International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences, Volume XL-5/W1, 2013
3D-ARCH 2013 - 3D Virtual Reconstruction and Visualization of Complex Architectures, 25 – 26 February 2013, Trento, Italy

LARGE SCALE URBAN RECONSTRUCTION FROM REMOTE SENSING IMAGERY

Georg Kuschk

Remote Sensing Technology Institute, German Aerospace Center (DLR), D-82234 Wessling, Germany
georg.kuschk@dlr.de

KEY WORDS: Stereo, Reconstruction, Remote Sensing, Meshing, Texturing, 3D Modeling

ABSTRACT:

Automatic large-scale stereo reconstruction of urban areas is increasingly becoming a vital aspect for physical simulations as well as for rapid prototyping large scale 3D city models. In this paper we describe an easily reproducible workflow for obtaining an accurate and textured 3D model of the scene, with overlapping aerial images as input. Starting with the initial camera poses and their refinement via bundle adjustment, we create multiple heightmaps by dense stereo reconstruction and fuse them into one Digital Surface Model (DSM). This DSM is then triangulated, and to reduce the amount of data, mesh simplification methods are employed. The resulting 3D mesh is finally projected into each of the input images to obtain the best fitting texture for each triangle. As verification, we provide visual results as well as numerically evaluating the accuracy by comparing the resulting 3D model against ground truth generated by aerial laser scanning (LiDAR).

1 INTRODUCTION

Large-scale stereo reconstruction of urban areas is increasingly becoming more important in varied scientific fields as well as in common life. Applications are ranging from physical simulations like flood simulation, the propagation of radio beams or sound, to 3D change detection, flight planning for unmanned aerial vehicles (UAV's), or intuitive and detailed navigation and 3D city visualization. Due to the large-scale attribute of the given problem, the stereo reconstruction problem needs to be addressed with remote sensing imagery arising from satellites or aerial sensors and furthermore needs to be fully automatic, as the manual effort would render the problem intractable.

While laser scanning (LiDAR) is a very accurate method for 3D reconstruction, it does not provide any color or texture information about the scene and requires a quite expensive hardware setup. In case of satellite platforms (very large scale reconstruction), laser scanning is not feasible at all. This paper is demonstrating the potential of dense stereo image matching as a cost-efficient way to create accurate and detailed textured 3D city models by the combination of current state-of-the-art computer vision methods. We further show that the accuracy of today's optical sensors and dense stereo algorithms is indeed sufficient enough to meet the requirements of the aforementioned applications - approximately 1m RMSE for aerial images (2km distance to the scene) and 3m RMSE for WorldView-2 satellite images (770km distance to the scene).

When acquiring the images, an initial pose is computed and stored along the corresponding image. In case of aerial images, data from the inertial measurement unit (IMU) and GPS information is used for estimating the initial pose. In case of satellite images, a Rational Polynomial Camera (RPC) model (see e.g. Grodecki, 2001) is initially provided for every scene. In both cases these initial estimates are not very accurate, with the average re-projection error typically > 1 pixel. Therefore we extract and match SIFT features (Lowe, 2004) between all images and optimize the relative camera poses using bundle adjustment (a survey can be found in Triggs et al., 2000) to achieve subpixel accuracy. If available, ground control point are added to the optimization process to further refine the absolute orientation of the cameras. If no initial pose is available, we compute the fundamental matrix for each image pair based on SIFT matches, and perform an image rectification on the two images.

For reconstructing the 3D scene, we use dense stereo matching to make use of all the image information and compute a depth for each single image pixel - compared to sparse multiview image matching methods based on SIFT- (Lowe, 2004) or SURF-like (Bay et al., 2006) features. As for each image pixel multiple depth hypotheses need to be tested, a computational efficient cost function is needed for the image matching, which additionally has to be robust to some radiometric changes between the images. For this case we chose a combination of the Census transform (Zabih and Woodfill, 1994) and Mutual Information (Viola and Wells III, 1997). As the pure data term of the cost function is still prone to noise and performs poorly over large disparity ranges and homogeneously textured regions, a simple pixel-wise Winner-takes-all strategy does not yield acceptable results, which is why we add additional smoothness constraints and solve the resulting optimization problem using the well established Semi Global Matching (Hirschmueller, 2005). The whole process of the 3D reconstruction is described in Section 2.1.

At this point, we have a heightmap for each input stereo image-pair, which is equivalent to a point cloud in the coordinate system of the reference image, each point also having a color assigned by it's projection in the corresponding reference image. An intermediate result would now be the fusion of all these point clouds into an even denser point cloud. However, for our purpose, we transform the different heightmaps (respectively their point clouds) into a common orthographic coordinate system (UTM coordinate system) and due to a large image overlap and therefore redundant height information, fuse the projected Digital Surface Models (DSMs) into one single DSM, yielding a higher accuracy because of reduced noise. Since all of our applications need a closed surface, we afterwards transform this DSM (or point cloud) into a meshed 3D model by Delaunay triangulation (Guibas and Stolfi, 1985).

