Image Acquisition for High Quality Architectural Reconstruction

Ojaswa Sharma* Nishima Arora† Himanshu Sagar‡

Indraprastha Institute of Information Technology - Delhi, India

ABSTRACT

In this work we propose a simple optimization based technique to compute camera poses for drone assisted automated image acquisition. We use this technique to create highly detailed 3D models of buildings using multi-view reconstruction. Our reconstructed models are great for use in virtual reality (VR) environments since they exhibit good amount of detail that is useful for creating realistic virtual walkthroughs. Creating a good 3D reconstruction with a set of nadir images is difficult since the vertical surfaces of buildings are not captured very well and are therefore not reconstructed accurately. Acquisition of non-nadir images require avoiding obstacles around the structure. Our technique is based on mathematical optimization, and is capable of calculating camera positions and orientations to maximally cover horizontal as well as vertical surface patches while avoiding obstacles around the building. We present a complete pipeline for a mostly automated and robust approach via a camera mounted quadcopter drone. We also validate our approach via a graphics-based simulation.

Keywords: Architectural reconstruction, UAV path planning, 3D multi-view reconstruction

Index Terms: Theory of computation—Discrete optimization; Human-centered computing—Virtual reality; Computing methodologies—Mesh geometry models

1 INTRODUCTION

Spatial data is central to any virtual reality (VR) system, and with the increasing popularity of VR based experiences, fast creation of high-fidelity geometric content has become a challenge. Such content is readily usable in 3D games, walkthroughs, and simulations. While it is usually created by 3D modelling artists, there is an increasing interest in semi-automatically synthesizing geometry. Such approaches can potentially create large 3D models in less time and with minimal human intervention. Modern ways to such modelling and creation primarily include terrestrial laser scanning (TLS), multi-view stereo (MVS) reconstruction, rule-based synthesis (procedural), and their variants [27]. Among these approaches, laser scanning provides highly detailed and accurate surface points, but is an expensive technology. Stereo reconstruction based methods are promising and recent scalable algorithms [13, 35] are capable of reconstructing large spaces. Procedural modelling algorithms are unique since they provide a way to accurately synthesize structures with internal details. Man-made structures like urban landscapes have been successfully created using rule-based grammer [11, 23, 26].

In stereo reconstruction, capturing correct geometry and appearance of an object requires that the images have good overlap and should have near orthogonal visibility of the object [19]. Large architectural structures are not very accessible for ground based image acquisition suited for a multi-view reconstruction. Such a manual acquisition process is not only time consuming, but may also not guarantee sufficient overlap for a correct reconstruction, and result in captured images with high perspective distortion. A convenient and popular choice has been aerial photogrammetric surveys that perform a horizontal flight to capture a set of nadir looking images. Such images capture detailed view of the region from top, but fail to cover the structures from sides. The resulting 3D reconstructions are not detailed enough for applications like a virtual walkthrough where the person moves near the ground surface. With the advent of camera mounted miniature UAV drones, it is now possible to manoeuvre into difficult spaces remotely and get novel views of structures that are otherwise very difficult to capture. This has opened-up a new area of exploration for optimal capturing of photographs to create 3D models that are detailed from all sides.

In this work we investigate a fast and practical approach for drone-based image capture planning. Such camera mounted drones usually have a tight budget for flight duration and number of waypoints. We propose computation of camera poses at various heights to ensure overlap and visibility for a highly detailed reconstruction. Our approach is well suited to capture near orthogonal images for 3D modelling based on discrete optimization of camera positions and orientations in presence of real-world obstacles.

2 RELATED WORK

2.1 Multi-view 3D Reconstruction

Multi-view reconstruction [17] is a widely used 3D reconstruction approach using photographs that capture an object from multiple viewpoints. Reconstruction from multiple views has two stages: *sparse reconstruction*, and *dense reconstruction*. In the sparse reconstruction stage, the algorithm takes a set of images covering an object from multiple viewpoints and performs an image matching operation to calculate common feature points between pairs of images. This process simultaneously optimizes external camera parameters and 3D positions of a sparse set of feature points. In dense reconstruction stage, the algorithm computes a dense set of 3D points as seen in multiple pairs of images. The dense point cloud is then triangulated into a 3D surface mesh and textured using the available image views. Recent advancements in precise texture mapping attempt to select best views for any mesh triangle to texture map with minimal illumination change and hide seams [12, 40]. Many open-source implementations like OpenMVG [24, 25], OpenMVS [2], COLMAP [35, 36], MVE [13], and CMPMVS [21] are able to perform large scale multi-view stereo reconstructions. Software like Open Drone Map [1], Pix4D Mapper [31], and Agisoft Photoscan [4] provide 3D reconstruction pipelines for drone based aerial surveys. Wolberg et al. [41] present a lightweight sketching based 3D modelling approach which utilizes structure from motion (SfM) along with precise geometry creation to generate textured 3D building models. For inaccurate 3D models, Aganj et al. [3] warp the images to fit the model. Camera poses computed during the sparse reconstruction stage of multi-view reconstruction can be used to texture map such models. Dellepiane et al. [9] propose an optical flow based approach to rectify local misalignments in texture mapping of 3D models during stereo reconstruction.

