Rapid urban 3D modeling for drone-based situational awareness assistance in emergency situations

Inge Coudron and Toon Goedemé

KU Leuven, EAVISE Research Group,
Jan Pieter De Nayerlaan 5, Sint-Katelijne-Waver, Belgium
{inge.coudron;toon.goedeme}@kuleuven.be

Abstract. Drones are becoming a necessary and invaluable tool in many industries, as well as in emergency response situations. They can assist emergency-services in hazardous situations to get better situational awareness. This may lead to an improved rescue-coordination, increased personal safety for agents in the field and less personal, physical and financial damages as a result of a faster and better intervention. Photo-realistic 3D models generated from the drone video data, for example, can provide situational awareness as it is easier to understand the scene by visualizing it in 3D. The 3D model can be viewed from different perspectives and provides an instant overview of the situation. In contrast to SLAM which is fast but sparse, and SfM-MVS which is dense but slow, we present a pipeline that produces a dense photo-realistic 3D model of the event site in near real time by fusing oblique images with pre-recorded, publicly available LiDAR datasets.

Keywords: computer vision, 3D model, large-scale reconstruction, real time processing, geo-registration

1 Introduction

As a first step towards supporting emergency response operations with improved situational awareness, this paper investigates a method for rapid 3D modeling of the emergency scene using drone images. The obtained 3D model can serve as the basis for a decision support system through which additional semantic information about the scene can be dynamically added and visualized. These include, for example, the location of other first responders and victims, possible hazards or a temperature texture overlay acquired with a thermal camera.
Two types of techniques have been widely used for the 3D reconstruction of outdoor environments, namely Structure from Motion (SfM) [4] and Simultaneous Localization and Mapping (SLAM) [3]. The former is traditionally performed in an off-line fashion on an unordered set of images potentially taken in different conditions. On the other hand, visual SLAM using only a camera, is supposed to work in real-time on an ordered sequence of images acquired from a fixed camera set-up. Both

techniques result in a sparse point cloud corresponding to the locations of the estimated feature points. However, the low spatial resolution of this sparse representation limits the desired level of detail required for interpreting the 3D model.
In order to obtain a more visually appealing 3D reconstruction, the sparse point cloud is often first densified using a Multi-view Stereo (MVS) approach such as PMVS [6] or SURE [10]. Next, a surface reconstruction algorithm computes a triangle mesh that can be textured. These last two steps are computationally expensive and can take up to several hours or days even on a modern desktop, depending on the number of images and their resolution. However, since emergency response is a very time-critical application of imaging and 3D reconstruction, these long processing times are not feasible in a (near) real-time environment.
To conclude, the sparse point cloud reconstructed by SfM or SLAM can be computed efficiently in (near) real-time, but it is not visually appealing. On the other hand, performing additional steps to create a textured mesh is computationally too expensive in case of emergency response applications. A lot of research has already been done to produce denser SLAM [3] or faster MVS [8], but none of it is conclusive. These contradicting demands can only be met by adding a priori information. Therefore, instead of using only the images from the drone, we propose to fuse these images with publicly available airborne LiDAR data from which a complete but untextured 3D model can be reconstructed in advance.
The focus in our application is on building emergencies such as fire. When an emergency call comes in, the drone is directed towards the building of interest. During the flight, a virtual 3D model of the emergency scene can already be constructed based on the LiDAR data. The drone flies around the building of interest and captures images at various viewpoints. These images are individually registered with the LiDAR data to infer the camera poses. In combination with the computed camera poses, the images can be used to texture the 3D model.

