Scene reconstruction using structured light

Roman Kapusta[*]

Slovak University of Technology Bratislava
Faculty of Electrical Engineering and Information Technology
Bratislava / Slovakia

Abstract

This paper will discuss digitization of 3D object using active method called the structured light method. This method is aimed at getting good quality reconstruction for low cost. The method has simpler algorithm then stereo vision, thus it works faster, and gives comparable results.

Keywords: structured light, 3D reconstruction, triangulation

1 Introduction

Three dimensional digitization is a long lasting research topic in computer vision. Nevertheless rapid evolution of shape digitizing and reconstruction methods has taken place in recent years. It is achieved thanks to the increasing power of desktop computers. Prices of accurate CCD sensors are falling down owing to growth of digital photography. The expense for methods is lowering and speed and accuracy are rising.

This progress is most influenced by low-end digitizing methods. These methods are not depending on special hardware, but on cheap and common equipment. Ordinary PC and digital camera are usually everything needed by low-end methods. This group of methods can get good results in spite of its low cost as we will show. Cost effective methods can be divided by their approach into passive and active.

Passive methods are based on two or more shots of the scene to recover 3D data (stereo vision, structure from motion). They are usable in great scale of problems where the precision is not the main goal. It is not an easy problem to find a correspondence between two images which is essential in the recognition process. This step consumes most of time spent on reconstruction. Moreover smooth surfaces without pattern are impossible to recover correctly. Such surface has no “corners” which are fundamental for traditional methods.

Active methods interact with the scene. Specific pattern is cast onto the scene, so the problem of finding corresponding corners becomes easier. One shot is sufficient to recover scene depth.. The object shape becomes more apparent thanks to the projected pattern but the texture of object is lost. Main drawback of this method is the need to darken whole scene and project light pattern. Described method is also called the structured light method.

This paper will give an overview of structured light method in details, its advantages and disadvantages. In chapter 3 entitled “Projective geometry” notation and important theory is explained. Chapter 4 with title “Triangulation” covers analytical details of used triangulation method. Chapter 5 deals with building of complete model from set of depth maps. Finally there are results and conclusion.

2 Structured light

This group of the digitizing methods is capable create depth map from single shot of the scene. Chia [5], Proesmans [8], Vuylsteke [13] developed digitizing techniques in this area.

The scene must be darkened and lit only by structured light, however we can use invisible infrared source of light so the object is not disturbed. The pattern is projected by strong reflector or by the fast moving laser beam onto the measured object. Problem with projectors is their small depth of field. Noise from other light sources can be reduced using monochromatic light and making the camera insensitive to other chroma by the filter. Original texture of the object cannot be fully recovered from a single shot. The second shot is usually taken from the same position and without structured light if the texture is needed.

We can solve problem with focus using the moving laser beam instead of the projector, but it increases the cost and add problems of the synchronization. The laser must project entire grid while a shutter of the camera is open, which can add noise to the picture. Other solution is to use more shots.

If the camera (or the light) differs too much from ideal collinear projection, this fact must be taken into account. This problem is usually solved using pre-calibration and filtering or following a more sophisticated way with an autocalibration. Of course the autocalibration is not possible with single shot of the scene.

Once we have a picture the first step is pattern recognition. Projected pattern must be recovered from

[*] kapusta@decef.elf.stuba.sk

the surface. This step is dependent on pattern of the light. Often regular grid pattern is used, thus recovery of pattern is not a problem on smooth surfaces.

The next step is to find original position of source of the light. In the general scene the position of light can be recovered automatically, but this is not true for some special constellations of the camera, the light and the object. Shape of the object has remarkable influence on these constellations (Pollefeys [7]). However automatic determination of the light position is quite difficult problem in any case.

Precision of structured light method is influenced by the camera and by projected pattern. Method is also called active triangulation.

3 Projective geometry

Conventions used in this paper are taken from Triggs [11].

A point in projective n-space ($\mathcal{P}^n$) is given by a *column* $(n+1)$-vector of coordinates $\mathbf{x} = (x_1 \ldots x_{n+1})^\top$. At least one of these coordinates should differ from zero. Affine points are given by vector $(\boldsymbol{x}\;\; u)^\top$, where u is nonzero scalar. Vector $\vec{\mathbf{v}} = (\boldsymbol{v}\;\; 0)^\top$ is asymptotic direction or ideal point.

