2007 Urban Remote Sensing Joint Event

Real Time Fully Automatic 3D-Modelling of HRSC Landscape Data

Rui Liu, Darius Burschka, Gerd Hirzinger
Institute of Robotics and Mechatronics
German Aerospace Center (DLR)
Oberpfaffenhofen, 82234 Wessling, Germany
Email: Rui.Liu@dlr.de

Bernhard Strackenbrock
Illustrated Architecture
Oberkrämer, Germany
Email: bs@illustrated-architecture.de

***Abstract*—This paper presents a fully automatic 3D-modelings-process of the landscape data obtained by the High Resolution Stereo Camera (HRSC) assembled on a airplane. The input of this 3D-modellingsprocess is the huge 2.5D point-clouds resulted from the photogrammetric preprocessing and the output is a group of simplified colored meshes. The 3D-modelingsprocess consists of six steps: tilling with overlap, mesh generation, mesh simplifying, mesh cut, mesh merger and texture mapping. The whole process is fully automatic and is performed in real time. Validation of the method has been done on the reconstruction of various famous regions of Bavaria with tourist features.**

I. INTRODUCTION

With the rapid development of computer technology and widely spreading of internet, digital landscape becomes a high topic in scientific area recently. Man uses airplane, airship, even satellite to gain the colored aerial views outdoors. Of course, the obtained data amounts are very large. Therefore, how to handle these data, to simplify them, to transform them to 3D models fast and automatically for further applications, becomes a challenging task.

This paper presents a fully automatic 3D-modellings-process of such 2.5D landscape points-clouds. And it is performed in real time. These landscape data are obtained with a High Resolution Stereo Camera (HRSC) [1] assemble on a flying airplane. This camera has been developed by the DLR Institute of Planetary Research for the exploration of the Mars surface. The airborne version HRSC-AX is currently used for capturing earth's landscape and cities from flight altitudes between 1500m to 5000 m [2]. The input data of our 3D-modellingsprocess are resulted from the photogrammetric pre-processing: correction, refining [3] and stereo-matching [4].

To handling these large datasets we designed an automatic modellingsprocess, which is performed in six steps: tilling with overlap, mesh generation, mesh simplifying, mesh cut, mesh merger and texture mapping. And the resulted 3D landscape is saved as a group of simplified meshes with corresponding textures in VRML2.0 format. The whole process is performed in real time.

The main contributions of our approach are listed below:

- **Real time:** the above six steps are integrated in one process, which can be completed in 20 ~ 30 minutes for one area with 283 million points (24480 x 11570). This accelerate the old process by factor 10 ~ 100.
- **Fully automatic:** the user only needs to give the tile size and sampling rate, and the other works are integrated in one process, which will be accomplished fully automatically. The resulted 3D colored model can be displayed in any position under any viewpoint. By the old approach of modeling the HRSC landscape data in our institute, man must predefine a group of virtual viewpoints, calculate the corresponding 2D image from every viewpoint, and show them in a proper sequence like a film. Because of the huge data amount, the calculation of the resulted 2D image from a certain viewpoint takes very long. Thus an online visualization can not be achieved by the old approach.
- **Enormous geometric reduction:** after the processing of our new method, normally over 99% points can be eliminated and the geometric features become sharp automatically. But in the method described above, all points must be kept for the further calculation.
- **Appropriate for the online visualization:** as the whole landscape is divided in tiles, it is suitable for a fast and dynamic loading process. So that an online visualization of large regions can be reached.
- **Better Interface to commercial software:** the format and size of our resulted files are compatible to other 3D modeling- and visualization-software, for example 3D studio, Navis etc, which can do the post processing for other advanced applications perfectly.

To validate this method, diverse experiments have been done. For example, the reconstruction of various regions of Bavaria, Germany: the city of Kelheim with the Hall of Liberation, the environment of Monastery Andechs and Wies-Church etc.

II. RELATED WORK

For the modeling of landscape data, LOD (Level of Detail) [5] is used to simplify the mesh as a traditional way. To

accelerate display and to fit the demand of dynamic environment, HLOD (hierarchical level of detail) [6] is introduced to generate meshes. This algorithm combines LOD computation algorithm with a hierarchical clustering.