In the next step, we address the problem of the complexity of the 3D model. For applications having limited res sources (e.g. navigation devices or web applications) or real time requirements (collision detection for UAV's) the resulting mesh needs to be simplified, while at the same time preserving the accuracy and interesting scene details. As controversial as this may sound compared to the earlier mentioned argument for a dense reconstruction, note the difference at which step we reduce the amount of information. For sparse multiview image matching, the amount of information gets already reduced during the detection and match-

ing of sparse image features, whereas in our approach the reduction takes place at a later step and uses all the dense depth information to detect and simplify local planar surfaces. And to finally get a visually appealing and realistically textured 3D model, we use images of the scene taken from different viewpoints to assign best fitting textures for all triangles of the 3D model. The process of the automatic 3D modeling is described in Section 2.3.
For evaluation of the algorithm's accuracy, we provide both visual and numerical results for two test scenes of urban areas as described in Section 3. Test scene A consists of an aerial image set of the city of Munich, while Test scene B consists of WorldView-2 satellite images of London. The results of both data sets are numerically evaluated against reference data obtained by airborne laser scanning (LiDAR) - see Section 4.

2 METHOD

In Figure 1, an overview of the complete workflow of the proposed method is shown - starting with a number of input images I_k together with their initial camera poses C_k.

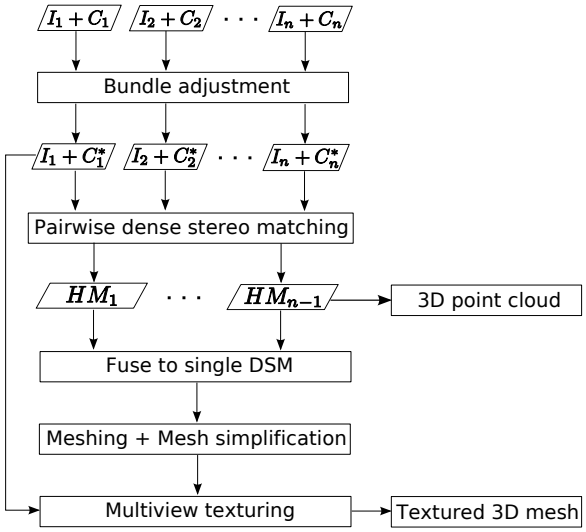


Figure 1: Workflow of the 3D reconstruction process with n input images I_k and camera poses C_k, refined camera poses C_k^* and $n-1$ pairwise heightmaps HM_k.

As already mentioned in the introduction, we first refine the initial poses to achieve subpixel accuracy for the dense stereo matching. To this end we extract and match SIFT features between all images and optimize the relative camera poses using bundle adjustment. If available, ground control point are added to the optimization process to further refine the absolute orientation of the cameras.

2.1 Dense Stereo Reconstruction

In classical dense stereo reconstruction, for every image pixel of the reference image $(x, y) \in I_1$ and a number of height/depth hypotheses $\gamma \in \Gamma$, a matching cost is computed by back-projecting the pixel into 3D space, projecting the resulting 3D point into the second image $\rightarrow (x', y') \in I_2$ and comparing the image information of the two images at their corresponding positions. The result is the so called *disparity space image* (Bobick and Intille, 1999), containing the *raw* matching costs.
If no information about the camera model would be available, the search space for matching one pixel in image I_1 would be the the whole image I_2. To reduce the search space from 2D to 1D, we need to establish an epipolar geometry between image pairs. If the cameras can be approximated by the pinhole camera model, the resulting epipolar geometry is mapping one image coordinate in the first image to a corresponding line in the second image. However, satellite images are obtained using a push-broom camera (the CCDs are arranged one-dimensional instead of a two-dimensional array) and the resulting epipolar lines of an image pair using the corresponding Rational Polynomial Camera (RPC) model are not straight, but curved (Oh, 2011).
We therefore pursue a different strategy and establish the epipolar geometry between two images I_1 and I_2 directly by evaluating the function $F_{(1,2)}(\mathbf{x}, \gamma)$, which projects a pixel $\mathbf{x}$ from I_1 to I_2 using the disparity γ, for every single pixel of $I_1 \in \Omega$ and every possible disparity $\gamma \in \Gamma$ individually. As the camera model may be very complex and computationally expensive and $\Omega \times \Gamma$ quite large, we compute $F_{(1,2)}(\mathbf{x}, \gamma)$ only for a sparse (and uniformly distributed) set of grid points in $\Omega \times \Gamma$ space and store the results in a lookup-table. For all other points we compute the projected pixel by trilinear interpolation of the nearest grid points.
By embedding this procedure in a plane-sweep approach (Collins, 1996), we furthermore get a rotational invariant cost function without introducing additional efforts. Given a disparity γ, we sweep over the reference image I_1, sample image I_2 at the corresponding image position (x', y') and copy the obtained color / intensity to an image $\tilde{I}_2$ at the same position (x, y) as in the reference image. When computing the matching costs of a disparity hypotheses γ and the whole image I_1, we simply evaluate the cost function at the same position (x, y), using the same local support window in both I_1 and I_2 (see Figure 2).
Note that by using this approach we have no need for an image rectification and avoid the errors induced by projective distortions and the involved sampling and interpolation.

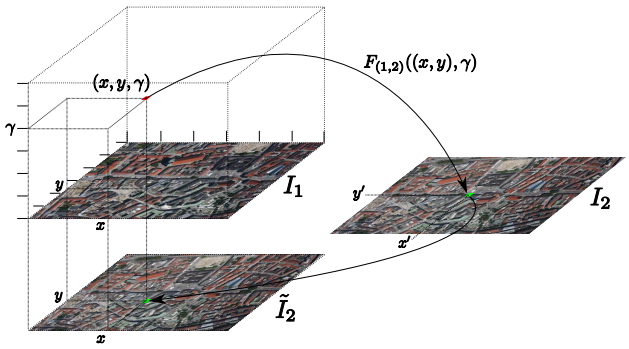


Figure 2: Computation of the disparity space image using a plane-sweep approach: For a coordinate (x, y) in image I_1 and disparity γ, obtain the corresponding coordinate (x', y') in image I_2 using the camera model $F_{1,2}$, sample the pixel color/intensity and copy it to the *warped* image $\tilde{I}_2$ at position (x, y). The matching cost can then be computed by comparing the pixel intensities in both images at position (x, y).