A *row* $(n+1)$-vector $\boldsymbol{\rho} = (\boldsymbol{n}\;\; d)$ specifies a plane with normal n-vector $\boldsymbol{n}$ and offset $-d$. Again at least one of coordinates should be nonzero. Plane in an Euclidean geometry has homogenous normal vector $\boldsymbol{n}$**.** Plane at infinity $\boldsymbol{\rho}_\infty = (0 \ldots 0\;\; 1)$ contains all the ideal points and no affine ones.

Point $\mathbf{x}$ lies on plane $\boldsymbol{\rho}$ if and only if $\boldsymbol{\rho} \bullet \mathbf{x} = 0$.

Plane $\boldsymbol{\rho}$ in projective space $\mathcal{P}^n$ is uniquely determined by n linearly independent points $\mathbf{m}_1 \ldots \mathbf{m}_n$. Plane is given by cross product of all points $\mathbf{m}_1 \ldots \mathbf{m}_n$: $\boldsymbol{\rho} = \mathbf{m}_1 \times \mathbf{m}_2 \times \ldots \times \mathbf{m}_n$.

Generalized cross product is a totally antisymmetric product which takes n vectors $\mathbf{v}_1 \ldots \mathbf{v}_n$ of length $(n+1)$ and the result is a vector of length $(n+1)$ that is orthogonal to all of the $\mathbf{v}_1 \ldots \mathbf{v}_n$.

Linear *transformations* are $(n+1)\times(n+1)$ matrices. Example of metric transformation matrix:

$$\mathbf{T}_A = \begin{pmatrix} \boldsymbol{R} & \boldsymbol{t} \\ \boldsymbol{0} & 1 \end{pmatrix} \tag{1.1}$$

where $\boldsymbol{t}$ is a translation vector and $\boldsymbol{R}$ is a rotation matrix. Transformation is acting by left multiplication on points $(\mathbf{x} \to \mathbf{Tx})$ and by right multiplication by the inverse on planes $(\boldsymbol{\rho} \to \boldsymbol{\rho} \bullet \mathbf{T}^{-1})$ so that point-plane products are preserved: $\boldsymbol{\rho} \bullet \mathbf{x} = (\boldsymbol{\rho} \bullet \mathbf{T}^{-1}) \bullet (\mathbf{T} \bullet \mathbf{x})$. To distinguish their different transformation laws points are called *contravariant* and planes *covariant*.

We work in projective 3D space ($\mathcal{P}^3$). A point is represented by 4-vector $\mathbf{x} = (x\;\; y\;\; z\;\; w)^\top$. Analogically a plane is represented: $\boldsymbol{\rho} = (n_x\;\; n_y\;\; n_z\;\; d)$. A line can be given by two points $\mathbf{x} + \alpha\vec{\mathbf{v}}$ or as intersection of two planes $\boldsymbol{\rho} \cap \boldsymbol{\sigma}$.

The cross product has usually only two operands limiting vectors to 3 coordinates. But as was told above, we work with 4-vectors so that cross product has 3 operands. In our formulas in next chapter only the simplest form of remarked cross product appears:

$$\vec{\mathbf{a}} \times \vec{\mathbf{b}} \times \mathbf{0} = \begin{pmatrix} a_1 \\ a_2 \\ a_3 \\ 0 \end{pmatrix} \times \begin{pmatrix} b_1 \\ b_2 \\ b_3 \\ 0 \end{pmatrix} \times \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \end{pmatrix} = \begin{pmatrix} a_2 b_3 - a_3 b_2 \\ a_3 b_1 - a_1 b_3 \\ a_1 b_2 - a_2 b_1 \\ 0 \end{pmatrix}, \tag{1.2}$$

where $\mathbf{0}$ is a 4-vector $(0\;\; 0\;\; 0\;\; 1)^\top$ representing point at space origin.

Formula (1.2) is numerically equivalent to well known:

$$\boldsymbol{a} \times \boldsymbol{b} = \begin{pmatrix} a_1 \\ a_2 \\ a_3 \end{pmatrix} \times \begin{pmatrix} b_1 \\ b_2 \\ b_3 \end{pmatrix} = \begin{pmatrix} a_2 b_3 - a_3 b_2 \\ a_3 b_1 - a_1 b_3 \\ a_1 b_2 - a_2 b_1 \end{pmatrix}. \tag{1.3}$$

4 Triangulation

We have one shot of the object lit by the structured light. The first step is to localize all key-points on the image. This step is strongly dependent on the specific problem area. Some research was done on methods capable to work with scenes containing sculptures. This is wide area covering most problems of real scenes. Temporarily key-point localization is made manually.

