Aerial Path Planning for Urban Scene Reconstruction: A Continuous Optimization Method and Benchmark

NEIL SMITH, King Abdullah University of Science and Technology
NILS MOEHRLE, TU Darmstadt
MICHAEL GOESELE, TU Darmstadt
WOLFGANG HEIDRICH, King Abdullah University of Science and Technology

Fig. 1. Proposed method and results of aerial path planning. We first compute an optimal aerial flight trajectory from an initially reconstructed nadir capture. A safe airspace (everything outside the purple region) in the lower left subfigure is created to allow the flight trajectory to descend much closer without collision. In the middle column we show the dense reconstruction of one of the field tests and on the right close ups of two of the reconstructed townhouses.

Small unmanned aerial vehicles (UAVs) are ideal capturing devices for high-resolution urban 3D reconstructions using multi-view stereo. Nevertheless, practical considerations such as safety usually mean that access to the scan target is often only available for a short amount of time, especially in urban environments. It therefore becomes crucial to perform both view and path planning to minimize flight time while ensuring complete and accurate reconstructions.

In this work, we address the challenge of automatic view and path planning for UAV-based aerial imaging with the goal of urban reconstruction from multi-view stereo. To this end, we develop a novel continuous optimization approach using heuristics for multi-view stereo reconstruction quality and apply it to the problem of path planning. Even for large scan areas, our method generates paths in only a few minutes, and is therefore ideally suited for deployment in the field.

To evaluate our method, we introduce and describe a detailed benchmark dataset for UAV path planning in urban environments which can also be used to evaluate future research efforts on this topic. Using this dataset and both synthetic and real data, we demonstrate survey-grade urban reconstructions with ground resolutions of 1 cm or better on large areas (30 000 m^2).

CCS Concepts: • **Computing methodologies** → **Motion path planning**; *Vision for robotics*; *3D imaging*; *Reconstruction*;

Additional Key Words and Phrases: UAV Path Planning, 3D Scanning, Urban Scene Reconstruction

ACM Reference format:
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. Graph.* 37, 6, Article 183 (November 2018), 15 pages.
https://doi.org/10.1145/3272127.3275010

1 INTRODUCTION

Ongoing advancements in unmanned aerial vehicles (UAVs) coupled with large-scale scene reconstruction are pushing the boundaries of what is possible in photogrammetric surveying and mapping. Compared to manned planes or satellite imagery, UAVs are much more affordable, easy to deploy, agile, and able to fly at low altitudes, giving them the ability to capture complex environments with both higher resolution and denser coverage than ever before.

In order to sufficiently capture large scene reconstructions, we find it necessary to improve on the state-of-art approaches in flight trajectory planning. Most UAV-based 3D scanning is currently performed either under manual control, or using simple, pre-programmed flight paths with the camera pointing in a fixed direction, making it hard to achieve complete and dense coverage over large urban environments. Capturing the many vertical faces of architectural buildings requires multiple passes of the UAV with the camera set to an oblique angle. Given the complexity of urban environments, capturing with a single oblique camera angle along a given flight path may not even be sufficient. Once holes or undersampled areas are discovered after hours of processing, it is often not possible to revisit the site and to conduct additional flights. Finally, performing

a (hopefully) exhaustive approach with multiple passes at different angles is typically not possible for practical reasons during capture (e.g., disruption of residents, cost of time in the field) or reconstruction (long processing times). We present an optimized camera path and view planning system that minimizes flight time while achieving complete and dense coverage (see Figure 1). Beyond following best practices in aerial capture and overlap (see, e.g. [Roberts et al. 2017]), we develop a novel reconstruction heuristic that takes into consideration criteria such as visibility, sample density, and obliqueness of views. We capture a nadir flyover at high altitude (100m) in only a couple minutes and extract from the small sample of images a geometric scene proxy and flyable airspace. We create an initial camera network of uniformly distributed views over the geometric scene proxy and then continuously optimize the views towards a predefined minimal reconstructability for all samples, yielding more accurate and complete reconstructions using significantly fewer images. The reconstruction metric is GPU accelerated allowing for fast iteration to an optimized flight trajectory that can even be performed in the field where time for flight is very limited. Our approach is able to scale path planning for survey-grade urban reconstruction with ground resolutions of 1 cm or better on large areas (30 000 m^2).

We perform extensive qualitative evaluations of our path planning approach using real UAVs in urban environments. In order to quantitatively evaluate our continuous optimization method, we design a new synthetic benchmark for image based reconstruction of large urban scenes. Although many benchmarks exist to examine SfM/MVS pipelines, they are either focused on indoor environments, or on those parts of exterior environments that are accessible to terrestrial laser scanning. In our work, we focus on full aerial capture of large urban or residential scenes. In these environments, it is usually not legally possible to perform repeated experiments with different flight paths, and even if this were possible, we would not have ground truth geometry to compare the reconstruction results for different flight paths. We instead propose using highly detailed 3D building models based off real-world locations and rendering textured images as input to the pipelines. Multiple geometrically complex buildings can be placed within an urban scene allowing for complex flight trajectories to be examined.

In this paper, we introduce a novel approach to flight trajectory planning validated using both a new benchmark and extensive real-world flight experiments. We provide the benchmark as a tool for experimentation and further development along with all the tools to generate new flights, reconstructions, and evaluation of flight trajectory performance. Our contributions are:

- an efficient and effective reconstructability heuristic and continuous GPU-accelerated optimization for view and path planning, which is fast enough for deployment in the field;
- an overall reconstruction approach validated with extensive synthetic and real-world tests that can achieve accurate and complete reconstruction using significantly fewer images;
- and a benchmark for evaluating image based reconstruction of geometrically complex and high variety urban and residential building scenes.

2 RELATED WORK

2.1 View Selection

The selection of optimal views for image-based scene reconstruction has been studied in different scenarios within the visual computing, photogrammetry, and robotics communities. A majority of this work within visual computing has primarily focused on heuristics to determine minimum view sets as well as best views for achieving a full reconstruction at high computational efficiency [Beder and Steffen 2006; Furukawa et al. 2010; Goesele et al. 2007; Mendez et al. 2016; Moreels and Perona 2007; Rumpler et al. 2011, 2016; Snavely et al. 2008]. Recently, Mendez *et al.* [2016] showed the importance of view selection by achieving state-of-the-art 3D reconstruction results in the Middlebury benchmark [Seitz et al. 2006] with a fraction of the images by incrementally choosing views with a good relation between baseline and convergence (absolute parallax). Fan *et al.* [2016] show advantages of path planning for table top 3D scanning and Wu *et al.* [2014] use NBV to improve 3D scanning using mobile robots (see also [Liu et al. 2018]). Goesele *et al.* [2007] select views based on the parallax angle between shared features, however, their heuristic further considers scale differences and they refine the selection on a per pixel level based on both triangulation angle and image content.

The robotics community has studied online exploration and reconstruction using UAVS with mounted cameras. This work focuses on sparse reconstruction of environments in real-time with *next best view* (NBV) planning as part of simultaneous localization and mapping (SLAM) [Dunn and Frahm 2009; Forster et al. 2014; Hepp et al. 2017; Leonard and Durrant-Whyte 1991; Liu et al. 2018; Mostegel et al. 2016; Vasquez-Gomez et al. 2014; Wenhardt et al. 2007]. Since these approaches rely on low resolution, high frame rate or RGBD cameras they must be close-range to achieve acceptable ground sample distance (GSD) and still are plagued with high noise in their reconstructions.

Photogrammetry traditionally approaches optimal view selection in an offline scenario with prior scene knowledge (*view planning*) [Alsadik et al. 2013; Hoppe et al. 2012; Olague and Mohr 2002; Pfeifer et al. 2012; Schmid et al. 2012]. In recent years UAVs have been more extensively used [Mostegel et al. 2016; Wendel et al. 2012]. A main focus has been on optimization of UAV videography [Galvane et al. 2017; Gebhardt et al. 2018; Roberts and Hanrahan 2016; Xie et al. 2018].

There are also several closely related works in aerial path planning for 3D scanning. Schmid *et al.* [2012], Hepp *et al.* [2017] and Roberts *et al.* [2017] calculate a surface of the scene after an initial capture, then generate candidate views that look along the surface normals. They take slightly different approaches to select next best views that increase coverage and exhibit a novel perspective (i.e. a different observation angle). Hoppe *et al.* [2012] capture a coarse mesh, then create a candidate view for each triangle that is fronto-parallel to the triangle and at the desired GSD. They iteratively select the view that observes the most triangles at a novel angle. Their approach models matchability by considering angles with respect to the surface but not directly with respect to other views. Improving on this early work, Hepp *et al.* [2017] and Roberts *et al.* [2017] employ submodularity to candidate view selection. They

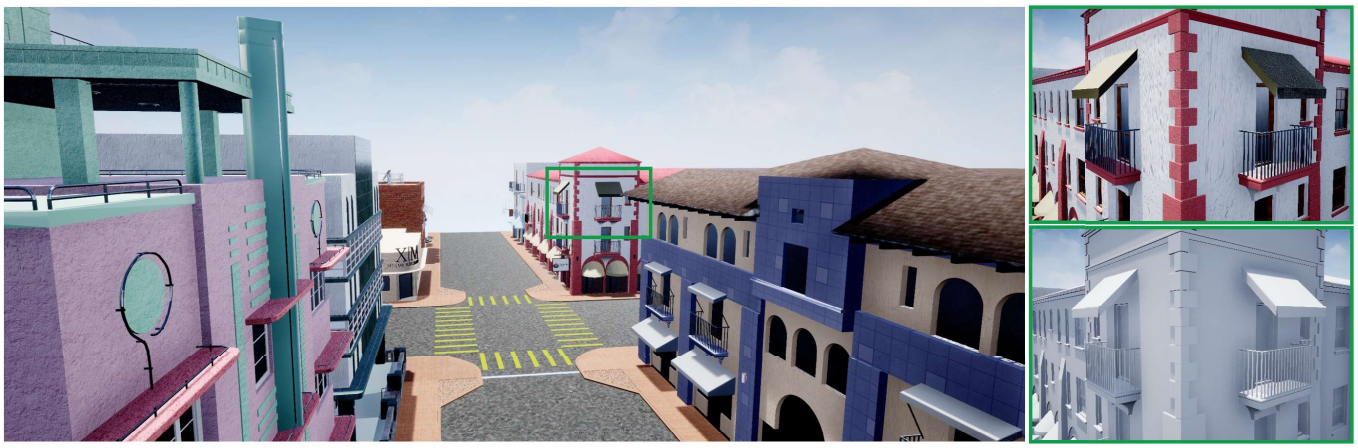


Fig. 2. CA-1 Dataset with natural looking colored textures and detailed close-ups of underlying geometry.

achieve a greater reward than traditional next best view and p-SPIEL Orienteering [Singh et al. 2009] implementations, and in less time. Similar to our work, Roberts *et al.* [2017] consider the parallax angle to determine the novelty of a view but unlike our approach not the matchability. Huang *et al.*'s [2017] approach is the first to reconstruct an online coarse proxy mesh from which a NBV path can be constructed. Below we compare our path planning approach to Roberts *et al.* [2017] and other NBV algorithms.