2 Related work

The most challenging task in the proposed workflow (Fig. 1) is registering the 2D images with the 3D LiDAR data. There are a number of different issues that make this task very difficult. To start with, there is a dimensionality gap. Furthermore, the datasets are likely captured under different circumstances such as different seasons, different traffic conditions or environmental changes due to rebuildings. The 2D-3D matching can be described as a camera pose estimation problem. The camera parameters, location and orientation, have to be determined from a single perspective image with respect to the 3D coordinates of a real world scene. Different approaches have already been suggested to solve this problem. These can be largely divided into two categories: feature-based or image-based.
The feature-based approaches try to match geometric features such as lines or points. In [13] , for example, line segments in both the 2D images and the 3D LiDAR data are extracted. For a given set of camera parameters, the 3D line segments are projected in 2D. Using a feature called

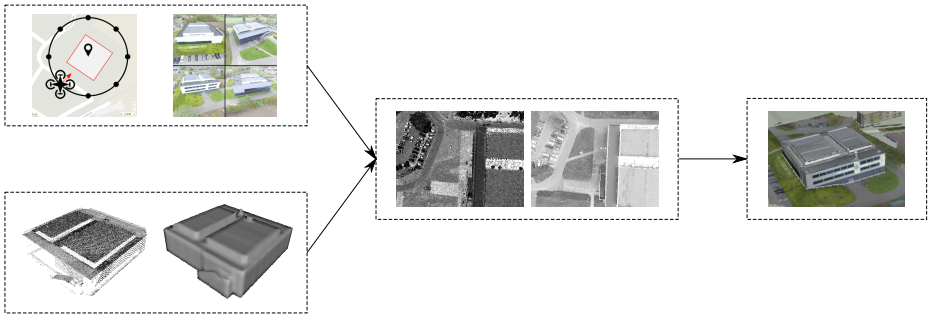


Fig. 1. Urban 3D modeling workflow: (Section 3.1) images from the building of interest are captured with a drone flying along a circular path with the camera pointing towards the building, (Section 3.2): a complete 3D model is generated in advance from pre-recorded publicly available LiDAR data, (Section 3.3) the camera pose for a single view is determined by maximizing the similarity between the captured image and a synthetically generated view from that pose. The registered oblique images can be mapped onto the 3D model using open source software such as *texrecon* [12]

3 connected segments (3CS),the line segments from the images and LiDAR data can be matched. Two matches are sufficient to compute the homography, but in order to make it more robust a RANSAC approach is used. The method works well on high altitudes where a lot of building contours are visible. However, it is expected to perform less good in our situation, where a single building takes up most part of the image and therefore drastically reduces the number of building contours that can be used for matching.

Another feature-based approach using point correspondences was proposed by [11]. They try to match 2D image features and 3D scene points, reducing it to a perspective-n-point problem. However, this method assumes the point cloud is obtained from an offline Structure-from-Motion reconstruction. As a consequence, every 3D point can be associated with at least two image descriptors such as SIFT or SURF. By extracting local features in the query image and matching them to the database descriptors, correspondences are established between the 2D features and 3D points. From these 2D-3D correspondences the camera pose can be estimated.

Image-based approaches on the other hand generally render the 3D points to generate 2D synthetic views in order to bridge the dimensionality gap. In [2], the camera pose estimation was reduced to a perspective-n-point problem as well. In this case, 2D-2D point correspondences between the query image and synthetic view were established by matching local features. The 3D positions of the synthetically generated can easily be determined by using the depth buffer information from the rendering procedure. Note that in this case terrestrial LiDAR data was used instead of airborne LiDAR data. Using airborne LiDAR data would probably have a negative impact on the algorithm since building facades,

which takes up most part in oblique images, are only represented sparsely leading to large gaps in the synthetic views.
The most similar to our approach is [9]. It tries to register the synthetic view with respect to the query image by minimizing a mutual information metric. The downhill simplex optimizer is utilized to infer the camera pose parameters. However, as suggested by [14], the downhill simplex optimizer only works best for small perturbations. Therefore, in our approach a different optimizer is used, that is more robust to coarse initializations.