Before an analytic triangulation can be utilized, the image should be free of all distortions added by the camera. The problems can be with principal point, aspect ratio, skew and barrel or fisheye distortion. The camera must be calibrated and all distortions must be measured. With this knowledge we can decrease any distortions on the image. Triangulation algorithm is sensitive mainly to

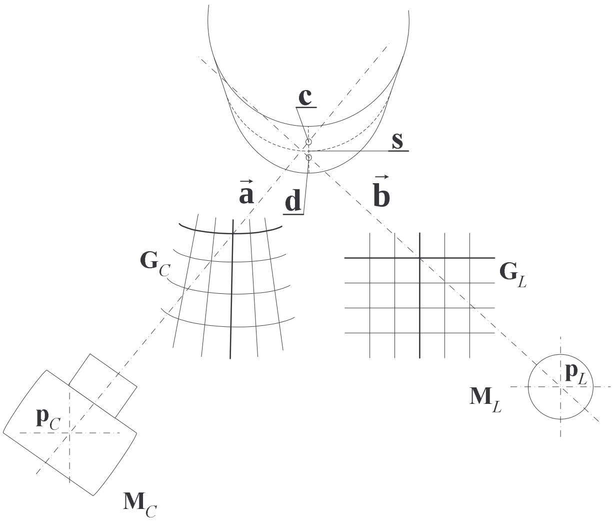


Figure 1: Schematic view of the triangulation process

nonlinear distortions like barrel and fisheye distortions. This kind of distortions is quite common on 35 mm lens of compact digital cameras. After the calibration takes place we can pretend that the image is ideal 2D collineation of its 3D model. Fortunately for many cameras distortions are close to zero, making calibration step unnecessary.

For the implementation of the next step we use our algorithm. More details can be found in (Kapusta [6]). Algorithms solving this area already exist but are not very accessible.

All information from previous step is stored in tensor $\boldsymbol{G}_C$. Additional information is given to create matrices $\mathbf{M}_C$ and $\mathbf{M}_L$. At the end of this chapter is discussed $\mathbf{M}_L$ autodetection.

At the beginning we have two matrices and two tensors. Matrices $\mathbf{M}_L$, $\mathbf{M}_C$ are the projection matrices of light and camera, they incorporate position $\mathbf{p}_C$, direction $\vec{\mathbf{d}}_C$ and field of view (vertical FOV_V and horizontal FOV_H). Matrix $\mathbf{M}_C$ can be obtained as scalar product of the scale + position matrix and rotation matrix:

$$\mathbf{M}_C = \begin{pmatrix} k_h & 0 & 0 & p_{CX} \\ 0 & k_v & 0 & p_{CY} \\ 0 & 0 & 1 & p_{CZ} \\ 0 & 0 & 0 & 1 \end{pmatrix} \bullet \left(1 + \frac{\sin\xi}{\lambda}\mathbf{L} + \frac{1-\cos\xi}{\lambda^2}\mathbf{L}^2 \right) \qquad (1.4)$$

$$\cos\xi = \frac{\vec{\mathbf{z}} \bullet \vec{\mathbf{d}}_C}{\left|\vec{\mathbf{d}}_C\right|}, \ \vec{\mathbf{r}} = \vec{\mathbf{z}} \times \vec{\mathbf{d}}_C \times \mathbf{0}, \ \mathbf{L} = \begin{pmatrix} 0 & r_3 & -r_2 & 0 \\ -r_3 & 0 & r_1 & 0 \\ r_2 & -r_1 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix},$$

$$\lambda = \left|\vec{\mathbf{r}}\right|, \ \vec{\mathbf{z}} = (0\ \ 0\ \ 1\ \ 0)^\top, k_h = \tan\frac{FOV_H}{2}, \ k_h = \tan\frac{FOV_H}{2}$$

where rotation matrix is computed as rotation about an arbitrary axis $\vec{\mathbf{r}}$.