2.1.1 Cost Function

As cost function for the image matching we chose the Census transform (Zabih and Woodfill, 1994) and afterwards perform a (very small) local aggregation of it using Adaptive support-weights (Yoon and Kweon, 2006).
The Census transform CT encodes the local image structure within a small patch around a given pixel. It is defined as an ordered set of comparisons of intensity differences and therefore invariant to monotonic transformations which preserve the local pixel inten-

sity order:

$$\xi(I(\mathbf{x}), I(\mathbf{x}')) = \begin{cases} 1 & \text{if } I(\mathbf{x}') < I(\mathbf{x}) \\ 0 & \text{otherwise} \end{cases} \quad (1)$$

$$CT(I, \mathbf{x}) = \bigotimes_{[i,j] \in D} \xi(I(\mathbf{x}), I(\mathbf{x} + [i,j])) \ , \quad (2)$$

with the concatenation operator $\bigotimes$ and an ordered set of displacements $D \subset \mathbb{R}^2$, which is normally chosen to be a 7×9 window. Because due to implementation issues, the resulting binary vector of length 63 fits into a 64 Bit variable. The matching cost of different Census vectors s_1, s_2 is then computed as their Hamming distance $d_H(s_1, s_2)$ – number of differing bits – where highest matching quality is achieved for minimal Hamming distance, and the costs are normed to the real-valued interval $[0, 1]$

$$cost(\mathbf{x}, \gamma) = \frac{d_H \left(CT(I_1, \mathbf{x}),\ CT(I_2, F_{(1,2)}(\mathbf{x}, \gamma)) \right)}{\max_{i,j}\{d_H(s_i, s_j)\}} \quad (3)$$

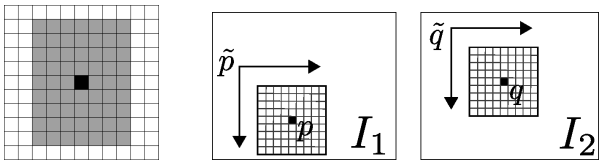
Using such a 7×9 support window for the Census transform increases the robustness of the matching function against mismatches, especially when searching through a large disparity range as is very common in remote sensing data.
On the other hand, this window-based matching suffers from the "foreground fattening" phenomenon when support windows are located on depth discontinuities, such as partially covering a roof top and the adjacent street. To limit this effect, we locally aggregate the cost function of Equation (3) using adaptive support-weights (Yoon and Kweon, 2006) for corresponding pixels p in I_1 and q in I_2:

$$C(p, q) = \frac{\sum_{\tilde{p} \in N_p, \tilde{q} \in N_q} [w(p, \tilde{p}) \cdot w(q, \tilde{q}) \cdot cost(\tilde{p}, \tilde{q})]}{\sum_{\tilde{p} \in N_p, \tilde{q} \in N_q} [w(p, \tilde{p}) \cdot w(q, \tilde{q})]} \quad (4)$$

The weights $w(p, q)$ are based on color differences $\Delta_c(p, q)$ and spatial distances $\Delta_d(p, q)$

$$w(p, q) = \exp\left(-\frac{\Delta_c(p, q)}{\gamma_c} - \frac{\Delta_d(p, q)}{\gamma_d} \right) \quad (5)$$

with $\gamma_d = 5$ (= radius of the support window) and $\gamma_c = 5.0$ for 8-bit images (respectively $\gamma_c = 20.0$ for 11-bit images). As this local aggregation favors fronto-parallel surfaces, we keep this radius relatively small (4 pixel), to keep a balance between increased accuracy along discontinuities and not "over-favoring" fronto-parallel surfaces.



(a) (b)

Figure 3: a) The 7×9 window D of the Census transform, b) The basic scheme for the adaptive support-weights.

2.1.2 Optimization In the resulting disparity space image (DSI) we now search for a functional $u(\mathbf{x})$ (the disparity map), which minimizes the energy function arising from the matching costs (called data term E_{data}), plus additional regularization terms. As the data term is prone to errors due to some incorrect and noisy measurements, one often needs some smoothness constraints E_{smooth}, forcing the surface of the disparity map to be locally smooth.

$$u(\mathbf{x}) = \underset{u}{\operatorname{argmin}} \left\{ \int_\Omega E_{data} + E_{smooth} \, d\mathbf{x} \right\} \quad (6)$$
$$= \underset{u}{\operatorname{argmin}} \left\{ \int_\Omega C(\mathbf{x}, u(\mathbf{x})) + \nabla u(\mathbf{x}) \, d\mathbf{x} \right\}$$

This energy is non-trivial to solve, since the smoothness constraints are based on gradients of the disparity map and therefore cannot be optimized pixelwise anymore. Various approximations for this NP-hard problem are existing, e.g. Semi-global Matching (Hirschmueller, 2005), energy minimization via graph cuts (Boykov et al., 2001) or minimization of Total Variation (Pock et al., 2008) - just to name three examples. Because of its simple implementation, low computational complexity and good regularization quality, we use semi-global matching for optimization, minimizing the following discrete energy term

$$E(u) = \sum_{\mathbf{x}} C(\mathbf{x}, u(\mathbf{x})) + \sum_{\mathbf{p} \in N_\mathbf{x}} P_1 \cdot T[|u(\mathbf{x}) - u(\mathbf{p})| = 1] + \sum_{\mathbf{p} \in N_\mathbf{x}} P_2 \cdot T[|u(\mathbf{x}) - u(\mathbf{p})| > 1] \quad (7)$$