3 Materials and methods

3.1 Drone-based image acquisition

For image acquisition a DJI Phantom 3 was used as flying platform. The Phantom 3 is equipped with a 12 megapixel camera with a resolution of 4000 × 3000 pixels. To ensure the building of interest is seen from all general view angles, the drone flies on a circular path around the building with the optical axis of the camera facing towards the center. The images in our dataset were captured at an altitude of approximately 35 meters under an angle of 30 degrees.

3.2 Surface reconstruction from airborne LiDAR data

The underlying 3D mesh onto which the texture will be mapped, is inferred from pre-recorded and publicly available airborne LiDAR data. The process for generating a triangle mesh from the point cloud data is shown in Fig. 2. First, the 3D point cloud is converted to a 2.5D Digital Elevation Model (DEM). Then, it is extruded into a voxel grid based on the elevation data. The result is again a 3D point cloud, but now the holes in the building facades, which are typical for airborne LiDAR data, are filled. From this 3D point cloud a proper triangle mesh can be computed.

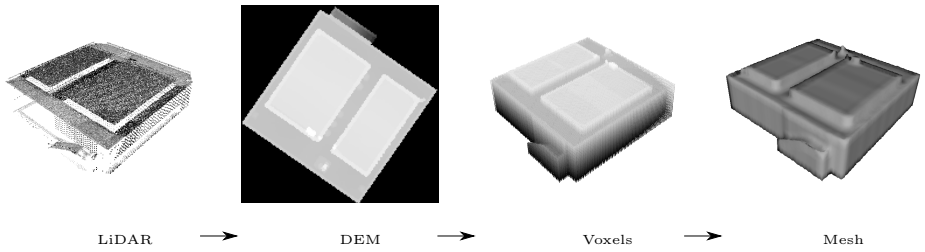


Fig. 2. Surface reconstruction from airborne LiDAR data: A Digital Elevation Model (DEM) is generated from the 3D points belonging to the building of interest. The 2.5D grid is extruded back into 3D to fill the holes in the facades. A watertight mesh is computed from the cleaned up point cloud.

Since the LiDAR data is georeferenced, we can use the building footprints from GIS data to extract the points belonging to the building of interest. First, the 3D points are projected onto a 2D grid. The resolution of the grid was chosen with respect to the point density of our LiDAR data. The average point density is 8 points per square meter. Therefore, the grid resolution was set to $\sqrt{(}8) \approx 0.35$. In each grid cell the highest elevation value is retained. Next, an inverse distance weighted square moving window of 7 cells is used to fill cells in the grid that have null values. Then, the 2D grid cells are extruded into a 3D voxel grid depending on the elevation. The actual mesh is generated by computing the concave hull of all occupied voxel centers. The same process can be repeated for other neighboring buildings in order to create a complete virtual urban 3D model as can be seen in Fig. 3.

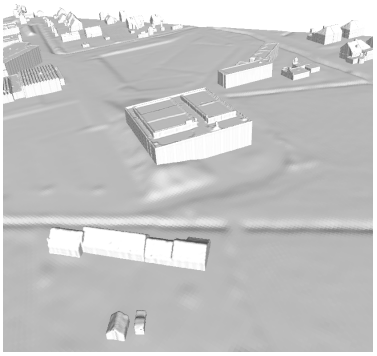


Fig. 3. Urban 3D model: A complete virtual 3D world is generated by performing the processing steps from the previous figure on all neighboring buildings.

3.3 Registration of LiDAR data and images

The basic elements of our image registration framework are described in Fig. 4. A similarity metric is computed to quantify the similarity between the query image and the synthetic view. The optimizer searches the camera pose that maximizes this similarity metric. In each optimization step, the new camera parameters are used to rerender the synthetic view. These three steps are repeated until a stop criterion is met.

