EXTRACTING SEMANTICALLY ANNOTATED 3D BUILDING MODELS WITH TEXTURES FROM OBLIQUE AERIAL IMAGERY

D. Frommholz, M. Linkiewicz, H. Meissner, D. Dahlke, A. Poznanska

DLR Optical Sensor Systems, Berlin-Adlershof, Germany -
(dirk.frommholz, magdalena.linkiewicz, henry.meissner, dennis.dahlke, anna-maria.poznanska)@dlr.de

KEY WORDS: Aerial, Camera, City, Extraction, Virtual Reality, Point Cloud, Texture

ABSTRACT:

This paper proposes a method for the reconstruction of city buildings with automatically derived textures that can be directly used for façade element classification. Oblique and nadir aerial imagery recorded by a multi-head camera system is transformed into dense 3D point clouds and evaluated statistically in order to extract the hull of the structures. For the resulting wall, roof and ground surfaces high-resolution polygonal texture patches are calculated and compactly arranged in a texture atlas without resampling. The façade textures subsequently get analyzed by a commercial software package to detect possible windows whose contours are projected into the original oriented source images and sparsely ray-casted to obtain their 3D world coordinates. With the windows being reintegrated into the previously extracted hull the final building models are stored as semantically annotated CityGML "LOD-2.5" objects.

1. INTRODUCTION

Oblique aerial cameras have been recently deployed for remote sensing tasks in urban areas. In addition to solely nadir-looking sensors these devices are equipped with a set of camera modules providing tilted (off-nadir) views of the scene captured which allows to take high-resolution images of the roofs as well as the façades of buildings. In both cases the viewing vectors converge on the respective surface normals minimizing projective distortions.

This property makes oblique cameras well suited for creating visualizations of virtual city models if their interior and exterior orientation is known as the bitmaps recorded can be accurately mapped on existing geo-referenced 3D data sets. For example, (Frueh et al., 2004) project oblique aerial imagery onto triangular meshes of city buildings obtained from a terrestrial laser scanner. A 2D-to-3D registration process based on line segments is used to refine the set of initial camera poses before the image patches are chosen for each face by evaluating the viewing angle, occlusion, neighborhood relationship and other criteria. For visualization purposes the resulting collection of colored triangles gets combined into a single texture atlas by a greedy placement algorithm. Similarly, (Wang et al., 2008) color LiDAR data of urban structures by projecting the virtual wall surfaces back into a set of tilted views and comparing the results against the edges of the façades inside the source bitmaps. In (Stilla et al., 2009) oblique thermal infrared imagery is used to texture a 3D model that has been previously constructed from nadir-looking aerial photographs covering the visible range of the spectrum. Their back-and-forth projection approach involves occlusion detection using a depth buffer and allows to estimate the actual metric resolution per pixel of the output bitmaps aiming at GIS applications.

Aside from texture mapping true 3D surface information of urban objects can be regained from oblique imagery itself using photogrammetric techniques. The general suitability of tilted aerial photographs for terrain reconstruction has been outlined as early as in (Schultz, 1994) where projectively distorted stereo scenes get processed by a dense matcher computing a weighted cross-correlation score. More recently (Rupnik et al., 2014) generate point clouds from oblique data sets from state-of-the-art oblique aerial cameras that have been oriented with an adapted bundle adjustment workflow. To obtain the 3D coordinates of a pixel a multi-resolution matching algorithm based on graph cuts is utilized, however, no high-level analysis is performed on the results yet to subsequently extract planar surfaces. This problem is addressed by (Xiao et al., 2012) who look for vertical lines in the tilted views to form façade patches. The patches are tested against a height gradient map obtained from densely matching overlapping image pairs. Eventually verified façades get integrated into a building box model limiting the form of the reconstructed structures to quadrilaterals if seen from the top.

This paper discusses an approach for the extraction of the hull of city buildings including their wall, roof and ground surfaces from true 3D point clouds. The points are solely generated from a set of oriented images recorded by a decent oblique aerial camera system. Except for being simple polygons no inherent limitation is imposed on the building outline. At the same time the resulting simplified virtual models get textured from the original images without resampling preserving their native resolution. During this process a texture atlas gets created that is subsequently used for rendering the models and to identify potential windows through object-based image analysis (OBIA). Following a vectorization stage the windows that have been found are reintegrated into the hull models to finally obtain semantically annotated "LOD-2.5" CityGML objects.