For the various surface simplification algorithms, a comprehensive survey is given in paper [7]. Among these methods, the QEM (Quadric Error Metrics) algorithm [8] shows an outstanding performance for its speed and accuracy. For the more, many researches combine 3D information with appearance attributes to get a more realistic effect for colored model [9, 10].

More and more efforts have been taken to find and reconstruct the man-made structures and natural structures automatically. For example, buildings, highways, trees etc. [11, 12, 13, 14, 15, 16]. And to enhance the 3D roaming velocity, the multi-thread technology and "Buffer-Quadtree" algorithm are presented in paper [17].

III. MAIN METHODE

The input of our 3D-modellingsprocess is the 2.5 D colored point clouds resulted from the photogrammetric preprocessing. Because one of such aerial views is normally several gigabytes, it overrides the capabilities of fast all of the 3D model-ling- and visualizationstools. To reduce the data amount and to prepare for a proper visualization further, we take a "divide-and-conquer" strategy: piecewise reading and modeling, then making the border of the adjacent parts seamless. So our approach of fully automatic 3D-modelling is consist of the following steps:

- Tilling with overlap
- Mesh generation
- Mesh simplifying
- Mesh cut
- Mesh merger
- Texture mapping

At the end we got a group of colored meshes organized in a special order for further applications.

The details of the six phases are described below:

A. Tilling with Overlap

In this stage, the whole area will be divided in tiles with a proper size. For a fast reading of input data in arbitrary areas with any resolution, the input data were saved as double-Tiff-format: one Tiff-file contains the height-information of every pixel, and the other holds the RGB colored information.

To reduce the distortion resulted from mesh-simplifying, we set an overlap between adjacent height-tiles and apply a mesh-cut action after simplifying.

The colored tiff file will be tiled and compressed with JPEG format. Although there is information loss after JPEG-compressing, the distortion is little and not relative to the position of pixel. So we do not set overlaps between colored-tiles. For a file with 12 MB in Tiff-format, it is reduced to 200~600 KB in JPEG-format. The compression ratio depends on the complexity of the picture content. The simple picture results in a high compression ration.

To generate height-tiles from a region with size of $W \times H$ pixels, the user can define the basis tile-size C, overlap μ and sampling rate s as input parameters. The whole region will be divided and saved in tiles automatically. Certainly the tiles at the boundary are smaller than the regular tiles. If we denote the size of regular tiles as $C_0 \times C_0$, the boundary tiles have obviously smaller edge lengths: C_1, C_2 and C_3 (See Figure 1).

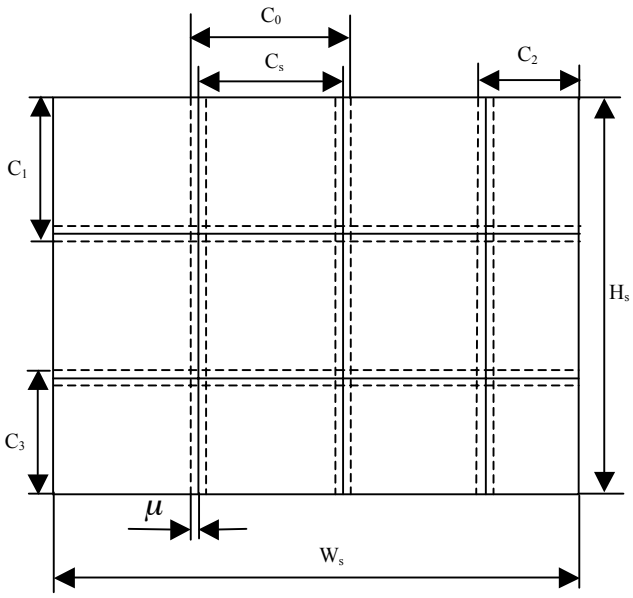

Figure 1. Tilling with overlap

And the calculations of them are:

$$\begin{cases} C_s = \lceil C/s \rceil \\ W_s = \lceil W/s \rceil \\ H_s = \lceil H/s \rceil \\ N_w = \lceil W_s/C_s \rceil \\ N_h = \lceil H_s/C_s \rceil \\ C_0 = C_s + 2\mu \\ C_1 = C_s + \mu \\ C_2 = W_s - C_s \cdot (N_w - 1) + \mu \\ C_3 = H_s - C_s \cdot (N_h - 1) + \mu \end{cases} \quad (1)$$

After tilling, the height-tiles will be meshed for the further processing.