*e-mail: ojaswa@iiitd.ac.in

†e-mail: nishima@iiitd.ac.in

‡e-mail: himanshu14046@iiitd.ac.in

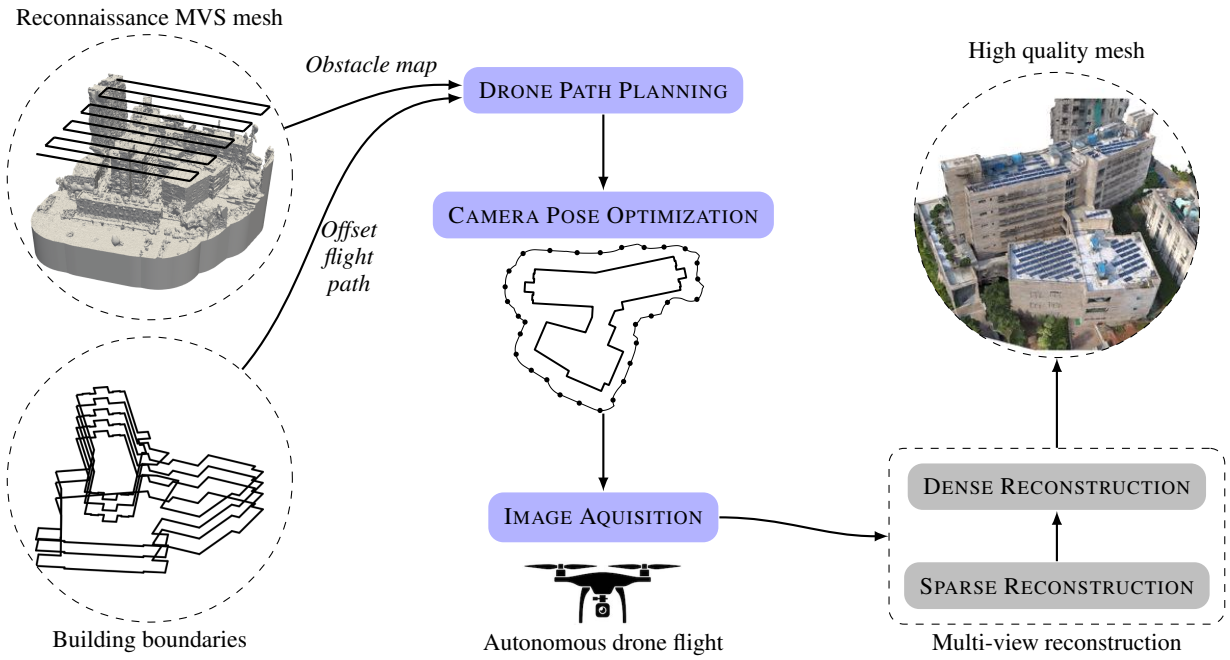

Figure 1: Building reconstruction workflow. We begin with a set of building boundaries for various levels, and a reconnaissance mesh created with multi-view stereo reconstruction from nadir images that serves as the set of static obstacles. Our optimization scheme solves for camera positions and orientations by maximizing coverage of surface patches while maintaining near orthogonal view to the nearby surface patches. These camera poses are used to capture images by a drone which are then processed by a standard multi-view reconstruction solver to give a textured mesh.

2.2 Image Capturing for Reconstruction

Multi-view stereo algorithms work for images with any orientations as long as the criteria for sufficient overlap, object coverage, and illumination are satisfied. Since it is usually difficult to estimate the coverage while photographing an urban landscape, a convenient choice is to acquire nadir images with the required overlap. This has the effect of covering the top of the landscape very well, but usually lacks coverage of vertical walls and overhangs. The resulting 3D reconstruction is thus an incomplete and inaccurate representation of the building. Ground based photography, on the other hand, is also not a viable solution since not all parts of an intricate building structure can be reached out to leading to photographs that only partially cover the entire building. This leads to the problem of *next best view* selection, which is extensively explored in the path planning and reconstruction literature [7, 30, 32]. The problem can be viewed in the context of types of sensors used and the planning required. In this work, we do not address the use of depth sensors, and restrict to a simple camera based sensing.

In a motorized positioning system for a structured light sensor arrangement, Fan et al. [28] provide an optimization based view selection and path planning algorithm for 3D object scanning. Recently camera mounted drones have been used for planning surveys that can take frontal photographs of buildings. Hoppe et al. [18] estimate optimal camera network by solving the Constraint Satisfaction Problem. Roberts et al. [34] present a submodular trajectory optimization for camera poses that avoids obstacles. The authors calculate optimal camera trajectory and orientations by maximizing coverage. A more recent approach by Smith et al. [37] proposes a continuous optimization approach of 5D camera poses. The authors suggest reconstructability heuristics and minimize an objective function for exploration of unseen regions in a scene while maximizing reconstructability heuristics. In our work, we take a different approach to optimizing camera positions and orientations. We select best camera views at any altitude while staying a certain fixed distance from the building walls to maximize coverage of the target surface. Our simplified approach is directly applicable to off-the-shelf drones with tight constraints (such as short flights and fixed number of waypoints), offers fast running times, and results in high-quality 3D reconstructions.

2.3 Obstacle Avoidance in Path Planning

In a real-life scenario, a building is usually surrounded by various obstacles including trees, poles and adjoining buildings. Path planning literature in robotics covers obstacle avoidance in great detail [8, 22]. The A* search algorithm [16] is one of the early algorithms successfully used for path planning on a raster grid. Voronoi diagram of empty space of obstacles results in a medial axis graph where a shortest path between any two points on the skeleton provides a clearance path [39]. Combined with the Delaunay Triangulations, these have been used to compute shortest paths in a polyhedral setting [10, 38]. Geraerts [14] introduces an Explicit Corridor Map constructed from the generalized Voronoi diagram to compute clearance paths for collision free motion in virtual environments. Álvarez et al. [5] apply the Fast Marching Method based for path finding in presence of obstacles. The authors employ a leader-followers scheme wherein the Eikonal equation governs time of arrival. The resulting path provides maximal clearance in narrow passages and in-between obstacles. We utilize this approach to conservatively replace potentially unsafe paths with clear routes for the drone.

3 Overview of Approach

Our reconstruction pipeline consists of computation of camera poses for height-wise optimal building surface coverage, drone assisted photography and 3D reconstruction. A drone with a gimbal mounted camera is used to photograph difficult parts of a building, which are otherwise unreachable from a ground camera. An initial flight path is computed from building boundaries which may be obtained either from building blueprints or cross-sections of reconnaissance mesh.

We address the challenge of manoeuvring the drone in difficult locations while carefully avoiding obstacles along its path. An overview of our approach is shown in Figure 1. The camera positions and orientations are computed via a discrete optimization process that maximizes total surface coverage while maintaining near orthogonal view of close by surfaces. The camera positions are generated on an obstacle-free path and we ensure that the drone remains on the same path even during flight legs in-between consecutive positions. The obstacles themselves are computed with multi-view stereo reconstruction from a reconnaissance survey of the region. Initially, the reconnaissance mesh is reconstructed from a set of nadir images and at the end a final mesh is reconstructed from a set images captured using the computed camera poses.

4 Drone Path Planning

Our automated image capturing approach employs a camera mounted quadcopter drone to take photographs of a building. We create floor-wise paths for the drone to traverse for photographing the building walls. The photographs are captured along an obstacle-free path looking at a wall. We deduce camera positions and orientations via an optimization that maximizes the total coverage such that any patch on a wall surface is seen by at least K cameras.

4.1 Building Boundaries

Building boundaries at various altitudes serve as an input for our algorithm to guide the drone. A flight mission for an altitude begins with a fixed starting point that determines the relative drone orientation for the first camera position. We carry out multiple such missions for different heights that cover the entire building vertically. The heights are calculated so as to keep a sufficient vertical overlap between photographs for a good reconstruction. Typically a minimum side-lap of 60% is recommended for multi-view stereo aerial surveys [29]. We compute the mission altitudes based on the desired vertical overlap between photographs (70% in our experiments) and the vertical coverage of the camera.