The first term is the matching cost, the second and third term penalties for small and large disparity discontinuities between neighboring pixels $\mathbf{p} \in N_\mathbf{x}$. The key idea is now not to solve the intractable global 2D problem, but to approximate it by combining 1D solutions from different directions $\mathbf{r}$, solved via dynamic programming

$$u(\mathbf{x}) = \underset{u}{\operatorname{argmin}} \left\{ \sum_{\mathbf{r}} L_\mathbf{r}(\mathbf{x}, u(\mathbf{x}) \right\} \quad (8)$$

The aggregated cost $L_\mathbf{r}(\mathbf{x}, u(\mathbf{x}))$ in Equation (8) represents the cost of pixel $\mathbf{x}$ with disparity $d = u(\mathbf{x})$ along the direction $\mathbf{r}$ and is being computed as

$$\begin{aligned} L_\mathbf{r}(\mathbf{x}, d) = & C(\mathbf{x}, d) \ + \ \min(L_\mathbf{r}(\mathbf{x} - \mathbf{r}, d), \\ & L_\mathbf{r}(\mathbf{x} - \mathbf{r}, d-1) + P_1 \ , \ L_\mathbf{r}(\mathbf{x} - \mathbf{r}, d+1) + P_1 \ , \\ & \min_k L_\mathbf{r}(\mathbf{x} - \mathbf{r}, k) + P_2) \ - \ \min_k L_\mathbf{r}(\mathbf{x} - \mathbf{r}, k) \end{aligned} \quad (9)$$

The penalties are set to default values $P_1 = 0.4$ and $P_2 = 0.8$ (with the raw costs normed to $[0, 1]$). For deeper insights the reader is referred to the paper of Hirschmueller, 2005.

2.2 DSM Post-Processing

After obtaining the disparity map for each image pair, there are most certainly some errors / outliers left due to mismatches or occlusions. To further reduce the number of outliers and increase the accuracy of the disparity map, we apply the following post-processing steps:

- **Consistency check**
  By exchanging the two stereo images I_1 and I_2, we compute a second disparity map D_2 (with the reference frame now being I_2) and compare each disparity of D_1 with the corresponding disparity in D_2 - the so called *left-right check*. Ideally, these disparities should be equal (uniqueness constraint). If they differ by a value larger than θ_{LR}, the dispar-

ity at this position is considered to be invalid

$$D_1(\mathbf{x}) = \begin{cases} \frac{D_1(\mathbf{x})+D_2(\mathbf{x})}{2} & \text{if } |D_1(\mathbf{x}) - D_2(\mathbf{x})| < \theta_{LR} \\ \textit{invalid} & \text{otherwise} \end{cases} \tag{10}$$

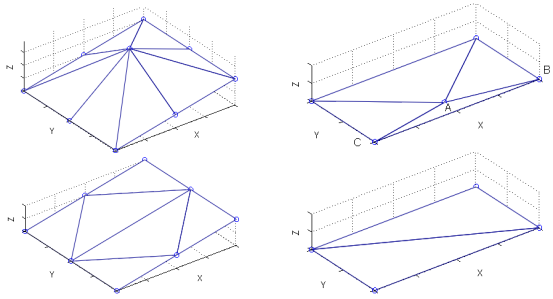
Note that this requires twice the time of the overall algorithm, since two disparity maps have to be computed.

- **Median filtering**
  Removing additional noise / small area outliers by filtering the disparity map with a 3×3 median filter
- **Interpolation**
  The invalidated disparities, resulting from occlusions and mismatches, need to be interpolated, which is mostly done based on the disparities in the local neighborhood. To this end, we implemented and use the multilevel interpolation based on B-splines from (Lee et al., 1997).
- **Subdisparity accuracy**
  By fitting a quadratic curve through the cost of the obtained disparity $C(\mathbf{x}, d)$ and the costs of the adjacent disparities (± 1) and computing the minimum of this curve, we refine the disparity map to subdisparity accuracy. Note that this is theoretically only valid for SSD-based cost functions, but nevertheless works fine for most cost functions.
- **Multiview DSM fusion**
  In the case of two input images, we just project the single disparity map into UTM coordinate system, discretize the result and again interpolate the hereby invalidated areas in the DSM (areas of the DSM which are occluded in the disparity map).
  For more than two input images, we compute a disparity map for each image pair and project all of them into a regular spaced and discretized grid given in UTM coordinate system. By collecting all the height values per DSM-pixel, we afterwards choose the median of these to be our final height value of the DSM at this position.

2.3 3D Modeling

2.3.1 Meshing We create a triangulated mesh of the DSM by simply connecting four incident pixels $(x, y), (x + 1, y), (x + 1, y + 1), (x, y + 1)$ into two triangles. Of the two possible triangulations we choose the one which better matches the height information by using an adaptive triangulation method minimizing the second derivative of the surface in the neighborhood of the square, as proposed in Grabner, 2002.