Synthetic 2D View Generation A direct point based rendering approach is applied for the synthetic view generation. To this end, we use OpenGL's GL_POINTS to render the 3D point cloud data. In OpenGL a calibrated camera can be simulated by computing the projection matrix from the intrinsic parameters of the camera according to Equation 1, where f_x, fy are the focal lengths, c_x, c_y the coordinates of the principal

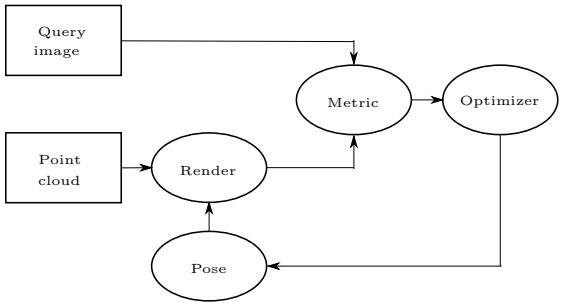
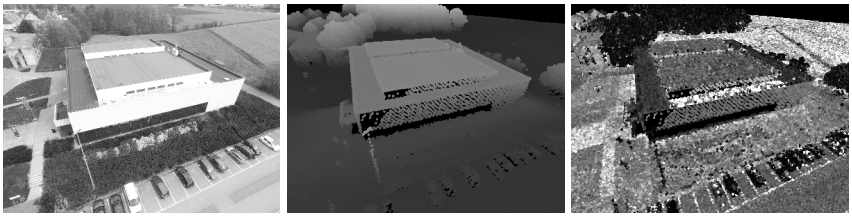


Fig. 4. Interactions among the components of our image registration technique

point, w, h the width and height of the image and f, n the far and near clipping plane. The modelview matrix represents the coordinate system transformation for the given camera pose. By setting these two matrices, the point cloud can be rendered as if seen from that camera viewpoint.

$$\begin{bmatrix} \frac{2 \cdot f_x}{w} & 0 & \frac{2 \cdot c_x}{w} - 1 & 0 \\ 0 & \frac{2 \cdot f_y}{h} & \frac{2 \cdot c_y}{h} - 1 & 0 \\ 0 & 0 & -\frac{f+n}{f-n} & \frac{-2 \cdot f \cdot n}{f-n} \\ 0 & 0 & -1 & 0 \end{bmatrix} \tag{1}$$

Since the resolution of the images is very large compared to the spatial resolution of the LiDAR data, rendering the point cloud using the original camera parameters would result in a very sparse image. Therefore, the images were resized to 10% of the original image size. Consequently, the camera parameters were rescaled accordingly. An example of such a synthetic 2D view generated using OpenGL is shown in Fig. 5.



(a) Query image: captured by the drone

(b) LiDAR based height-encoded image

(c) LiDAR based intensity-encoded image

Fig. 5. Synthetic 2D view generation from 3D LiDAR data

Similarity metric calculation An important design choice in image registration is the selection of an appropriate similarity measure. In this work, mutual information was chosen as similarity measure since it

has proven to be accurate and robust for multi-modal matching [7]. It measures the mutual dependence between the underlying intensity distributions of the images. The larger this measure the stronger the dependence. The mutual information between an image A and B is calculated as:

$$MI(A,B) = \sum P_{A,B}(a,b) \log \frac{P_{A,B}(a,b)}{P_A(a) \cdot P_B(b)} \tag{2}$$

The joint probability matrix can be computed as the normalized 2D histogram, were the normalization factor equals the number of histogram bins N. The sum of the rows and the columns, respectively, gives the marginal probability distributions of image A and B:

$$\begin{aligned} P_{A,B}(a,b) &= \frac{h(a,b)}{N} \\ P_A(a) &= \sum_b P_{A,B}(a,b) \\ P_B(b) &= \sum_a P_{A,B}(a,b) \end{aligned} \tag{3}$$

Intensity based optimization In this work, an evolutionary algorithm is used for image registration. Evolutionary algorithms have been proven to be a promising solution, since they are able to perform a robust search in complex search spaces like those arising in image registration [14]. The structure of a classic evolutionary algorithm is as follows:

