Improving architectural 3D reconstruction by plane and edge constraining

H. Cantzler[1], R.B. Fisher[1] and M. Devy[2]

[1] Division of Informatics, University of Edinburgh,
Edinburgh, EH1 2QL, UK
[2] LAAS-CNRS, 31077 Toulouse, France

{helmutc,rbf}@dai.ed.ac.uk, michel@laas.fr

Abstract

This paper presents new techniques for improving the structural quality of automatically acquired architectural 3D models. Common architectural features like parallelism and orthogonality of walls and edges are exploited. The location of these features is extracted from the model by using a probabilistic technique (RANSAC). The relationships among the planes and edges are inferred automatically using a knowledge-based architectural model. A numerical algorithm is used to optimise the orientations of the features. Small irregularities in the model are removed by projecting the triangulation vertices onto the features. Planes and edges in the resulting model are aligned to each other. The techniques produce models with improved appearance. We show results for synthetic and real data with consideration of noise.

Keywords

Surface geometry, Shape, Scene analysis, Constrained architectural reconstruction

1 Introduction

The process of 3D reconstruction of scenes is often affected by noise in the measurements. Furthermore, inaccuracies are created by view merging, segmentation and surface fitting. Ways to improve the reconstruction are to use more sophisticated methods like photogrammetry techniques or to increase the number of views (possibly from different sensors). Another way is to identify and exploit properties of the scene to improve the model. 3D reconstruction of industrial parts has used prior knowledge to improve models for a long time [14, 17]. Architectural scenes are also particularly suitable for the constraints since the geometry is typically very structured [3]. Architectural constraints can be used for camera calibration as well as 3D reconstruction from single [18, 10] and multiple [6, 1] intensity images or for image based modelling of buildings [4]. Features used for architectural constraints are typically straight lines, large coplanar regions and the parallelism and orthogonality of lines or planes. These kinds of features can be easily found

BMVC 2002 doi:10.5244/C.16.2

in architecture scenes. Architectural constraints for improving 3D models have been used to automatically straighten edges in 3D models [5]. The work presented in this paper combines parallelism and orthogonality constraints on planar regions (typically walls, floors and ceilings) and edges. Planes are flattened and edges are straightened. We enforce the alignment of planes and edges by applying orientation constraints. Triangulated nearly planar regions are made coplanar and edges straight. We apply the constraints to the data following meshing. Zabrodsky concluded in [19] that corrections following meshing generally give a greater improvement. The method is independent of the calculation of the 3D structure unlike the work presented in [18, 10, 6, 1] where constraints are used in combination with reconstruction from intensity images.

This work consists of three steps. First, architectural features are extracted from previously triangulated 3D models (Section 2). We use RANSAC techniques [7] to find planes (similar to that given in [2]) and edges in the model. The next step is the automatic extraction of the relationships between the extracted scene features. Few papers have dealt with the automatic extraction leaving it to the user to specify them [14, 17]. The interpretation of the scene is formalised as a constraint satisfaction problem [16]. Liedtke used a semantic net for interpretation of architectural scenes [11]. His interpretation is hypothesis driven. Hypotheses are verified or falsified by matching the 3D objects against the image. In our work we match the planes against a semantic net of a generic house by using a backtracking tree search (Section 3). The semantic net concentrates on the definition of the 3D objects and its relations. We check the interpretations only by verifying the relationships between the 3D objects. We then analyse the edges in the models by clustering. Edges with similar orientation are clustered together. We obtain the three principle orientations of the architectural model from the clusters. Constraints are assigned to almost-regularities like parallel or perpendicular walls or edges. The last and final step consists of applying the architectural constraints (Section 4). The original model is fitted to the new constrained model. Optimising the model can be done in a number of ways (*e.g.* numerically [2, 17] or evolutionary [14]). We use the Downhill Simplex method [12]. It is a robust numerical multidimensional minimisation technique. After finding the parameter vector for the optimised model the vertices are projected onto the features. The result is a model with fewer irregularities (*e.g.* edges on walls) and aligned features (*e.g.* parallel walls).

2 Feature extraction

At all stages of the process, the model is a mesh consisting of vertices $V = \{(x, y, z)'\}$, edges $E = \{(v_1, v_2)\}$ and triangles $T = \{(v_1, v_2, v_3)\}$. The first step of the process is to extract planes and edges from the raw triangulated model. Before starting the extraction the model is normalised. The model is scaled and translated to fit into a unit sphere at the origin. A robust RANSAC algorithm [7] is then used to obtain a set of planes and edges.