2.3.2 Simplification Using the naive meshing from above, the triangulated mesh roughly contains $2n$ triangles, if n is the number of pixels in the DSM. As our aerial image data for example has a resolution of 0.2m, our 3D model would have about 25 triangles per m^2 or $25 \cdot 10^6$ triangles per km^2 - which is simply too much for our purposes. Therefore, we further employ a two step mesh simplification to reduce the amount of triangles needed to represent the 3D model, while at the same time preserving its dominant features and surface properties.
The first step aims at simplifying planar structures. To this end, we iterate over all vertices and fit a 3D plane through its adjacent neighbors using the least squares method. Using a quad-edge data structure as in Guibas and Stolfi, 1985 allows an efficient search for the neighboring vertices.
If the minimum distance of the currently visited vertex to the fitted plane is $< \Delta_{plan}$, the vertex is deemed to be redundant and marked for removal (see Figure 4). As this would sometimes remove the corners of steep building walls, we add a further constraint that the vertex gets only removed, if the height difference to all of its adjacent vertices is $< \Delta_{disc}$. These two parameters depend on the grid resolution δ of the initial DSM and were chosen to be $\Delta_{plan} = \delta$ and $\Delta_{disc} = 10\delta$. An evaluation of the influence of this parameter δ is conducted in Section (4), Figure (10).
The second step of our mesh simplification is removing nearly collinear triangles. If for any triangle (A, B, C), $\overline{AB} + \overline{AC} < \overline{BC} \cdot \Delta_{coll}$ (with $\Delta_{coll} > 1$) the vertex A will be removed. We chose to remove only very collinear triangles ($\Delta_{coll} = 1.01$).



(a) (b)

Figure 4: a) Planar mesh simplification, b) Collinear mesh simplification

2.3.3 Multi-view texturing Using a single image for texturing is done straightforward: We project each triangle of our 3D model into the image plane, check whether the 2D projection faces towards the given camera and store the projected 2D image coordinates as normalized texture coordinates.
However, images of the scene taken from different viewpoints allow us to extract the texture of parts of the scene hidden from a single view, like for example the facades of buildings (see Figure 5). In that case we have to devise a quality measure Q for the projection $\pi(t_i, I_k)$ of a triangle t_i into each image I_k available for texturing. Of all these K projections, we then choose the one with the best quality measure for texturing the triangle t_i

$$k = \underset{k}{\text{argmax}} \{ \, Q(\, \pi(t_i, I_k) \,) \, \} \tag{11}$$

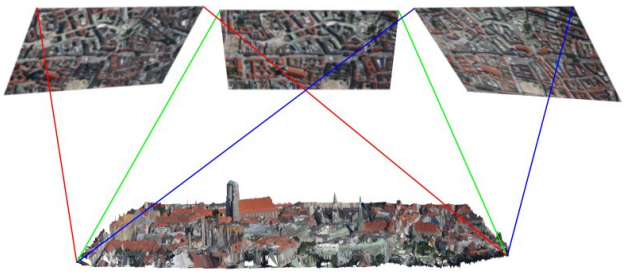


Figure 5: Multi-view texturing

To reduce the effect of perspective distortion, the angle between the normal vector of the 3D triangle and the looking vector of the camera should be minimal. However, at the same time the 2D projection of the triangle should have maximal size (to capture fine details) and it should be least occluded by other triangles (especially for urban areas containing large buildings and narrow streets). In practice, we found it sufficient to optimize just for the last two requirements, since our aerial image data was taken from roughly the same distance to the scene and therefore its a valid assumption, that the size of the 2D projection of a triangle correlates with the angle between its normal and the looking vector of

the camera.
We therefore extract a texture for a triangle t_i from the image I_k, where its projection on the image plane is maximal and simultaneously least occluded by other projected triangles. To solve these requirements efficiently, we compute the texture quality of each triangle for an image I_k using the following z-buffering approach:

- Sort all 3D triangles of the model according to the distance to the camera center
- Beginning with the furthest triangle, compute the projection of all triangles onto the image plane and render them using a unique identifier / unique color each
- Sweep over the rendered image and compute the visibility of each triangle in term of its remaining visible pixels:

$$Q(\pi(t_i, I_k)) = \sum_{(x,y)\in\pi} \left\{ \begin{array}{ll} 1 & (x,y) \text{ visible} \\ 0 & \text{otherwise} \end{array} \right\} \quad (12)$$

Using equation 11, we obtain the view where the best texture quality is to be expected and store the corners of the triangle's 2D projection as its texture coordinates.

3 DATA

For evaluation, two test sites of complex urban areas were chosen (see Figure 6 and 7), both covering an area of 2000×2000 pixel. The two test sites were captured with different remote sensing sensors and ground truth data was obtained by airborne laser scanning (LiDAR) in both cases.

3.1 Aerial image data - Munich

The aerial image data was taken of the inner city of Munich, using the 3K+ camera system (Kurz et al., 2007). The flight altitude (or distance of the camera to the scene) is $\approx$ 2km above ground with a ground sampling distance (GSD) of $\approx$ 0.2m per pixel. 17 partially overlapping images (converted to gray scale, 8 Bit per pixel) were used for the 3D reconstruction, and for texturing, the RGB images themselves were used. The resulting DSM is resampled to 0.2m and, having a size of 2000×2000 pixel, covers an area of 400m $\times$ 400m.