2. BUILDING EXTRACTION

In order to extract the buildings of an urban scene both the oriented oblique and nadir images taken with the aerial camera are preprocessed into (tilted) digital surface models (DSMs). For this purpose the input data is passed to a dense stereo matcher based on the semi-global matching algorithm (Hirschmüller, 2008). The SGM implementation generates graylevel bitmaps that orthogonally encode the height information of the scene above the XY coordinate plane. As the surface information from the façades pictured by the oblique views will get lost during the reprojection of the height values the tilted cameras require to be rotated by the average angular displacement to approximately point into the nadir direction before applying SGM and back-rotated later on to retain the original geometry. Following the matching stage the surface models are smoothed by a bilateral filter comprising

of two Gaussian functions to reduce the noise without affecting high-frequency details (Tomasi and Manduchi, 1998). Furthermore, under the assumption that city buildings appear as elevated regions surrounded by lower ground, those DSM pixels that form the terrain are identified and removed by repeatedly moving windows of varying size over the DSM bitmaps as described in (Mayer, 2004). This process that basically creates a digital elevation model also eliminates most of the vegetation that is likely to disturb any upcoming processing steps and provides an estimate for the ground height around the buildings as a by-product.

After preprocessing the remaining DSM pixels are transformed into genuine 3D points and merged into a single point cloud. The resulting high-resolution samples are reduced to limit the memory consumption and projected onto the XY plane subdivided by a regular grid. The spacing of the grid needs to be adjusted empirically with respect to the residual noise of the points. For each grid cell its density, spatial distribution and characteristic height histogram is computed and evaluated to identify potential façade pieces (Linkiewicz, 2012). On the set of cell elements $C = \{p_i = (x_i, y_i)\}$ that is considered a part of the façade the regression line is estimated from the 2×2 covariance matrix

$$\mathbf{M_{cov}}(C) = \begin{bmatrix} cov(x_i, y_i) & cov(x_i, y_i) \\ cov(y_i, x_i) & cov(y_i, y_i) \end{bmatrix} \quad (1)$$

of the confined point set with

$$cov(x, y) := \frac{1}{n-1} \sum_{k=1}^{n} (x_k - \overline{x})(y_k - \overline{y}) \quad (2)$$

as outlined already in (Pearson, 1901). Effectively performing PCA the Eigenvector of $\mathbf{M_{cov}}$ that belongs to the greatest Eigenvalue denotes the direction of the best-fit line through C, and a measure for its confidence can be obtained from the ratio

$$\rho := \frac{\sigma_{min}}{\sigma_{max}} \quad (3)$$

of the square roots of the Eigenvalues, that is, the ratio of the standard deviations of the projected 3D points in the direction of the Eigenvectors. Alternatively, if the statistical analysis of the grid cell indicates the presence of outliers, the main direction of a potential façade piece is determined by the RANSAC method using a consensus set of two projected points (Fischler and Bolles, 1981). Beside an upper count of iterations the RANSAC estimator terminates if the percentage of the points of C that are within a maximum distance to the regression line running through the consensus set is above a threshold which depends on the grid spacing.

If the line segments have been determined for individual grid cells the computation of the regression line will be extended to groups of adjacent cells in order to form façade fragments of the same orientation. For this purpose the line direction of every cell in a local 8-neighborhood is pair-wisely compared against the line direction of its center element using a fixed angular threshold. In case the directions match the respective cells will be added to the same cell set. This growth process repeats until no more adjacent cells are left to be assigned. If a stable state is reached a common regression line will be estimated through the point set of every cell group. The resulting linear façade fragments will be intersected if their endpoints are locally adjacent within the grid forming a closed two-dimensional contour.