Position of camera and light can be obtained back:

$$\mathbf{p}_C = \mathbf{M}_C \bullet \mathbf{0}, \quad \mathbf{p}_L = \mathbf{M}_L \bullet \mathbf{0}. \qquad (1.5)$$

Tensors $\boldsymbol{G}_L$, $\boldsymbol{G}_C$ are $m \times n$ matrices of 3-vectors. $\boldsymbol{G}_C$ represents 2D positions of $m \times n$ key-points on projective plane of camera. $\boldsymbol{G}_L$ has cognate meaning for light, but values of $\boldsymbol{G}_L$ can be computed by simple formula for regular grid.

Rays from the camera ($\vec{\mathbf{a}}_{ij}$) and the light ($\vec{\mathbf{b}}_{ij}$) can be computed as vectors:

$$\vec{\mathbf{a}}_{ij} = \mathbf{M}_C \bullet \mathbf{H} \bullet \boldsymbol{G}_C(i,j), \quad \vec{\mathbf{b}}_{ij} = \mathbf{M}_L \bullet \mathbf{H} \bullet \boldsymbol{G}_L(i,j)$$

$$i \in 1\ldots m, \ j \in 1\ldots n, \ \mathbf{H} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{pmatrix}, \qquad (1.6)$$

where $\mathbf{H}$ is mapping from $\mathcal{P}^2$ to $\mathcal{P}^3$, setting Z-axis to 1.

Now points $\mathbf{c}$ and $\mathbf{d}$ can be computed. Point $\mathbf{c}$ is lying on line $\mathbf{p}_C + \alpha\vec{\mathbf{a}}$ and is nearest possible to line $\mathbf{p}_L + \alpha\vec{\mathbf{b}}$, conversely point $\mathbf{d}$ is lying nearest to $\mathbf{c}$:

$$\mathbf{c}_{ij} = \mathbf{p}_C + \vec{\mathbf{a}}_{ij}\left(\frac{\left(\vec{\mathbf{b}}_{ij} \times \vec{\mathbf{n}}_{ij} \times \mathbf{0}\right) \bullet \vec{\mathbf{v}}}{\left(\vec{\mathbf{b}}_{ij} \times \vec{\mathbf{n}}_{ij} \times \mathbf{0}\right) \bullet \vec{\mathbf{a}}_{ij}} \right), \qquad (1.7)$$

$$\mathbf{d}_{ij} = \mathbf{p}_L + \vec{\mathbf{b}}_{ij}\left(\frac{\left(\vec{\mathbf{a}}_{ij} \times \vec{\mathbf{n}}_{ij} \times \mathbf{0}\right) \bullet (\text{-}\vec{\mathbf{v}})}{\left(\vec{\mathbf{a}}_{ij} \times \vec{\mathbf{n}}_{ij} \times \mathbf{0}\right) \bullet \vec{\mathbf{b}}_{ij}} \right), \qquad (1.8)$$

where $\vec{\mathbf{n}}_{ij} = \vec{\mathbf{a}}_{ij} \times \vec{\mathbf{b}}_{ij}$ and $\vec{\mathbf{v}} = \mathbf{p}_L - \mathbf{p}_C$. There is again used $\mathbf{0} = \left(0\ 0\ 0\ 1\right)^\top$ to have correct cross product for 4-vectors (cross product have n operands for $(n+1)$-vectors). Result is average between both points $\mathbf{c}_{ij}$ and $\mathbf{d}_{ij}$:

$$\mathbf{s}_{ij} = \frac{\mathbf{c}_{ij} + \mathbf{d}_{ij}}{2} \qquad (1.9)$$

We can little bit optimize formula (1.8) using fact that $\left(\mathbf{a} \times \mathbf{b} \times \mathbf{0}\right) \bullet \mathbf{c} = \text{-}\left(\mathbf{c} \times \mathbf{b} \times \mathbf{0}\right) \bullet \mathbf{a}$:

$$\mathbf{d}_{ij} = \mathbf{p}_L + \vec{\mathbf{b}}_{ij}\left(\frac{\left(\vec{\mathbf{a}}_{ij} \times \vec{\mathbf{n}}_{ij} \times \mathbf{0}\right) \bullet \vec{\mathbf{v}}}{\left(\vec{\mathbf{b}}_{ij} \times \vec{\mathbf{n}}_{ij} \times \mathbf{0}\right) \bullet \vec{\mathbf{a}}_{ij}} \right). \qquad (1.10)$$

