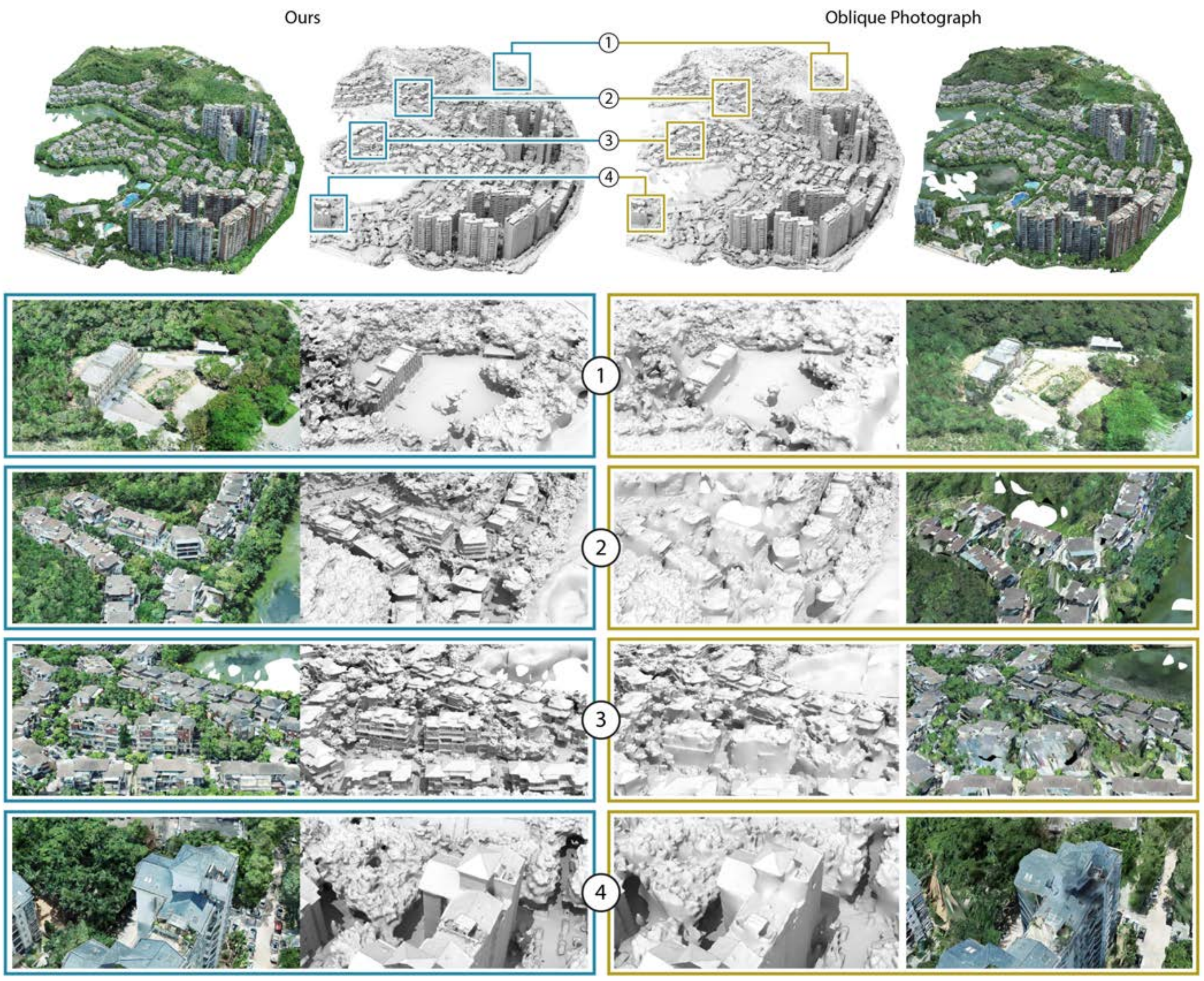


Fig. 14. The scene High-rise is quite large in size, 0.5km^2 ($0.71\text{km} \times 0.87\text{km}$), and covers buildings and other structures with much variation in height. The tallest building is $140m$ high, so the aerial path for oblique photography (OP) is set at a height of 170m, while our path can direct the drone to adaptively fly between 80m and 170m elevations, to best capture the scene details. The top row shows the entire 3D scene reconstructions produced from images captured using our path (top left) and OP (top right). Due to the complexity of the scenes, we show blown-up views at the bottom to allow a visual comparison between the fine details reconstructed. Also, we show plain renderings without colors for better assessment of the reconstructed geometry.

Lionel Heng, Alkis Gotovos, Andreas Krause, and Marc Pollefeys. 2015. Efficient visual exploration and coverage with a micro aerial vehicle in unknown environments. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*. 1071–1078.

Benjamin Hepp, Debadeepta Dey, Sudipta N Sinha, Ashish Kapoor, Neel Joshi, and Otmar Hilliges. 2018a. Learn-to-score: Efficient 3d scene exploration by predicting view utility. In *Proceedings of the European conference on computer vision (ECCV)*. 437–452.

