Low Cost High Quality 3D Virtual City Models

Milan FTÁČNIK & Peter BOROVSKÝ & Martin SAMUELČÍK

Dept. of Computer Graphics and Image Processing; Faculty of Mathematics, Physics and Informatics
Comenius University of Bratislava

1 ABSTRACT

There are several projects aimed at creating the 3D virtual city models for bigger cities all over the world. In order to bring these models closer to smaller cities to be used e.g. in urban planning or regional development the trade-off between cost and quality of the model should be solved. From this point of view we discuss the input data available at low cost, the relatively simple work flow in creating the model and the quality parameters of the final model.

We discuss the low cost input data such as DTM and terrestrial images taken with ordinary digital cameras with special focus on the cadastral data providing the building footprints that can be used for reconstruction. The simple work flow includes the data-processing providing the block flat-roof model and also the silhouette based building reconstruction method. Special focus in the work flow is given to relatively simple texturing and rendering to obtain the photorealistic large scale model. The quality parameters of the model are discussed, mainly the navigation, speed, accuracy, manageability of the model data and subjective feeling of the user. The proposed procedures can achieve many of them in the final model as can be seen from the examples of low cost high quality 3D virtual model of Bratislava.

2 INTRODUCTION

In more than 100 cities all over the world the 3D virtual city models are being created, solving the problems of modeling, texturing, storing and rendering of such models for various purposes. Mainly the bigger cities are using virtual models in order to better address the problems of urban planning, regional development, virtual tourism, flood prevention, saving of the cultural heritage and many others. Sooner or later the models will be available also for smaller cities that need to address similar problems. For them the cost of the model is the issue on one hand and the quality of the model on the other. This paper offers one possible solution to the problem how to create low cost model with satisfactory quality e.g. in urban planning.

3 INPUT DATA FOR LOW COST MODELS

The classification of the input data is here provided mainly in accordance to the cost of their acquirement. At low cost one can have digital terrain model (DTM), terrestrial images with low end camera and also the cadastral data. Usually the aerial photos cost more, in this paper we take in this category also the height photos taken by the low end camera from the high buildings in the city.

3.1 DTM

High-quality samples and lower-quality DTMs with larger coverage can be found freely on the internet [USGS02]. Therefore we treat them as a base for low cost virtual city reconstruction giving the model realistic terrain which it stands on. One can get DTM also as a by-product of the photogrammetric processing of the aerial photos if they are available. The details of the DTM´s role in the model are discussed in the section 5.

3.2 Terrestrial images

This data can be taken with low end camera from the ground usually providing at least two views on the building. They can be used for the refinement of the building´s model or for more precise texturing of the building. Some software need special high end camera, see e.g. [Graz03], some are able to use the ordinary camera with the calibration before modeling. In section 7 we present the modeling method using the silhouettes extracted from the terrestrial images providing the more detailed model mainly for the symmetrical buildings. In general the terrestrial images bring more details to the model, like windows, doors, protrusions etc.

3.3 Cadastral data

This is the information source containing 2D vector data like street index or the building footprints, which are very helpful for the city modeling. Cadastral data are usually operated by the particular cadastral authorities. In some cases they are collected and maintained directly by the cities themselves [Petzold03]. The analogue cadastral data exist for every region and can be relatively simply transformed to the digital form if they are yet not available like that. Apart from their content, they are valuable for the accuracy and relatively large area coverage. The usage of this data type in low cost 3D city modeling is discussed in the section 6.

3.4 Aerial photos

This is the prevailing source of data for the creation of the 3D city models. They can be obtained either from the specialized companies photographing the city with the professional camera providing all parameters of the photos and usually offering also the photogrammetric processing (at the prize of 110 EUR for the square area 100x100 m^2) or by the “height“ photos taken from the higher buildings in the city like churches or chimneys or other dominants. For the purposes of the low cost modeling we use them mainly for texturing but also their modeling role (not using photogrammetry) as discussed in this paper. The merging of aerial and terrestrial data in automatic generation of 3D model is discussed in [Takase03] and [Fruh03] using also the laser scan data which is relatively new source of higher cost data which do not need photogrammetric processing.

4 WORK FLOW IN THE CREATION OF THE MODEL

The proposed work flow (Figure 1) consists of the line leading to flat roof block model located on the textured DTM. The block model is created from the building footprints (from the cadastral data) by elevating them to the average height and then texturing them by mapping the aerial (or height) photos onto it as well as on the DTM. This results to the photorealistic large-scale model of the city, which can be then refined either in large scale or in the small scale. In large scale it means to add the information about the real building heights to the block model or to find the height by manual elevation of the footprint until it fits with the texture taken from the aerial photos. For the refinement in small scale the silhouette based modeling method is being proposed. It uses terrestrial images and special information called azimuth connected with the orientation of the image. If the city decides to use other types of input data for modeling (either aerial of laser scanned) they can provide alternative (more precise) models in the area chosen which can be georeferenced in the created large scale model by the proposed work flow.