The building boundaries may be derived either from digital blueprints of the building or from horizontal cross-sections of the reconnaissance mesh. Here we use georeferenced digital blueprint boundaries as input. We create an offset flight path $\mathcal{P}_o$ by expanding a boundary by a distance d. This flight path ensures that the drone during its flight is at a distance d from the building walls. However, there might be obstacles on this path which we resolve in the next stage.

4.2 Obstacle-free Path

Path $\mathcal{P}_o$ can be sampled to generate camera positions if there are no obstacles around a building; however, in a real-life scenario buildings are surrounded by a multitude of obstacles including trees, poles, buildings, and other structures. While the current off-the-shelf drones are capable of obstacle avoidance, this is still in research and requires ideal operating conditions. In this work, we classify obstacles as static and kinematic and propose to handle static obstacles as part of the path planning stage. Since most of the objects fall under the static category, majority of the obstacle avoidance burden shifts to path planning. Therefore, we rely on a drone's built-in obstacle avoidance for kinematic objects and as a safety feature. We deduce an obstacle-free path $\mathcal{P}$ from the initial path $\mathcal{P}_o$ and a set of static obstacles detected at the flight altitude under consideration. These static obstacles are taken from a reconnaissance mesh. After this, an obstacle-free path is computed in 2D using the Fast marching Method [5].

A reconnaissance mesh of the surrounding area is computed from nadir aerial photographs using state-of-the-art multi-view stereo method. We capture these nadir photographs at a height of 70 m from the ground level at 4K resolution to get a high quality 3D reconstruction. Such a mesh captures all the buildings and static obstacles like trees and poles as blobs. This mesh is georeferenced using the geotagged photographs during reconstruction. It serves as a base mesh for all obstacle computations and registration of high quality reconstructions in later stages.

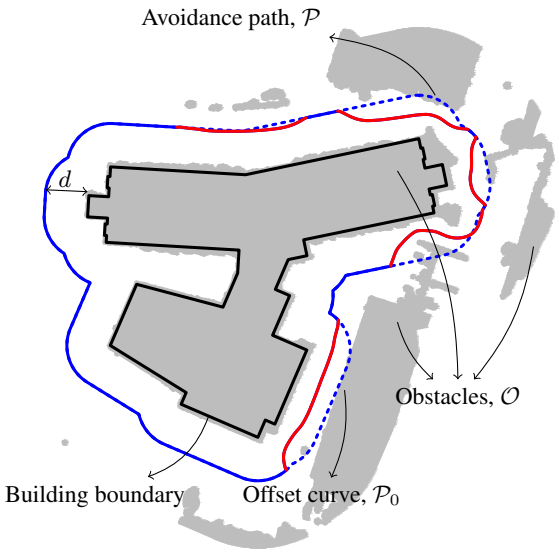

Figure 2: Avoidance path for drone flight. An offset curve $\mathcal{P}_0$ (shown in blue) maintains a safe horizontal distance d from the building boundary. Intersection with the obstacles detects unsafe segments (dashed blue) that are replaced with the safe ones. An obstacle-free path $\mathcal{P}$ consists of subpaths from $\mathcal{P}_0$ (shown in solid blue), and the computed obstacle-free subpaths (shown in solid red).

In order to construct a 2D obstacle map $\mathcal{O}$ at a given altitude h, we intersect the mesh with two planes at heights $(h-1)$ meter and $(h+1)$ meter. This is performed to take care of changes in height of upto ± 1 meters while the drone is airborne. An intersection in this case is a set of disconnected polygons (possibly concave). An obstacle map is created by taking a union of these two sets of polygons.

We now calculate subpaths in $\mathcal{P}_o$ that intersect with obstacles in $\mathcal{O}$. These subpaths are the minimum set of regions of $\mathcal{P}_o$ that must be replaced with obstacle-free pathways (shown as dashed blue subpaths in Figure 2). For the purpose of computing these subpaths, we dilate the obstacle map by a certain distance to take care of GPS positioning error in the drone. Usually this is about 2 meters in our experiments, thus we dilate the obstacle polygons by twice that distance. The starting and ending point of these subpaths are then used to calculate new subpaths using the Fast Marching method proposed by Álvarez et al. [5]. This approach uses a variant of the Fast Marching Method to search for a path amidst obstacles in a binary obstacle grid map. The search is performed by first creating a distance map and then computing the shortest distance via the gradient descent method. We split $\mathcal{P}_0$ into subpaths that are free from obstacles and the ones that are not. An overall safe path is reconstructed by replacing the unsafe subpaths by obstacle-free curve segments (shown as red subpaths in Figure 2). This path serves as an obstacle-free path $\mathcal{P}$, which by construction is maximally away from obstacles on both sides.

4.3 Computation of Camera Positions and Orientations

We use path $\mathcal{P}$ to sample optimal camera positions and compute orientations. These camera poses will be used to capture photographs for 3D reconstruction. Therefore, we would like to ensure that the entire surface at a particular height is covered well from these generated views. In addition, the drone hardware usually imposes

a limit on the number of waypoints that it can handle in a flight mission. In order to limit the number of points and also cover the area such that each patch on the surface is seen by N cameras, we solve an optimization problem whose solution gives us the desired positions and camera orientations.

Given a closed boundary $\mathcal{B}(t)$ at height h parameterized by curve length $t \in [0, \tau]$, we consider a set of N candidate camera positions $\{\mathbf{p}_1, \cdots, \mathbf{p}_N\}$ lying on the obstacle-free path $\mathcal{P}$ around the boundary. Let us also denote by $\mathcal{V}(\mathbf{p}_i, \mathcal{B}(t))$ the visibility function that returns a subset $\mathbf{t}_i = \cup_{j \in J}[t_i^{(j)}, t_i^{(j)}]$ of the parameter t for points on $\mathcal{B}(t)$ that are visible from camera position $\mathbf{p}_i$. Depending on the boundary shape and the camera pose, $\mathbf{t}_i$ may contain more than one disconnected ranges of t. In order to calculate optimal camera positions and orientations, we minimize the length of uncovered boundary as seen by the N cameras such that any point on the boundary is seen at least by K cameras. We choose the number of cameras N to be the average number of views needed for the boundary to be seen by K cameras as

$$N = \min\left(K\left\lceil\frac{\tau}{2d\tan(\theta/2)}\right\rceil, N_{max}\right),$$

where d is the offset distance between path $\mathcal{P}$ and boundary $\mathcal{B}$, θ is the vertical field of view of the camera (see Figure 2), and N_{max} is the maximum waypoint limit imposed by the drone hardware for a flight mission. In practice, since d does not remain constant and variably decreases at places of obstacles, we decide N based on $K + 1$ cameras rather than K to introduce some redundancy in the optimization. At the moment N is not a parameter of our optimization.