2.2 Scene Reconstruction Benchmarks

Several existing benchmarks evaluate image based scene reconstruction [Aanæs et al. 2016; Dai et al. 2017; Knapitsch et al. 2017; Merrell et al. 2007; Richter et al. 2017; Schöps et al. 2017; Seitz et al. 2006; Strecha et al. 2008; Treible et al. 2017; Waechter et al. 2017]. Most similar to this work are the EPFL benchmark [Strecha et al. 2008], UNC dataset [Merrell et al. 2007], ICL-NUIM dataset [Handa et al. 2014] and Tanks and Temples Benchmark [Knapitsch et al. 2017]. Knapitsch *et al.*'s [2017] benchmark is the most extensive providing complete indoor and outdoor scenes. Ground-truth is acquired through LiDAR scanning and images or video are provided for testing. In particular, the Lighthouse, Temple and Palace are currently the only complete outdoor scenes of richly detailed buildings available to the research community. In general there are, however, currently no ground-truth datasets available that provide full urban or residential scenes for image based scene reconstruction. Since there is no way to generate new image positions from these benchmarks, their applicability to flight trajectory evaluation is limited. The need for much larger-scale scenes is apparent and well demonstrated by early work by Snavely *et al.* [2006; 2008] and their later work on BigSfM[1]. Game engines such as Unity and UE4 are also starting to be used to provide large-scale synthetic scenes for computer vision problems such as image based reconstruction, segmentation and depth estimation [Battaglia et al. 2013; Dosovitskiy et al. 2017; Gaidon et al. 2016; Hepp et al. 2017; Mueller et al. 2016; Müller et al. 2018; Papon and Schoeler 2015; Qiu et al. 2017; Richter et al. 2017; Roberts et al. 2017; Ros et al. 2016; Shah et al. 2017]. In particular, Roberts *et al.* [2017] and Hepp *et al.* [2017] independently use scenes from the the Grasslands fantasy level in the Unreal Engine 4 (UE4) to generate synthetic images for evaluating their trajectory optimization. Although exporting models from the UE4 editor is possible, these models are not water-tight and have many intersecting faces, overlapping meshes, and generally a low polygon count. These drawbacks make geometric validation of reconstructions difficult. To the best of our knowledge, the currently proposed benchmark is the first to provide complete datasets of urban and residential scenes that allow full exploration of image based scene reconstruction.

[1] http://www.cs.cornell.edu/projects/bigsfm/

3 METRICS FOR UAV PATH PLANNING

In the following, we will discuss metrics for evaluating the quality of UAV flight paths for the purpose of dense geometry reconstruction. These metrics will inform both the path planning algorithm (Section 4) as well as the synthetic benchmark (Section 5).

3.1 Reconstruction Quality Measures

One way of evaluating flight paths for 3D reconstruction is to perform a 3D reconstruction from all images captured along the full flight path, and compare the resulting models to some ground truth model. This approach measures the performance of the full pipeline, including not just path planning but also bundle adjustment, reconstruction with structure-from-motion or multi-view stereo, and post-processing. This can be both a blessing and a curse – on the one hand we obtain realistic estimates of overall system performance, but on the other hand we lose the ability to attribute performance differences to the path planning as opposed to other system components. This concern can be alleviated with careful design of the benchmark dataset, as discussed in Section 5.

Given a reconstruction and a ground truth model, we apply a quantitative approach for measuring both accuracy and completeness of the reconstruction. First, to compute the geometric accuracy we determine the closest point on the ground truth model's surface for each of the reconstruction's vertices and report the result for the 90% and 95 % threshold (95 % of the points have a distance of less

than x m). In contrast to Middlebury [Seitz et al. 2006] our ground truth has been modeled, not scanned, resulting in a very clean mesh with large differences in triangle size. However, the reconstruction pipelines still create much denser reconstructions (>20M triangles). Therefore, we uniformly sample points on the ground truth surface and determine the coverage for each of these samples (using as many samples as the reconstruction has vertices). We then report the completeness at a 0.075 m threshold. The second measure allows us to determine how much of the area was reconstructed within various degrees of accuracy.

3.2 Reconstructability Heuristics

In addition to the above quality metrics, we introduce a reconstructability heuristic for predicting reconstruction quality using only a simple proxy geometry. The heuristic ensures that the image positions and angles used for the image based reconstruction are sufficient to accurately reconstruct the model. This same heuristic allows us to produce optimized flight plans that ensure complete coverage of the real-world models. To efficiently predict if and how well a point can be reconstructed given a camera network, we develop a heuristics based on distance, observation angle and pairwise parallax angle using the following principles that apply independently of the specific MVS algorithm used:

(1) Triangulation error increases with distance and decreasing parallax,
(2) The ability to triangulate points by stereo matching ("matchability") of images decreases for shallower observation angles and larger parallax,
(3) Any surface needs to be seen by at least two views in order to be reconstructable.

Principles 1 and 2 can also be more formally related since the depth error ϵ_z is related to the matching or disparity error ϵ_d in baseline stereo via

$$\epsilon_z = z^2/(bf) \cdot \epsilon_d, \tag{1}$$

with baseline b, focal length f and depth z [Gallup et al. 2008]. We use these principles as motivation for our reconstructability heuristic which consists of several independent terms that each depend on two views and determine its parameters empirically. More specifically, we model the pairwise view contribution of views (V_1, V_2) to the reconstructability of a sample s (position and normal) as

$$c(s, V_1, V_2) = w_1(\alpha) w_2(d) w_3(\alpha) \cos\theta. \tag{2}$$

As shown in Figure 3, α is the parallax angle between the views and θ is the shallower of both observation angles. The distance $d = \max(\|\overline{sV_1}\|, \|\overline{sV_2}\|)$ is the larger of the two distances between s and the views.

$w_1(\alpha)$ models the parallax dependency of the triangulation error from Principle 1 as

$$w_1(\alpha) = \left(1 + \exp\left(-k_1 \cdot (\alpha - \alpha_1)\right)\right)^{-1}. \tag{3}$$

To model the shape of w_1, we use a logistic function. w_1 also contains an implicit distance component since α decreases with increasing d. The distance also contributes to

$$w_2(d) = 1 - \min\left(d/d_{\max}, 1\right), \tag{4}$$

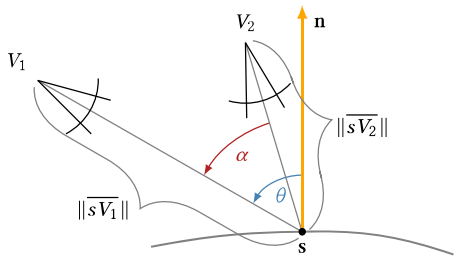


Fig. 3. Two views observing a surface point s and the corresponding input measures for the heuristics.

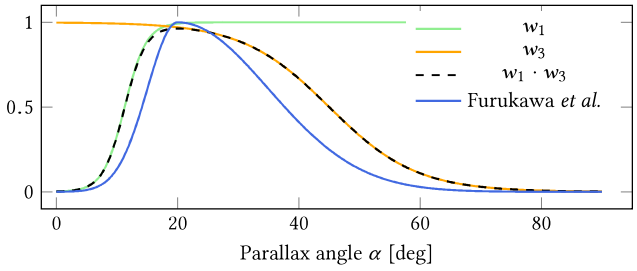


Fig. 4. Combining our parallax weighting functions results in a shape similar to the function proposed by Furukawa *et al.* [2010].

which favors closeup views and thus also handles sample scale as addressed by typical MVS algorithms such as MVE [Goesele et al. 2007]. The parameter $d_{\max}$ has to be chosen based on the desired GSD.

The parallax dependency of the matchability from Principle 2 is modeled as

$$w_3(\alpha) = 1 - \left(1 + \exp\left(-k_3 \cdot (\alpha - \alpha_3)\right)\right)^{-1}, \tag{5}$$

again using a logistic function. In addition, the observation angle is directly contained in Equation 2 as $\cos\theta$.

We determine the parameters for the reconstructability and the depth reconstruction error after extensive experimentation using MVE and SMVS on the Strecha benchmark. Specifically, we constrain $w_1(0) \approx 0$, $w_3(0) \approx 1$, and evaluate results as described in Section 5.5 yielding the parameters $k_1 = 32$, $\alpha_1 = \frac{\pi}{16}$, $k_3 = 8$ and $\alpha_3 = \frac{\pi}{4}$. These parameters are then fixed for all experiments. Interestingly, this yields a dependency of $w_1 \cdot w_3$ on the parallax angle α that is very similar to the corresponding function postulated by Furukawa *et al.* [2010] (see Figure 4). We compared both weighting functions on the Strecha benchmark but found none to be superior to the other.

The last component of our heuristic is a binary visibility function $v(s, V)$ that evaluates the visibility of sample $\mathbf{s}$ in view V based on the scene approximation. Using the pairwise view contributions $c(s, V_1, V_2)$ and the visibility function $v(s, V)$ we then define our heuristic as

$$h(s, \mathcal{V}) = \sum_{\substack{i=1\ldots|\mathcal{V}| \\ j=i\ldots|\mathcal{V}|}} v(s, V_i) v(s, V_j) c(s, V_i, V_j), \tag{6}$$

following Principle 3 with the summation of pairwise view contributions.

4 PATH PLANNING APPROACH

Our approach is based on a two-stage scanning concept that follows a similar high-level structure as Schmid *et al.* [2012]: We first capture the area of interest using a standard grid pattern with nadir views and use the captured data to reconstruct an approximation of the scene geometry, which we call the "proxy" in the following. Based on the proxy, we perform a view and path planning step, yielding a series of 5D camera poses at which we would like to capture images during the production flight. We finally use a standard image-based 3D reconstruction pipeline to reconstruct detailed scene geometry.