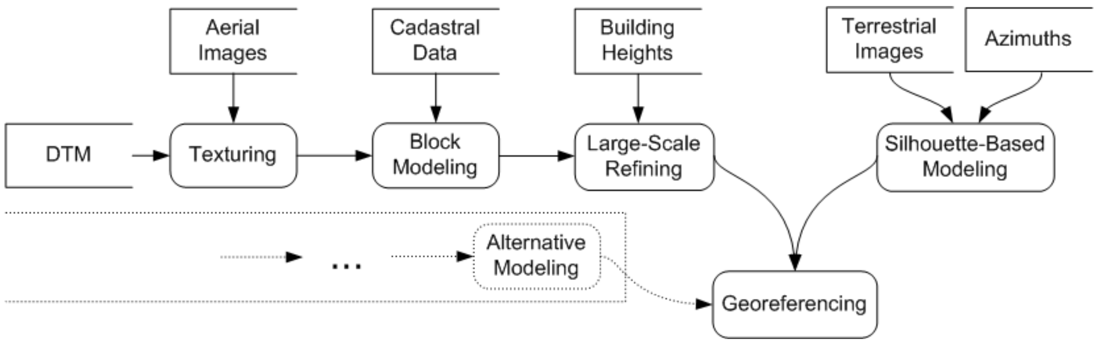

Figure 1: Dataflow diagram for the processing stages with mandatory input described in the text.

5 DIGITAL TERRAIN MODEL

Despite the wide offer of a contemporary DTM-processing applications ([TEC03] shows an exhausting list), we have developed our own software solution, exploiting a virtual city creation-specific approach. Description of its capabilities can be found in [Boro03]. Figure 2 shows a brief overview of the modeling features added to the standard DTM viewer functionality, especially the pin-planning tool (helping to sketch new plans) and a color fill technique we call flood painting. Next sections analyze DTM "cheap" texturing methods designed to enrich a low-cost terrain model with visual details applicable to the city buildings, too. This is far beyond the scope of the pure terrain viewers and it leads to the photorealistic city model(er)s, gaining results like the one shown in the Figure 6.

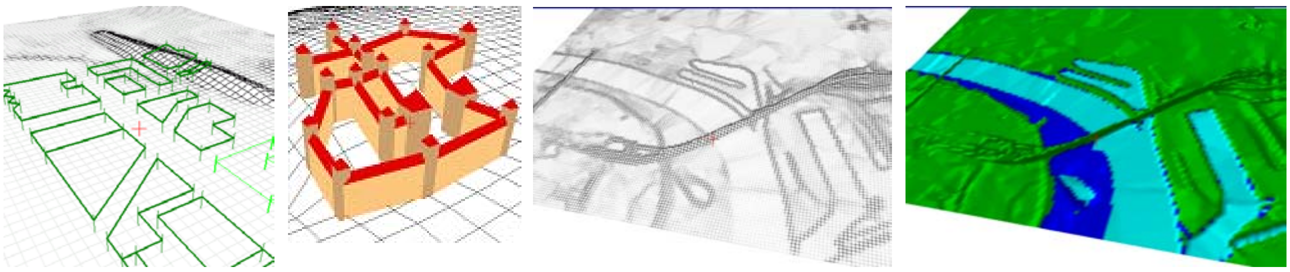

Figure 2: Planning a virtual city on the DTM, medieval city walls as a plan representation, Bratislava river harbor DTM and its flood levels.

5.1 DTM Texturing

Texturing is a common photorealism technique. Otrophotomaps are usually applied as textures in the terrain visualization systems. Texture mapping (2D texture space ↔ 3D object space association) is very simple in the orthophotomap case: ground point of *(x,y,z)* coordinates, where *z* denote the height, would have *(f(x),g(y))* texture coordinates, where *f*, *g* are simple functions modifying the texture space so the texture domain copies terrain region it covers (planar texture projection approach, illustrated in the left part of the Figure 3).

We have proposed more accurate method for aerial photograph texturing. It is an inverse simulation of the photography process: light rays carrying the texture color information are cast from the camera to the terrain surface. Therefore we refer to it as a raycasting, an algorithm suitable both for terrain texturing discussed in this section and building texturing, discussed in section 5.3.

Our method precisely maps the texture onto the terrain surface, as it is illustrated in the right part of the Figure 3. If the photograph was taken by the system similar to the ideal pinhole camera (which is true for standard survey systems), raycasting resembles the central perspective texture projection (texture domain is set to the reversed image from the positive photoplane), which can substitute our method. Nevertheless, raycasting is more general than the usual perspective texture projection, because it can simulate more complex or less precise camera systems and remove any known optics distortion. Theoretically, it can simulate any higher-distorting "low-cost" camera, originally not intended for the aerial photography. Moreover, input parameters for our method (focal distance, photoplane dimensions, etc.) can be easily transcripted from the camera calibration protocol.

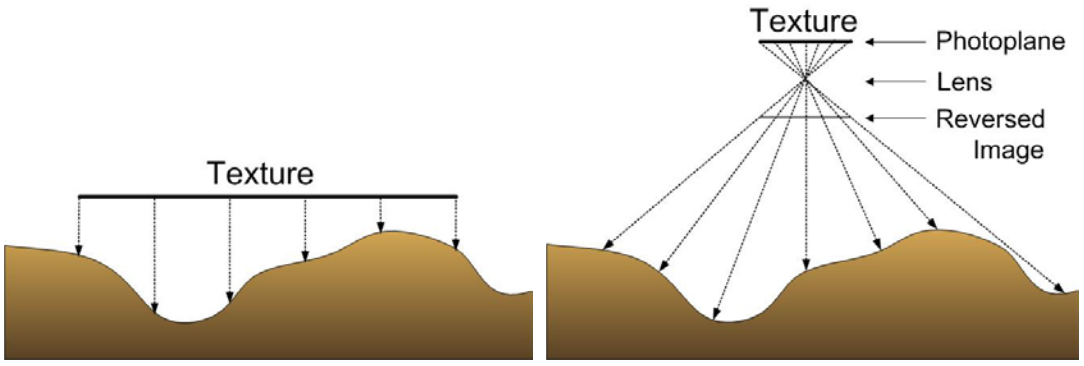

Figure 3: Difference between planar projected ortophotomap and the aerial image cast from the pinhole camera.