Following façade reconstruction the roof surface gets modeled from the nadir image set recorded by the aerial camera. This is because the current implementation is based on a planar extension of the linear regression algorithm used to find the wall surface contours which cannot handle depth continuities on the house tops yet. To determine the roof regions the nadir 3D point cloud as of SGM is subdivided into a regular grid again where the spacing must be chosen empirically. However, no projection is applied, and those cells which are not within a polygonal outline from the previous stage are discarded eliminating the possibility of modeling overhangs for now. Afterwards the 3×3 covariance matrix is computed similarly to equation 1 and 2 for each grid cell $D = \{q_i = (x_i, y_i, z_i)\}$ as

$$\mathbf{M_{cov}}(D) = \begin{bmatrix} cov(x_i, x_i) & cov(x_i, y_i) & cov(x_i, z_i) \\ cov(y_i, x_i) & cov(y_i, y_i) & cov(y_i, z_i) \\ cov(z_i, x_i) & cov(z_i, y_i) & cov(z_i, z_i) \end{bmatrix} \quad (4)$$

The Eigenvector that belongs to the smallest Eigenvalue of this matrix marks the normal vector $\mathbf{n}$ of the current cell plane whose flatness τ is described by the ratio

$$\tau := \frac{\sigma_{\mathbf{n}}}{\sigma_2 + \sigma_3} \quad (5)$$

where the σ's once more denote the standard deviations of the confined 3D points in direction of the corresponding Eigenvectors. Roof cells are identified by applying a planarity threshold and will get merged into the same labeled cell group if their normals roughly point into the same direction.

To derive the roof geometry from the cell groups their spatial relationship needs to be computed first. As a prerequisite the remaining gaps that might have been caused by structures like chimneys or satellite dishes are closed because initially one cannot guarantee that the house top is fully covered by roof cells after the regression process. Therefore the labeled cell groups get projected into the XY plane and a breadth-first region growing algorithm is applied until all grid elements within the building contour have been uniquely assigned. The grown sets are subsequently tested for potential connections by shifting a 3×3 window over the projected area. If the window covers two or more different labels then this will indicate a topological edge or node between two adjacent cell groups respectively and stored in a table.

However, the sliding window does not determine the relationship between the façades of the building and the house top. To add this information to the topology the polygonal outline of the structure obtained during wall surface reconstruction is scanned discretely. If a particular side of the contour entirely joins a single roof cell group it will be considered an edge adjacent to the corresponding façade. If two or more different labels are found along a side of the polygon instead a façade-roof node resulting for instance from an arris will be assumed. Similarly, to have a node in the topology where two wall surfaces and exactly one roof cell group intersect the number of roof cell groups at the corner points of the contour is computed by circular sampling as depicted in figure 1.

After the deduction of the spatial relationship the world coordinates of the roof elements are derived from the joint topological tables which do not contain any positional information on their own. For this purpose the planes forming the house top are constructed by fitting a regressions plane to the 3D points confined in the respective roof cell groups. Also, the façade planes get erected perpendicularly to the XY plane containing their projection as a part of the building contour. Having the geometric entities the intersection of adjoint pairs of roof planes or a roof plane with a nearby wall surface yields the geographic position and direction of the topological edges, that is, the roof ridge, arris and the outer edges of the house top. However, as the resulting and

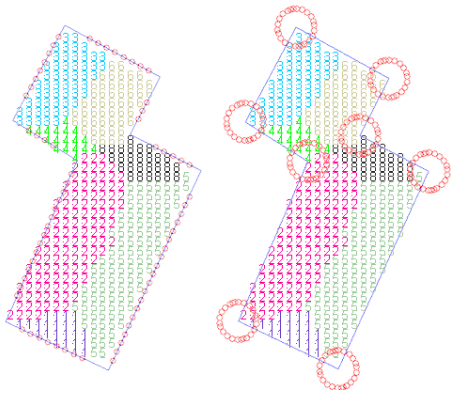

Figure 1. Topology analysis between roof regions/façades (left) and at façade corners (right)

possibly skew lines are infinite yet they need to be limited by the spatial equivalent of the topological nodes before they eventually can form the skeleton of the building to be modeled. Therefore the node coordinates are computed as the geometric center of the shortest connecting line among all combinations of pairs of lines derived from the topological edges, and the edge lines get adjusted accordingly.