B. Mesh Generation

As the heights of pixels are saved as matrix-format in tiff-file, by connecting the shorter diagonal of every rectangle grid, we get a regular grid mesh. But it is very dense. Actually, we need not many triangles in flat area. So we simplifying the grid mesh in the next step.

C. Mesh Simplifying

An efficient mesh simplifying will be applied in this step. Geometric characters become more outstanding, for example, edge becomes sharper. And the data amount can be reduced over 99% without distinguish distortion.

We use the modified quadric error metric algorithm to do mesh-simplifying. The quadric error metric algorithm [8] has proven to yield good result in practice. Basically, it uses edge collapse method by assigning each vertex-pair with an optimal contraction error and collapses the vertex-pair with the minimum error iteratively.

To computer the optimal contraction error **e** of one vertex pair ($\mathbf{v_1}$, $\mathbf{v_2}$), a quadric error function is defined for every vertex **v** as the sum of its squared distance to a set of planes $\Phi = \{p\}$.

$$e(v) = \sum_{p \in \Phi} (p^T v)^2 = v^T (\sum_{p \in \Phi} pp^T) v = v^T Q v \qquad (2)$$

Q is the quadric error matrix, which is computed as:

$$Q = \sum_{p \in \Phi} (pp^T) \qquad (3)$$

With $\boldsymbol{p} = [a\ b\ c\ d]^T$ as the normalized form of the plane defined by $ax + by + cz + d = 0$.

If the contraction is ($\mathbf{v_1}$, $\mathbf{v_2}$) $\rightarrow$ **v**, to computer the optimal position of **v**, the union set of the adjacent planes of $\mathbf{v_1}$ and the adjacent planes of $\mathbf{v_2}$ should be taken as the planes set $\mathbf{\Phi}$. But the author of QEM simply adds the two plane set for a fast computation of **Q**. That is, $\mathbf{Q} = \mathbf{Q_1}+\mathbf{Q_2}$ with $\mathbf{Q_1}$ is derived from all adjacent planes of $\mathbf{v_1}$ and $\mathbf{Q_2}$ is derived from all adjacent planes of $\mathbf{v_2}$.

And the minimal error **e** is reached by solving

$$\begin{cases} \partial e/\partial x = 0 \\ \partial e/\partial y = 0 \\ \partial e/\partial z = 0 \end{cases} \qquad (4)$$

To update the vertex-pair with the minimum cost, it uses a heap structure to store all valid pairs for contraction. And re-heap the involving pairs after each contraction. Thus it needs huge memory to hold the mesh with large size, and it contracts only one vertex-pair in every iterative step.

We improve the quadric error metric technique in the following way: firstly, we combine the change of the colors with the distance error function; secondly, we take a careful segmentation of the whole area at first and put only the vertex-pairs of complex objects to the heap structure to determine the best contraction position. For the big planar regions we replace the heap-sort structure with a bucket-sort structure and use a relative cheap error function to do edge collapse, which can contract more than one pairs in one step.

The following examples show a little area with several houses. The original grid mesh is generated from HRSC 2.5D data with resolution of 20 cm. It has 359,552 triangles. After simplification, only 0.5% triangles are remained. However, the geometric characters become more outstanding, and the appearance of the textured 3D model is improved considerably (See Figure 2).