How to accurately relate the given texture to the DTM? In case of knowing the camera exterior parameters (position & orientation) our software solution fixes the virtual camera in the proper position above the terrain. In case of missing or incomplete exterior parameters, we developed two camera-positioning methods: manual and semiautomatic. Manual method allows user of the modeling software to interactively move and rotate the virtual camera, unless the raycasted texture fits to the terrain. He should choose some control objects (texture points/segments for which the terrain location is known) for evaluating the exact texture placement.

Manual camera-positioning method is very quick (it takes only about a minute in our software to fit an aerial texture to the ground) and as precise as the user wants. We appreciated it for adjusting vaguely set values like camera altitude (very suitable for GPS systems with approximate localization). However, it has two significant drawbacks: it is user-limited and it does not handle the camera interior parameters (e.g. focal distance). Even we can design a user interface capable of changing interior parameters interactively (normally, they are manually typed from the available camera calibration documents), tedious control of at least 14 parameters (pinhole camera exterior parameters plus focal distance, texture dimensions, position and rotation in the photoplane), simultaneously by one user would be far from an intuitive approach, with no guarantee of quick result.

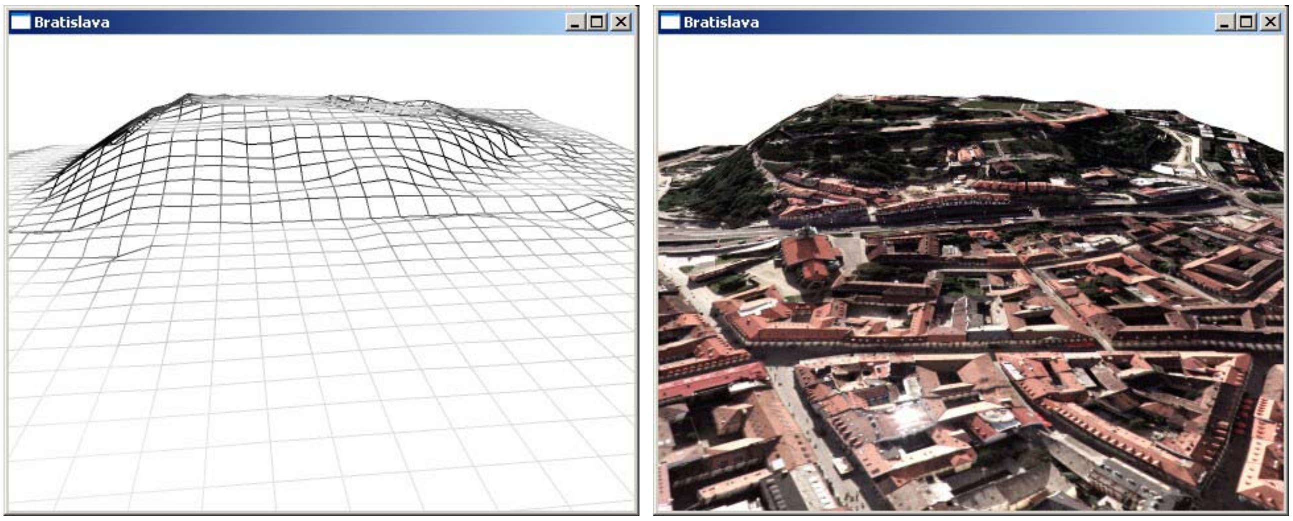

Figure 4: DTM wireframe visualization and the aerial photograph raycasted from the manually positioned virtual camera.

5.2 Texture Mapping Optimization

Raycasting in the software allowing the user to interactively set the camera parameters achieves satisfactory results in case of unknown/incomplete camera parameters. While trying to extend the interactivity from the basic parameters like camera position to the all other parameters, we found it hard to control and even evaluate the result, e.g. changing one parameter could lead to better texturing in one control point, but worse in some other.

Due to complexity of parameter setting, we proposed more simple and user-friendly method and built an application that computes the camera parameters. Basic idea is that each texture mapping can be set up with couple of pairs consisting of 3D scene points and their corresponding (u,v) coordinates. At least four such pairs (with scene points not lying in the same plane) determine the most simple central perspective texture mapping. The more parameters influence the mapping, the more pairs are needed for definite specification. The user interferes the computation only once; for setting the (u,v) and corresponding scene coordinate pairs. This is the only one possible source of an inaccuracy; result can be as precise as the user input. Therefore, we let the user to specify as much coordinate pairs as possible. Since we are combining more input pairs, we can achieve results with even less than one texel accuracy.

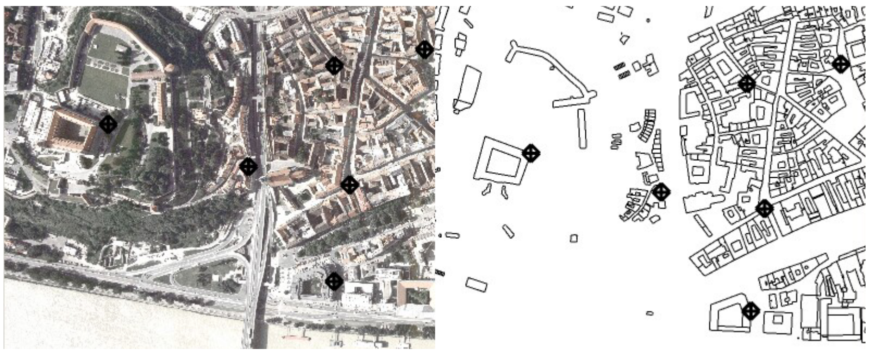

Figure 5: Corresponding texture↔scene points specified by the user (false colors and widths applied due to the print issues). For the scene coordinate specification our application uses ground data (the right half). Hence, we got rid off the third dimension and allow user to work conveniently in 2D.