4.1 Geometric Scene Proxy

The proxy is an important scene representation used for two purposes: planning optimal view parameters and determining safe airspace for the UAV flight path (see Figure 1). Therefore, we acquire an initial nadir grid pattern with 80/80 image overlap, which consistently produces a complete nadir capture. We then use MVE [Fuhrmann et al. 2015] to reconstruct a full 3D scene from the nadir images. Since it is only captured from nadir viewpoints, sides of buildings, inset windows, and other unseen features are only partially reconstructed. We use a height map from the 3D reconstruction to better fill these gaps while preserving reconstruction geometry compared to simply meshing using Poisson Surface Reconstruction (which creates inaccurate geometry around gaps). We extract the height map by aligning the z axis with the gravity vector and calculating the maximum z value for the discretized (x, y) plane using a pixel size of two times the median GSD. We apply a 3×3 median filter to remove outliers and fill holes by iteratively assigning each invalid pixel with more than two valid neighbors the median of its neighbors. To extract a closed surface mesh we compute an auxiliary point cloud consisting of one sample (3D position and normal) per height map pixel as well as samples at height discontinuities. The samples on height discontinuities are spaced equally with the same resolution as the depth map with horizontal normals oriented towards the depth map gradient. We extract the surface using Floating Scale Surface Reconstruction [Fuhrmann and Goesele 2014] with a uniform scale (height map pixel width).

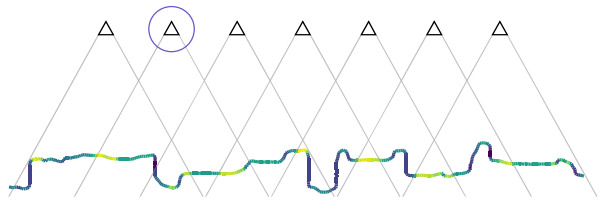
To estimate the safe airspace we use the same approach as for noise removal, but replace the median filter with a dilation operator. The dilation of the height map uses a maximum kernel in combination with a ball structuring element [Schmid et al. 2012], whose radius is the desired minimum distance. This creates a closed surface boundary between safe and potentially dangerous airspace.

4.2 Camera Network Optimization

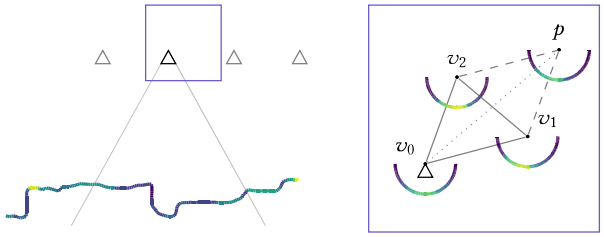
Given a scene approximation we first create an initial camera network with a given and fixed number of uniformly distributed views. We then discretize the scene approximation's surface with a set of uniformly distributed samples $\mathcal{S}$ and optimize their reconstructability through incremental improvements of the view parameters (Figure 5). Specifically, we minimize the objective function O using

$$\arg\min_{\mathcal{V}} O = \arg\min_{\mathcal{V}} \lambda|\mathcal{U}| + \sum_{s\in\mathcal{S}\setminus\mathcal{U}} (\max(h_{\max} - h(s,\mathcal{V}), 0))^2 . \quad (7)$$

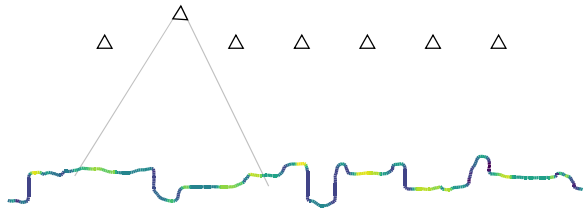
$\mathcal{U} \subset \mathcal{S}$ is the set of samples s that are not visible in any view. The first term encourages exploration of unseen samples with a factor λ



(a) Initialization: Calculate the observation ray list per sample point and evaluate the reconstructability scores. Then select a view for optimization.



(b) Optimization: Left: First remove all observation rays originating from the view to optimize and update the reconstructability scores. Right: Then evaluate the vertices of the view's simplex ($\triangle v_0 v_1 v_2$), project the worst one (v_0) across the hyperplane of the others and select the best position and orientation from the new simplex ($\triangle v_1, v_2, p$) – in this case v_2. The convolved spherical contribution histograms used to determine the best orientation for a position are visualized as arcs around the vertices.



(c) Update: Add the new observation rays for the optimized view. This concludes one iteration, the next iteration starts by selecting another random view (Section 4.2) for optimization.

Fig. 5. Visualization of our algorithm on a 2D slice of the dataset. The colors encode reconstructability of the samples.

(we use $\lambda = 1$ in all our experiments). The second term maximizes the samples' reconstructability heuristics. It is clamped at $h_{\max}$ since adding views will typically not improve the ability to reconstruct a given surface once $h(s, \mathcal{V})$ reaches a certain threshold.

The complexity of the objective function is $O(|\mathcal{S}||\mathcal{V}|^2)$, since evaluating the heuristic for each sample involves pairwise view terms, but more importantly it requires $O(|\mathcal{S}||\mathcal{V}|)$ evaluations of the visibility function (assuming that we cache each combination of sample and view), which is computationally the most expensive part. However, it is possible to efficiently calculate the change of O for a single view in $O(|\mathcal{S}||\mathcal{V}|)$ with $O(|\mathcal{S}|)$ visibility function evaluations by caching observations as described in the implementation section. On larger scenes this is even possible for multiple views, because the influence of a view is spatially limited (see Equation 4). We call a set of such views independent and optimize them in parallel. For each

view, we treat the optimization of viewing position and orientation separately as follows:

For a fixed viewing position, the orientation has only a binary influence on the reconstructability of a sample since it only affects visibility. We can thus collect per sample improvements in a spherical histogram for each view, convolve these spherical histograms with a kernel that represents the viewing frustum and determine the orientation that maximizes the contribution. The kernel size depends on the field of view of the camera. For a full frame camera with 35 mm focal length the kernel size is ca. 15% of a sphere's surface. The convolved spherical histograms are thus smooth, relaxing the requirements for the input histogram resolution.

Although occlusions and a limited field of view introduce discontinuities, the objective function is relatively smooth in practice due to the smooth weighting functions of the heuristics and the large number of samples visible in each view. We therefore use the downhill simplex method to optimize the view position. To do this, we first create randomly oriented simplices around the uniformly distributed initial views and initialize with nadir view orientation. We then iteratively select random sets of mutually independent views and optimize their position and orientation in parallel. For the optimization of a specific simplex, we first reevaluate the change of O for all of its vertices since they may have changed due to updates of other views. When evaluating the change of O for a new candidate position of a vertex, we first determine the optimal view direction for the candidate position as above. All other details follow a standard downhill simplex optimization approach.

The downhill simplex method adjusts its step size depending on the success of a move. The simplex expands if a better position has been found and shrinks if the new position is worse than the previously known positions. The objective function potentially changes in between iterations, because we optimize other views. It is therefore important to maintain an appropriate simplex size. We encourage a regular update of the simplex size by increasing the selection probability for views that have been optimized less often than others before. We ensure that each view stays within the flyable airspace by penalizing vertices that get close to its boundary and prohibiting simplex expansions if a vertex would leave the airspace. Finally, we terminate the optimization once the improvement in the last 100 iterations falls below a certain threshold.

4.3 Implementation Details

We use stratified sampling on the triangle mesh of the geometry proxy to reach an average sampling density of 25 samples per square meter and exploit the parallel nature of our per sample heuristic through the extensive use of GPU computing. The evaluation of the objective function as well as the computation and convolution of spherical histograms is implemented in CUDA. We spawn multiple threads with separate streams when working with larger scenes to optimize multiple views simultaneously. Figure 5 visualizes our algorithm in 2D on a slice through the dataset. We do not illustrate the step size adjustments of the simplex downhill method for simplicity.

The evaluation of our heuristic requires the distance and observation direction of each view that observes a sample point as well as the sample point's normal (to determine the observation angle). This information can be stored as a list of observation rays, that have been rotated to a local coordinate system based on the sample's normal. We only compute these lists completely in a single step at the beginning of our optimization and maintain them through updates: We remove all observation rays of a view before we optimize it and add the new observation rays after it has been optimized.

We use spherical coordinates to parameterize orientations with a resolution of 2.5° (the accuracy of a typical camera gimbal used on UAVs is approximately 1°) and the vertices of a subdivided icosahedron (three subdivisions) to calculate spherical histograms.

We determine visibility via ray casting using a bounding volume hierarchy based on axis-aligned bounding boxes and constructed using the surface area heuristic (SAH) [Wald 2007]. We reduce the field of view by 1° when performing visibility tests during optimization to compensate for orientation and, to some extent, position errors during capturing as well as approximation errors of our spherical histograms.

4.4 Trajectory

The final step of our algorithm is the extraction of an optimized flight trajectory from the camera network. We first find a short path through all view positions by solving the corresponding traveling salesman problem with a random restart 2-opt algorithm. This algorithm is suitable for the GPU [O'Neil et al. 2011] but we are currently running a CPU version. We convert this into a smooth path using the global interpolation method for B-spline curves generating a knot vector based on the chord length method [Shene 1998]. Although we ensure that all views are reachable from the flyable airspace the spline could theoretically violate the safety margin. Through the planning of the spatial trajectory we can detect and manually reroute in such cases.

5 BENCHMARK DATASET

To allow for quantitative evaluations of our flight paths as well as enable comparisons with future work on this topic, we developed a new synthetic benchmark dataset specifically tailored towards the UAV path planning problem in urban environments.

5.1 Building Scenes

The urban and residential scenes in the benchmark are made from a large asset library of buildings, walls, bridges and streets. The models start as highly detailed polygonal meshes produced by professional modelers. They are visually realistic when rendered. Since the models are polygonal and not solids or NURBS they quite often have interior, duplicated or intersecting faces, which can affect the reconstruction score during evaluation. Each model is examined and cleaned. As a final step we re-topologize the model so that the visual appearance when rendered reflects the underlying geometry.

The assets are arranged to create urban scenes to be used in the benchmark. Since there is full control over the buildings we can reconfigure many of the attributes to create a large diversity of scenes for evaluation. In total, we use 34 buildings to make four urban scenes each reflecting different architectural styles (see Table 1). We also design one large urban scene with 60 less complex buildings to examine large-scale flight planning. Each final scene is exported as a single water-tight triangular mesh. The multi-building urban scenes

Fig. 6. Datasets with natural looking colored textures and close-ups of underlying geometry. *Top*: Scene NY-1, *Middle*: Scene UK-1, *Bottom*: Scene GOTH-1.