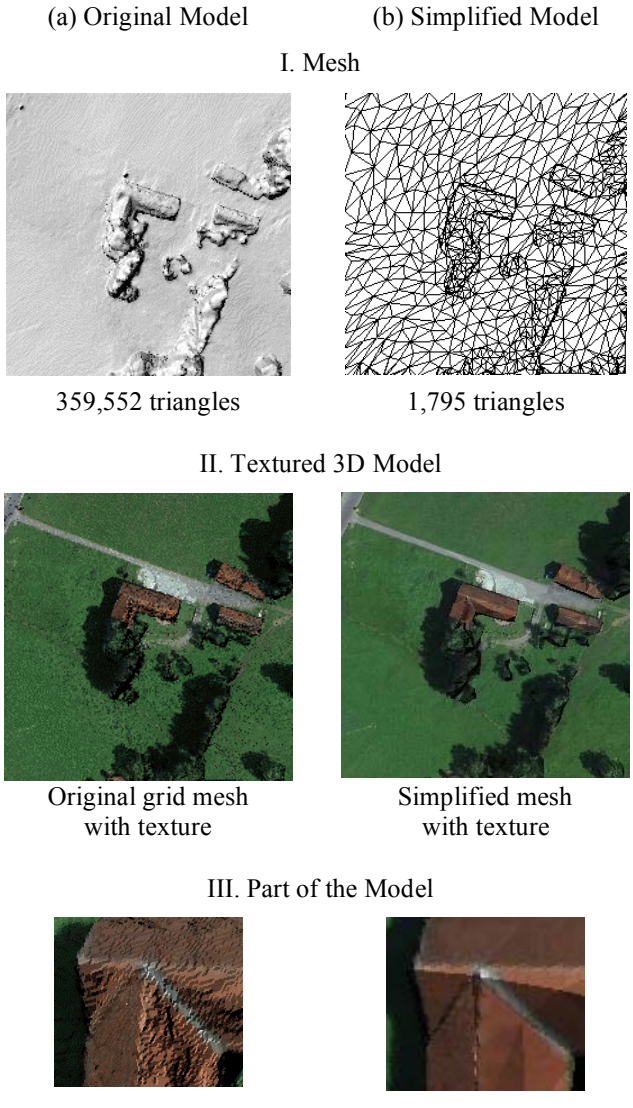

Figure 2. Mesh Simplifying

D. Mesh Cut

In this step, the bounding box of each tile will be calculated automatically according to the tile size and the position of it. Exactly along the middle axis of the overlapping area, every mesh will be cut neatly. That means, the projection of all resulted boundary edges from one cut action should lie in one straight line.

Although there are six cutting planes in one bounding box, we don not take the top and bottom cutting planes into account, because every tile has at most four neighbors.

Every cutting-plan $\boldsymbol{\varphi}$ has one inside and one outside about the bounding-box. Triangles across the cutting plane will be newly triangulated. There are generally two cases: (a) A triangle has only one vertex A inside $\boldsymbol{\varphi}$, then we need only substitute the outside vertices B and C with the intersection points B' and C'; (b) If a triangle has two vertices ***a*** and ***c*** inside $\boldsymbol{\varphi}$, we need re-triangulate the Quadrilateral (***a, a', c', c***) (See Figure 4).

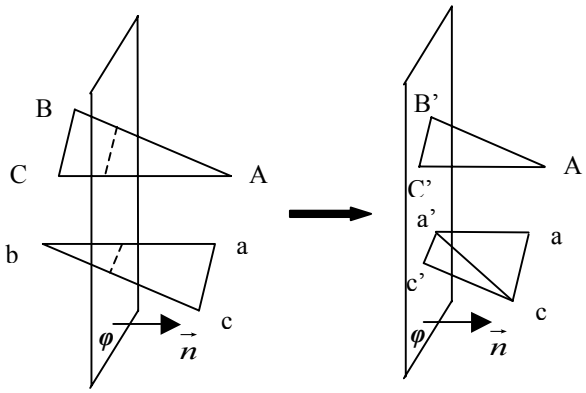

Figure 3. Mesh cut

Due to sufficient overlaps, there are only tiny gaps between the adjacent meshes after cutting, and the gaps should be exactly on the cutting plans. That is, if the view direction is perpendicular to the aerial photo-graph, no gaps are visible. Because the projection of all boundary edges concerning to one cutting-plan should lie on one line. This reduces the distortion resulted from the merger action.

E. Mesh Merger

After neatly cutting of the simplified meshes, we apply a merger action to "stitch" the adjacent meshes point-to-point.

To facilitate the discussion, we define the following notation. If a ***stitch action*** does $v_1 \rightarrow v_2$, that is, merge point v_1 to v_2, we call v_1 ***stitching point*** and v_2 ***stitched point***.

Generally, the stitch action has two forms: ***stitch-to-point*** and ***stitch-to-edge***, which will be applied in three situations (See Figure 4):

(a) The stitched point is at a corner;

(b) The stitched point is at not at a corner, but the distance between the stitched and stitching points is under a certain threshold;

(c) The rest situations.