3.2 Satellite image data - London

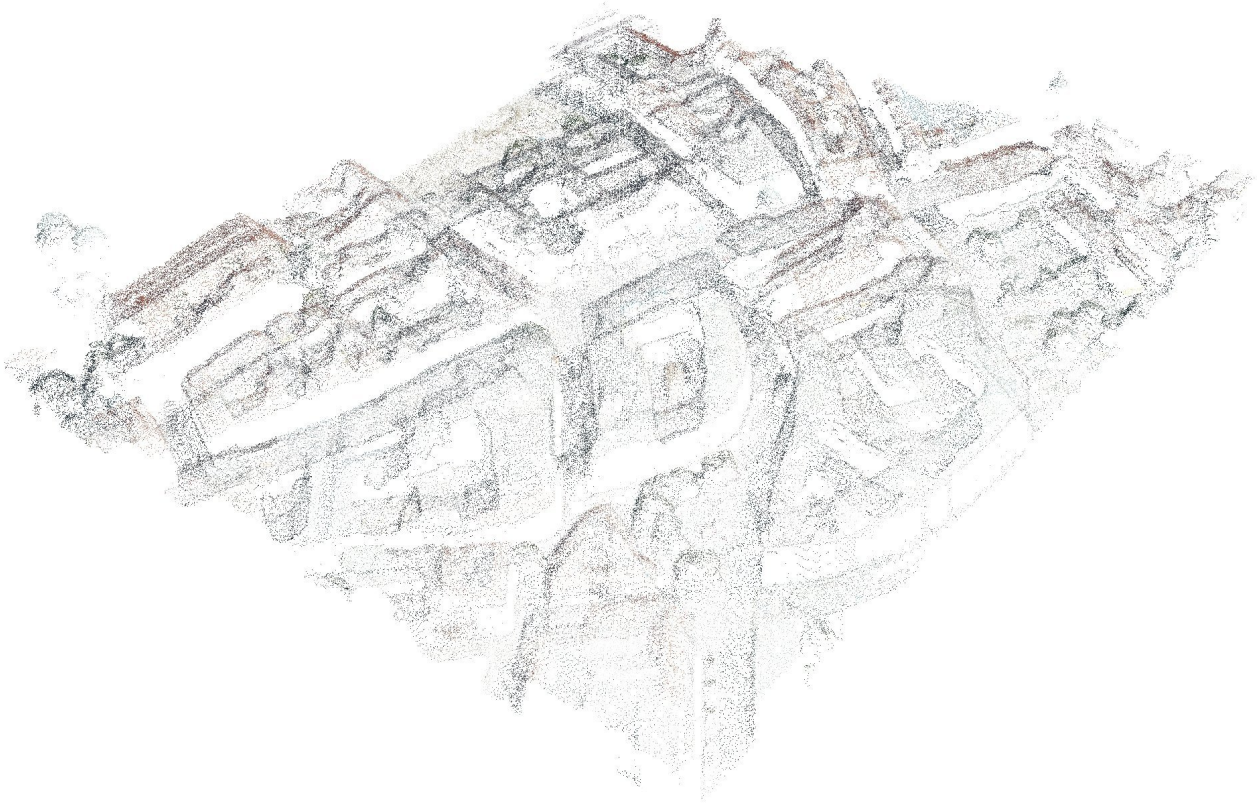
The satellite image data was taken of the inner city of London, using the WorldView-2 satellite. The flight altitude (or distance of the camera to the scene) is $\approx$ 770km above ground with a ground sampling distance (GSD) of 0.5m per pixel. 8 overlapping images (obtained during one pass) of the panchromatic sensor (11 Bit per pixel) were used for the 3D reconstruction. For texturing, RGB images were computed by pansharpening the 0.5m GSD panchromatic images with the 2.5m GSD multispectral channels. The resulting DSM is resampled to 0.5m and, having a size of 2000×2000 pixel, covers an area of 1km $\times$ 1km.

4 RESULTS

For both test areas, reference data was obtained by airborne laser scanning (LiDAR), having a positional resolution of about 1m, a positional accuracy of 0.5m RMSE and a vertical accuracy of 0.25m RMSE according to the provider of the data. Due to the different resolution of the DSMs and the LiDAR point cloud, we compute the error metrics as Euclidean distance between the LiDAR points and the triangulated surface of the DSMs and 3D models.
In Table (1) we provide the numerical results of the accuracy evaluation. As can be seen, the aerial test area produces more accurate results, which is to be expected since the GSD is 0.2m compared to the 0.5m GSD of the satellite test area. Furthermore, the 3D models are less accurate than the original DSM, which also is to be expected as the amount of data is reduced from $4,000,000$ pixel to $221,000$ and $103,000$ vertices. This actually means a data reduction of 94.5% for the aerial test area and 97.4% for the satellite test area. Additionally we provide visual results of the textured 3D models in Figure (8) and (9).

Figure 6: Aerial image data - inner city of Munich (Marienplatz)