Algorithm 1 Evolutionary algorithm

```
INITIALIZE population with random candidate solutions
EVALUATE each candidate solution
while Termination condition not reached do
  SELECT individuals for the next generation
  RECOMBINE pairs of parents
  MUTATE the resulting offspring
  EVALUATE each candidate solution
end while
```

The construction of the actual evolutionary algorithm can be divided into three parts: (1) the definition of an appropriate structure to represent the solution; (2) determination of the fitness function; and (3) the design of the genetic operators. In our work a candidate solution encodes the 6-DOF camera pose consisting of the 3-DOF orientation and 3-DOF position. As described earlier, mutual information is used to evaluate the fitness of a candidate. The genetic operators are the core of the evolutionary algorithm. They define how to generate new solutions from the existing ones. First, we reduce the number of individuals by keeping only 40% of the solutions based on their fitness value. This is in fact called a 'rank selection'. Next, two parent solutions are selected based

on the closest Euclidean distance of their variables. An offspring of these parents is calculated as the average of the variables of both parents. Finally, each offspring is mutated by adding random perturbations to the original variables.

4 Experimental results

We show the results of our algorithm with a collection of five images taken from a building on our campus. As no GPS/INS data was available, the initial poses were selected manually. For this a georeferenced orthomosaic image and the corresponding digital elevation model was used. First, matching feature points were selected in the query image and orthomosaic. Using the raster information from the orthomosaic and the digital elevation model, the 3D coordinates of the point correspondences could be computed. Finally, solving the Perspective-n-Points problem yielded an initial camera pose, which also serves as the ground truth pose.
Several experiments were conducted. First, different variations of the similarity measure were compared. To this end, the mutual information is calculated while the camera parameter being evaluated is varied. The other parameters are held constant at their correct values. We experimented with the number of histogram bins N used for calculating the mutual information. As suggested in [7], using Sturges Rule for calculating the number of histogram bins not only makes the image registration more efficient but also more accurate. The same conclusion can be drawn for our image set as can be seen in Fig. 6 where the similarity measure is plotted for small perturbations from the estimated ground truth pose. As the top graph of Fig. 6 shows, using 256 bins results in a shift of the maximum, which is the optimal point, for certain degrees of freedom. As the LiDAR data also contains intensity values, we also explored the mutual information between the query image and synthetic intensity image. Fig. 6 shows that this metric is too noisy for accurate registration. To conclude, the metric using only depth information and 18 histogram bins performed best.
Next, the robustness of our algorithm was tested by artificially perturbing the camera positions. The translation parameters were randomly perturbed by maximally $\pm 5m$ and the rotation parameters were perturbed by maximally $\pm 3^\circ$. For each of the images 100 coarse initializations were simulated this way. The mean deviation of the optimized camera parameters from the estimated ground truth pose is shown in Fig. 7. It was apparent by visual inspection that the order of magnitude of the deviations is close enough for texture mapping.
Finally, our approach was visually compared with current state-of-the-art open source Structure from Motion approaches, namely Multi-View Environment (MVE) [5] and OpenDroneMap (ODM) [1]. For the texture mapping of the oblique images onto our virtual 3D model, the same library [12] was used as in MVE and ODM. Both approaches produced a 3D model in about half an hour. With our approach, it takes on average $4s$ to register a single oblique image to the LiDAR data. In contrast to the Structure from motion approaches, we don't require any overlap between