How to compute texture parameters from the user input? Instead of explicit computation of 14 parameters, which would not always lead into definite solution in case of extra input coordinate pairs, we have chosen the stochastic approach. Our algorithm tries to find the best solution while it generates parameters on the random base. Particular solutions are compared by an error function: each input coordinate pair produce an error, which is a difference between input and computed (u,v) coordinates. The error function is an average of all errors.

We exploit the well-known hill-climbing algorithm to find the optimal solution with minimal error. In each step, algorithm moves parameters randomly and checks if the error function gives better results. It searches the ε-surrounding of the actual parameters until it finds better result (smaller error mean), which then becomes an actual parameter set. If it did not succeed after preset number of guesses, greater ε is chosen. Number of steps as well as the initial threshold error mean is given by the user. We refer reader to the more in-depth details in [Boro03].

5.3 From DTM To Building Texture Mapping

Our raycasting method maps the texture on any object in the virtual scene, in general. We have demonstrated it on the terrain example, however it is obvious how to extend it to building models usually made from triangle meshes: each vertex in the mesh asks for its (u,v) coordinates, regardless it is visible from the camera position. Unpleasant artifacts in invisible faces can be suppressed by involving more photographs from different angles, so the object is fully texture-covered. Fortunately, city buildings represent objects coverable by only a few photographs. Moreover, in most cases, each building façade (at least its upper part) should be visible somewhere from the air and the overall city model can be viewed as a height field defined all over the modeled area. City model semiautomatically textured by couple of aerial photographs therefore yields satisfactory photorealistic results in the flythrough applications. For such virtual flythrough just a simple building block models as described in the next section is sufficient.

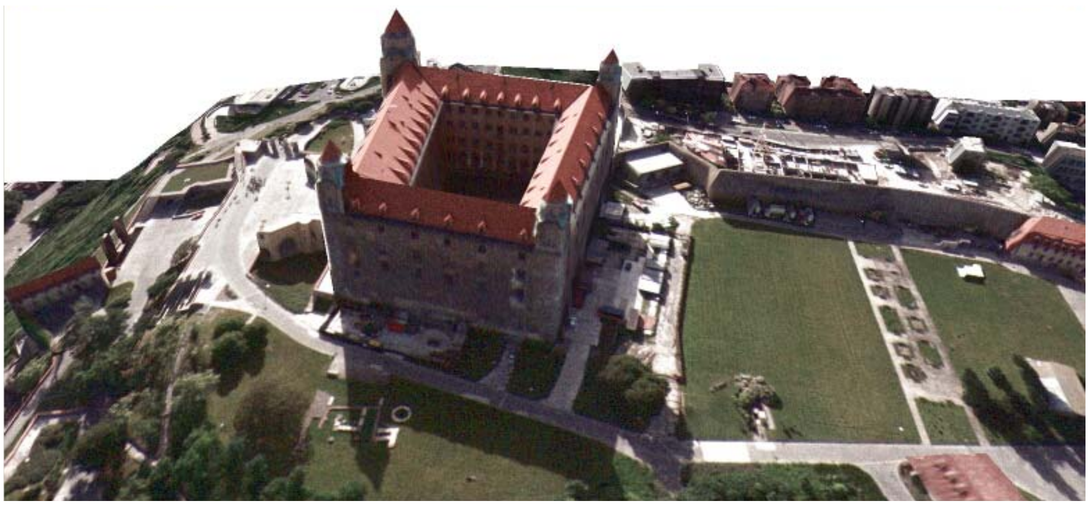

Figure 6: Semiautomatic texture mapping was used on the DTM and the castle model input shown in the Figure 8. Source texture is an aerophoto taken from about 2100 meters above the terrain.

Recently, in [Fruh03] a semiautomatic texturing technique adapting the Lowe's algorithm, essentially same to ours, appeared. They use it for laser scans, where each 3D mesh face, simultaneously photographed and scanned, associates the best texture among the all possibilities after setting couple of corresponding texture↔scene points. Our approach differs in three main issues. At first, we can use any reasonable texture, even a distorted photograph found on the Internet. Next, only one aerial photograph alone can be applied to the large city area. If it is an orthophotomap, only a ground and the building roofs would be textured, since the ortophoto contains no facades. If it is a photograph which captured building from a different angle, it can texture also all visible facades as it is shown in the Figure 7. Compromise between coverage and the angle most suitable for facades is a photograph captured from an appropriate high building, which makes data acquisition relatively cheap. Since the façades invisible or stretched in the photograph are textured with high perspective distortion (Figure 7), more photographs taken from different angles are needed to apply.

Finally, we have designed an user interface that abstracts from the third dimension, since we use a precise 2D map (e.g. cadastral data like street map or building ground plans) for fixing the (u,v) coordinates to the ground (Figure 5). Our application computes the third dimension from the DTM. This makes texture mapping as easy as clicking the corresponding points in two pictures.