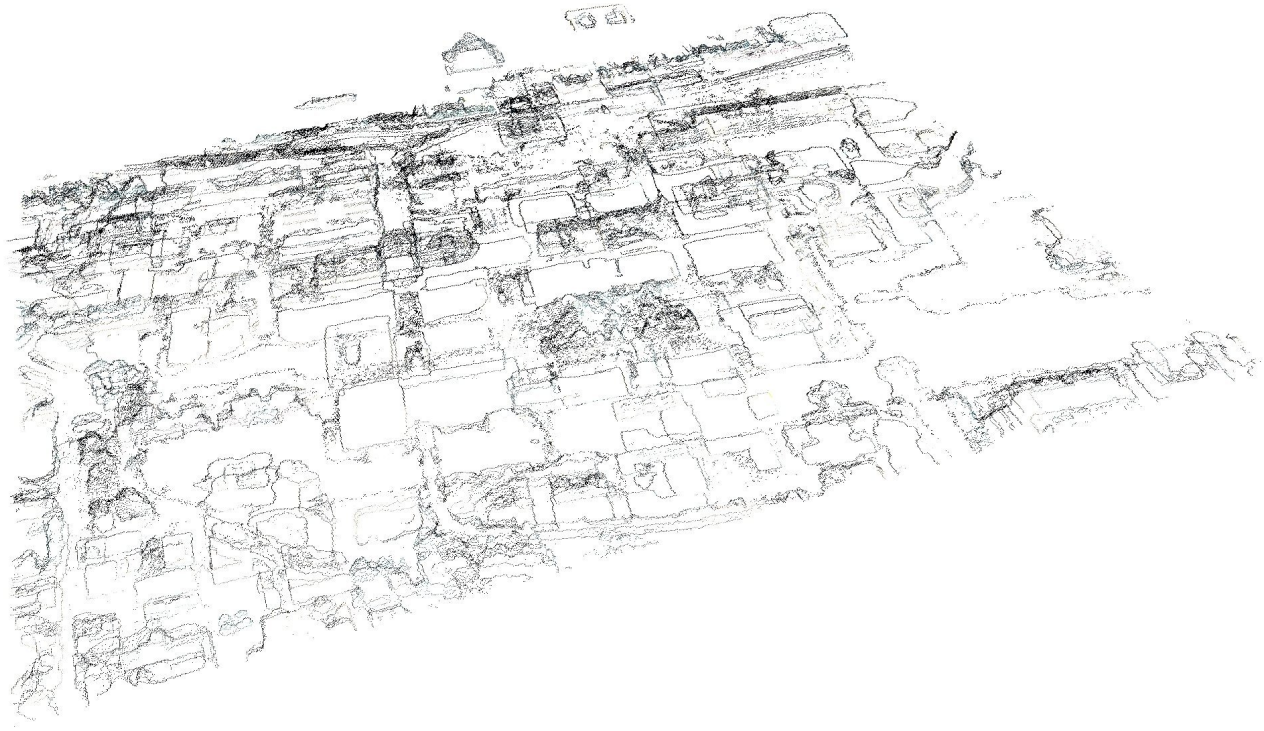
Figure 7: Satellite image data - inner city of London (Canary Wharf)



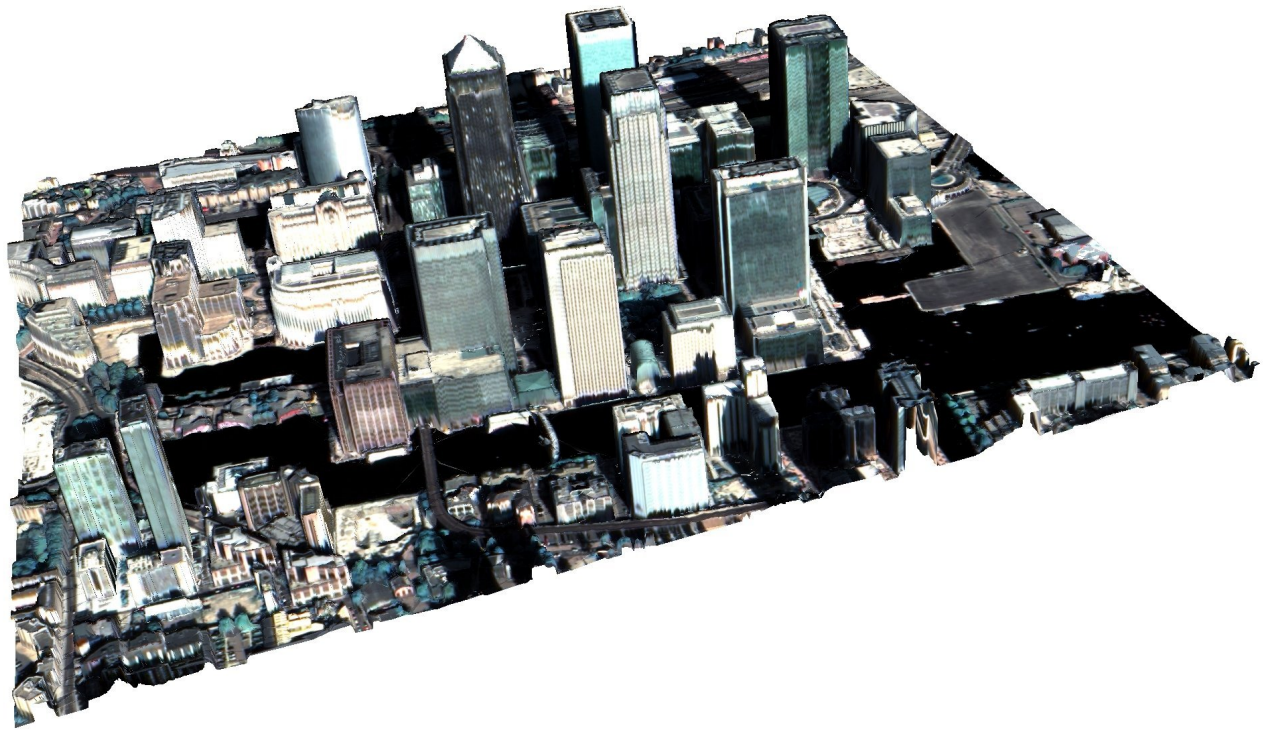
(a) Aerial image data - Thinned point cloud

(b) Aerial image data - Textured mesh

Figure 8: Resulting 3D model of the aerial image data



(a) Satellite image data - Thinned point cloud



(b) Satellite image data - Textured mesh

Figure 9: Resulting 3D model of the satellite image data

Table 1: Properties of the two data sets and accuracy of the resulting DSMs and 3D models - Mean Absolute Error (MAE), Root Mean Square Error (RMSE), Normalized Median Absolute Deviation (NMAD)