We consider the optimization problem of minimizing the uncovered length by calculating the total length deficit $\mathcal{U}$. Let us define the length of boundary which is seen by at least k cameras as

$$\mathcal{L}_k = \int_0^\tau \chi_k(t)\,\mathrm{d}t, \tag{1}$$

where, the characteristic function $\chi_k(t) : [0, \tau] \mapsto \{0, 1\}$ indicates if t appears in the intersection of any k distinct intervals out of the N intervals generated by the visibility function $\mathcal{V}$. Thus, for $k > 1$ we define

$$\chi_k(t) = \begin{cases} 1 & \text{if } t \in \bigcap_{i \in e} \mathbf{t}_i \text{ for at least an } e \in S_k, \\ 0 & \text{otherwise,} \end{cases} \tag{2}$$

where S_k is a set of all k-combinations of elements from the set $S = \{1, \cdots, N\}$, and

$$\chi_1(t) = \begin{cases} 1 & \text{if } t \in \mathbf{t}_i, i \in S, \\ 0 & \text{otherwise.} \end{cases} \tag{3}$$

In essence, $\chi_1(t)$ indicates if boundary point $\mathcal{B}(t)$ is seen at least by 1 camera, $\chi_2(t)$ indicates if boundary point $\mathcal{B}(t)$ is seen at least by 2 cameras, and so on.

We define *length deficit* for k-cameras as the length of boundary not seen at least by k cameras. The total length deficit $\mathcal{U}$ is then computed as the sum of length deficits for different number of camera visibilities, which can be written as

$$\mathcal{U} = \sum_{k=1}^{K}\left(\tau - \int_0^\tau \chi_k(t)\,\mathrm{d}t\right). \tag{4}$$

The visibility function $\mathcal{V}$ depends on the shape of the boundary $\mathcal{B}$ and has no analytical form. Therefore, we minimize $\mathcal{U}$ in a discrete setting. From a computational perspective, we subdivide the boundary $\mathcal{B}$ into n_b small patches of size Δt with patch centres at $\{t_1, \cdots, t_{n_b}\}$, and sample the path $\mathcal{P}$ into n_p positions $\{\mathbf{p}_1, \cdots, \mathbf{p}_{n_p}\}$. Since $\mathcal{V}$ is computationally expensive, we precompute the visibility between a point $\mathbf{p}_i$ and a boundary point $\mathcal{B}(t_j)$ in a matrix $\mathbf{V} \in \mathbb{R}^{n_p \times n_b}$ where $\mathbf{V}_{ij}$ is the cosine of angle between the vector $(\mathbf{p}_i - \mathcal{B}(t_j))$ and the normal vector $\mathbf{n}_j$ at the boundary point.

We first considered a different form of equation (4) without summation that considered length deficit for only single k value. In our experiments, optimising for a single k value led to skewness in the distribution of camera positions (these were not well distributed around the boundary and resulted in more oblique views). We noticed that optimising over all k values (i.e, with a summation) leads to a more uniform distribution of camera positions (thus giving us more frontal views). In both cases, the optimization objective was met, however the reconstruction results would vary due to nature of captured images (oblique vs. frontal).

We use genetic algorithm [15] to solve for camera positions and orientations that minimize coverage deficit $\mathcal{U}$ (or maximize coverage) where any boundary patch is to be seen by at least K cameras. The N positions are taken from the set of n_p sampled positions from $\mathcal{P}$. We impose lower and upper bound constraints on the camera orientations. Any camera orientation vector $\mathbf{v}_i$ will at most make an angle δ with the normal vector η_i to the boundary at camera position $\mathbf{p}_i$. That is to say, for $\phi = \cos^{-1}(\mathbf{v}_i \cdot \eta_i)$,

$$-\delta <= \phi <= \delta.$$

A frequency count of visibility (`true` if $\mathbf{V}_{ij} > 0$) along columns of $\mathbf{V}$ can be used to calculate χ_k and $\mathcal{U}$ during the optimization process. Our variable set for genetic optimization consists of parameters t and ϕ with their respective lower and upper bounds. We use a mutation strategy to randomly generate directions that are adaptive with respect to the last successful or unsuccessful generation. For the next generation, children are generated from two parents using a scattered crossover.

Figure 3 shows a set of camera positions and directions generated by our optimization procedure. We observed that about 200 generations were sufficient to reach the optima. Our termination criterion consisted of about 100 generations of insignificant change in the fitness function $\mathcal{U}$. It can be observed that our algorithm ensures visibility by K cameras for most of the boundary patches (~90% in the shown example), while the remaining patches are mostly interior and oblique. For these remaining patches, minimizing $\mathcal{U}$ amounts to optimizing visibility by $K - 1$ cameras, and so on. The same cannot be ensured if, for example, we sample the camera poses regularly along the obstacle-free path since the boundary may have intricate turns and folds that cannot be handled by a simple strategy.

4.3.1 Obstacle Avoidance between Waypoints

The above optimization results in camera positions sampled along an obstacle-free path. However, the drone moves in a straight line path in-between any two consecutive positions. The closed path $\mathcal{P}$ is usually concave in nature and therefore, a straight line path between any two points can intersect with nearby obstacles. Therefore, we check if the line segment connecting any two consecutive waypoints intersect with the obstacles in $\mathcal{O}$. We replace each such segment with an approximation of $\mathcal{L}$ between the segment endpoints. Such an approximation can be computed using a simplification algorithm [33] upto a given threshold distance from the original curve.

4.4 Roof Coverage

The roof of a building can be easily covered by a set of nadir images captured at a certain height. We follow a conservative strategy and compute camera positions only for points lying inside the roof polygon, which helps us keep the total number of waypoints well

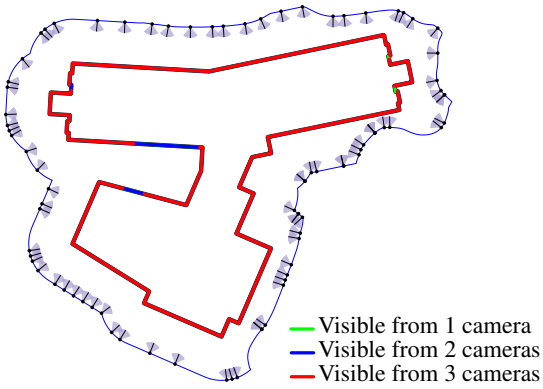

Figure 3: Camera pose optimization with a total of N cameras. Here, camera positions are shown as black dot and orientations as arrows. Our optimization distributes N cameras along the obstacle-free path $\mathcal{P}$ so as to maximise coverage while simultaneously keeping the viewing angle within limit.

within the hardware limitation of the drone. To reduce flight time, we alternate between the scan directions (ie., left to right, followed by right to left and so on). The camera heading is maintained to capture images aligned to the direction of motion of the drone (which we fix to be the larger principal axis of the roof polygon shape). Figure 4 shows computed waypoints for the roof polygon. In a building with varying height of the roof, we do not maintain a constant capturing height from the roof, however this can be easily incorporated in our pipeline. In the current setting, we consider the roof polygon to be the union of all floor boundary polygons. This gives us the largest roof area that needs to be covered from top.