2.1 Plane extraction

The algorithm hypothesizes a number of random planes from triples of points in V. The distance of a triangle centroid to the hypothetical plane is calculated by computing the difference between the distance of the plane to the origin D and the dot product between

the triangle centroid $C = (c_x, c_y, c_z)'$ and the unit plane normal $N = (n_x, n_y, n_z)'$. Triangles that satisfy the following inequality belong to the hypothetical plane.

$$|C \cdot N - D| < tolerance \tag{1}$$

The size of a hypothetical plane is calculated by adding up the areas of the triangles that satisfy (1). The hypothesis that creates the largest plane is selected. Each plane is represented by its minimal description, which is the surface normal represented by azimuth and elevation angles with respect to a reference vector and the distance to the origin. The exact number of features in a scene is not known. So, we repeat the RANSAC algorithm until the size of the resulting feature falls under a certain threshold. (An EM algorithm [13] could instead have been used to select the number of planes and fit them, but we chose a simpler technique to focus on the reconstruction issues.)

The plane extraction gives reasonable results. However, it sometimes produces a plane that consists of small disconnected patches distributed over the scene. An architectural plane (*e.g.* a wall) is not usually separated by a large gap. However small gaps frequently occur for example due to the presence of pipes or decorations. Therefore, the planes are analysed by single linkage clustering [9] to ensure that the triangles of a plane are closely connected. The cluster technique starts with the individual triangles and groups them together to form larger and larger clusters (hierarchical clustering). The distance between two clusters is defined as the minimal Euclidean distance of any two triangles belonging to different clusters (nearest neighbor method). The clustering terminates after reaching a certain distance. This distance specifies how far apart parts of the plane can be.

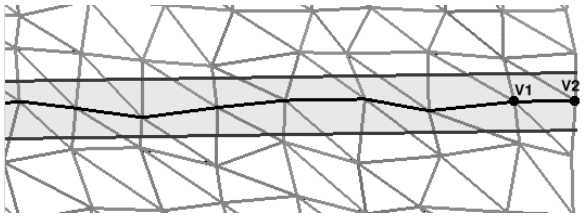

Figure 1: A fold edge goes horizontally thought this mesh. A hypothetical edge is created with the vertices V_1 and V_2. All edges that lie in the 3D corridor (grey area) belong to the hypothetical edge (middle dark line).

2.2 Edge extraction

The edge extraction starts with filtering the edges of the 3D model. Only jump edges (belonging to a single triangle) or fold edges, which separate non-coplanar triangles, are used. The algorithm creates random edges from the filtered edge set. The distance of a vertex to the hypothetical edge is calculated with the starting point $S = (s_x, s_y, s_z)'$ and the unit orientation $O = (o_x, o_y, o_z)'$ of the edge. All triangle edges with their vertices (V_1, V_2) that satisfy the following inequality belong to the hypothetical edge (see figure 1).

$$||(V_i - S) - O((V_i - S) \cdot O)|| < tolerance \tag{2}$$

The length of a hypothetical edge is calculated by adding up the lengths of the matched triangle edges. The hypothesis that creates the longest edge is selected. We represent every edge by its starting point S and orientations as α and β. We repeat the process until the length of the resulting edges falls under a given threshold.

3 Scene interpretation

3.1 Plane labelling

A model of an architectural scene is described in a general semantic net (see figure 2). The model entities (walls, roof and floor) are represented as nodes in the net. The nodes are connected via different types of relationships (arcs). A semantically meaningful description is assigned to the scene features by matching them to the semantic net. A backtracking tree search is used to find the best match. The algorithm takes as input a set of plane features F, a set of possible model labels L and a set of binary model relationships R which limits the possible labelling. The tree search starts with the first feature from F and assigns all labels from L. A second feature is fetched from F and all labels are assigned. At this level some of the labels might be ruled out because they violate the observed scene relationships. This process continues until all features have been labelled. A consistent labelling then exists if each feature is assigned a single valid label that is also arc consistent with adjacent nodes. The relationships between features are used to select appropriate geometrical constraints for enforcing parallelism or orthogonality later in the optimisation process.