It is useful to quantify the error of our reconstruction. Since we do not have digital reference model to compare result with, we must get along with data we have computed. It is evident that for absolutely precise input the equation $\mathbf{c}_{ij} = \mathbf{d}_{ij} = \mathbf{s}_{ij}$ should be valid. Noise, distortions or bad position of light add errors and greater error means greater distance between $\mathbf{c}_{ij}$ and $\mathbf{d}_{ij}$. The error can be evaluated as $\left|\mathbf{c}_{ij} - \mathbf{s}_{ij}\right| = \left|\mathbf{d}_{ij} - \mathbf{s}_{ij}\right|$, or better with square $\left(\mathbf{c}_{ij} - \mathbf{s}_{ij}\right)^2 = \left(\mathbf{d}_{ij} - \mathbf{s}_{ij}\right)^2$. For total error we can write:

$$\sigma = \sum_{j=1}^{n}\sum_{i=1}^{m} \frac{\left(\mathbf{c}_{ij} - \mathbf{s}_{ij}\right)^2}{mn} \qquad (1.11)$$

Automatic placement of the light is based on minimizing total error σ. It is generally difficult problem to be solved exactly. Problem is solved iteratively, but there is always probability to fall into local minimum far from actual light position.

Another way of light placement is to find epipolar geometry of the scene.