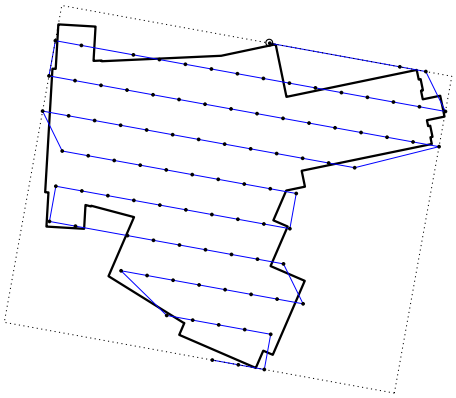

Figure 4: Waypoints for covering building roof (starting point is shown as an encircled dot). Principal axis of roof polygon is chosen as flight direction.

5 Image Capture and 3D Model Reconstruction

Once camera positions and orientations are computed, images are captured at multiple altitudes by flying the drone autonomously. We used the DJI Phantom 4 quadcopter drone for image acquisition which is equipped with a forward looking vision-based sensing system. We use the built-in obstacle avoidance as a secondary safeguard while the drone is airborne. We therefore keep the drone look forward (i.e, along the heading) during the entire flight in a mission. At any waypoint, we rotate the drone to align the camera with the computed view direction.

The camera captures geotagged images (i.e., they have GPS position in the EXIF metadata), however the positions are not very accurate (errors of about 2 to 5 meters are common in consumer grade GPS receiver). Furthermore, the altitude cannot be relied upon and has been observed to have much higher error (of about 10 to 20 meters). The sparse reconstruction stage of multi-view reconstruction followed by bundle adjustment procedure resolves correct positions and orientations of the cameras. A dense reconstruction followed by meshing and texturing results in a high quality 3D reconstructed model of the building.

6 Results

Our technique yields a set of camera poses at a set of altitudes. The computed poses ensure visibility of any building patch by at least 3 cameras at a given altitude. Our optimization framework allows for choosing the number K of cameras that directly see a patch (we set K to 3 in our experiments). In our computations, we subdivide boundary polygon into patches of 1 meter in length. The value of d is the distance of camera from the building surface, which is an important parameter in determining overall quality of reconstruction. A lower value of d will result in detailed close-up images that will enhance the quality of reconstruction, however it may not be feasible or otherwise safe to fly the drone arbitrarily close to the building due to low GPS accuracy and hardware limitations. In our experiments we considered flying at 8 meters from the boundary wall of a building to be safe, except along the regions where obstacles had to be resolved (where the distance could be less, but not closer than 2 meters from either the wall or the obstacles). The safe path $\mathcal{P}$ is also discretized into points every 1 meter for computation of the visibility matrix. The DJI Phantom 4 drone has a waypoint limit of 99 points for a mission and a maximum flight time of 20 minutes in a single battery charge. These two limitations led to the design choice of subdividing the entire building mission into floor-wise missions, where each mission does not exceed the limit in its number of camera waypoints. Number of camera waypoints is a parameter in our optimization which can be explicitly controlled. In the set of camera poses, we also insert a few dummy waypoints to avoid obstacles along the flight path between a pair of consecutive camera positions (as discussed in Sect. 4.3.1). In our experiments, we chose a vertical overlap of 70% between photographs and compute mission altitudes accordingly. An additional horizontal mission covers the roof as discussed in Sect. 4.4. Figure 5 shows a set of camera poses for all the missions of a survey.

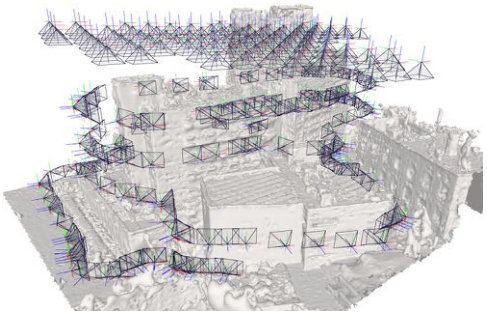

Figure 5: Computed camera poses for roof and 4 horizontal missions.

We conducted a reconnaissance survey of the entire campus area in a horizontal flight at an altitude of 70 meter from the ground and captured photographs with 70% overlap in a grid layout. The reconstructed georeferenced mesh serves as a base mesh for us and we use this to compute obstacles and to perform comparison with our 3D reconstructions. We show results of our reconstruction for two buildings: *academic* and *residence*. The academic building has an irregular boundary and is surrounded by a number of obstacles (trees, buildings, and construction structures). The total height of

Table 1: Running times (in sec.) for mission planning.

	Mission # (waypoints)	*Obstacle map*	*Obstacle-free path*	*Pose optimization*	*Overall*
Academic	1 (47)	48.8	8.1	48.9	105.8
	2 (82)	35.2	6.7	164.8	206.7
	3 (53)	30.9	4.3	40.2	75.4
	4 (53)	30.1	3.7	37.3	71.1
Residence	1 (31)	20.8	6.3	39.0	66.1
	2 (29)	17.7	5.3	33.0	56.0
	3 (45)	22.3	2.2	27.8	52.3
	4 (45)	16.7	2.1	27.5	46.3
	5 (45)	12.5	2.1	27.3	41.9
	6 (45)	21.0	1.7	28.9	51.6
	7 (41)	12.2	1.2	23.0	36.4

Table 2: Running times (in hours) for 3D reconstructions.

Building	*Sparse*	*Densify*	*Mesh*	*Texture*	*Overall*
Academic	1.29	1.25	0.33	3.96	6.83
Residence	0.22	0.73	0.22	0.82	1.99

this building is about 32 meters from ground level. A total of four horizontal missions covered the building from sides, while one roof mission covered the top. This is a complicated building to survey given a number of large obstacles very close to the rear half of the building. Many of such obstacles were from ongoing construction activity (e.g. barricades, and construction frames). Under low GPS precision, the drone relies heavily on its vision based avoidance for manoeuvring in between tight spaces. Figure 6 shows a comparison of the base mesh with our reconstruction. The geometrical and textural enhancements resulting from using our camera poses is clearly visible in the close-up views. Residence is a 45 meter high building with 12 storeys and has multiple apartments on every floor. We used a total of seven horizontal missions and one for the roof to cover the entire building. This building was fairly accessible except for an ongoing construction of an adjacent tall building with a minimum clearance of about 7 meters in-between. We observed that the 3D reconstruction from nadir images is particularly bad near the top. With optimal coverage, our reconstruction does not suffer from this problem and provides a much detailed textured mesh as shown in Figure 7. These reconstructed 3D models are very well suited for creating high quality virtual reality walkthroughs.