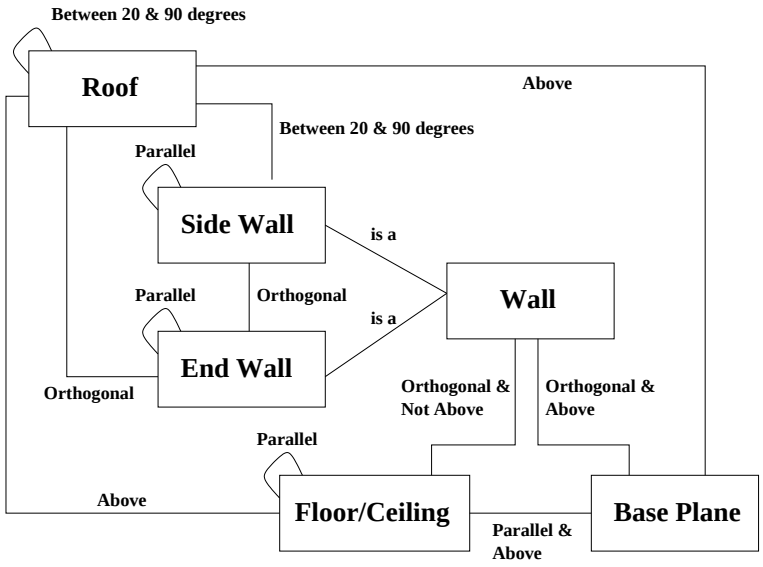

Figure 2: The model of the architectural scene is represented by a semantic net. Nodes represent the model entities and are linked by architecturally meaningful relationships.

The model-entities (labels) and the relationships among the entities represent the

knowledge of a typical architectural outdoor scene. Possible labels are L = {*Side Wall, End Wall, Base Plane, Ceiling/Floor, Roof, No Architectural Feature*}. The architectural relationship between two features and their labels are checked (*e.g.* horizontal and vertical walls are almost perpendicular). Angle relationships between two features are checked with a certain tolerance (3 degrees). The "Above" relationship is satisfied if 99% of the vertices of one plane are above a second plane defined by surface normal and distance. *No Architectural Feature* does not have any relation with a normal feature and can therefore be assigned everywhere. The final labelling is obtained by finding the solution that maximises the number of architectural labels.

The semantic net models a reasonable subset of all houses. It includes the interior and exterior structure of houses. The model can include an arbitrary number of walls which can be placed parallel or orthogonal to each other. They can be on the same level or on different ones (then seperated by a *Floor/Ceiling*). The base plane is below all other parts of the building. It represents the ground on which the house stands. The roof is modelled as a typical sharp roof. Errors in the scene description are resolved by labelling them as *No Architectural Feature*. The semantic net can be easily extended with features like windows and doors. These features can be modelled as parallel and close to the actual walls. However, the previous plane detection concentrates on finding big planes. So, modelling windows and doors is not necessary at this step.

3.2 Edge grouping

Each edge that was found has an orientation O and a starting point S. Edges with almost-equal orientation are grouped together by complete linkage clustering [9]. The distance between clusters are determined by the greatest distance between any two edges in different clusters (opposite to the nearest neighbor method). This leads to small very compact clusters. We use the angle between the orientations of the edges as the similarity measurement.

The majority of architectural structures consists of linear elements that form a three-dimensional structural frame [3]. The frame defines the three principal directions of a building. We use the clusters found previously to find the three directions of the architectural frame. The three clusters that are most orthogonal to each other are selected. Orthogonality constraints are created between the orientations which are used in the optimisation process. We find the three directions O_1, O_2 and O_3 from the set of clusters by minimising:

$$\sum_{i=1}^{2}\sum_{j=i+1}^{3}\left(\left|\frac{\pi}{2}-\arccos\left(\frac{|O_i\cdot O_j|}{\|O_i\|\ \|O_j\|}\right)\right|\right) \tag{3}$$

4 Model optimisation

Optimising the model by enforcing the constraints found previously is formulated as a nonlinear programming problem. There are many algorithms which are designed to search spaces for an optimum solution. Some of them become ill-conditioned and fail with nonlinear problems. We use the Downhill Simplex method [12]. It is a numerical multidimensional minimisation technique. This method requires only function evalua-

tions, not derivatives. Like any technique which uses only function evaluations, this technique is not very efficient in terms of computational performance. However, it is very robust and leads to a quick solution if we already know a set of parameters for a solution close to the optimum.

The evaluation function for the optimisation technique consists of the squared residuals of the vertices and the constraint functions. It uses a parameter vector $\vec{p}$ that concatenates all the parameters for the individual planes and edges. The parameters for each plane includes the surface normal as two angles (2 floats) and the distance (1 float). An edge consists of the starting point S (3 floats) and the orientation as two angles (2 floats). By keeping the number of parameters for each individual feature small, the size of the parameter set is kept small and so gives better computational performance.