To complete the reconstruction process the missing ground surface is obtained from the contour as of the façade cell regression. Its z-value is set to the height estimate from the terrain removal process following SGM. The finite roof, wall and ground polygons of the modeled buildings subsequently get written to separate files using the open Alias/Wavefront OBJ data format where the semantic designations are kept in specifically formatted comments. Also, the geometry gets written to an LOD-2 CityGML file that can be used with GIS databases.

3. TEXTURE MAPPING

For attractive visualizations incorporating the reconstructed buildings their roof, wall and ground surfaces must be colored preferably from the oriented aerial imagery. In a process called texture mapping each point of a model polygon is assigned a normalized 2D position inside a texture bitmap pointing to the pixel which effectively provides the spectral intensities at that vertex. The render engine will then be able to interpolate any other pixel that belongs to the interior zone of the polygon from the colors of its nodes. Computing the texture map for each surface of the building comes in two steps. First, the aerial image that pictures a particular polygon best needs to be determined. Second, because real-world render engines do not efficiently handle hundreds or thousands of texture bitmaps at once due to memory and I/O limitations, those parts of the chosen image that actually contribute to the surface must be placed compactly in a single digital image called texture atlas.

To find the image that optimally colors a particular part of the building model in the first step the entire oriented imagery from the aerial camera is initially classified into nadir and oblique bitmaps depending on the tilt angle of the underlying sensor. The normal vectors of the nadir views are then compared against the surface normals which are implicitly given in form of a consistent vertex order enforced during the reconstruction run. If both vectors point into the same direction the texture source will be discarded as the polygon is assumed to be hidden. Otherwise, in case of opposing normal directions, the angle of intersection between the view and surface normal is tested against a user-defined off-nadir threshold. If the image passes this test the polygon vertices subsequently will be projected into the potential texture bitmap. If the resulting 2D positions (s_i, t_i) lie inside the dimensions of the image it will be accepted as a valid color source with the (s_i, t_i) being the source texture coordinates of the surface. In addition a quality score will be derived from the bounding box of the (s_i, t_i) penalizing projective distortions that grow with the distance from the camera's principal point. Ultimately, if more than one nadir bitmap is found suitable to picture a certain roof or wall surface the image with the highest quality score will be taken.

If there is no nadir image candidate that can color a model polygon the oblique images will be tried. Similarly to the nadir case a quality measure will be computed for the tilted views if the view vector and surface normal point into opposing directions and the projected polygon vertices fall into the bitmap's dimensions. However, since the vertical building façades may particularly benefit from taking pixels far off the principal point a different formula is chosen for the score. To reward lower incidence angles and consequently minimize projective distortions the quality $0 \leq q \leq 1$ for oblique texture sources is defined as

$$q := |\mathbf{n_p} \cdot \frac{1}{k} \sum_{i=1}^{k} (\mathbf{v_i} - \mathbf{c})| \tag{6}$$

where $\mathbf{n_p}$ denotes the roof or wall surface normal, $\mathbf{v_i}$ is the projected surface vertices in world coordinates and $\mathbf{c}$ is the center of projection of the camera. If due to the flight parameters or camera setup neither a nadir image nor a tilted view picture a particular roof or wall surface it will be marked as untextured.

After the building polygons have been assigned a texture image and the respective surface vertices have been attached source texture coordinates (s_i, t_i) the actual color information is ready to be written to the texture atlas in a two-step process. The placement stage initially computes a compact layout for the texture patches comprising the pixels inside the (s_i, t_i) area for every model polygon. However, since finding the perfect free spots for the entire patch set within the dimensions of the rectangular atlas bitmap without wasting space is known to be NP-hard only an approximate solution can be efficiently computed for this problem in combinatorial optimization. Thus, as a basic algorithm, new texture patches get placed inside the atlas one by one. To find the next free spot for the current patch P_i the vacant positions inside the destination bitmap are sequentially traversed. Once a free position has been found it will be attempted to paste P_i in there. This is accomplished by losslessly rotating its pixels in steps of $90°$ to efficiently use the space available inside the texture atlas and intersecting the resulting shapes with the set of texture patch polygons placed before. If there is no overlap the atlas position and rotation angle for P_i will be stored in a dynamic list and the algorithm will proceed with the next patch. However, in case P_i cannot be pasted anywhere the destination bitmap will get resized horizontally and vertically to the next power of two of pixels as demanded by most graphics cards.