Table 1. Key statistics of the scenes in the synthetic benchmark datasets: area, number of buildings, maximum building height, and number of triangles, as well as number of images and flight distance for our flight plan.

	A [m^2]	#Bldg	Ht [m]	#Tri	#Img	Dist [m]
NY-1	4,674	10	24.50	123k	433	2807.6
GOTH-1	12,255	5	46.50	594k	588	4213.2
CA-1	6,490	11	16.00	3048k	743	3939.3
UK-1	27,565	8	41.00	624k	923	7819.2
Old-City	28,896	60	34.00	5k	587	6017.2

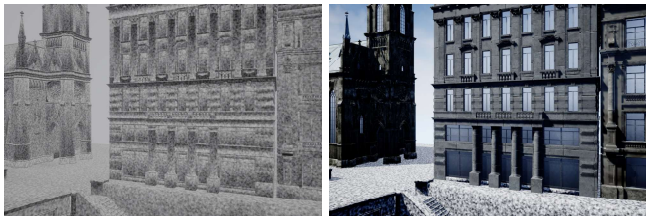


Fig. 7. Rendered camera image used in reconstruction from Goth-1 Dataset, textured using Perlin Noise (left) and natural looking colored textures (right).

provide a new ground truth benchmark at a scale and complexity larger than past datasets (see Supplementary Material for detailed description and figures of each urban scene).

5.2 Texturing and Image Extraction

Homogeneous surface texture is problematic for any image based reconstruction pipeline and additional views will not help 3D reconstruction algorithms deal better with textureless regions. Since we are interested specifically in benchmarking the path planning aspect of the system, we try to isolate this aspect and ensure that the dataset has rich texture everywhere. Under these idealized circumstances, most SfM/MVS pipelines have similar performance (see Section 6, Table 4).

For analysis we create two textured variants of the benchmark: one with random noise texture and the second with naturally looking texture (see Figure 7). By texturing the mesh with simplex noise [Perlin 2002], we are able to provide a detailed and easy to reconstruct texture with constant resolution. Using this texturing technique, we prevent bias and the introduction of non-geometric artifacts that would affect adversely the reconstruction accuracy. We next import the datasets into Unreal Engine 4 to take advantage of its high quality real-time rendering engine. The random noise texture model is imported as is and is set to not cast shadows or reflections. For the naturally looking textured variant we re-apply physical based materials [Karis and Games 2013], which allow for a more accurate shading model and typically provide a more natural looking appearance to all the surfaces of each dataset. Multiple light sources and a skydome are introduced to create realistic lighting and shadows under a noon daylight setting. The ground plane is textured with a blended material of concrete and simplex noise to prevent false positive matches in the reconstruction pipelines from repeated tiling of the concrete texture over large surface areas. All rendering is done in UE4 using global illumination with dynamic shadows, distance field ambient occlusion, and multi-pass reflections (for further detail on texturing and lighting see the supplementary material and video). A movable camera is setup within each scene to a 30 mm focal length with rendering at a resolution of 6000×4000 pixels. We extract subsequent images for the two textured variants using our plugin tool that incrementally moves a camera to a list of waypoints and saves each image to disk.

This synthetic dataset allows us to capture synthetic views through rendering and compare the reconstructions against the geometric model itself. In the results section we compare how SfM/MVS pipelines perform differently using these two approaches.

5.3 Alignment

We evaluate the reconstruction quality using the methodology of the Middlebury MView benchmark [Seitz et al. 2006]. I.e., we must first align the reconstruction pipeline's dense point cloud and mesh output to the ground truth model. We randomly select four well distributed camera positions from the reconstruction pipeline and use a Helmert transformation to align to their corresponding camera positions from the computed flight trajectory. This approach allows for an automatic and accurate alignment of the reconstruction and is an advantage afforded by having known camera positions from the synthetic benchmark. As a final refinement we use the iterative closest points (ICP) approach [Besl and McKay 1992] with scaling to ensure that a close alignment is achieved. If the input reconstruction maintains a fair level of accuracy the final alignment is typically below an RMS error of 0.003 m.

5.4 Benchmarking Tool for Evaluation

We setup an automated benchmarking tool to assist in conducting the evaluations presented in this paper and for future work. We provide for download the coarse geometry proxy for each dataset along with original nadir images captured on a 80/80 nadir grid. Either the proxy mesh or nadir images (once reconstructed into a model) can be used to compute a trajectory using our approach or others (e.g. Roberts *et al.* [2017]). The benchmark datasets are packaged within a release project of UE4 that can be downloaded and used for evaluation. Images from the computed trajectory are generated by loading a set of camera poses into the UE4 project at runtime (the nadir grid of images can also be generated using this approach). The images can then be processed with any SfM/MVS pipeline and evaluated with the downloaded benchmark tools.

5.5 Reconstruction Pipelines

In this paper, we evaluate several image based open-source and commercial structure-from-motion pipelines. The open-source software systems include MVE [Fuhrmann et al. 2015], and COLMAP [Schönberger and Frahm 2016]; Pix4D [Strecha et al. 2008] allows for comparison to commercial applications. The benchmark presented here provides new challenges to all the pipelines: primarily in large-scene completion, repetitive structures and high camera angle disparity. In contrast to many of the open-source work, the commercial software is continually being improved with growing focus on aerial capture and camera extrinsics from GPS is increasing

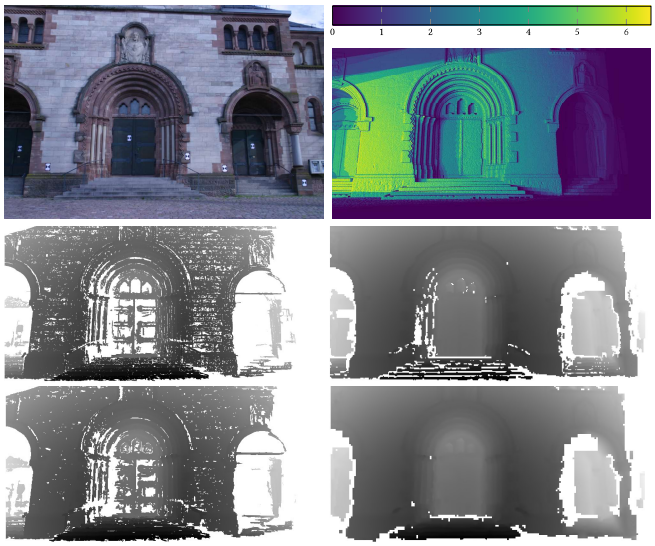


Fig. 8. Input image (*top left*), reconstructability (*top right*) and corresponding depth maps for the *Fountain-P11* dataset (cropped). *Bottom rows:* MVE depthmaps *(left)* and SMVS depthmaps *(right)* reconstructed with Scale 1 *(middle row)* and Scale 2 *(bottom row)*. Note that MVE struggles with weakly textured surfaces. Completeness of the depth map improves with scale.

in importance; therefore, we provide the camera extrinsics from the simulated GPS to all pipelines that support it.

6 EVALUATIONS

We first evaluate the reconstructability heuristics and flight planning performance on the synthetic benchmark. The synthetic benchmark allows for evaluation of state-of-the-art reconstruction pipelines on large-scale urban scenes with ground truth measurements.

6.1 Reconstructability Heuristics

To evaluate the heuristics, we use two fundamentally different multi-view stereo algorithms to reconstruct two different benchmark scenes, *Fountain-P11* and *Herz-Jesu-P8* from Strecha et al. [2008]. Specifically, the two algorithms are the variational approach by Semerjian [2014] as implemented in SMVS [Langguth et al. 2016] (called "SMVS" in the following), and the region-growing approach of Goesele *et al.* [2007] as implemented in MVE [Fuhrmann et al. 2015] ("MVE"). Both of these can be applied to differently scaled input images. Scale n refers here to 2^{-n} times the resolution of the input images. The examples in Figure 8 show, e.g., that MVE struggles with the reconstruction of weakly textured surfaces such as the door and that completeness of the depth map increases on the larger scale (i.e., lower resolution images). SMVS has no issue with these surfaces but the fact that it optimize patches is visible at depth discontinuities (e.g. the steps).

Figure 9 plots the mean and variance of the depth error vs. the reconstructability heuristic for the corresponding ground truth surface point on the *Fountain-P11* dataset. Overall, both the error and its variance decrease with increasing values of the reconstructability heuristics. Likewise, the fraction of unreconstructed points drops

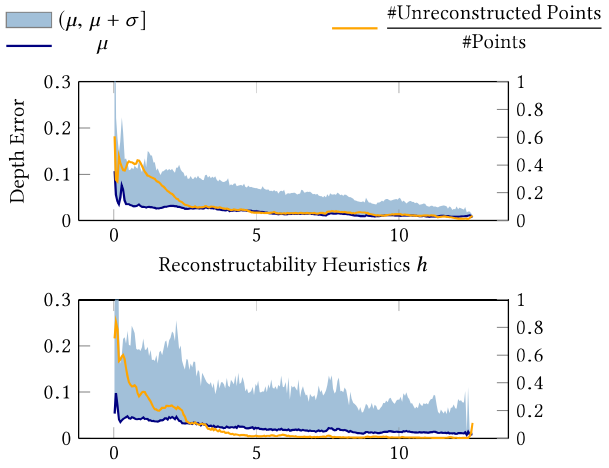


Fig. 9. Error statistics aggregated over all pixels and images in the datasets (leaving out border pixels to avoid errors with patch-based algorithms). The mean and variance of the depth error are denoted as μ and σ, respectively. *Top*: SMVS with Scale 1 on *Fountain-P11*, *Bottom*: MVE with Scale 2 on *Fountain-P11*. Peaks on the right are caused by small sample counts.