The squared residual is the squared geometric distance from the vertices to their feature (plane or edge). We have a set of features parameterised $F = \{F_i(\vec{p})\}$. Each feature i has a set of vertices $\{V_{i,j}\}$. The residual of every feature is normalised with its number of vertices N_i. Thus, model size does not affect results.

Every constraint is represented by a constraint function c(). The values of these functions correspond to the degree that the constraints are satisfied. As an example the constraint function for enforcing parallelism looks like this:

$$c_{parallel}(\vec{p}) = |\arccos(\frac{|O_i \cdot O_j|}{\| O_i \| \quad \| O_j \|})| \tag{4}$$

The constraint function can be seen as a penalty function. The constraint functions are added up to give the global constraint error. λ is a weight factor which scales the constraints to the residuals. λ can go to ∞ to ensure exact constraint satisfaction. However, λ is kept fairly small to find a good fit to the original data.

$$\sum_i \frac{1}{N_i} \sum_j dist(F_i(\vec{p}), V_{i,j})^2 + \lambda \sum_i c^{(i)}(\vec{p}) \tag{5}$$

Minimising this gives the optimised model parameters. We now project the vertices of the planes onto their planes. We calculate the new coordinates $V_p = (x_p, y_p, z_p)'$ of the vertex with the original vertex $V = (x, y, z)'$, the unit surface normal of the plane $N = (n_x, n_y, n_z)'$ and the distance D of the plane to the origin as:

$$V_p = V - tN \tag{6}$$

where

$$t = \frac{V \cdot N - D}{N \cdot N} \tag{7}$$

Analogously, we project the vertices of the edges onto their edges. The new position V_p of the vertex is calculated with the original vertex V, the start vertex of the edge $S = (s_x, s_y, s_z)'$ and the unit orientation of the edge $O = (o_x, o_y, o_z)'$.

$$V_p = O((V - S) \cdot O) + S \tag{8}$$

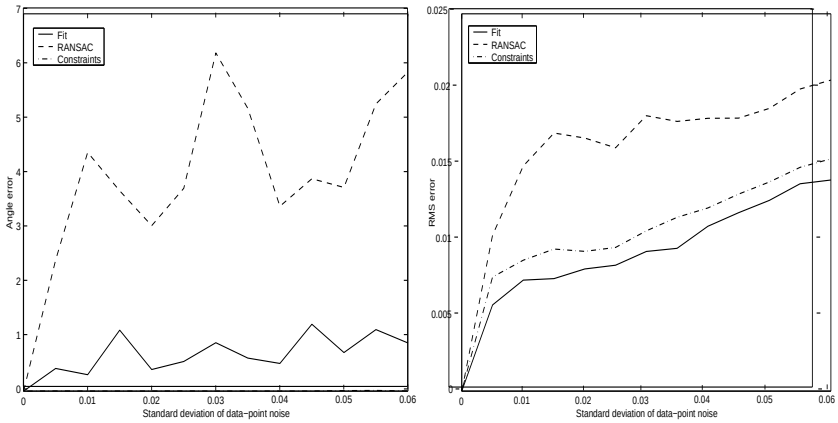

Figure 3: Results for the experiment with the synthetic model. The left graph shows the angle error in degrees versus the noise level. The graph on the right shows the mean squared residual versus the noise.

5 Experimental results

The proposed technique described above is general. It is independent of the way the 3D model was created (*i.e.* from range or intensity data) and of model properties like variance of the triangle size. It has been applied to several triangulated models. We will here present results for a synthetic model and for one reconstructed real model.

First, we applied the described technique to a synthetic model. The model consists of a perfect mesh of three walls at 90 degrees (1323 vertices & 2400 triangles). Two walls are parallel. A varying amount of Gaussian distributed 3D noise is added to the vertices. The walls are constrained by three constraints. Additionally, three orthogonality constraints are used for the three principal directions. The first graph shows the constraint error from feature extraction (top curve), improving the fit (middle curve) and application of the constraints (bottom curve, near noise level axis). Improving the fit is done without using any constraints in the evaluation function. The constraint error from feature extraction is a result of the random nature of RANSAC. Specially, the orientation of the three principle directions varies much, because fewer points are used to fit the lines and estimate the three directions. Improving the fit using all data points from the features gives much better results. Finally, using the constraints gives a constraint error close to zero. The second graph shows the mean squared residual after feature extraction (top curve), improving the fit (dashed curve) and constraining the model (solid curve). The parameters obtained from RANSAC show the biggest error. The mean residuals from improving the fit and from applying the constraints are close together and are both significantly below the the RANSAC curve. The two graphs show that applying constraints improves the orientation of walls and edges without significantly worsening the fit.