Even though the outlined basic algorithm creates tightly packed texture atlases its worst-case runtime is still quadratic in the number of pixels and also in the number of building surfaces. To increase its speed in practice the search for free places inside the destination bitmap is accelerated by a two-dimensional block availability map (BAM). The BAM stores scanline segments of pixels where the texture patches can be placed at and is updated every few loop cycles (see figure 2). Effectively implementing run-length compression on its free space only those parts of the atlas bitmap need to be checked for a vacant spot for which a BAM entry exists. Furthermore, to limit the amount of expensive

polygon-polygon intersection tests when the algorithm attempts to paste a particular P_i, the previously placed texture patches to test P_i against are referenced by the cells of regular grid covering the entire atlas. As a consequence the rotated P_i must not be overlapped with the union $P := \bigcup_{k=1,k\neq i}^{n} P_k$ but very likely with just a small subset $\tilde{P} \subseteq P$.

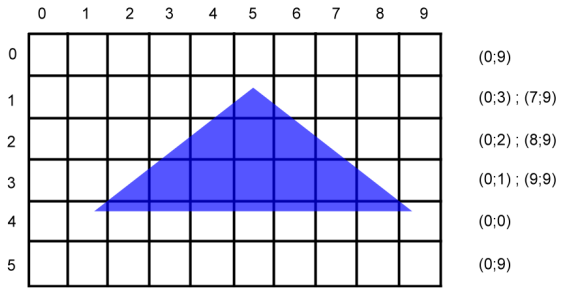

Figure 2. The BAM stores free scanline segments, here for a 10×6 texture atlas containing a single triangular texture patch

Beyond the runtime the placement stage also anticipates render issues that will occur with graphics adapters which implement mipmaps, that is, pyramids of downscaled textures for different display resolutions. As the mipmap generator typically utilizes digital low-pass finite impulse response filters for resampling nasty transitions from the texture patches to the background color of the atlas bitmap are likely to appear when the building model is visualized. To circumvent this problem the patches P_i are added a safety margin before they eventually will be placed. This safety margin is calculated by offsetting the contour of each texture patch polygon along its outwardly oriented vertex normals and results in extended source texture coordinates $(s_i, t_i) \rightarrow (\tilde{s_i}, \tilde{t_i})$.

Once the layout has been computed the render stage writes the pixels from the respective aerial images to their designated places inside the texture atlas. For this purpose the bounding box around the extended texture coordinates $(\tilde{s_i}, \tilde{t_i})$ is computed. Those pixels that are inside the bounding box and the polygon described by the tupels get rotated and copied to the corresponding texture atlas position. However, because the input data set can easily include thousands of bitmaps that can exceed a hundred megapixels each in size it is unfeasible to load them completely into memory for the copy process. To save time the texture patches are first sorted by their source bitmap in order to avoid opening the same file more than once. Subsequently only those pixels that actually need to be transferred to the texture atlas are being read using an image I/O subsystem that provides on-demand access based on concepts that have been similarly implemented in decent operating systems. The resulting problem of numerous but local write operations on the destination bitmap is defused with a write cache comprising a few hundred target scanlines.

After the texture patches have been completely written the coordinates of their vertices inside the texture atlas excluding the polygon offset will be normalized by the dimensions of the destination image as this is a requirement imposed by render engines and most 3D file formats. The resulting positions $(\hat{s}_i, \hat{t}_i)$ and a reference to the atlas bitmap are subsequently added to the respective building model. In addition to the texture atlas the index of the surface, its semantic designation and the rotation applied during the placement stage are kept in an artificial multichannel bitmap as filled polygons comprising the area defined by the $(\hat{s}_i, \hat{t}_i)$. Because this semantic layer is exclusively used for the object-based classification the contour offset does not need to taken into account. Also, to preserve the link between the final texture coordinates and the aerial images, the source texture coordinates (s_i, t_i) and the index of the corresponding model surface are stored in a text file.