Benjamin Hepp, Matthias Nießner, and Otmar Hilliges. 2018b. Plan3D: Viewpoint and Trajectory Optimization for Aerial Multi-View Stereo Reconstruction. *ACM Trans. on Graphics* 38, 1 (2018), 4:1–4:17.

A. Hornung, B. Zeng, and L. Kobbelt. 2008. Image Selection for Improved Multi-View Stereo. In *Proc. of IEEE Conf. on Computer Vision & Pattern Recognition*. 1–8.

Chong Huang, Fei Gao, Jie Pan, Zhenyu Yang, Weihao Qiu, Peng Chen, Xin Yang, Shaojie Shen, and Kwang-Ting Tim Cheng. 2018a. Act: An Autonomous Drone Cinematography System for Action Scenes. In *Proc. of IEEE Int. Conf. on Robotics & Automation*. 7039–7046.

Rui Huang, Danping Zou, Richard Vaughan, and Ping Tan. 2018b. Active Image-based Modeling with a Toy Drone. *Proc. of IEEE Int. Conf. on Robotics & Automation* (2018), 1–8.

Mustafa Devrim Kaba, Mustafa Gokhan Uzunbas, and Ser Nam Lim. 2017. A reinforcement learning approach to the view planning problem. In *Proc. of IEEE Conf. on Computer Vision & Pattern Recognition*. 5094–5102.

Sertac Karaman and Emilio Frazzoli. 2011. Sampling-based algorithms for optimal motion planning. *The international journal of robotics research* 30, 7 (2011), 846–894.

Arno Knapitsch, Jaesik Park, Qian-Yi Zhou, and Vladlen Koltun. 2017. Tanks and Temples: Benchmarking Large-scale Scene Reconstruction. *ACM Trans. on Graphics (Proc. of SIGGRAPH)* 36, 4 (2017), 78:1–78:13.

Tobias Koch, Marco Körner, and Friedrich Fraundorfer. 2019. Automatic and semantically-aware 3D UAV flight planning for image-based 3D reconstruction. *Remote Sensing* 11, 13 (2019), 1550.

Andreas Krause and Daniel Golovin. 2014. Submodular function maximization.

Q. Kuang, J. Wu, J. Pan, and B. Zhou. 2020. Real-Time UAV Path Planning for Autonomous Urban Scene Reconstruction. In *Proc. of IEEE Int. Conf. on Robotics & Automation*. 1156–1162.

Steven M LaValle. 1998. Rapidly-exploring random trees: A new tool for path planning. (1998).

Massimo Mauro, Hayko Riemenschneider, Alberto Signoroni, Riccardo Leonardi, and Luc Van Gool. 2014. A Unified Framework for Content-aware View Selection and Planning Through View Importance. In *Proc. of British Machine Vision Conference*.

J. Maver and R. Bajcsy. 1993. Occlusions as a guide for planning the next view. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 15, 5 (1993), 417–433. https://doi.org/10.1109/34.211463

Oscar Mendez Maldonado, Simon Hadfield, Nicolas Pugeault, and Richard Bowden. 2016. Next-best Stereo: Extending Next Best View Optimization for Collaborative Sensors. In *Proc. of British Machine Vision Conference*.

Pierre Moreels and Pietro Perona. 2007. Evaluation of features detectors and descriptors based on 3D objects. *Int. J. of Computer Vision.* 73, 3 (2007), 263–284.

Christian Mostegel, Markus Rumpler, Friedrich Fraundorfer, and Horst Bischof. 2016. Uav-based autonomous image acquisition with multi-view stereo quality assurance by confidence prediction. In *Proc. of IEEE Conf. on Computer Vision & Pattern Recognition*. 1–10.

Kourosh Naderi, Joose Rajamäki, and Perttu Hämäläinen. 2015. RT-RRT* a real-time path planning algorithm based on RRT. In *Proceedings of the 8th ACM SIGGRAPH Conference on Motion in Games*. Association for Computing Machinery, New York, NY, USA, 113–118.

Lasse Damtoft Nielsen, Inkyung Sung, and Peter Nielsen. 2019. Convex decomposition for a coverage path planning for autonomous vehicles: Interior extension of edges. *Sensors* 19, 19 (2019), 4165.

Daryl Peralta, Joel Casimiro, Aldrin Michael Nilles, Justine Aletta Aguilar, Rowel Atienza, and Rhandley Cajote. 2020. Next-best view policy for 3D reconstruction. In *Proc. of Euro. Conf. on Computer Vision*. 558–573.

Mike Roberts, Debadeepta Dey, Anh Truong, Sudipta Sinha, Shital Shah, Ashish Kapoor, Pat Hanrahan, and Neel Joshi. 2017. Submodular Trajectory Optimization for Aerial 3D Scanning. In *Proc. of Int. Conf. on Computer Vision*.

Mike Roberts and Pat Hanrahan. 2016. Generating Dynamically Feasible Trajectories for Quadrotor Cameras. *ACM Trans. on Graphics (Proc. of SIGGRAPH)* 35, 4 (2016), 61:1–61:11.