Next we show the application of the constraints to a Bavarian farmhouse reconstructed by the European Commission Joint Research Centre (JRC) [8, 15]. The model is shown in figure 5). The model was reconstructed from multiple range data scans (12504 vertices & 16589 triangles). This is a full 3D model. The original solid model shows small edges on

the walls. The optimised model has these edges projected onto the wall (see figure 4 for a close view of a wall). The plane extraction finds the 4 walls of the house and two planes for the roof (total 1856 vertices). The low number of plane vertices in comparison to the total number of vertices results from the fact that the planes consist of relatively few big triangles and that model details like windows consist of many small triangles. The plane extraction preserves features like the windows and doors (see figure 5). 63 edges are extracted and 27 of them are grouped together in 8 clusters. All edges in one cluster are considered to be parallel to each other. The parameter vector consists of 223 variables (18 for the 6 planes, 189 for the 63 edge starting points and 16 for the 8 cluster orientations). The initial parameters obtained from RANSAC give us angle errors that are no more than 1.3 degrees off. The angle errors of the plane and edge orientation vary from the optimum by 0.4 and 0.7 degrees on average in the original model. The planes are constrained by 10 constraints and the edge orientations by three. After optimisation all angle errors differ less than 0.01 degrees from the optimum. The result in figure 5 shows the model with removed irregularities and aligned planes and edges. The average disparity of the moved vertices is 0.21% of the model diameter. The optimisation step took 165 seconds on an Intel Pentium III with 600MHz.

Figure 4: A close view of a wall of the farmhouse. On the left is the unconstrained model. Surface ripples between the windows are most easily seen in the circled areas. On the right is the optimised model with fewer irregularities.

6 Conclusion

Previous work used architectural constraints mainly for scene reconstruction from intensity images. This work shows how architectural constraints can be used for improving the reconstruction of full 3D models independent of the sensor data. Only 3D information is used. The constraints make architectural features more regular in terms of their architectural properties. We exploit common architectural features like walls and their relationships to each other.

Initially, a RANSAC technique obtains a set of planes and edges from the 3D data. We automatically discover the constraints between the planes by using a tree search strategy. Even conservatively loose thresholds on angles and position lead to a correct labelling of the planes in the scene. Parallel edges are grouped together by clustering. The model parameters are optimised with a robust numerical optimisation algorithm.

The experimental results show how imperfections like small irregularities on planes and the orientations of walls and edges are corrected. As a result orientations of planes and edges are corrected. The visual appearance of the model is enhanced.

References

[1] C. Baillard and A. Zisserman. A plane-sweep strategy for the 3d reconstruction of buildings from multiple images. *ISPRS Journal of Photogrammetry and Remote Sensing*, 33(B2):56–62, 2000.

[2] A. Bartoli. Piecewise planar segmentation for automatic scene modeling. *Conference on Computer Vision and Pattern Recognition, Hawaii*, pages 283–289, 2001.

[3] F.D.K. Ching. *Architecture: Form, Space and Order*. Van Nostrand Reinhold, New York, 1996.

[4] P.E. Debevec, C.J. Taylor, and J. Malik. Modeling and rendering architecture from photographs: A hybrid geometry- and image-based approach. *SIGGRAPH 96*, pages 11–20, 1996.

[5] P. Dias, V. Sequeira, J.G.M. Goncalves, and F. Vaz. Combining intensity and range images for 3d architectural modelling. *International Symposium on Virtual and Augmented Architecture, Dublin*, pages 139–145, 2001.

[6] A. Dick, P. Torr, and R. Cipolla. Automatic 3d modelling of architecture. *British Machine Vision Conference, Bristol*, pages 372–381, 2000.

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

[8] The Bavarian Farmhouse from the European Commission Joint Reseach Centre (JRC). *http://mortimer.jrc.it/*.

[9] A.K. Jain and R.C. Dubes. *Algorithms for Clustering Data*. Prentice Hall, 1988.

[10] D. Liebowitz, A. Criminisi, and A. Zisserman. Creating architectural models from images. *Eurographics*, 18(3):39–50, 1999.