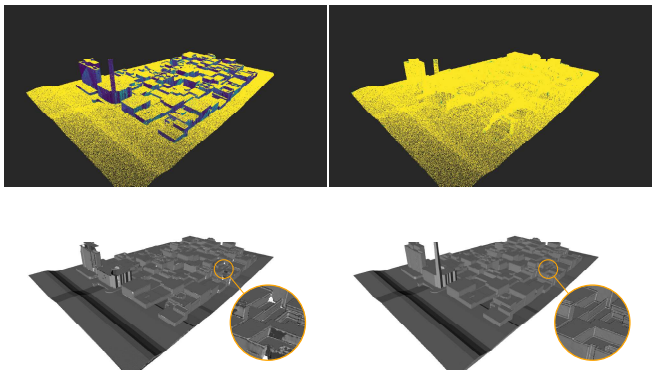


Fig. 10. *Top row*: Reconstructability evaluated on *Initial* and *Optimized* views (clamped to $[0, 3]$ and colormapped). *Bottom row*: Resulting reconstructions of the MVE pipeline with 287 nadir *left* and optimized views *right*.

with increasing heuristics values. However, depth discontinuities can pose a problem for both algorithms but are not modeled by our heuristics. Overall, these results confirm a good correlation between our heuristic and actual reconstruction quality for both of these very dissimilar MVS pipelines.

6.2 View and Path Planning on Synthetic Data

We first evaluate our path planning on the largest of our synthetic scenes, *Old-City* (see Figure 10), which has all faces fully exposed with no overhangs. Using our path planning approach, the optimized views achieve high accuracy and completeness with only a minor increase in views compared to a nadir capture with 80% forward and 80% side overlap (visualized in Figure 10). However, it is clear that there is a strong correlation between allotted camera views, accuracy and completeness. We use a 30 mm focal length, set $d_{\max} = 100$ m

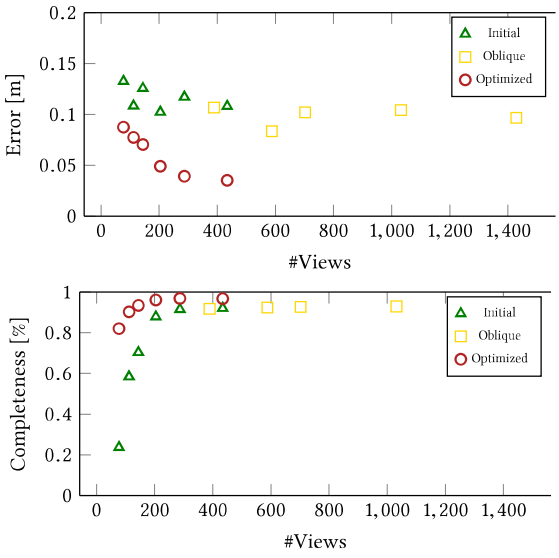


Fig. 11. Results of our view planning in combination with the MVE pipeline on the *Old-City* dataset for varying number of views and configurations. *Initial* refers to a regular grid with nadir views, *Oblique* to a regular grid with a mix of nadir and 45° views, and *Optimized* to our method.

and optimize for a $h_{\max} = 3.0$. As baseline we use a regular grid with nadir views at 75 m altitude (labeled *Initial*). We also compare in Figure 11 a second baseline (labeled *Oblique*) containing 20 % nadir views and 20 % views pitched at 45° for the four cardinal directions. All views are rendered at a resolution of 3000×2000 pixels.

Figure 11 of the *Old-City* scene shows a comparison of how different flight trajectories with varying number of views impact accuracy and completeness. The accuracy of the *Initial* and *Oblique* reconstructions hardly drops under 0.1 m, while in contrast, our *Optimized* solution is more than twice as accurate for the datasets with 204 and more views. Likewise, the completeness of *Optimized* solutions is better than 96 % for 204 or more views whereas none of the other methods reaches 93 % even with significantly more images. Note that in general these last few percentage points of completeness are particularly hard to achieve. Figure 10 shows renderings of the reconstructability and reconstructed models. Note the improvement in the geometry.

6.3 Algorithm Runtime and Path Efficiency

Next, we compare the run time efficiency of our path planning approach (OURS) to Roberts *et al.* [2017]'s submodular approach (SUB) and their next best view (NBV) implementation. We use the *GOTH-1* and *CA-1* datasets rendered with simplex noise texture for all comparisons (see supplemental for more results). Computation of the proxy mesh from synthetically generated nadir images takes ca. 30 minutes with MVE. Timing starts at provision of the proxy mesh to the approach. Each approach is tasked to compute a single flight plan to capture the entire area (since both GOTH-1 and CA-1 are large areas the actual UAV would require multiple flights to complete). All images captured are combined and reconstructed offline using COLMAP (this last step of computing the final mesh averages between 8-32 hours). In Table 2, we show the run-time results for computation of the graph (only SUB and NBV), trajectory and orientation. The amount of images computed is in brackets next to each run. Our approach computes camera trajectory and orientation at the same time and optimizes over the proxy mesh directly. SUB and NBV must also first compute a visibility graph that is computationally expensive and bound by the available cores of the CPU and memory (see supplemental material workstation specifications). SUB in comparison to NBV can set a solver limit. We found for the large datasets a minimum of 7 minutes is required to achieve a global reward above 50%. Both NBV and SUB's full runtime from providing a proxy mesh to results takes 2.6-4.7 hours with the orienteering representing only the last step. In comparison, our approach in total only takes 4-6 minutes from proxy mesh input to final results, this is 2-2.5 times faster than just the orienteering steps by comparison. OURS also scales much better with more complex areas such as *CA-1* where total computation time drops to only 4.14 minutes in comparison to a 54% increase in computation for NBV and SUB. These runtimes highlight the advantages of the parallelization and scalability on the GPU of our approach. In real world applications our approach can be performed in the field, and requires a total down time of less than 45 minutes between the nadir scan and the execution of the optimized flight plan.

Table 2. Timing Results of GOTH-1 and CA-1 Dataset comparing different methods. Timing calculated for computation of Graph nodes (Graph), Camera Trajectory (Traj) and Orientation (Ori). Total travel distance is calculated in meters (Dist) and estimated flight duration (t1) and simulated flight time (t2) in minutes.

Method(#Img)	Graph	Traj	Ori	Total	Dist[m]	t1	t2
GOTH-1							
NBV(607)	144.0	10.0	3.0	157.0	**3817.7**	20.2	28.7
NBV(761)	144.0	11.0	5.0	160.0	4785.2	25.4	30.13
SUB(605)	144.0	7.0	5.0	156.0	3829.8	20.2	26.4
SUB(753)	144.0	7.0	6.0	157.0	4789.5	25.1	34.2
OURS(588)	—	**6.0**	—	**6.0**	4213.2	**19.6**	**22.4**
CA-1							
NBV(741)	264.0	11.0	5.0	280.0	3824.7	24.7	28.9
SUB(743)	264.0	7.0	7.0	278.0	3834.5	24.8	30.5
OURS(728)	—	**4.1**	—	**4.1**	**3638.6**	**24.3**	**25.1**
OURS(743)	—	**4.1**	—	**4.1**	3939.3	24.8	25.7

Flight path length and time estimated to capture the dataset is also presented in Table 2. The primary bottleneck for UAV path planning is the speed at which images can be captured and UAV battery duration; drone physics and thrust limits are rarely taxed in image acquisition. We assume like Roberts *et al.* [2017] that images are captured at a rate of 2 seconds and total individual flight durations are 8 minutes. This image capture rate limits the flight speed of the UAV to a speed that will allow the camera to successfully trigger by the time the next waypoint is reached. For shorter distances between captures the UAV will have to fly much slower while it may fly up to reasonable UAV flight speed of 5 m/s for further distances (still far below thrust limits of the UAV). Modern UAV flight

controllers handle this internally, incrementing speed according to distance between waypoints. For the *GOTH-1* and *CA-1* dataset we allow NBV, SUB, and OURS to calculate the optimal amount of images if the UAV flies for a maximum of 32 minutes (4 flights). As the global optimum scores for NBV and SUB were low for GOTH-1 we additionally ran them for 5 flights. For comparing flight duration, we take two approaches. The first is to assume average flight duration is 2s between each image allowing the UAV to vary its speed accordingly. Second we run the flight paths as auto missions using the Sim4CV [Müller et al. 2018] plugin with our setup in UE4. The second approach better estimates the additional time required for slowing down when cornering, ascending, or descending which must account for inertia and other flight dynamics.

For the GOTH-1 dataset our approach requires only 588 images but with a longer flight length of 4213.2 m. In comparison, for the same amount of flights, NBV and SUB require slightly more images and time to acquire the scene but at a slightly shorter flight length. NBV has the smallest flight length and number of images but results in the poorest reconstruction results (see Table 3). In order for NBV and SUB to achieve improved capture accuracy and completeness, five flights on the *GOTH-1* dataset are required.

For *CA-1* dataset 4 flights are calculated. All three approaches estimate a similar number of images and travel distance to acquire the scene. The flight simulation indicates our flight duration is shorter, which is a product of our algorithm interpolating a smooth continuous path after trajectory optimization (see Section 4.4). The other approaches have many abrupt 90 degree turns and sharp elevation changes at corners that affect UAV flight speed leading to longer flight times.

In summary, NBV, SUB, and OURS all are highly optimized with respect to flight path duration and length. OURS requires fewer images to achieve higher scores of reconstructibility but takes greater liberty in flight path length to more optimally place its cameras.

6.4 Comparison to State-of-the-Art

We next compare accuracy and completeness on *GOTH-1* and *CA-1*, which are both medium size urban scenes (see Table 3). Our approach is able to achieve better accuracy and completeness on both datasets compared to NBV and SUB. Our results confirm that SUB does outperform NBV and in less computation time. The primary limitation of graph based approaches is the requirement to select from candidate views that are restricted to an evenly spaced voxel grid. This limits the potential placement of the cameras and in many cases oversampling occurs in areas while in other areas large gaps remain. Our approach, with the same number of images (743), has greater flexibility in camera placement dropping much lower and closer to the scene geometry (2.5 m) allowing it to capture complex areas such as under the arches in *GOTH-1* and the balconies and storefronts of *CA-1*. This allows for example in *CA-1* a 21% higher completeness score for our approach. We also run our approach limiting camera placement to no closer than 4 m to compare with the inherent limitation of SUB and NBV for CA-1 (note for GOTH-1 our approach did not require placing cameras closer than 4 m). This leads to less images needed (728) and shorter flight path, while maintaining similar if not better accuracy. Completeness drops as would be expected but is still above both SUB and NBV.