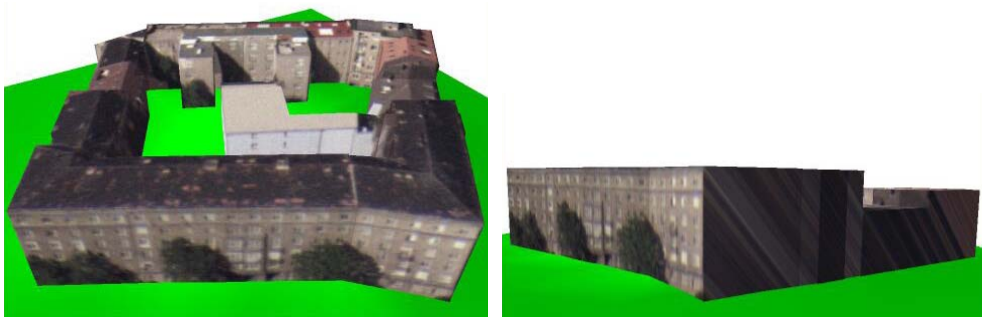
Figure 7: One texture mapped to the structure from its front side (left) leads to a perspective distortion from the other sides (right).

6 LARGE SCALE BLOCK MODELING

Ideal city modeling input would be a dataset containing all city structures in a 3D vector format (Figure 8). However, such an input is very costly. We found out that large city structures satisfactory for virtual flythroughs and photorealistic pictures taken from farther distances may be modeled from cadastral data, which includes vector data like street index or ground plans, suitable for determining the texture mapping accuracy (Figure 5) or the building shapes in two dimensions. Despite their usual 2D nature, they can help to build a large city model with minimal effort. Only requirement is to have the software exploiting their biggest advantage: large area coverage. In this section we focus on the building ground plans as the most essential part of the cadastral data. We introduce a simple block-modeling technique (one of the results is shown in the Figure 7) and also the idea of constructing 3D models from the aerophotographs, where the footprints can specify the first level of detail.

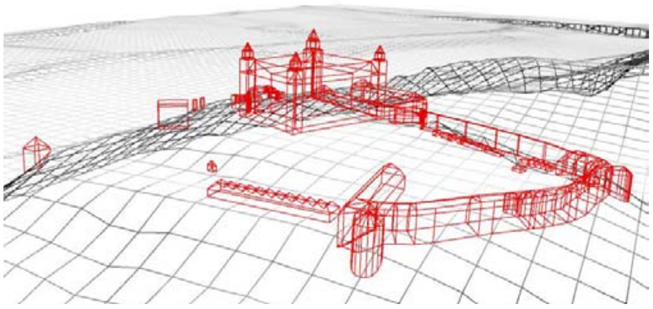
Figure 8: Bratislava Castle roof plans wireframe achieved by effortful manual photogrammetric processing.

We are mapping footprints on the terrain and exploiting their features for 3D modeling in two ways. At first, we use them as the control objects for our texture raycasting method described above. The relation between DTM and footprint coordinates is usually known, since the DTM data inputs are expected to have the latitude/longitude dataset specification and their georeferencing is straightforward. Hence, mapping of the building plans onto the terrain model is as accurate as the DTM and cadastral data precision.

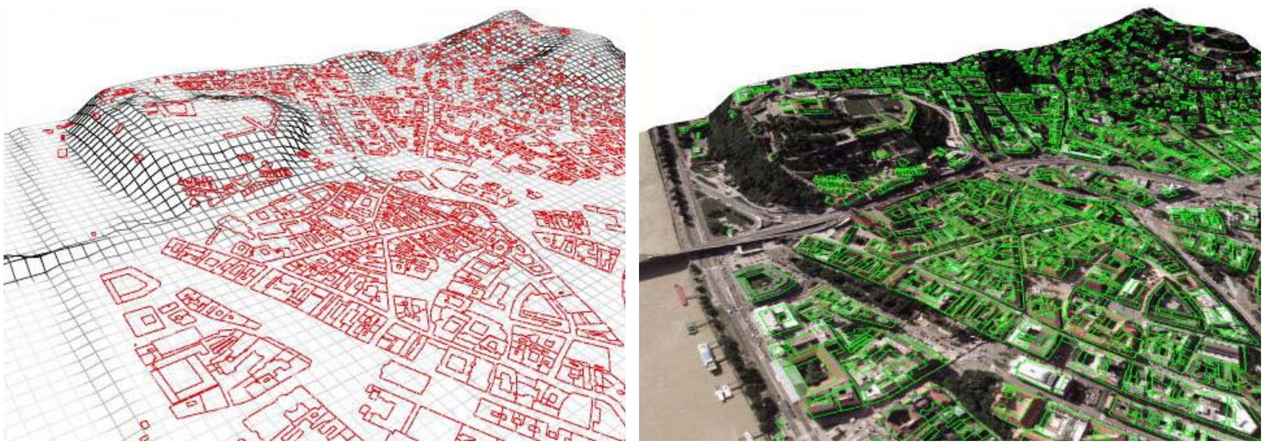
Figure 9: Building footprints and the aerial photograph mapped onto the terrain model, controlled by overlapping ground plans.

Most important footprints exploitation is a rough city block model specification: having the 3D building ground plan and its (average) height we assemble the model with exact ground lines and flat roof. Although the roofs in this model are less precise (than e.g. the roofline models in [Graz03]), the model is very competitive for the walls accuracy, since the real walls are usually perpendicular to the terrain. The building plan – the building contour – is sufficient for the wall geometry and topology specification. On the other hand, roof contour datasets are less accurate for the wall specifications, because an ordinary building roof is often hanging over the wall, setting an offset to the building contour. The best way is to combine both ground and roof plans as it is show in the Figure 10. However, the flat-roof models are still good enough for the overall city snapshots or virtual flythroughs.