4. WINDOW CLASSIFICATION

In order to find out whether it is possible to extract windows from an unrectified texture atlas that originally gets created for visualization purposes the result of the mapping stage undergoes object-based image analysis (OBIA) using the commercial eCognition software package by Trimble. For this purpose, using a custom classification rule set that focuses on the façade windows for now, those texture patches that describe the roof and ground surfaces of the reconstructed buildings are initially being removed by looking up their role in the semantic layer. The remaining set of wall surface patches then gets separated into discrete objects through eCognition's Contrast Split Segmentation algorithm.

To subsequently identify the window areas in a single façade object the Multiresolution Segmentation producing very fine and detailed segments is performed. This allows to capture every surface detail despite the presence of a low image contrast. The segments that have been found will be further classified as initial window objects if the mean intensity difference compared to a surrounding volume exceeds a certain threshold and is negative to accommodate the diffuse reflection of the light rays inside the building. Those window objects that share a common border will get merged and successively tested against geometric constraints like their minimum area, width and density to remove incorrectly classified small items and irregular dark elements or shadows. Since the merged objects still may not yield the correct extent and shape they are further grown into adjacent segments under similarity and homogeneity constraints for the window seed, candidate and target. Following region growing the results once more get tested against the geometric features and unsharply compared against a rectangular shape. To close any gaps in the classification outcome the window objects are processed by both morphological filters and the island filter before the resulting regions eventually get uniquely labeled and written to a bitmap that is of the same size as the texture atlas.

Having the label image congruently covering the texture atlas and the text file linking the texture atlas to the aerial source images it will now be possible to re-integrate the extracted windows into the reconstructed buildings. To convert the labeled pixel sets from OBIA into polygons that can be transformed into world space the contours of the window areas are retrieved first using a tracing algorithm based on the Moore neighborhood with a modified stopping criterion (Reddy et al., 2012). The resulting amount of contour vertices is reduced by the Ramer-Douglas-Peucker method (Douglas and Peucker, 1973) with a maximum simplification error of one pixel. Because each contour lies inside a corresponding texture patch its coordinates can be computed relative to the patch and back-rotated according to the information from the congruent semantic layer from the mapping stage. Further, utilizing the text file that assigns a texture patch its aerial source image and its source texture coordinates (s_i, t_i) the relative contour coordinates can be translated into the aerial source image as well. With the known camera orientations, if the translated window contours are now being ray-cast, their 3D coordinates will be obtained from the intersections with the respective wall surfaces and added to the reconstructed buildings as polygonal objects. The modified building models are now written again as semantically annotated OBJ and CityGML code effectively lifting the latter to the inoffical "LOD-2.5".