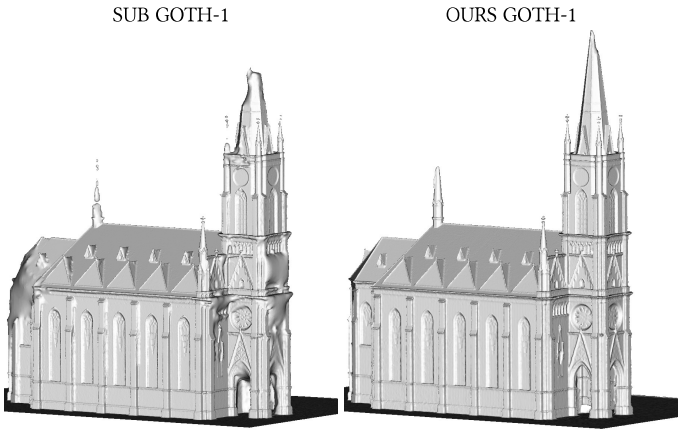


Fig. 12. Visual Comparison of GOTH-1's Cathedral reconstructed by Roberts *et al.* [2017] approach and Ours.

It is also important to highlight for these other methods that as areas become larger the voxel grid becomes more expensive. For *GOTH-1* the graph node just barely fit in system memory, larger areas would require an increase in voxel distance to remain within memory limits further reducing potential to select optimal candidate views and much longer computation times to search through the nodes to select best candidates.

Table 3. Comparison of reconstruction quality for different flight paths on the GOTH-1 dataset.

	GOTH-1				
	Error[m] 90%	Error[m] 95%	Comp[%] 0.075 m	Comp[%] 0.050 m	Comp[%] 0.020 m
NBV (607)	0.027	0.051	44.41	40.50	32.51
NBV (761)	0.037	0.051	47.81	43.59	30.19
SUB (605)	0.039	0.047	49.24	44.86	31.20
SUB (753)	0.020	0.039	51.63	47.40	39.70
OURS (588)	**0.019**	**0.028**	**58.09**	**54.05**	**45.70**
	CA-1				
NBV (741)	0.027	0.081	53.06	48.25	42.10
SUB (743)	0.020	0.049	51.45	48.61	42.80
OURS (728)	**0.014**	**0.029**	57.58	54.32	48.65
OURS (743)	**0.014**	0.032	**64.92**	**62.29**	**56.36**

Next, we evaluate performance of the reconstruction pipelines on the two versions (simplex noise and color) of all four urban scenes using our computed flight trajectories. For each scene, we set an initial target of 1.0cm GSD with a resolution of 6000×4000 pixels and 30mm focal length that matches the drone camera (Sony NEX-7). The airspace limit is set to 2.5m which still allows descent below most canopies and arches but prevents potential collision. The number of calculated initial images starts with these estimates and nadir overlap of 80/80 and image count is only increased based on the evaluation of the reconstruction heuristic (see above and Figure 10). The majority of view points maintain an altitude below the limit of 50 m but safely above the buildings to achieve a large

Table 4. Results of synthetic benchmark detailing reconstruction error and completeness measures

	MVE+FSSR			COLMAP			Pix4D		
	Error[m] 90%	Error[m] 95%	Comp[%] 0.075 m	Error[m] 90%	Error[m] 95%	Comp[%] 0.075 m	Error[m] 90%	Error[m] 95%	Comp[%] 0.075 m
NY-1 (Noise)	**0.019**	**0.033**	**51.00**	0.024	0.047	44.62	0.837	1.541	35.57
GOTH-1 (Noise)	**0.010**	**0.017**	**59.49**	0.021	0.028	58.08	0.953	1.676	59.85
CA-1 (Noise)	0.024	0.048	63.65	**0.014**	**0.032**	**64.92**	1.109	1.478	65.04
UK-1 (Noise)	**0.030**	0.119	37.52	0.032	**0.046**	**37.90**	2.304	4.185	42.03
Mean	**0.021**	0.054	**52.92**	0.023	**0.038**	51.38	1.30	2.220	50.62
NY-1 (Color)	**0.025**	**0.052**	44.28	0.029	0.054	**47.87**	0.803	1.642	45.90
GOTH-1 (Color)	**0.021**	**0.028**	50.36	0.026	0.043	**55.82**	0.679	1.202	54.06
CA-1 (Color)	**0.035**	**0.079**	40.44	0.065	0.190	**51.70**	2.982	10.95	46.29
UK-1 (Color)	**0.035**	**0.088**	**35.06**	0.106	0.138	32.82	2.815	8.182	24.58
Mean	**0.029**	**0.062**	42.53	0.056	0.106	**47.05**	1.820	5.494	42.71

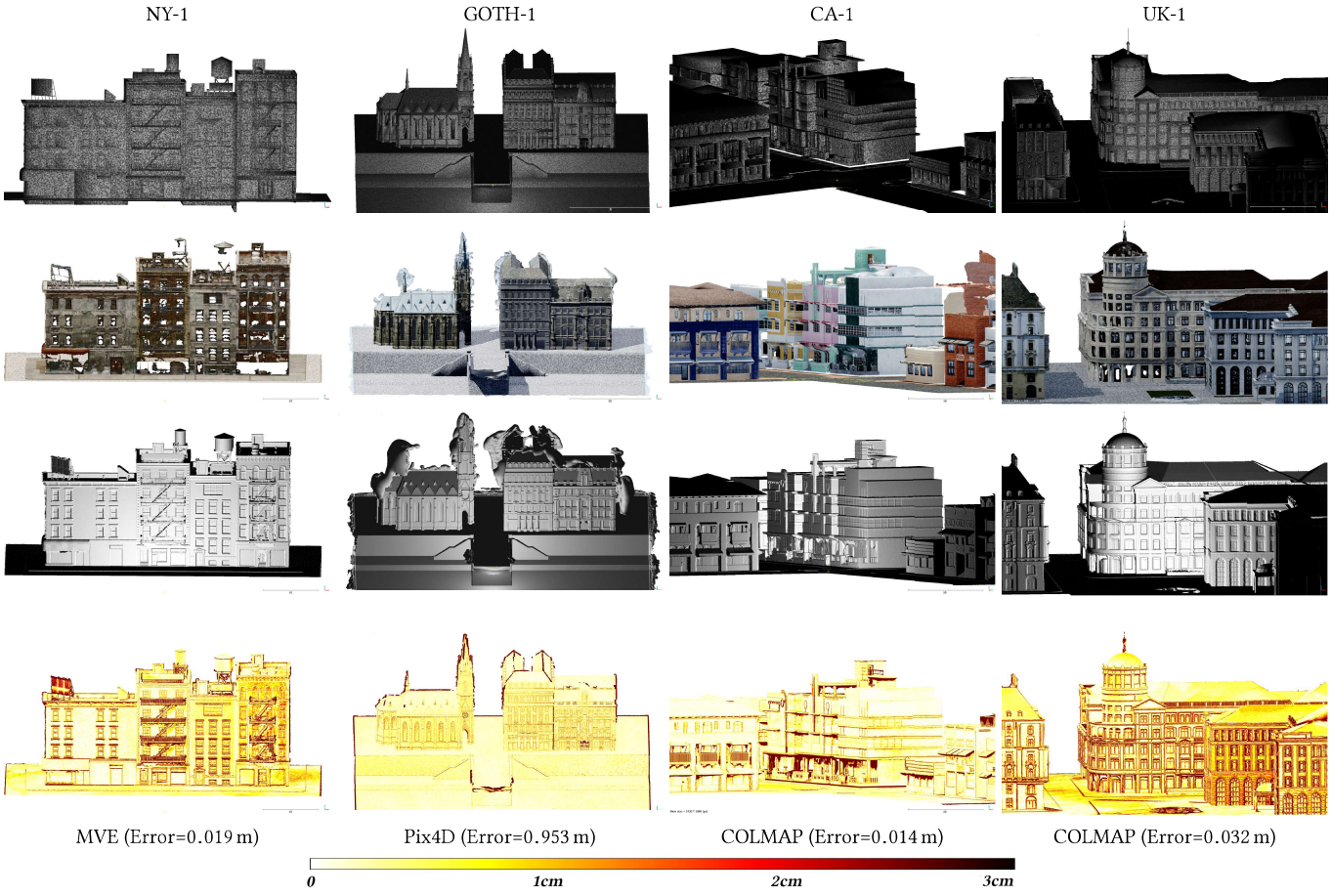


Fig. 13. Quantitative results of large synthetic scene reconstructions. *Top Row*: input synthetic models with textured Simplex noise, *Second Row*: reconstruction of the naturally textured models from the SfM pipelines, *Third Row*: reconstruction of the Simplex noise textured models from the SfM pipelines, *Fourth Row*: error relative to ground truth point cloud.

area coverage at the desired GSD. A small percentage of cameras ascend higher to capture roof tops or descend lower to capture below balconies or overhangs. In Table 1 we note the size of each scene and the correlating number of images required to achieve high accuracy and completeness. It can be seen that more images are required for smaller areas with higher building density.

Table 5. Real test datasets with total area size, targeted ground sample distance (GSD), max allowable altitude of the UAV (Max Alt.), number of images acquired, and shutter speed of the camera.

	Area $[10^3 m^2]$	GSD [cm]	Max Alt. [m]	#Img	Shutter [s]
Housing	5.4	1.0	25	390	1/1250
Apartments	12.3	1.0	30	637	1/1250
School	32.0	1.5	40	724	1/1250
Stadium	51.9	2.0	100	183	1/1000

A summary of the benchmark performance for each reconstruction pipeline is presented in Table 4. To compute the accuracy we determine the closest point on the ground truth model for each of the reconstructed vertices and report the minimum error for the 90 % and 95 % thresholds (i.e. 90 % of the points have a distance of less than x m). We report the completeness for a 0.075 m threshold. In contrast to Middlebury datasets, our benchmark consists of full scale buildings that are measured in meters and abut each other with many faces that cannot be seen due to occlusion and the flight limitations we impose on the UAV flyable airspace; therefore, the overall accuracy and completeness score threshold is lower across the board. Our ground truth has also been modeled, not scanned, resulting in a very clean mesh with large differences in triangle size, in order to compensate for this we uniformly sample points on the ground truth's surface and determine the coverage for each of these samples (using as many samples as the reconstruction has vertices).

We note that MVE tends to produce the lowest error, while COLMAP performs very well in terms of completeness. Overall, the results for the noise texture are more similar than for the natural texture. This indicates that indeed the noise texture is effective at suppressing differences due to reconstruction algorithms, and is therefore more suitable for comparing path planning methods in isolation (for a detailed discussion of the performance of the different pipelines please see the supplementary material).