[11] C.-E. Liedtke, O. Grau, and S. Growe. Use of explicit knowledge for the reconstruction of 3d object geometry. *International Conference on Computer Analysis of Images and Patterns, Prague*, pages 580–587, 1995.

[12] J.A. Nelder and R. Mead. A simplex method for function minimization. *Computer Journal*, 7:308–313, 1965.

[13] B. North and A. Blake. Using expectation-maximisation to learn dynamical models from visual data. *British Machine Vision Conference, Essex*, pages 669–679, 1997.

[14] C. Robertson, R.B. Fisher, N. Werghi, and A. Ashbrook. Fitting of constrained feature models to poor 3d data. *Proceedings Adaptive Computing in Design and Manufacture, Plymouth*, pages 149–160, 2000.

[15] V. Sequeira, K. Ng, E. Wolfart, J.G.M. Goncalves, and D.C. Hogg. Automated reconstruction of 3d models from real environments. *ISPRS Journal of Photogrammetry and Remote Sensing*, 54:1–22, 1999.

[16] D.L. Waltz. *Generating Semantic Descriptions from Drawings of Scenes with Shadows*. PhD thesis, AI Lab, MIT, 1972.

[17] N. Werghi, R.B. Fisher, A. Ashbrook, and C. Robertson. Shape reconstruction incorporating multiple non-linear geometric constraints. *Computer-Aided Design*, 31(6):363–399, 1999.

[18] M. Wilczkowiak, E. Boyer, and P.F. Sturm. Camera calibration and 3d reconstruction from single images using parallelepipeds. *International Conference on Computer Vision, Vancouver*, pages 142–148, 2001.

[19] H. Zabrodsky and D. Weinshall. Using bilateral symmetry to improve 3d reconstruction from image sequences. *Computer Vision and Image Understanding*, 67(1):48–57, 1997.

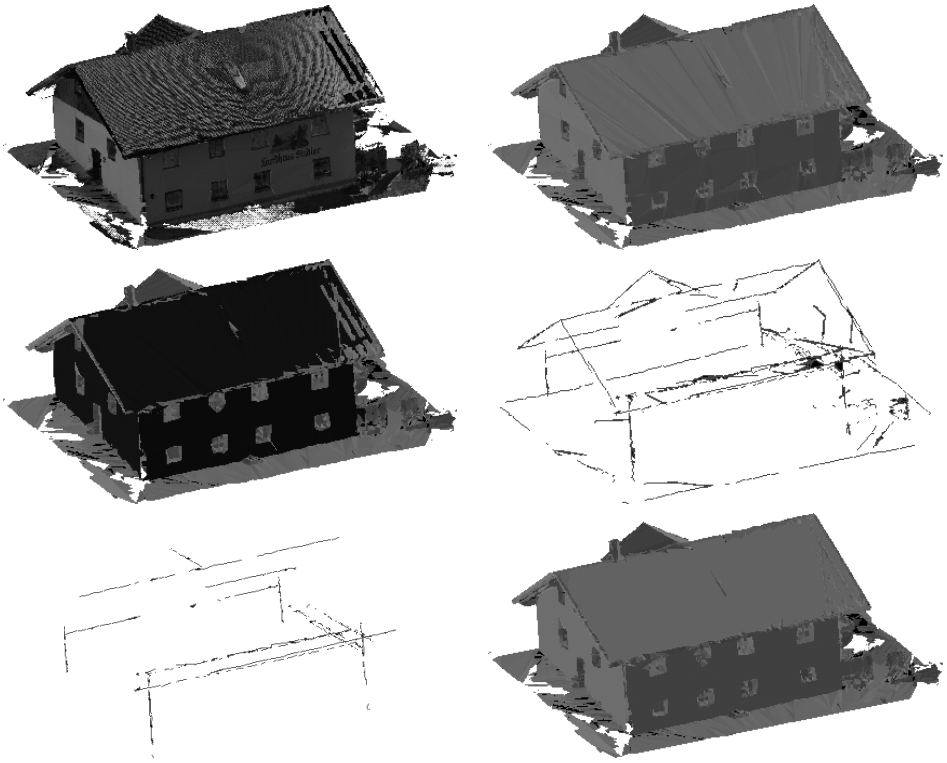

Figure 5: Top: The original textured model (left) and solid model (right) of the Bavarian farmhouse reconstructed by the European Commission Joint Research Centre (JRC). Middle: The extracted planes in darker colour (left) and edges (right). Bottom: The edges of the three principal directions (left) and the resulting model with flattened, straightened and aligned planes and edges (right).