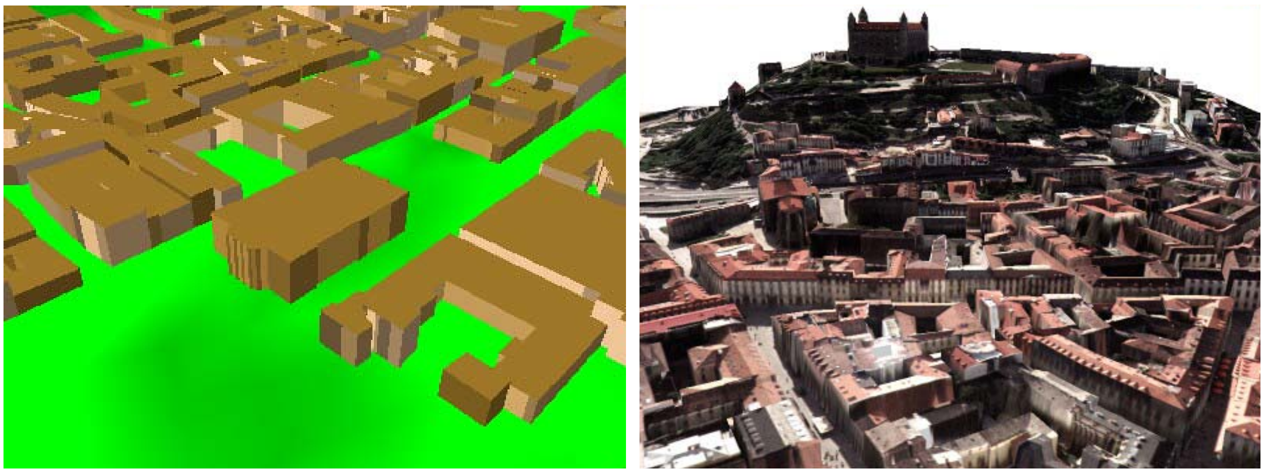
Figure 10: Flat-roofed building block models made from the footprints and the city visualization. The only one exception in the right picture is a castle surrounding, made from the roof plans from the Figure 8, all the other buildings are flat-roofed.

Ground plans dataset does not contain any altitude information. Anyway, we are still able to estimate the heights in large city areas, because they are often based on a typical regular pattern. We use a pseudorandom algorithm, which generates the building height with regard to a dominant (average) height in the location, following the expected architectural style. It generates the number of floors and adjusts the elevation according to the neighborhood. Surprisingly pleasant results are gained in the homogeneous areas as it is shown in the Figure 10, where no real heights were applied and the model construction took only as long as the data fetching.

6.1 Refining 3D Models from Aerial Photographs

Up to now, we described the texturing and the rough block-modeling stages from our dataflow diagram (Figure 1). We are able to achieve very accurate texturing, however the pseudorandom building height estimation devalues its precision in high details. One solution is to acquire the necessary altitude data, which requires a survey if they are not available. We offer a more comfortable solution, based on the texturing accuracy. Having the terrain and the camera set in the virtual scene, user of our software can easily adjust particular model height by stretching it into its eligible shape as it is illustrated in the Figure 11.

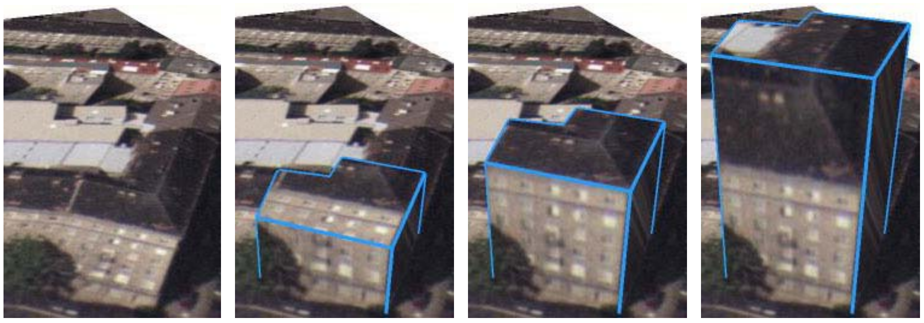
Figure 11: Modeling the geometry from the only one aerophoto texture: none, underestimated, real and overestimated building heights.

Adjusting the height from one suitable photograph is very simple due to the vertical facades. Extracting any other geometry information needs slightly sophisticated modeling interface, e.g. the sloping roof on the building from the Figure 11 would require to set a roofline somewhere in the middle of the box top and elevate it into the space unless the texture fits on the edges of the temporary roof shape. Same principle can be applied to all geometry controllable by the texture: texture mapping is conserved, the model changes, and the texture rays incident new shape.

For exact geometry determination, the user would need to raycast two photographs in general (2 intersecting rays define one 3D point) and the modeling would resemble the photogrammetry techniques. However, we enriched the standard stereoscopic photogrammetry by the use of an ordinary photograph (instead of a highly specialized hardware and its outputs, see the reasoning in the section 5.1) as well as the footprints and other auxiliary data. Moreover, merely one photograph is sufficient in many cases (photogrammetry needs 2 or even more). The fastest large-scale block model refinement requires to estimate at least the building heights, which is the most essential geometry information missing in 2D cadastral data. The fast interactive method illustrated in the Figure 11 substitutes the mandatory height data from our dataflow diagram. Terrestrial photography can be involved to achieve finer texturing and hence detailed geometry. An alternative method (not requiring the DTM and other space dependencies) of model refining is introduced in the next section.