Figure 13 shows the input textured models, reconstructions of several of the scenes sampled by different reconstruction pipelines and their accuracy score colormap. The optimized flight trajectory allows for the drone to fly much lower and closer to the mesh and capture hard to observe areas that are not possible if flying at a fixed altitude. This is apparent in CA-1 with many low canopies (ca. 25), GOTH-1's bridge tunnel and arches and NY-1's fire escapes. These are all captured efficiently using our approach.

6.5 Field-Test

We select four large, real-world scenes to evaluate the path planning approach in the field. The real-world tests have multiple tall buildings in non-uniform orientations and heights with challenging overhangs, balconies and inset doorways. For each scene we first capture the encompassing area in a nadir flight at an altitude of 100 m (≈ 0.025 m GSD) and copy the images to a field laptop. Typically less than 50 images are needed to capture the area allowing for reconstruction using Colmap/Pix4D in less than thirty minutes. It requires less than seven minutes to then generate a geometric scene proxy, flyable airspace and compute the optimal flight trajectory (see Table 2 for timings). In Table 5, we list the size and images captured for each scene along with camera setup for the UAVs. Similar to the example in Figure 10, we can visualize the reconstructibility of the scene to get an immediate insight on the potential success of a flight trajectory prior to testing in the field. From the computed 5D flight trajectory, we are able to generate detailed waypoint missions that include continuous updates on camera pitch, roll, and yaw. Images are captured once the UAV reaches a waypoint. The multi-rotor has a limited flight time of 12 minutes and averages 150-250 images a flight. The trajectory is automatically broken into multiple flight missions, uploaded to a cloud server and then synced from a tablet to the UAV. Since the trajectory is optimized to not overlap, it would be possible in future work to split the total capture between multiple UAVs at the same time rather than running the missions sequentially.

We present qualitative results of the real-world tests in Figure 14. There are several clear advantages to the optimized flight trajectory noticeable in the results of the field test. First, in comparison to the nadir flights, vertical surfaces are well constructed and dense. Second, significantly fewer flights are required to capture these complex areas while still maintaining consistent GSD. The optimized flight trajectory allows the capture of ground surfaces at proper distances while also compensating for building roofs by ascending to much higher altitudes. At the same time, the vertical faces of buildings are captured much closer allowing much higher GSD to be achieved then capturing oblique images at only one fixed height. Similar to the synthetic benchmark, the path planning is flexible including descent close to ground level to capture hard to reach faces.

The current limitations of field capture are primarily related to the sensitivity of the reconstruction pipelines to photometric appearance. Different methods have different amounts of tolerance for appearance issues such as non-uniformity of shading and lack of texture. All the buildings captured primarily have a stucco texture that appears homogeneous at a GSD of more than 1 cm. In addition, direct sunlight on the surface of these buildings leads to washed out images. Finally, the dark shadowed inset walls as seen in the *Apartment* dataset are poorly reconstructed despite camera angles directly viewing these areas. All of these problems are also seen in our colored versions of the benchmark.

There are also hardware based limitations to the field acquisition that impact reconstruction. For example, desired camera positions cannot be reached with an accuracy of more than 1 m to 2 m unless differential GPS is used, although this is not critical at larger capture distances. Second, even though absolute angles are used for camera angles, drift on the gimbal and the magnetometer compass is apparent even over short flight durations of 10 min. These are issues we plan to address in future work as we focus more on integration of our approach in regular field operations.

Despite these limitations, the optimized flight trajectories allow us to capture the desired area in significantly less time and with less images. Moreover, the quality of the reconstruction is higher with fewer artifacts from improperly computed point positions.

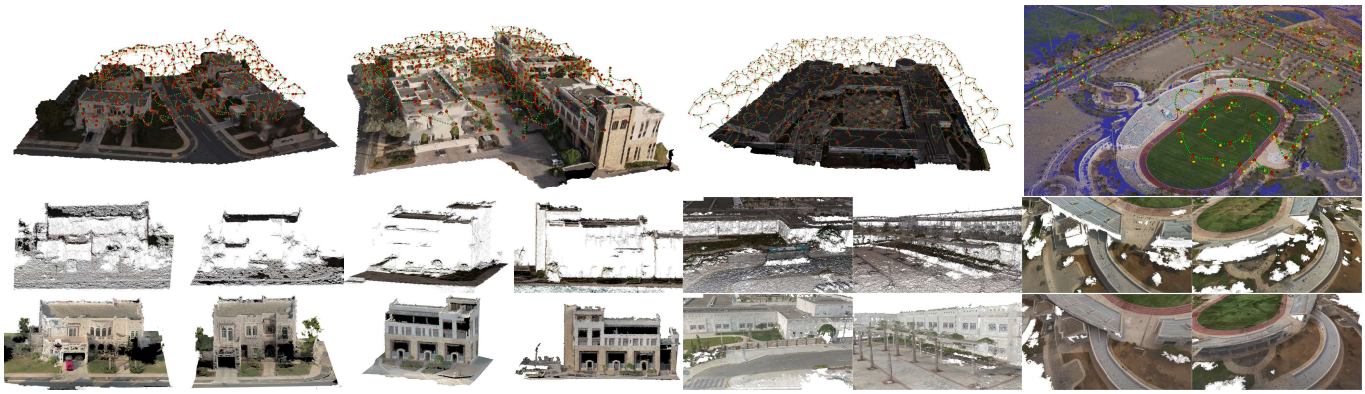


Fig. 14. Reconstruction results of our field tests, the four areas and their optimal flight trajectories *Top Row*, the result of nadir capture *Middle Row*, and optimal trajectory result *Bottom Row*.

6.6 Limitations and Future Work

Our approach starts with a nadir capture and reconstructs the scene proxy as a 2.5D surface from a height map. This means that our scene approximation cannot represent overhanging structures such as under bridges, canopies, or store fronts. Similarly, the flyable airspace which assumes a margin of GPS error restricts the UAV from approaching tight spaces such as narrow alleys and may not properly represent small light posts or hanging wires not reconstructed from the nadir capture. In the future, we would like to complement the offline planning with simulation of online UAV path planning using the synthetic benchmark in UE4. The simulation will allow us to explore methods to focus capture on the hard to reach areas which cannot be fully approximated using a height map or safely flown using only gps-guided UAVs.

Another limitation of our current implementation we hope to address in the future is its inability to adapt the number of views. For example, removing views with negligible contributions or splitting views that observe multiple optima where minimum reconstructibility is not reached.

Finally, as seen in the real-world scenes and colored variant of the benchmark, the ground sample distance from architectural buildings, dynamic lighting, and shadows impact the amount of features that can be detected, especially on homogeneously colored surfaces, which ultimately leads to lower overall reconstruction accuracy and completeness. In future work, we hope to introduce into path planning a consideration of varying surface texture and lighting, which we can accomplish using the synthetic benchmark where full control over texture and lighting can be evaluated.

7 CONCLUSION

We introduce a novel view and path planning approach of large urban scenes for multi-rotor UAVs. In order to achieve this, we develop a new reconstructability heuristic to formulate the capture planning as a continuous optimization problem. Our optimization combines a search over view orientations with an optimization of view positions in a parallel approach, yielding more accurate and complete reconstructions using significantly fewer images. We evaluate the efficacy of our approach on synthetic and real world scenes.

In order to quantitatively evaluate path planning, we present a new synthetic benchmark for image-based reconstruction techniques on large urban scenes. The benchmark addresses the current lack of large ground truth urban scenes for quantitative analysis. A major advantage of the benchmark is the ability to provide an environment to evaluate path planning and how camera position, orientation and distance from complex architectural features impact reconstruction performance. We make the benchmark available for the research community to explore new techniques in planning and image-based reconstruction with the synthetic benchmark (https://vccimaging.org/Publications/Smith2018UAVPathPlanning).

ACKNOWLEDGMENTS

We would like to thank FalconViz for support with this project. This work was in part supported by King Abdullah University of Science and Technology under VCC Center Competitive Funding and Individual Baseline Funding.

REFERENCES

Henrik Aanæs, Rasmus Ramsbøl Jensen, George Vogiatzis, Engin Tola, and Anders Bjorholm Dahl. 2016. Large-Scale Data for Multiple-View Stereopsis. *IJCV* 120, 2 (01 Nov. 2016), 153–168.

Bashar Alsadik, Markus Gerke, and George Vosselman. 2013. Automated Camera Network Design for 3D Modeling of Cultural Heritage Objects. *J. Cultural Heritage* 14, 6 (2013).

Peter W. Battaglia, Jessica B. Hamrick, and Joshua B. Tenenbaum. 2013. Simulation as an engine of physical scene understanding. *Proc. National Academy of Sciences* 110, 45 (2013), 18327–18332.

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

P. J. Besl and N. D. McKay. 1992. A Method for Registration of 3-D Shapes. *IEEE Trans. PAMI* 14, 2 (1992), 239–256.

Angela Dai, Angel X. Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Nießner. 2017. ScanNet: Richly-annotated 3D Reconstructions of Indoor Scenes. In *Proc. CVPR*.

Alexey Dosovitskiy, German Ros, Felipe Codevilla, Antonio Lopez, and Vladlen Koltun. 2017. CARLA: An Open Urban Driving Simulator. In *Proc. Ann. Conf. Robot Learning*. 1–16.

Enrique Dunn and Jan-Michael Frahm. 2009. Next Best View Planning for Active Model Improvement. In *Proc. BMVC*.