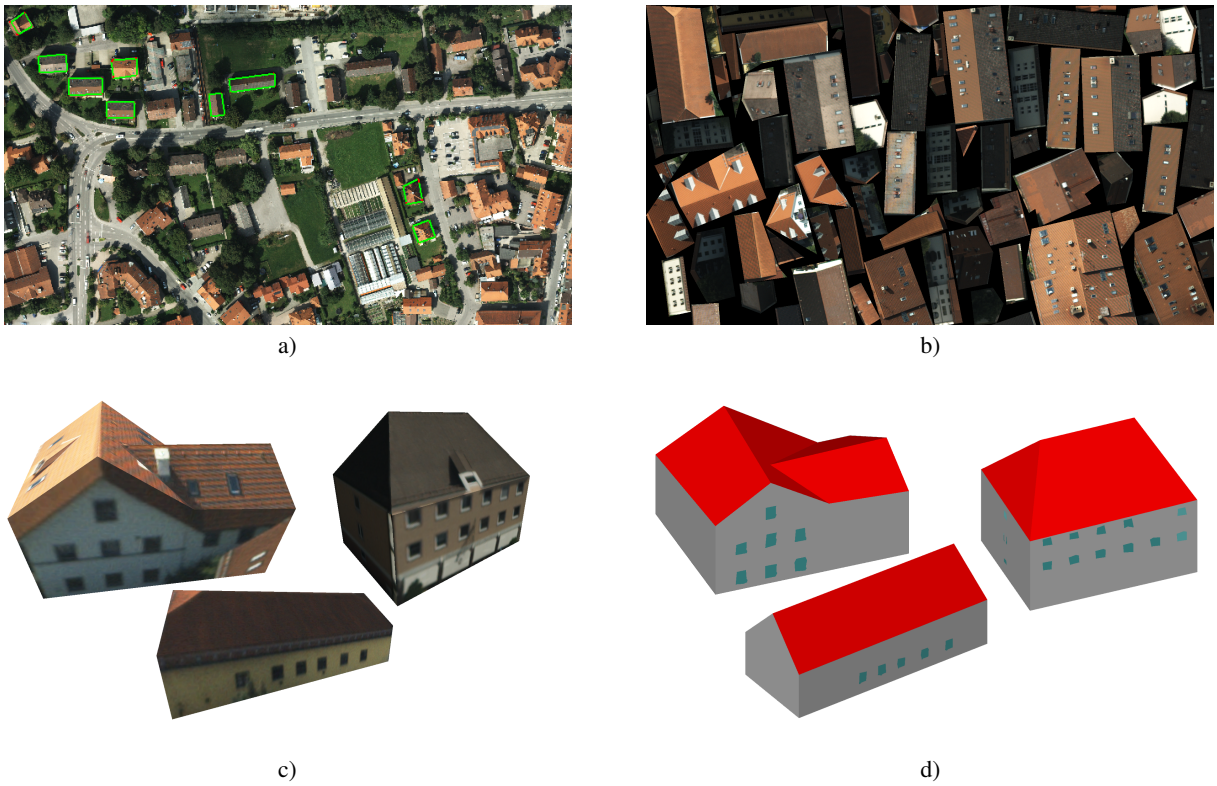

Figure 3. Results for the Murnau scene a) Contours of reconstructed buildings, b) Texture atlas, c) Textured building models, d) Same building models as CityGML showing the semantic designations of the surfaces and the windows in different colors

5. RESULTS

In a proof-of-concept for the proposed method a flight campaign over the city of Murnau/Germany was conducted using the DLR MACS HALE aerial camera (Lehmann et al., 2011) with a 90-60 image overlap. This device is equipped with one nadir-looking and two oblique sensors pointing to the left and right at an angle of 36°. An on-board GPS receiver and inertial measurement unit (IMU) register the approximate position and rotation of the camera. Following bundle adjustment to refine the initial orientation values a total of 4295 images were converted to digital surface models by SGM at a spatial resolution of 7 cm per pixel. Furthermore, a set of 24 buildings was put into the reconstruction pipeline implemented in MATLAB except for the texture mapper that is written in C++. The runtime for the reconstruction process was less than five minutes per building while texture mapping took approximately 3 minutes in total. Figure 3 depicts the output of different stages of the proposed workflow.

To evaluate the positional accuracy the footprints of the resulting models were compared against the Automated Land Records (ALK) of Murnau. Reflecting the considerable residual noise from SGM and the inherent simplification errors an average distance of 80.2 cm and a median deviation of 39.9 cm has been found between the extracted contours and the data from the cadastrial map. Since no reference data is available for the windows their area has been determined manually from the façade patches of the texture atlas first and then opposed to the classification outcome. Based on this ground truth the correctness has been computed as 69% and the completeness as 73% which is a substantial result but not extraordinarily good. The main issue with window extraction is the lack of appropriate geometric features which consequently cannot be exploited because the texture atlas does not get rectified in order to avoid any quality degradation due to resampling. Relying solely on spectral properties however fails when for instance windows and neighboring surfaces like shutters get merged. This erroneous consolidation influences the shape of objects from OBIA and possibly causes their omission as shown in figure 3c and 3d. One possibility to overcome the limitation to spectral features would be to introduce projective invariants into the classification rule sets which needs to be further investigated.