7 SILHOUETTE-BASED MODELING

We now present a method for reconstructing building from terrestrial images. This approach can be used with a good precision for symmetrical buildings and also in the case we do not need a model with a high count of polygons. It is suitable for replacing particular buildings from the large-scale model (places of higher interest) with more detailed geometry, constructed from scratch.

As an input we need at least two terrestrial images (ordinary photos) of the building. For each terrestrial image we also need an angle of the directional vector. We call this angle azimuth (e.g. if we are looking in the image on the building from the north, the azimuth is equal to 0). If the footprint of the building is given, we can use it as another optional input parameter for a more accurate reconstruction. Because as input we need only few images and few azimuths to create reconstruction and we don't need images with high resolution, this method uses only low cost inputs. Some buildings can be reconstructed from only one image.

Output of this method is a 3D model of the building. We can also use given images to texture the model, making it to look more realistic. We store the output in VRML file so it can be viewed via Internet. Because resulting model doesn't have high count of polygons and also images don't need to have high resolution, downloading and viewing of the model is fast. It also has its inner format for storing geometry and topology of the model.

7.1 Related Work

There are many papers related to image-based building reconstruction. Most of them are based on finding projection matrices for each image, detection of corresponding objects (points, lines, windows, etc.). As a result one can have point clouds [Metro97] or polygonal model [Debevec96]. The visual hull method (intersecting of silhouette cones) is usually used for reconstruction of small objects from a set of images using a volumetric intersection [Dyer01], [Rama00]. Another algorithm with a similar approach is used for generating models of trees from terrestrial images using B-rep [Shly01]. The exact position of the camera must be given. Then the silhouette is extruded into the cone with its top at the camera location and the intersection is calculated.

7.2 Method Description

First we have to find a silhouette in each given image. User does this manually because in some images it isn't possible to find it automatically. Then the algorithm extrudes a silhouette in the direction of azimuth given for each image. It extrudes silhouette into a prism so the opposite lines will be parallel. If the footprint of the building is given, we get more accurate result by extruding that footprint in the direction of z axis. By intersecting of these prisms we get a visual hull of the building that can be treated as its approximate 3D reconstruction. This still leaves some scaling errors, but if images are taken from a longer distance from the building and the viewing direction is perpendicular to the facade, the result is more accurate. Also the algorithm is more successful in the reconstruction of symmetrical buildings like towers. For more accurate results some steps can be added to the process. Images taken by low-cost cameras are often warped by the perspective projection, so it is suitable to unwarp the image first. At the end of the algorithm this image can be back projected onto model in the direction of azimuth.

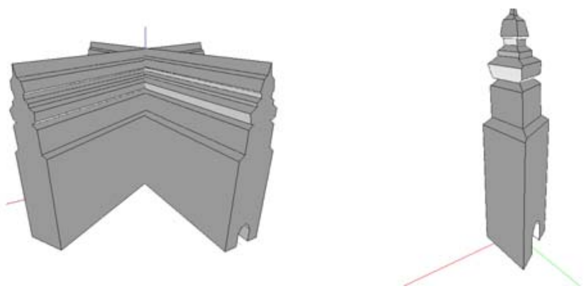

Figure 12: Example of silhouettes extruded into prisms and their intersection (St. Michael's Tower in Bratislava).

As we can see in the Figure 13, models of symmetrical buildings with satisfactory quality are gained. Based on the low cost input, resulting models with mapped textures yield high quality. Additionally, the creation time is short; it takes only a few minutes to achieve presented results.

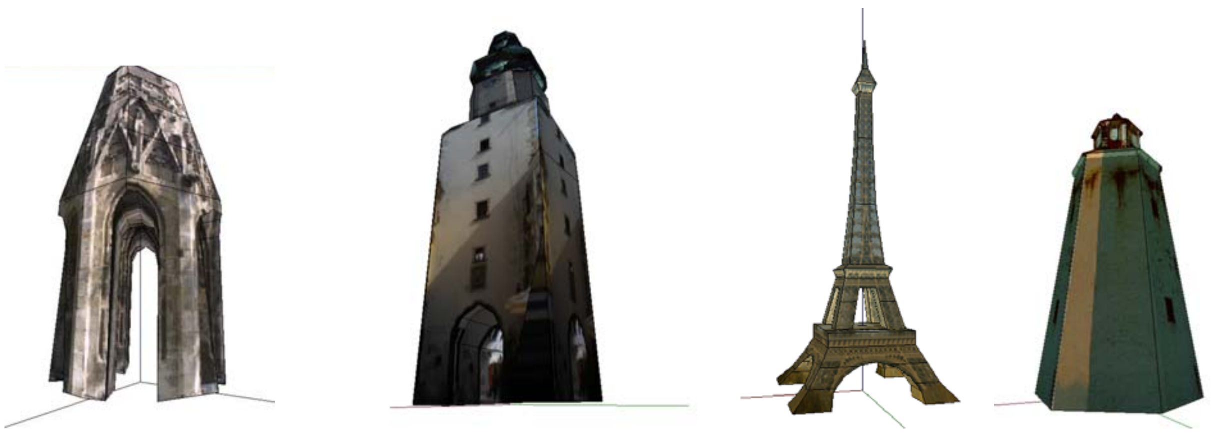

Figure 13: Silhouette-based models: towers from Janko Kral orchard and St. Michael's Gate (Bratislava), Eiffel Tower in Paris and a lighthouse.