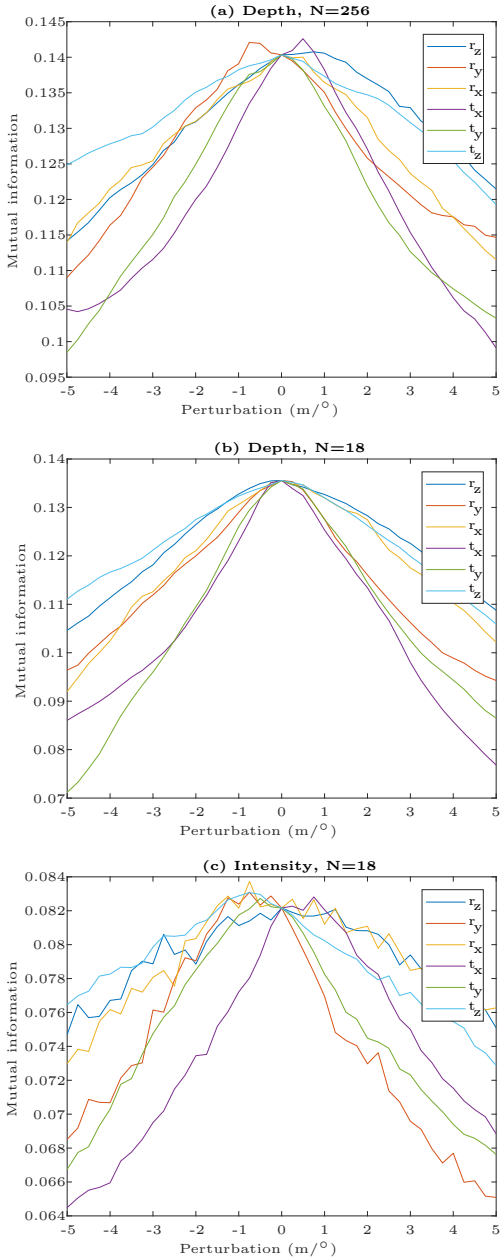


Fig. 6. Plots of the mutual information as each of the camera parameters (r_x: angle of rotation around the x-axis, t_x: translation in the direction of the x-axis, etc.) are perturbed from the estimated ground truth pose. (a) Mutual information between the query image and height-encoded LiDAR image using 256 histogram bins,(b) Mutual information between the query image and height-encoded LiDAR image using 18 histogram bins, (c) Mutual information between the query image and intensity-encoded LiDAR image using 18 histogram bins

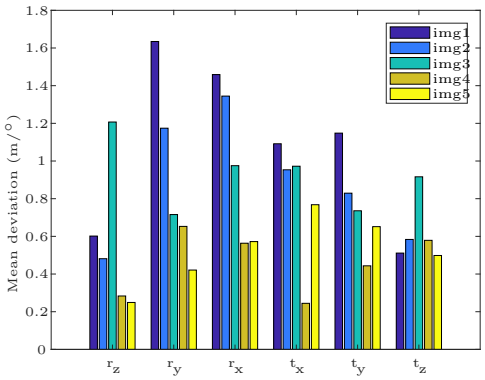
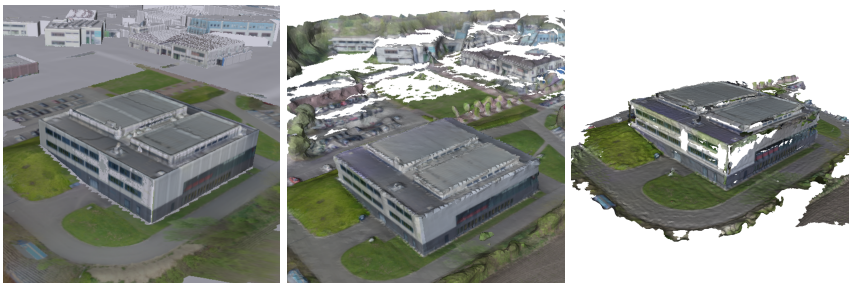


Fig. 7. Plots of the mean deviation of the camera parameters ($(r_x$: angle of rotation around the x-axis, t_x: translation in the direction of the x-axis, etc.) found by the optimization algorithm from the estimated ground truth pose for each image in our image set.

the images, as each image is registered individually. As a result we only require 5 images to texture the complete 3D model, which reduces the processing time significantly. Including the texture mapping, the total reconstruction will take on average $29s$. If we compare the models visually we can see another advantage (see Fig. 8) of our approach over the others. 3D models generated with SfM-MVS typically suffer from holes. As our model is pre-generated from LiDAR data, some parts of the mesh may not be textured, but at least all underlying structures are visible and accurate.