Our entire pipeline is written in Matlab and uses the open-source gptoolbox [20] library for geometry processing, and path planning toolkit from Álvarez et al. [5]. The drone controller is written in Java as an Android application using the DJI mobile SDK. We use open-source libraries OpenMVG [24, 25] and OpenMVS [2] for multi-view reconstruction. Our optimization code is composed of three stages: computation of obstacle maps from mesh, creating obstacle-free path for drone, and optimizing camera poses along the path. Running times for these stages are shown in Table 1 along with mission details for both buildings. Our computation of camera poses and waypoint missions takes a few minutes on a computer with 2.4 GHz Intel Xeon processor, 64 GB memory, and Nvidia Tesla K40 GPU. Multi-view reconstructions for these buildings took a few hours as shown in Table 2. The source code for our implementation is publicly available for research purpose at `https://github.com/ojaswa/DroneAutoCapture`.

6.1 Evaluation

In order to evaluate improvement in quality with our approach, we reconstruct a 3D digital building model and compare the accuracy of reconstruction. A digital 3D model $\mathcal{M}_{GT}$ serves as ground truth for comparison (which otherwise is difficult to obtain for a real building). Our approach to evaluation is to first use a 3D digital building model to create waypoints for both nadir camera poses and optimized camera poses. We then create a scene in the Unity game engine and capture images from a virtual camera within the game level using scripting capabilities of the game engine. The captured images are then used to reconstruct both the nadir model $\mathcal{M}_{Nadir}$ and the optimal model $\mathcal{M}_{Optimal}$.

Building boundaries are generated by slicing the digital model with a set of planes at certain heights to cover the building vertically. The heights are calculated to consider 70% vertical overlap between successive levels. We match the field of view of actual camera and the hardware limits of the quadcopter drone. Figure 8 shows the digital model, and the two reconstructions (nadir and optimal). The bottom row shows zoomed-in parts of the meshes which indicates the quality of resulting surface in both reconstructions.

For a qualitative measure to compare the reconstructed meshes, we use the Hausdorff distance [6] that indicates how closely the points on boundaries of two shapes match. The Housdorff distance between two shapes S and S' is given by

$$d_H(S, S') = \sup_{x \in S} \inf_{x' \in S'} d(x, x'),$$

where $d(\cdot, \cdot)$ is a distance metric. In our comparisons, we use the mean and RMS Hausdorff distance d_H (with Euclidean distance metric) computed w.r.t to the reference mesh $\mathcal{M}_{GT}$. A lower value of d_H (close to zero) indicates a better reconstructed surface (see Table 3). To understand the magnitude of error in reconstruction, the table also includes length of bounding box (BBox) diagonal of the concerned meshes. Further, we also show per-vertex error of the reconstructed surface mesh as distance measured from the corresponding point (projected) on the original surface mesh in Figure 9. The accompanying histograms show distribution of these errors, where the Y-axis represents error values (in absolute mesh distance units) while the X-axis has the frequency. A wider error distribution of $\mathcal{M}_{Nadir}$ indicates that the mesh deviates more from the digital model compared to $\mathcal{M}_{Optimal}$. This is further supported by a higher Hausdorff distance value for the nadir mesh.

Table 3: Reconstruction error between surfaces measured using mesh Hausdorff distance.

Reconstruction	*BBox diagonal*	*Mean d_H*	*RMS d_H*
Nadir mesh	647.229919	0.719164	1.053784
Optimal mesh	649.962341	0.358802	0.724885

Given that more details are visible in images taken from closer distances, some spurious blobs appear in $\mathcal{M}_{Optimal}$ (these are colored as high error regions in Figure 9). We observed that these blobs appear in areas where a sharp or thin feature is photographed against sky or a flat surface. Such features are more prominent in close-up views rather than in long-distance or nadir views and are picked up by the stereo reconstruction pipeline. In our experiments, such geometry is generated when there are multiple small objects clustered in the scene. A correct reconstruction in such regions may require an even closer photography. Camera distance from building surface is a parameter in our algorithm and may be changed based on the application domain requirements.

6.2 Differences with Related Approaches

The two closely related approaches with ours are recent works by Roberts et al. [34] and Smith et al. [37]. These approaches use path planning and view selection for autonomous aerial image capture using a drone. Both methods (like ours) rely on an approximate mesh of the scene to compute a set of optimized views. Likewise,

(a) (b) (c) (d)

Figure 6: Reconstruction results on academic building that has an irregular shape with a number of obstacles around. (a) Reconnaissance mesh reconstructed from a set of nadir images, (c) Our mesh reconstructed from optimized camera positions and orientations, (b) and (d) Close-up views of different areas show geometric and radiometric quality of the two meshes.

(a) (b) (c) (d)

Figure 7: Reconstruction results on residence building that has 12 storeys, (a) Reconnaissance mesh reconstructed from a set of nadir images show loss of geometric details at the top, (c) Our mesh reconstructed from optimized camera positions and orientations, (b) and (d) Close-up views of different areas show geometric and radiometric quality of the two meshes.

all of these methods take into consideration hardware constraints like flight time and battery life.

The method of Roberts et al. [34] proposes a coverage model to pick most useful camera trajectories to yield a high-quality 3D reconstruction. The algorithm first discretizes a set of camera positions and then calculates an optimal orientation for every position that maximizes coverage. The trajectory selection problem is then solved as an integer linear optimization utilizing the submodularity of the coverage function. The candidate camera positions in this method are restricted to lie on uniformly sampled points from the scene bounding box within the permissible free space above the scene. This helps the authors in limiting the possible choices for camera positions for optimization. For the examples shown, the grid size is chosen between 3.5 m - 4.5 m. Given that the target shape could be concave, at such large distances it becomes important to handle obstacles that may come in between two successive camera positions on an optimal trajectory, however the method does not discuss this explicitly. In contrast, our method allows for a smaller change in camera positions (around 1m or less) which enables finer position computation albeit only at the concerned altitude. Also, we appropriately handle the case of obstacles falling in-between successive camera positions as discussed in section 4.3.1 since we always move the drone (similar to [34]) in a straight line path in-between waypoints.