Xinyi Fan, Linguang Zhang, Benedict Brown, and Szymon Rusinkiewicz. 2016. Automated View and Path Planning for Scalable Multi-Object 3D Scanning. *ACM Trans.*

Graph. 35, 6 (Nov. 2016).
Christian Forster, Matia Pizzoli, and Davide Scaramuzza. 2014. Appearance-Based Active, Monocular, Dense Reconstruction for Micro Aerial Vehicles. In *Robotics: Science and Systems*.
Simon Fuhrmann and Michael Goesele. 2014. Floating Scale Surface Reconstruction. *ACM Trans. Graph.* 33, 4 (2014).
Simon Fuhrmann, Fabian Langguth, Nils Moehrle, Michael Waechter, and Michael Goesele. 2015. MVE - An Image-Based Reconstruction Environment. *Computers & Graphics* 53 (2015).
Yasutaka Furukawa, Brian Curless, Steven M. Seitz, and Richard Szeliski. 2010. Towards Internet-Scale Multi-View Stereo. In *Proc. CVPR*.
Adrien Gaidon, Qiao Wang, Yohann Cabon, and Eleonora Vig. 2016. Virtual worlds as proxy for multi-object tracking analysis. In *Proc. CVPR*. 4340–4349.
David Gallup, Jan-Michael Frahm, Philippos Mordohai, and Marc Pollefeys. 2008. Variable Baseline/Resolution Stereo. In *Proc. CVPR*.
Quentin Galvane, Christophe Lino, Marc Christie, Julien Fleureau, Fabien Servant, François-Louis Tariolle, and Philippe Guillotel. 2017. Directing Cinematographic Drones. *CoRR* abs/1712.04216 (2017). arXiv:1712.04216 http://arxiv.org/abs/1712.04216
Christoph Gebhardt, Stefan Stevsic, and Otmar Hilliges. 2018. Optimizing for Aesthetically Pleasing Quadrotor Camera Motion. *ACM Trans. Graph.* 37, 4, Article 90 (Aug. 2018), 11 pages.
Michael Goesele, Noah Snavely, Brian Curless, Hugues Hoppe, and Steven M. Seitz. 2007. Multi-View Stereo for Community Photo Collections. In *Proc. ICCV*.
A. Handa, T. Whelan, J. McDonald, and A. J. Davison. 2014. A benchmark for RGB-D visual odometry, 3D reconstruction and SLAM. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*. 1524–1531. https://doi.org/10.1109/ICRA.2014.6907054
Benjamin Hepp, Matthias Niessner, and Otmar Hilliges. 2017. Plan3D: Viewpoint and Trajectory Optimization for Aerial Multi-View Stereo Reconstruction. *CoRR* abs/1705.09314 (2017).
Christof Hoppe, Andreas Wendel, Stefanie Zollmann, Katrin Pirker, Arnold Irschara, Horst Bischof, and Stefan Kluckner. 2012. Photogrammetric Camera Network Design for Micro Aerial Vehicles. In *Proc. Computer Vision Winter Workshop*, Vol. 8.
Rui Huang, Danping Zou, Richard Vaughan, and Ping Tan. 2017. Active Image-based Modeling. *CoRR* abs/1705.01010 (2017). arXiv:1705.01010 http://arxiv.org/abs/1705.01010
Brian Karis and Epic Games. 2013. Real Shading in Unreal Engine 4. (2013).
Arno Knapitsch, Jaesik Park, Qian-Yi Zhou, and Vladlen Koltun. 2017. Tanks and Temples: Benchmarking Large-Scale Scene Reconstruction. *ACM Trans. Graph.* 36, 4 (2017).
Fabian Langguth, Kalyan Sunkavalli, Sunil Hadap, and Michael Goesele. 2016. Shading-Aware Multi-View Stereo. In *Proc. ECCV*.
John J Leonard and Hugh F Durrant-Whyte. 1991. Simultaneous Map Building and Localization for an Autonomous Mobile Robot. In *Proc. IROS*.
Ligang Liu, Xi Xia, Han Sun, Qi Shen, Junzhan Xu, Bin Chen, Hui Huang, and Kai Xu. 2018. Object-Aware Guidance for Autonomous Scene Reconstruction. *ACM Trans. Graph.* 37, 4 (2018), 104:1–104:12.
O. Mendez, Simon Hadfield, Nicolas Pugeault, and Richard Bowden. 2016. Next-Best Stereo: Extending Next-Best View Optimisation for Collaborative Sensors. In *Proc. BMVC*.
P. Merrell, P. Mordohai, J. M. Frahm, and M. Pollefeys. 2007. Evaluation of Large Scale Scene Reconstruction. In *Proc. ICCV*. 1–8.
Pierre Moreels and Pietro Perona. 2007. Evaluation of Features Detectors and Descriptors Based on 3D Objects. *IJCV* 73, 3 (2007).
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. CVPR Workshops*.
Matthias Mueller, Neil Smith, and Bernard Ghanem. 2016. A Benchmark and Simulator for UAV Tracking. In *Proc. ECCV*. 445–461.
Matthias Müller, Vincent Casser, Jean Lahoud, Neil Smith, and Bernard Ghanem. 2018. Sim4CV: A Photo-Realistic Simulator for Computer Vision Applications. *IJCV* (24 Mar 2018). https://doi.org/10.1007/s11263-018-1073-7
Gustavo Olague and Roger Mohr. 2002. Optimal Camera Placement for Accurate Reconstruction. *Pattern Recognition* 35, 4 (2002).
Molly A O'Neil, Dan Tamir, and Martin Burtscher. 2011. A Parallel GPU Version of the Traveling Salesman Problem. In *Proc. PDPTA*.
Jeremie Papon and Markus Schoeler. 2015. Semantic Pose using Deep Networks Trained on Synthetic RGB-D. *CoRR* abs/1508.00835 (2015). http://arxiv.org/abs/1508.00835
Ken Perlin. 2002. Improving Noise. *ACM Trans. Graph.* 21, 3 (2002).
Norbert Pfeifer, Philipp Glira, and Christian Briese. 2012. Direct Georeferencing with on Board Navigation Components of Light Weight UAV Platforms. *International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences* (2012).
Weichao Qiu, Fangwei Zhong, Yi Zhang, Siyuan Qiao, Zihao Xiao, Tae Soo Kim, Yizhou Wang, and Alan Yuille. 2017. UnrealCV: Virtual Worlds for Computer Vision. *ACM Multimedia Open Source Software Competition* (2017).
Stephan R. Richter, Zeeshan Hayder, and Vladlen Koltun. 2017. Playing for Benchmarks. In *Proc. ICCV*.
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. ICCV*.
Mike Roberts and Pat Hanrahan. 2016. Generating Dynamically Feasible Trajectories for Quadrotor Cameras. *ACM Trans. Graph.* 35, 4 (2016).
German Ros, Laura Sellart, Joanna Materzynska, David Vazquez, and Antonio Lopez. 2016. The SYNTHIA Dataset: A Large Collection of Synthetic Images for Semantic Segmentation of Urban Scenes. In *Proc. CVPR*.
Markus Rumpler, Arnold Irschara, and Horst Bischof. 2011. Multi-View Stereo: Redundancy Benefits for 3D Reconstruction. In *Proc. Workshop of Austrian Assoc. for Patt. Recog.*, Vol. 4.
Markus Rumpler, Alexander Tscharf, Christian Mostegel, Shreyansh Daftry, Christof Hoppe, Rudolf Prettenthaler, Friedrich Fraundorfer, Gerhard Mayer, and Horst Bischof. 2016. Evaluations on Multi-Scale Camera Networks for Precise and Geo-Accurate Reconstructions from Aerial and Terrestrial Images with User Guidance. *Computer Vision & Image Understanding* (2016).
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. Inteligent & Robotic Sys.* 65, 1-4 (2012).
Johannes Lutz Schönberger and Jan-Michael Frahm. 2016. Structure-from-Motion Revisited. In *Conference on Computer Vision and Pattern Recognition (CVPR)*.
Thomas Schöps, Johannes L. Schönberger, Silvano Galliani, Torsten Sattler, Konrad Schindler, Marc Pollefeys, and Andreas Geiger. 2017. A Multi-View Stereo Benchmark with High-Resolution Images and Multi-Camera Videos. In *Proc. CVPR*.
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. CVPR*, Vol. 1.
Ben Semerjian. 2014. A New Variational Framework for Multiview Surface Reconstruction. In *Proc. ECCV*.
Shital Shah, Debadeepta Dey, Chris Lovett, and Ashish Kapoor. 2017. AirSim: High-Fidelity Visual and Physical Simulation for Autonomous Vehicles. In *Proc. Field and Service Robotics*.
Ching-Kuang Shene. 1998. Curve Global Interpolation. (1998). https://www.cs.mtu.edu/~shene/COURSES/cs3621/NOTES/INT-APP/CURVE-INT-global.html
Amarjeet Singh, Andreas Krause, and William J. Kaiser. 2009. Nonmyopic Adaptive Informative Path Planning for Multiple Robots. In *IJCAI*.
Noah Snavely, Steven M Seitz, and Richard Szeliski. 2006. Photo Tourism: Exploring Photo Collections in 3D. *ACM Trans. Graph.* 25, 3 (2006).
Noah Snavely, Steven M. Seitz, and Richard Szeliski. 2008. Skeletal Graphs for Efficient Structure from Motion. In *Proc. CVPR*, Vol. 1.
Christoph Strecha, Wolfgang von Hansen, Luc van Gool, Pascal Fua, and Ulrich Thoennessen. 2008. On Benchmarking Camera Calibration and Multi-View Stereo for High Resolution Imagery. In *Proc. CVPR*.
W. Treible, P. Saponaro, S. Sorensen, A. Kolagunda, M. O'Neal, B. Phelan, K. Sherbondy, and C. Kambhamettu. 2017. CATS: A Color and Thermal Stereo Benchmark. In *Proc. CVPR*. 134–142.
J. Irving Vasquez-Gomez, L. Enrique Sucar, Rafael Murrieta-Cid, and Efrain Lopez-Damian. 2014. Volumetric Next-Best-View Planning for 3D Object Reconstruction with Positioning Error. *Int. J. Adv. Robotic Sys.* 11 (2014).
Michael Waechter, Mate Beljan, Simon Fuhrmann, Nils Moehrle, Johannes Kopf, and Michael Goesele. 2017. Virtual Rephotography: Novel View Prediction Error for 3D Reconstruction. *ACM Trans. Graph.* 36, 1, Article 8 (Jan. 2017), 11 pages.
Ingo Wald. 2007. On Fast Construction of SAH-Based Bounding Volume Hierarchies. In *Proc. Symp. Interactive Ray Tracing*.
Andreas Wendel, Michael Maurer, Gottfried Graber, Thomas Pock, and Horst Bischof. 2012. Dense Reconstruction on-the-fly. In *Proc. CVPR*.
Stefan Wenhardt, Benjamin Deutsch, Elli Angelopoulou, and Heinrich Niemann. 2007. Active Visual Object Reconstruction using D-, E-, and T-Optimal Next Best Views. In *Proc. CVPR*.
Shihao Wu, Wei Sun, Pinxin Long, Hui Huang, Daniel Cohen-Or, Minglun Gong, Oliver Deussen, and Baoquan Chen. 2014. Quality-driven Poisson-guided Autoscanning. *ACM Trans. Graph.* 33, 6, Article 203 (Nov. 2014), 12 pages. https://doi.org/10.1145/2661229.2661242
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. Graph.* 37, 4 (2018), 88:1–88:13.