8 THE PARAMETERS OF QUALITY

The quality of the created model is usually discussed only with respect to the accuracy of the model, which strongly depends on the scale of the input data. The DTMs found on the Internet are usually provided as the grids with around the 20 x 20 m^2 resolution, the cadastral data are provided in the scale of 1:1000 up to 1:2000, the aerial photos have usually the resolution 20-40 cm per pixel, all of them satisfactory for the necessary accuracy. The other parameters of quality that we treat as important are mainly the navigation in the model, speed (with special focus on Internet application), manageability of the model data and subjective feeling of the user.

The large scale model, which can be created at low cost by the proposed methods provides mainly the bird's-eye navigation in the model, where the subjective feeling of the user is very realistic and it strongly depends on the quality of the textures provided by the aerial photographs. We have developed a special viewer for rendering the overall model in the real time, including also the limited walk through mode that enables the views close to above the ground.

To achieve the reasonable rendering speed of we have to solve the problems of rendering, culling of the objects, levels of detail (LOD) of the model. This is much more urgent, when the model should be accessible via the Internet. If the models should be used in real time, the fillrate must be around 25-30 fps. Even in the large-scale model proposed in this paper this is achievable.

Manageability of the model data means to provide the user with the desired part of the model in reasonable time. Usually two databases are being used: one for storage of the modeling data including the GIS data and the second for the data used for the viewer.

The feeling of the user is a subjective parameter of quality reflecting the photorealistic view of the model. Even the rough low-cost models achieve very positive acceptance in large scale. Moreover, the proposed refinement methods support the photorealistic effect.

9 CONCLUSION AND FUTURE WORK

The methods for the creation of the low cost 3D city model have been presented. They include the photorealistic texturing of the DTM and also texturing of the rough flat roof block model, created from the building footprints, taken from the cadastral data. Methods for further refinement of the model have been proposed, including the large-scale geometry refinement by interactive fitting of the building height with the underlying texture and also the silhouette based modeling using the low cost terrestrial images, which provides geometry refinement on smaller scales.

In the future we shall continue working with the geometry and texture refinements, we want to implement our image-based modeling ideas from the section 6.1, on the other hand we would like to examine the automatic extrusion methods in our silhouette-based modeling and in general, we are seeking for more automated techniques, that would assist (but not artificially substitute, as it is often seen in some automated projects, resulting in much more limitations than the authors wanted) the user to quickly (re)construct desired virtual city. Accurate georeferencing and the combination of all the methods in the final low cost model are also the part of our future work as well as the elaboration of parameters of quality in the existing city models.

AKNOWLEDGEMENTS

This work has been supported by APVT grant No. APVT-20-02550. Vector data and the aerophotographs have been provided thanks to EUROSENSE, Ltd and the City of Bratislava. Terrestrial photographs were captured by the members of Virtual Bratislava team.

REFERENCES

[Petzold03] Petzold, B.: 3D City Models – Nice Toy or Basic Information for the City Council?, FIG Working week, Paris, April 13-17, 2003
[Takase03] Takase, Y, Sho, N., Sone, A. and Shimiya, K.: Automatic generation of 3D city models and related applications, International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences, Vol. XXXIV-5/W10, 2003
[USGS02] USGS (U.S. Geological Survey) Geographics Data download from the EROS (Earth Resource Observation Systems) data centre, http://edc.usgs.gov/geodata
[TEC03] U.S. Army Topographic Engineering Centre, Survey Of Terrain Visualization Software, February 2003, www.tec.army.mil/TD/tvd/survey
[Graz03] VRVis 3D City Modeling, http://www.vrvis.at/ar2/city_model/city.html
[Boro03] Borovský, P.: Large-Scale Virtual City Models and Tgeir Visualization, rigorous thesis, FMFI UK Bratislava, 2003, www.sccg.sk/~borovsky/papers/rigorous.htm
[Fruh03] Früh , Ch., Zakhor, A.: Constructinng 3D City Models by Merging Aerial and Ground Views, In IEEE CG&A 6/23, pp. 52-57, 2003.
[Dyer01] Dyer, Ch. R., Volumetric Scene Reconstruction from Multiple Views, in Foundations of Image Understanding, pp. 469-489, Kluwer Boston,
[Rama00] Ramanathan, P., Steinbach, E., and Girod, B., Silhouette-based Multiple-View Camera Calibration, Proc. Vision, Modeling and Visualization 2000, pp. 3-10, Saarbruecken, Germany, November 2000.
[Debevec96] Debevec, P. E., Taylor, C. J. and Malik, J., Modeling and Rendering Architecture from Photographs: A hybrid geometry- and image-based approach, In Computer Graphics (SIGGRAPH '96 Proceedings), volume 30, pp. 11-20, New Orleans, Louisiana, 1996.
[Shly01] Shlyakter, I., Rozenoer, M., Dorsey, J. and Teller, S., Reconstructing 3D Tree Models from Instrumented Photographs, in IEEE CG&A 3/21, pp. 53-61, May/June 2001.
[Metro97] Klaus, A., Bauer, J., Karner, K. and Schindler, K., MetropoGIS: A Semi-Automatic Documentation System, in PCV02, Graz, 2002.
Ma, W. Y. and Manjunath, B. S., Edge Flow: A Framework of Boundary Detection and Image Segmentation, Proc. IEEE Conf on Computer Vision and Pattern Recognition, 1997