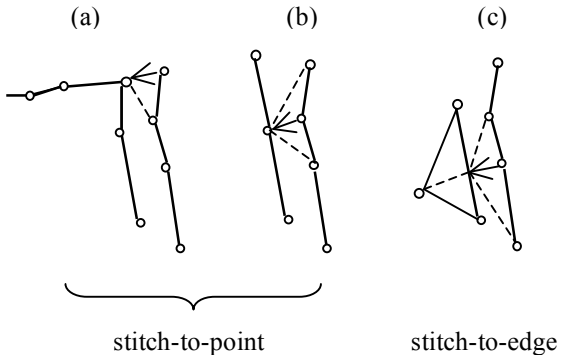

Figure 4. Stitch Action

Our algorithm of mesh merger is described below:

1. Find boundary vertices of both meshes on their common cutting plane and denote them as BV_1 and BV_2.
2. **For each** boundary vertices w_i of BV_2,
   - Find the nearest point v_j in BV_1
   - **if** dist(w_i, v_j) < ε **or** v_j is at corner
     - stitch w_i to v_j
     - denote v_j as stitched
   - **else**
     - choose the proper split-edge e from the two adjacent edges of v_j
     - calculate the projection u_i of w_i on e
     - shift w_i to u_i
     - split e
     - put u_i in BV_1
     - denote u_i as stitched
3. **For each** unstitched vertices v_i of BV_1, do stitch action like step 2.

Figure 5. Algorithm for merge of meshes on a certain cutting plan

As step 2 and 3 of our algorithm guarantee that all the boundary vertices of one mesh are stitched with a boundary vertex of the other mesh, the resulted mesh should be seamless.

F. Texture Mapping

As the colored-tiles are saved as a group of JPEG files and every tile has the same size of its corresponding mesh-tile, the texture mapping can be achieved by projection mesh to image. It means that the texture coordinates $\boldsymbol{T}$ (x_t, y_t) of each 3D point

P (*x, y, z*) is obtained by projection it to the image plane. The values are ranged from 0 to 1. The principle of texture mapping is showed in Figure 6.

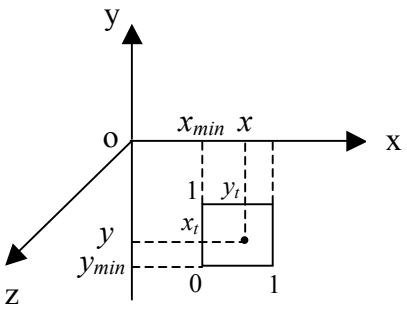

Figure 6. Texture mapping

If the resolution of the aerial views is *r*, the edge length of the regular tiles should be $(r \cdot C)$. The calculation of x_t and y_t is showed below.

$$\begin{cases} x_t = (x - x_{\min})/(C \cdot r) \\ y_t = (y - y_{\min})/(r \cdot C) \end{cases} \quad (5)$$

And the binding of JPEG files with 3D models is achieved by defining the URL of "Image Texture" in VRML 2.0 format.

IV. Experiments and Results

To verify our method, we have reconstructed various regions in Bavaria, Germany. Most of them contain very famous cultural historic buildings, for example the Hall of Liberation, Monastery Andechs etc. Both the overview of the regions and partial models of distinguished sight-seeing are showed in Figure 7.

We have processed the data on a computer with Intel Core2 Duo Processor E6600 (2.4GHz) and 2048 MB RAM. The runtime to processing various models is showed in Table 1.

TABLE I. Process Time of Diverse Regions

Region	**Area Size** n x m (pixels^2)	**Resolution** (m)	**Output** (Triangles)	**Runtime** (min)
Kehlheim	24480 x 11570	0.20	3,623,330	26,5
Andechs	6000 x 6000	0.15	463,333	3,6
Wiekirche	6000 x 6000	0.20	465,557	3,7

V. Conclusion

For the further development, firstly we should improve the texture mapping techniques. At now, we just do a simple 3D-to-2D projection, which may result mapping error between texture and 3D models.

Secondly, we want to combine the automatic structure detection and reconstruction in our mesh simplification, so that the man-made objects can be represented more accurately.

And at the third, we are searching for a good structure to store our tiles for an interactive online visualization. Some wonderful works have been done by other researchers.