To assess the performance of the mapping stage the generated texture patches have been projected onto the reconstructed buildings and visualized. Due to the high overlap and the low-density residential areas of Murnau occlusions do rarely occur for the test scenario. However, in presence of high-rise buildings the process will probably benefit from occlusion detection based on the DSM. This can be efficiently implemented by spatially sorting and ray-casting the polygons of the 3D model as seen from the aerial camera that provides the source image respectively. Furthermore, since the building surfaces currently have to project completely into one bitmap to get textured, the mapping algorithm needs to be extended to support multiple source images at once in order to adequately handle dense residential areas or business districts. Implicating changes on the placement strategy this modification also induces the need for radiometric adjustments between two or more adjacent pixel subsets that form a texture patch.

6. CONCLUSION

This paper has presented an approach for the reconstruction of the hull of buildings exclusively from the oriented imagery recorded by a modern oblique aerial camera system. In a proof-of-concept it has been shown that both its nadir and oblique views can be used for mapping textures on virtual models of an urban scene without any loss of quality. The additional information provided

by the oblique sensors helps to regain the surfaces of the structures including their semantic designation. Despite some limitations it has also been generally demonstrated that the same unrectified texture atlas that gets used for the visualization of the reconstructed buildings can also serve as a source for object-based image analysis to classify façade elements like windows. Nevertheless, the applicability of the proposed method in urban scenarios with different architectural styles remains to be further evaluated with the increasing market penetration of oblique cameras and the availability of the respective images.

REFERENCES

Douglas, D. H. and Peucker, T. K., 1973. Algorithms for the reduction of the number of points required to represent a digitized line or its caricature. *Cartographica - The International Journal for Geographic Information and Geovisualization* 10(2), pp. 112–122.

Fischler, M. A. and Bolles, R. C., 1981. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM* 24(6), pp. 381–395.

Frueh, C., Sammon, R. and Zakhor, A., 2004. Automated texture mapping of 3D city models with oblique aerial imagery. *3D Data Processing, Visualization and Transmission Proceedings* pp. 396–403.

Hirschmüller, H., 2008. Stereo processing by semi-global matching and mutual information. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 30(2), pp. 328–341.

Lehmann, F. et al., 2011. MACS - Modular Airborne Camera System for generating photogrammetric high-resolution products. *Photogrammetrie, Fernerkundung, Geoinformation.* 6, pp. 423–434.

Linkiewicz, M., 2012. Extraktion von senkrechten Fassadenebenen aus 3D-Punktwolken von Schrägluftbildern. Master's thesis, Beuth Hochschule für Technik Berlin.

Mayer, S., 2004. Automatisierte Objekterkennung zur Interpretation hochauflösender Bilddaten in der Erdfernerkundung. PhD thesis, Humboldt-Universität zu Berlin, Mathematisch-Naturwissenschaftliche Fakultät II.

Pearson, K., 1901. On lines and planes of closest fit to systems of points in space. *Philosophical Magazine* 2, pp. 559–572.

Reddy, P. R., Amarnadh, V. and Bhaskar, M., 2012. Evaluation of stopping criterion in contour tracing algorithms. *International Journal of Computer Science and Information Technologies* 3, pp. 3888–3894.

Rupnik, E., Nex, F. and Remondino, F., 2014. Oblique multi-camera systems - orientation and dense matching issues. *Int. Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences* 40 (3/W1), pp. 107–114.

Schultz, H., 1994. Terrain reconstruction from oblique views. *ARPA Image Understanding Workshop Proceedings* pp. 1001–1008.

Stilla, U., Kolecki, J. and Hoegner, L., 2009. Texture mapping of 3D building models with oblique direct geo-referenced airborne IR image sequences. *Archives of the ISPRS 38-1-4-7/W5*.

Tomasi, C. and Manduchi, R., 1998. Bilateral filtering for gray and color images. *IEEE 6th International Conference on Computer Vision* pp. 839–846.

Wang, Y., Schultz, S. and Giuffrida, F., 2008. Pictometrys proprietary airborne digital imaging system and its application in 3D city modelling. *Int. Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences* 37, pp. 1065–1070.

Xiao, J., Gerke, M. and Vosselman, G., 2012. Building extraction from oblique airborne imagery based on robust façade detection. *ISPRS Journal of Photogrammetry and Remote Sensing* 68, pp. 56–68.