The approach of Smith et al. [37] incorporates both novelty of a camera view and matchability in their reconstructability heuristic. The authors present a comprehensive comparison and benchmarking on both real and synthetic datasets. In their method, the camera pose estimation and path planning is modeled as a continuous optimization problem. A set of regularly spaced overhead camera poses are used to reconstruct a height map for free space calculation. This is motivated by the fact that nadir views will only partially reconstruct unseen structures. However, a height map cannot represent overhanging structures which the authors have mentioned as a limitation of their method. Further, a trajectory is modeled as a smooth path by fitting B-spline curves between waypoints that could potentially violate the safety margin for obstacles. Since our method utilizes horizontal cross-sections (or floor plans), it is possible to handle buildings with changing floor areas along the height.

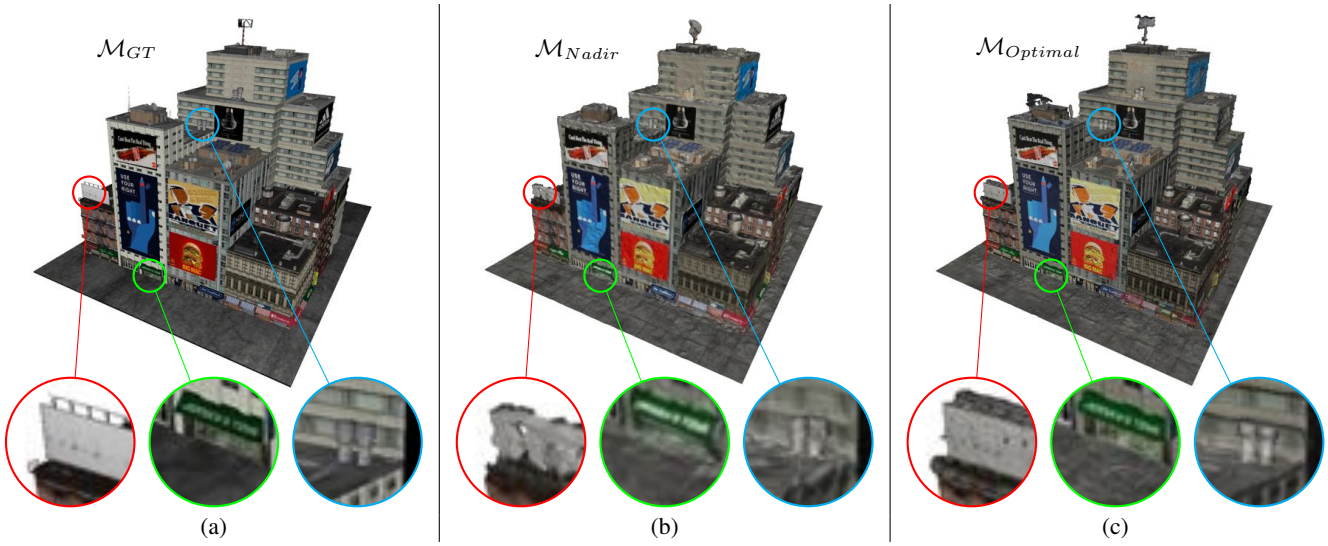

Figure 8: Reconstruction evaluation using 3D model $\mathcal{M}_{GT}$. (a) This model serves as ground truth for evaluating reconstruction quality. (b) A classical reconstruction approach using nadir photographs resulting in mesh $\mathcal{M}_{Nadir}$. (c) Our approach uses pose optimization resulting in mesh $\mathcal{M}_{Optimal}$. The closeups show improvement in quality in terms of structure and appearance.

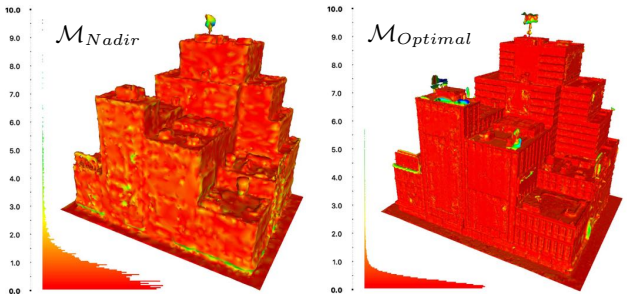

Figure 9: Per-vertex errors expressed as distance from the reference 3D digital model M_{GT}. The errors are shown for nadir and optimal reconstructions to indicate deviation from the reference surface. The histogram on left shows a colormap as well as area-weighted distribution of these errors that indicate spread and extent of the deviations.

6.3 Discussion and Challenges

Our approach to image capture considers several constraints imposed by the quadcopter hardware. The design choice to split the entire survey into multiple horizontal contours takes into account the limited battery life (maximum of 20 minutes per flight) and the upper limit of 99 waypoints per mission. Further, the choice of using blueprints is motivated by the fact that these are usually available for most modern buildings. In a case when blueprints are unavailable, our approach will work just fine by using cross sections from a reconnaissance mesh, however that will require identification of target building surface. The definitive advantage with architectural blueprints is that they provide with precise building boundaries to create an offset path for the drone.

We compute precise GPS positions and orientations of the camera during the optimization process, however the actual camera parameters at the time of image acquisition differ. Various factors like horizontal GPS error (which is usually 2-5 meters under good conditions) and wind introduce deviations in both the position and orientations. We find that the strategy of optimizing poses such that 3 or more cameras cover any surface patch introduces redundancy which is helpful in hiding errors arising from sources beyond control. Certain high-end drones do support RTK GPS that can give a better positional accuracy (of up to 5 cm). Factors like wind also contribute to fast battery discharge during the flight. We also observed that in practice, it is much easier to fly if clearance between obstacles and the building is more than double the horizontal GPS error. The built-in vision system of drones is still experimental and cannot be always relied upon.

7 Conclusion

In this work, we have explored how detailed 3D building models can be captured using an optimization framework that ensures maximal coverage of the building surface at any altitude while maintaining near orthogonal viewing angles. Our approach works in presence of multiple obstacles and calculates precise drone flight path. Since we consider real obstacles in our technique, our drone flight paths are safe and reliable. Our approach is faster than existing methods and is applicable to resource constrained drones, which is more practical in the field.

Acknowledgments

This research was supported by the Science and Engineering Research Board (SERB) of the Department of Science and Technology (DST) of India (Grant No. ECR/2015/000006).

References

[1] Open Drone Map. `http://opendronemap.org`.

[2] OpenMVS. Open multi-view stereo reconstruction library. `http://cdcseacave.github.io/openMVS`.

[3] E. Aganj, P. Monasse, and R. Keriven. Multi-view texturing of imprecise mesh. In *Asian Conference on Computer Vision*, pp. 468–476. Springer, 2009.

[4] Agisoft. Agisoft Photoscan. `http://www.agisoft.com/`.

[5] D. Álvarez, J. V. Gómez, S. Garrido, and L. Moreno. 3D robot formations path planning with fast marching square. *Journal of Intelligent & Robotic Systems*, 80(3):507–523, 2015.