Further more, the data compression and transfer techniques are also an interesting topics for us.

And what is very excited for us is the fusion of models generated from different sensors on different scales, for example, how to combine the models from laser scanning or panorama camera with the HRSC landscape models.

Of course, to manage these huge amount of data, the intelligent database technique and the modeling of special musters, for example trees etc, are required.

Acknowledgment

Thanks for Mr. Johann Heindl and Dr. Heiko Hirschmüler, they give us useful advices and sufficient input data to test our modeling process.

References

[1] F. Wewel, F. Scholten, and K. Gwinner. "High resolution stereo camera (HRSC)-multispectral 3D-data acquisition and photogrammetric data processing," Canadian J. Remote Sensing, 26:466–474, 2000.

[2] G. Hirzinger et al. "Photo-Realistic 3D-Modelling - From Robotics Perception Towards Cultural Heritage," International Workshop on Recording, Modelling and Visualization of Cultural Heritage, Ascona, Switzerland, 2005.

[3] F. Scholten, K. Gwinner and F. Wewel. "Angewandte digitale Photogrammetrie mit der HRSC-AX," Photogrammetrie Fernerkundung Geoinformation, 5:317–332, 2002.

[4] H. Hirschmüller, F. Scholten, G. Hirzinger. "Stereo Vision Based Reconstruction of Huge Urban Areas from an Airborne Pushbroom Camera (HRSC)", in Lecture Notes in Computer Science: Pattern Recognition, Proceedings of the 27th DAGM Symposium, 30 August - 2 September 2005, Vienna, Austria, Volume 3663, pp. 58-66.

[5] D. Luebke, M. Reddy, J. Cohen, A. Varshney, B. Watson, and R. Huebner. "Level of Detail for 3D Graphics," Morgen Kaufmann Publisher, 2003.

[6] C. Erikson, D. Manchoca, D., and W.V. Baxter III. "HLODs for Fast Display of Large Static and Dynamic Environments," Symposium on Interactive 3D Graphics '01 Proceedings, 111-120, March, 2001.

[7] D. P. Luebke. "A Developer's Survey of Polygonal Simplification Algorithms," IEEE Computer Graphics &Applications, May, 2001.

[8] M. Garland, P.S. Heckbert. "Surface Simplification using Quadric Error Metrics," Proc. of the ACM SIGGRAPH' 97, pp. 209-216, Los Angeles, California, 1997.

[9] M. Garland, P.S. Heckbert. "Simplifying Surfaces with Color and Texture Using Quadric Error Metrics", Proceedings of IEEE Visualization '98, pp. 263-269, North Carolina, 1998.

[10] H. Hoppe. "New Quadric Metric for Simplifying Meshes with Appearance Attributes", Proceedings of IEEE Visualization '99, pp. 59-66, San Francisco, 1999.

[11] A. Gruen, O Kübler , P. Agouris (Eds.). "Automatic Extraction of Man-Made Objects from Aerial and Space Images," Birkhäuser Verlag, Basel, 1995.