	Munich 3K+	London WV2
Area [pixel]	2000×2000	2000×2000
GSD [m]	0.2	0.5
Area [m]	400×400	1000×1000
Vertices in 3D model	$221,000$	$103,000$
Vertices / m^2	1.38	0.10
Vertices / pixel	0.06	0.03
DSM - MAE [m]	0.71	1.17
DSM - RMSE [m]	1.44	2.07
DSM - NMAD [m]	0.52	0.78
3D Model - MAE [m]	0.86	1.69
3D Model - RMSE [m]	1.51	2.44
3D Model - NMAD [m]	0.77	1.62

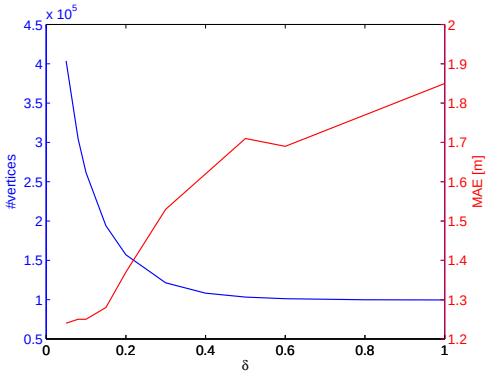


Figure 10: Influence of the mesh simplification parameter δ on the accuracy of the resulting 3D model (London data set). The correlation between the reduced number of vertices and increasing mean absolute error is clearly visible.

5 CONCLUSION

In this paper we presented a complete, fully automatic and model-free workflow for the 3D reconstruction of large scale (urban) areas. As our focus is on fully automatic processing chains, we can process the input image data very fast, around the clock and without the need for additional user-guided input. Of course, with manual interaction the accuracy normally is better than a fully automatic approach, so if there is need for higher accuracy, the resulting 3D models can be refined later on by further (semi-) manual processing. We additionally point out that, except from the input images themselves, no additional data like building footprints, special building models / primitives, road maps, etc. is required.

The accuracy was shown to be in the range of about 1m mean absolute error and about 2m root mean square error (mainly in height) for images with a GSD (or pixel size) of 0.2-0.5m, taken from 2km and 770km distance to the scene. Compared to our LiDAR ground truth's horizontal accuracy of 0.5m RMSE and vertical accuracy of 0.25m RMSE, the results of our image-only reconstruction is not that far off.

ACKNOWLEDGMENTS

The author would like to thank European Space Imaging (EUSI) for providing the WorldView-2 data of London.

REFERENCES

Bay, H., Tuytelaars, T. and Van Gool, L., 2006. Surf: Speeded up robust features. ECCV pp. 404–417.

Bobick, A. and Intille, S., 1999. Large occlusion stereo. International Journal of Computer Vision 33(3), pp. 181–200. occlusion detection.

Boykov, Y., Veksler, O. and Zabih, R., 2001. Fast approximate energy minimization via graph cuts. IEEE Transactions on pattern analysis and machine intelligence pp. 1222–1239. Graph Cuts.

Collins, R., 1996. A space-sweep approach to true multi-image matching. In: cvpr, Published by the IEEE Computer Society, p. 358.

Grabner, M., 2002. Compressed adaptive multiresolution encoding. Journal of WSCG 10(1), pp. 195–202.

Grodecki, J., 2001. Ikonos stereo feature extraction - rpc approach. In: Proc. ASPRS Annual Conference, St. Louis, pp. 23–27.

Guibas, L. and Stolfi, J., 1985. Primitives for the manipulation of general subdivisions and the computation of voronoi diagrams. ACM Transactions on Graphics 4(2), pp. 74–123.

Hirschmueller, H., 2005. Accurate and efficient stereo processing by semi-global matching and mutual information. In: Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on, Vol. 2, IEEE, pp. 807–814.

Kurz, F., Mueller, R., Stephani, M., Reinartz, P. and Schroeder, M., 2007. Calibration of a wide-angle digital camera system for near real time scenarios. In: Proc. ISPRS Hannover Workshop 2007-High Resolution Earth Imaging for Geospatial Information, pp. 1682–1777.

Lee, S., Wolberg, G. and Shin, S., 1997. Scattered data interpolation with multilevel b-splines. Visualization and Computer Graphics, IEEE Transactions on 3(3), pp. 228–244.

Lowe, D., 2004. Distinctive image features from scale-invariant keypoints. International Journal of Computer Vision 60(2), pp. 91–110.

Oh, J., 2011. Novel Approach to Epipolar Resampling of HRSI and Satellite Stereo Imagery-based Georeferencing of Aerial Images. PhD thesis, The Ohio State University.

Pock, T., Schoenemann, T., Graber, G., Bischof, H. and Cremers, D., 2008. A convex formulation of continuous multi-label problems. Computer Vision - ECCV 2008 pp. 792–805.

Triggs, B., McLauchlan, P., Hartley, R. and Fitzgibbon, A., 2000. Bundle adjustment - a modern synthesis. Vision algorithms: theory and practice pp. 153–177.

Viola, P. and Wells III, W., 1997. Alignment by maximization of mutual information. International journal of computer vision 24(2), pp. 137–154.

Yoon, K. and Kweon, I., 2006. Adaptive support-weight approach for correspondence search. Pattern Analysis and Machine Intelligence, IEEE Transactions on 28(4), pp. 650–656.

Zabih, R. and Woodfill, J., 1994. Non-parametric local transforms for computing visual correspondence. Computer Vision - ECCV 1994 pp. 151–158. census transform, rank transform.