[6] N. Aspert, D. Santa-Cruz, and T. Ebrahimi. MESH: Measuring errors between surfaces using the Hausdorff distance. In *IEEE International Conference on Multimedia and Expo*, vol. 1, pp. 705–708, 2002.

[7] J. E. Banta, L. Wong, C. Dumont, and M. A. Abidi. A next-best-view system for autonomous 3D object reconstruction. *IEEE Transac-*

tions on Systems, Man, and Cybernetics-Part A: Systems and Humans, 30(5):589–598, 2000.

[8] H. M. Choset. *Principles of robot motion: theory, algorithms, and implementation*. MIT press, 2005.

[9] M. Dellepiane, R. Marroquim, M. Callieri, P. Cignoni, and R. Scopigno. Flow-based local optimization for image-to-geometry projection. *IEEE Transactions on Visualization and Computer Graphics*, 18(3):463–474, 2012.

[10] D. Demyen and M. Buro. Efficient triangulation-based pathfinding. In *AAAI*, vol. 6, pp. 942–947, 2006.

[11] B. Domínguez, Á. García, and F. Feito. Semiautomatic detection of floor topology from cad architectural drawings. *Computer-Aided Design*, 44(5):367 – 378, 2012.

[12] M. Eisemann, B. De Decker, M. Magnor, P. Bekaert, E. De Aguiar, N. Ahmed, C. Theobalt, and A. Sellent. Floating textures. *Computer Graphics Forum*, 27(2):409–418, 2008.

[13] S. Fuhrmann, F. Langguth, N. Moehrle, M. Waechter, and M. Goesele. MVE–an image-based reconstruction environment. *Computers & Graphics*, 53:44–53, 2015.

[14] R. Geraerts. Planning short paths with clearance using explicit corridors. In *Robotics and Automation (ICRA), 2010 IEEE International Conference on*, pp. 1997–2004. IEEE, 2010.

[15] D. E. Goldberg. *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1st ed., 1989.

[16] P. E. Hart, N. J. Nilsson, and B. Raphael. A formal basis for the heuristic determination of minimum cost paths. *IEEE transactions on Systems Science and Cybernetics*, 4(2):100–107, 1968.

[17] R. Hartley and A. Zisserman. *Multiple view geometry in computer vision*. Cambridge university press, 2003.

[18] C. Hoppe, A. Wendel, S. Zollmann, K. Pirker, A. Irschara, H. Bischof, and S. Kluckner. Photogrammetric camera network design for micro aerial vehicles. In *Computer vision winter workshop (CVWW)*, vol. 8, pp. 1–3, 2012.

[19] A. Hornung, B. Zeng, and L. Kobbelt. Image selection for improved multi-view stereo. In *IEEE Conference on Computer Vision and Pattern Recognition, 2008.*, pp. 1–8, 2008.

[20] A. Jacobson et al. gptoolbox: Geometry Processing Toolbox, 2016. http://github.com/alecjacobson/gptoolbox.

[21] M. Jancosek and T. Pajdla. Multi-view reconstruction preserving weakly-supported surfaces. In *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3121–3128, 2011.

[22] S. M. LaValle. *Planning algorithms*. Cambridge university press, 2006.

[23] R. Lewis and C. Séquin. Generation of 3D building models from 2D architectural plans. *Computer-Aided Design*, 30(10):765 – 779, 1998.

[24] P. Moulon, P. Monasse, and R. Marlet. Global fusion of relative motions for robust, accurate and scalable structure from motion. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 3248–3255, 2013.

[25] P. Moulon, P. Monasse, R. Marlet, and Others. OpenMVG. An open multiple view geometry library. `https://github.com/openMVG/openMVG`.

[26] P. Müller, P. Wonka, S. Haegler, A. Ulmer, and L. Van Gool. Procedural modeling of buildings. *ACM Trans. Graph.*, 25(3):614–623, 2006.

[27] P. Musialski, P. Wonka, D. G. Aliaga, M. Wimmer, L. v. Gool, and W. Purgathofer. A survey of urban reconstruction. In *Computer graphics forum*, vol. 32, pp. 146–177. Wiley Online Library, 2013.

[28] Fan, Xinyi and Zhang, Linguang and Brown, Benedict and Rusinkiewicz, Szymon. Automated view and path planning for scalable multi-object 3D scanning. *ACM Transactions on Graphics (TOG)*, 35(6):239, 2016.

[29] Leberl, F and Irschara, A and Pock, T and Meixner, P and Gruber, M and Scholz, S and Wiechert, A. Point Clouds: Lidar versus 3D Vision. *Photogrammetric Engineering & Remote Sensing*, 76(10):1123–1134, 2010.

[30] R. Pito. A solution to the next best view problem for automated surface acquisition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21(10):1016–1030, 1999.

[31] Pix4D. Pix4D Mapper. `http://www.pix4d.com`.

[32] C. Potthast and G. S. Sukhatme. A probabilistic framework for next best view estimation in a cluttered environment. *Journal of Visual Communication and Image Representation*, 25(1):148–164, 2014.

[33] U. Ramer. An iterative procedure for the polygonal approximation of plane curves. *Computer graphics and image processing*, 1(3):244–256, 1972.

[34] M. Roberts, D. Dey, A. Truong, S. Sinha, S. Shah, A. Kapoor, P. Hanrahan, and N. Joshi. Submodular trajectory optimization for aerial 3D scanning. In *International Conference on Computer Vision (ICCV)*, 2017.

[35] J. L. Schönberger and J.-M. Frahm. Structure-from-motion revisited. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[36] J. L. Schönberger, E. Zheng, M. Pollefeys, and J.-M. Frahm. Pixelwise view selection for unstructured multi-view stereo. In *European Conference on Computer Vision (ECCV)*, 2016.

[37] N. Smith, N. Moehrle, M. Goesele, and W. Heidrich. Aerial path planning for urban scene reconstruction: a continuous optimization method and benchmark. In *SIGGRAPH Asia 2018 Technical Papers*, p. 183. ACM, 2018.

[38] J. A. Storer and J. H. Reif. Shortest paths in the plane with polygonal obstacles. *Journal of the ACM (JACM)*, 41(5):982–1012, 1994.

[39] O. Takahashi and R. J. Schilling. Motion planning in a plane using generalized voronoi diagrams. *IEEE Transactions on robotics and automation*, 5(2):143–150, 1989.

[40] M. Waechter, N. Moehrle, and M. Goesele. Let there be color! large-scale texturing of 3D reconstructions. In *European Conference on Computer Vision*, pp. 836–850. Springer, 2014.

[41] G. Wolberg and S. Zokai. Photosketch: a photocentric urban 3D modeling system. *The Visual Computer*, pp. 1–12, 2017.