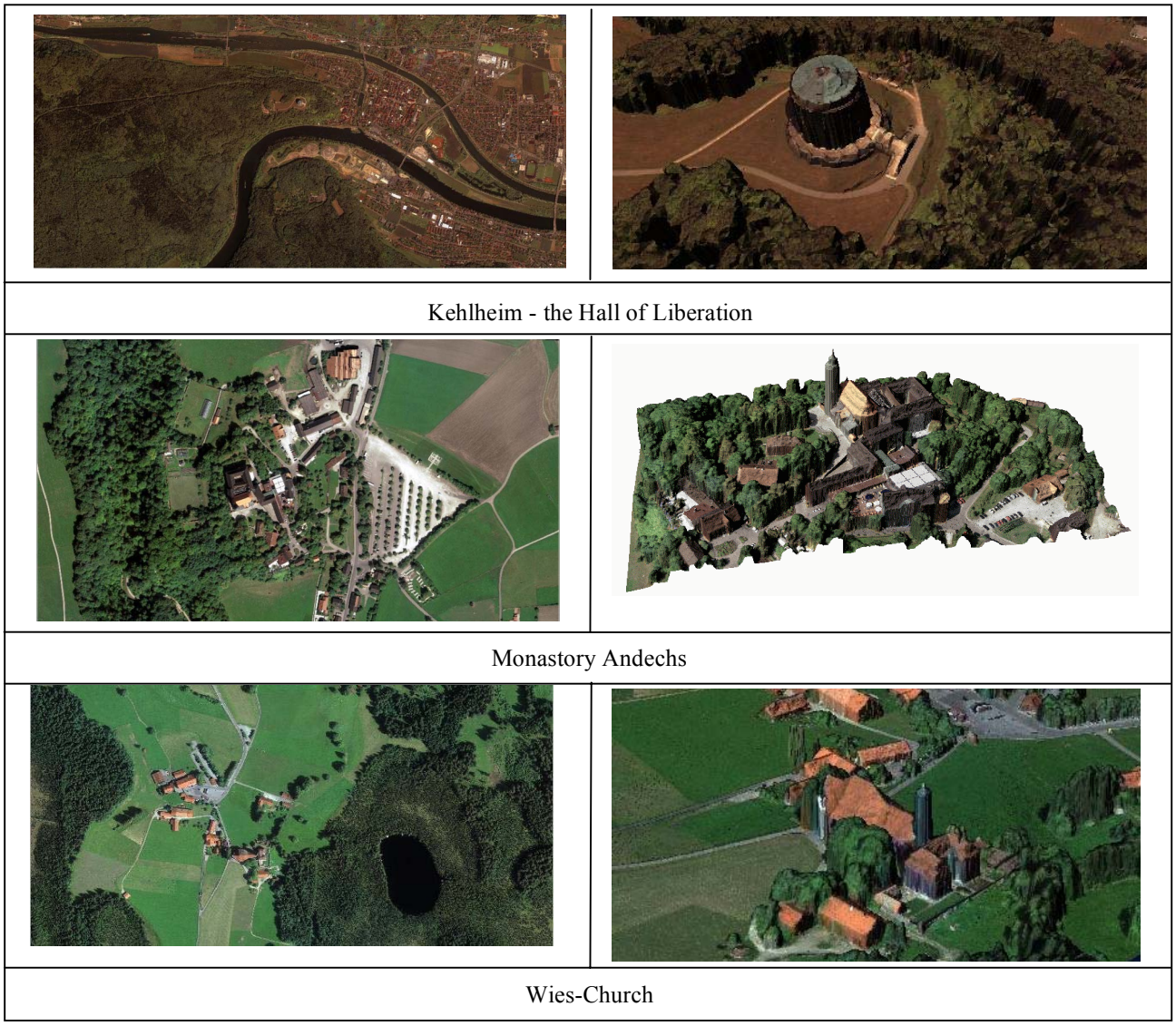

Figure 7. Various reconstructed regions

[12] A. Gruen, E. Baltsavias, O. Henricsson (Eds.), "Automatic Extraction of Man-Made Objects from Aerial and Space Images (II)". Birkhäuser Verlag, Basel, 1997

[13] G. Sithole, G. Vosselman, "Automatic structure detection in a point-cloud of urban landscape", Proceedings of the 2nd GRSS/ISPRS Joint Workshop on Remote Sensing and Data Fusion over Urban Areas (URBAN2003), pp. 67-71, Berlin, Germany, 2003.

[14] M. Niederöst. "Detection and reconstruction of buildings for a 3-D landscape model of Switzerland," Proceedings of UM3/2000 Workshop, Tokyo 2000.

[15] I. Laptev, H. Mayer, T. Lindeberg, W. Eckstein, C. Steger and A. Baumgartner. "Automatic extraction of roads from aerial images based on scale space and snakes," Machine Vision and Applications (2000) 12: 23–31

[16] M. Gerke, B.M. Straub and A. Koch. "Automatic Detection of Buildings and Trees from Aerial Imagery Using Different Levels of Abstraction," Publikationen der Deutschen Gesellschaft für Photogrammetrie und Fernerkundung, Band 10, Eckhardt Seyfert (Hrsg.), pp.273-280, 2001.

[17] Q. Li, X. Zuo, B. Yang. "3D Highway Landscape Modeling and Visualization," The International Archives of Photogrammetry and Remote Sensing, Vol. 34, Part (5), Kunming, 2002.