Markus Rumpler, Arnold Irschara, and Horst Bischof. 2011. Multi-view stereo: Redundancy benefits for 3D reconstruction. In *Proc. of Workshop of the Austrian Association for Pattern Recognition*, Vol. 4.

Korbinian Schmid, Heiko Hirschmüller, Andreas Dömel, Iris Grixa, Michael Suppa, and Gerd Hirzinger. 2012. View planning for multi-view stereo 3D reconstruction using an autonomous multicopter. *J. of Intelligent & Robotic Systems* 65, 1-4 (2012), 309–323.

Lukas Schmid, Michael Pantic, Raghav Khanna, Lionel Ott, Roland Siegwart, and Juan Nieto. 2020. An Efficient Sampling-based Method for Online Informative Path Planning in Unknown Environments. *IEEE Robotics and Automation Letters* 5, 2 (2020), 1500–1507.

Steven M Seitz, Brian Curless, James Diebel, Daniel Scharstein, and Richard Szeliski. 2006. A Comparison and Evaluation of Multi-View Stereo Reconstruction Algorithms. In *Proc. of IEEE Conf. on Computer Vision & Pattern Recognition*. 519–528.

Magnus Selin, Mattias Tiger, Daniel Duberg, Fredrik Heintz, and Patric Jensfelt. 2019. Efficient autonomous exploration planning of large-scale 3-D environments. *IEEE Robotics and Automation Letters* 4, 2 (2019), 1699–1706.

Ruben M. Smelik, Tim Tutenel, Rafael Bidarra, and Bedrich Benes. 2014. A Survey on Procedural Modelling for Virtual Worlds. *Computer Graphics Forum* 33, 6 (2014).

Neil Smith, Nils Moehrle, Michael Goesele, and Wolfgang Heidrich. 2018. Aerial Path Planning for Urban Scene Reconstruction: A Continuous Optimization Method and Benchmark. *ACM Trans. on Graphics (Proc. of SIGGRAPH Asia)* 37, 6 (2018), 183:1–183:15.

Noah Snavely, Steven M. Seitz, and Richard Szeliski. 2006. Photo Tourism: Exploring Photo Collections in 3D. *ACM Trans. on Graphics (Proc. of SIGGRAPH)* 25, 3 (2006), 835–846.

Noah Snavely, Steven M. Seitz, and Richard Szeliski. 2008. Modeling the World from Internet Photo Collections. *Int. J. of Computer Vision.* 80 (2008), 189–210.

P. Whaite and F. P. Ferrie. 1997. Autonomous exploration: driven by uncertainty. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 19, 3 (1997), 193–205. https://doi.org/10.1109/34.584097

Christian Witting, Marius Fehr, Rik Bähnemann, Helen Oleynikova, and Roland Siegwart. 2018. History-aware autonomous exploration in confined environments using MAVs. In *Proc. of IEEE Int. Conf. on Intelligent Robots & Systems*. 1–9.

Ke Xie, Hao Yang, Shengqiu Huang, Dani Lischinski, Marc Christie, Kai Xu, Minglun Gong, Daniel Cohen-Or, and Hui Huang. 2018. Creating and Chaining Camera Moves for Quadrotor Videography. *ACM Trans. on Graphics (Proc. of SIGGRAPH)* 37, 4 (2018), 88:1–88:13.

Hao Yang, Ke Xie, Shengqiu Huang, and Hui Huang. 2018. Uncut Aerial Video via a Single Sketch. *Computer Graphics Forum (Proc. of Pacific Conf. on Computer Graphics & Applications)* 37, 7 (2018), 191–199.

Xiaohui Zhou, Ke Xie, Kai Huang, Yilin Liu, Yang Zhou, Minglun Gong, and Hui Huang. 2020. Offsite Aerial Path Planning for Efficient Urban Scene Reconstruction. *ACM Trans. on Graphics (Proc. of SIGGRAPH Asia)* 39, 6 (2020), 192:1–192:16.

A DETAILS ON D_i^j AND $\omega_i(h)$

For self-contained-ness, we give further details of D_i^j and $\omega_i(h)$ in Eq. (1), which are from [Roberts et al. 2017]. For disk D_i^j, its radius decays exponentially as the associated camera moves away from surface point s_i. Here, a cone is formed with D_i^j as its base and s_i as its tip, and D_i^j's radius is adjusted by the cone's apex half angle:

$$\theta_i^j = \theta_{\max}\, 2^{\frac{-\max(t_i^j - t_0, 0)}{t_{\text{half}}}}, \tag{7}$$

where $\theta_{\max}$ is the maximum apex half angle in radians; t_i^j is the distance from camera to s_i in meters; t_0 is a threshold distance in meters, below which θ_i^j does not change; and t_{half} is the decay half-life of the half angle in meters. In the implementation, we set $\theta_{max} = \frac{\pi}{90}$ radians, $t_0 = 20m$, and $t_{\text{half}} = 25m$. For weight function $\omega_i(h)$, it is set to have a cosine-weighted falloff, i.e., $\omega_j(h) = cos(\alpha_h)$, where α_h is the angle between the hemisphere pole and the vector from the hemisphere origin (s_i) to the hemisphere location h.