(a) Ours (b) ODM (c) MVE

Fig. 8. Comparison of different 3D modeling approaches

5 Conclusions and future work

We have presented our custom pipeline for the registration of 3D LiDAR data and 2D oblique images. The registration of multi-modal 2D/3D datasets is inherently difficult due to fundamental different nature of the data. To overcome this problem, the dimensionality gap was first bridged by rendering the 3D point cloud into 2D for the camera poses computed in each optimization step. The similarity between these multi-modal images was measured using mutual information. Our experimental results show that this choice is indeed suitable for multi-modal registration. We further demonstrated the robustness of our approach by randomly perturbing the camera parameters from the ground truth.
Future research plans include reconsidering using the intensity values from the LiDAR data for matching. The results revealed that on our image set this metric did not demonstrate adequate quasi-convexity. This is probably due to the noisy nature of the rendered synthetic 2D view. One possibility is to apply a preprocessing filter on the rendered image, that removes the noise such as anisotropic diffusion filter. Another possibility is to use a variant of the mutual information metric such as Quadratic Mutual Information. We will also explore the applicability of our approach for registering thermal images with the LiDAR data as this is certainly relevant for improving situational awareness for firefighters.

References

1. S. Fitzsimmons B. Dakota and P. Toffanin. Opendronemap, 2017.
2. Christoph Bodensteiner, Marcus Hebel, and Michael Arens. *Accurate Single Image Multi-modal Camera Pose Estimation*, pages 296–309. Springer Berlin Heidelberg, Berlin, Heidelberg, 2012.
3. J. Engel, T. Schöps, and D. Cremers. LSD-SLAM: Large-scale direct monocular SLAM. September 2014.
4. David Forsyth and Jean Ponce. *Computer vision: a modern approach*. Upper Saddle River, NJ; London: Prentice Hall, 2011.
5. Simon Fuhrmann, Fabian Langguth, and Michael Goesele. Mve: A multi-view reconstruction environment. In *Proceedings of the Eurographics Workshop on Graphics and Cultural Heritage*, GCH '14, pages 11–18, Aire-la-Ville, Switzerland, Switzerland, 2014. Eurographics Association.
6. Y. Furukawa and J. Ponce. Accurate, dense, and robust multi-view stereopsis. In *2007 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8, June 2007.
7. Rosin P. Marshall D. Legg, P. and J. Morgan. Improving accuracy and efficiency of registration by mutual information using sturges histogram rule. *Medical Image Understanding and Analysis*, pages 26–30, 2007.
8. A. Locher, M. Perdoch, and L. V. Gool. Progressive prioritized multi-view stereo. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3244–3252, June 2016.

9. A. Mastin, J. Kepner, and J. Fisher. Automatic registration of lidar and optical images of urban scenes. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 2639–2646, June 2009.
10. Mathias Rothermel, Konrad Wenzel, Dieter Fritsch, and Norbert Haala. Sure: Photogrammetric surface reconstruction from imagery. 2013.
11. Torsten Sattler, Bastian Leibe, and Leif Kobbelt. *Improving Image-Based Localization by Active Correspondence Search*, pages 752–765. Springer Berlin Heidelberg, Berlin, Heidelberg, 2012.
12. Michael Waechter, Nils Moehrle, and Michael Goesele. Let there be color! — Large-scale texturing of 3D reconstructions. In *Proceedings of the European Conference on Computer Vision*. Springer, 2014.
13. Lu Wang and U. Neumann. A robust approach for automatic registration of aerial images with untextured aerial lidar data.
14. Xiujie Zhang, Yi Shen, and Shiyong Li. A hybrid genetic algorithm for medical image registration with downhill simplex method.