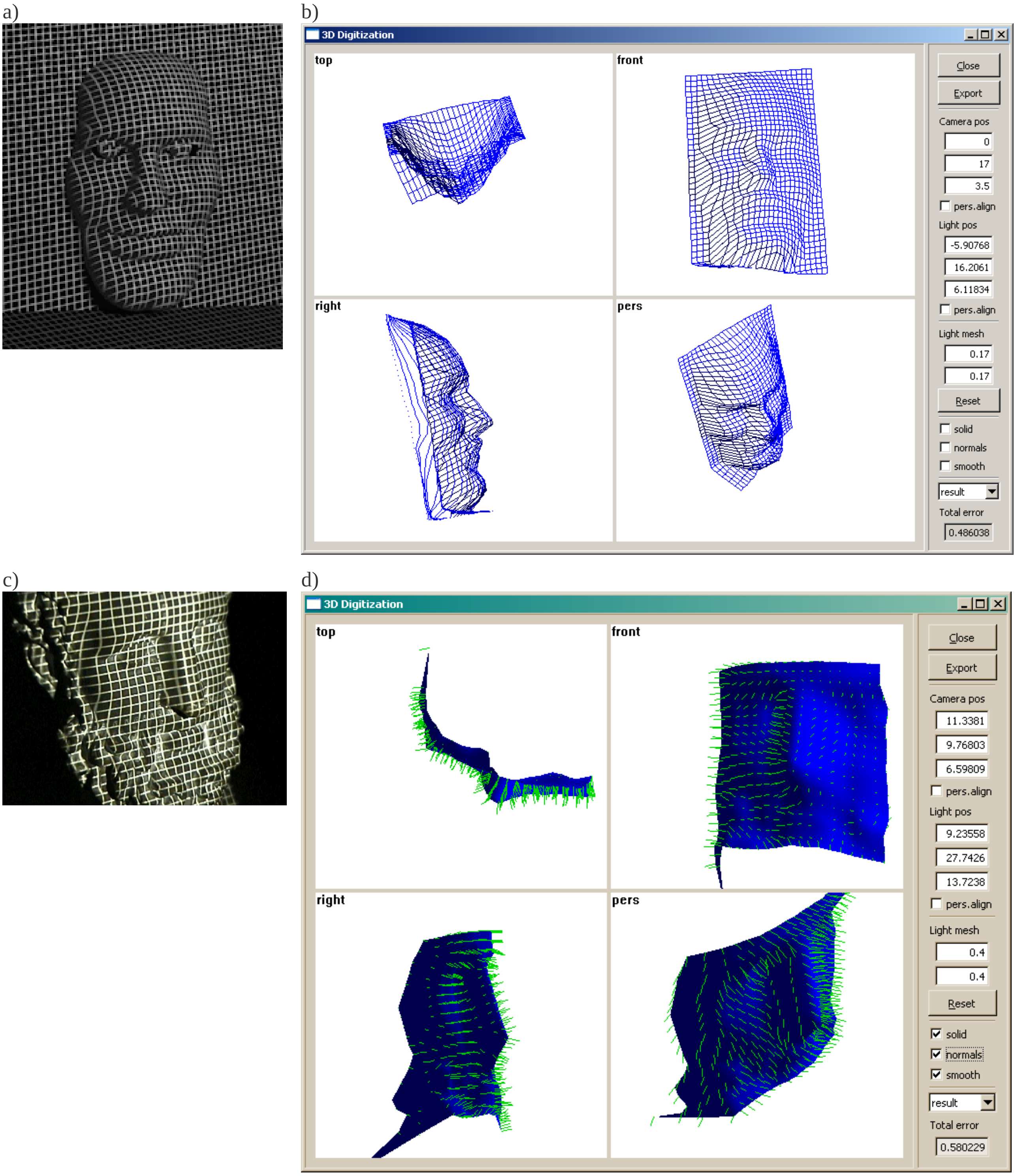
5 Model building

From one shot we get only one depth map. For complete model reconstruction, it is necessary to have more depth maps from various directions or even various distances. Depth maps alone usually do not provide information about the camera position the shot was taken from. The autopositioning of depth maps is very difficult, because a lot of ambiguity and a large amount of possibilities how depth maps can be arranged. At least the approximate positions must be known before the model building can begin. Many of following methods can automatically correct small deviations of the given positions and better align model parts.

The surface reconstruction from range data or depth maps has been active area of research for several decades. The strategies have proceeded along two basic directions: reconstruction from unorganized points and reconstruction that preserves structure in acquired data (Curless [2]).

Methods working on unorganized sets of points are implicit methods (Hoppe [4], Bajaj [1]). Although implicit methods are generally applicable, they do not use convenient information usually gathered during scanning process (such as surface normal and reliability estimates). Methods work well in smooth areas but they are not usually robust in regions with high curvature.

Structure preserving methods can be divided into polygonal methods and implicit methods. Polygonal algorithms (Soucy and Laurendeau [10], Turk and



b)

Camera pos: 0, 17, 3.5
☐ pers.align
Light pos: -5.90768, 16.2061, 6.11834
☐ pers.align
Light mesh: 0.17, 0.17
☐ solid
☐ normals
☐ smooth
result
Total error: 0.486038

d)

Camera pos: 11.3381, 9.76803, 6.59809
☐ pers.align
Light pos: 9.23558, 27.7426, 13.7238
☐ pers.align
Light mesh: 0.4, 0.4
☑ solid
☑ normals
☑ smooth
result
Total error: 0.580229

Figure 2: a) synthetic source image b) result obtained using triangulation from single shot c) photograph of lit sculpture d) result with normals.

Levoy [12], Rutishauser [9]) usually perform better than methods working with unorganized points, but still do not provide good solution in areas of high curvature.

Implicit methods can be further subdivided into discrete-state voxel methods and continuous-valued voxel methods. Methods using continuous-valued voxels usually provide better results. Hilton [3] has developed method using weighted signed distance functions for merging range images. Curless [2] enhanced Hilton's algorithm to cope with sensors uncertainty, incremental updating, hole filling, and made it more space efficient.

6 Results

We made a program for the proving described analytic formulas and our approach. User has the control on all important parameters of the scene which are used to fill input matrices. User can see in real time how the result is changing when he changes parameters or moves the light source or the camera.

Input images can be divided into two groups. First group are synthetic images or virtual photographs (Figure 2a) rendered by the computer. Second group are photographs (Figure 2c) of real objects.

We use 6×6 cm slides to project different grids onto sculpture. Slides were taken from laser printed templates. The objective with high focal length was used to achieve minimal distortion on slides. Final photos are taken using digital camera and slide projector.

Because of some distortion at projected pattern and digital camera, real images have greater error than synthetic. It was confirmed by experimental results (Table 1). Most visually apparent errors reveal on corners of grid, because of small barrel distortion detected on used camera.

Source of image	average total error
synthetic (rendered)	0.6
taken by camera	1.1

Table 1: total error compared on 4 pictures, 2 synthetic and 2 taken by camera.

Direct comparison between original synthetic model and reconstructed counterpart proves that used method gives accurate results when no distortion is present at the pattern and the camera.

7 Conclusions

This method can give satisfying results. Capability to digitize model in real time should be its highest advantage, but automatic grid detection must be finished.

Limitations of this method are quite understandable from the principle how it works. Transparent or reflective surfaces cannot be digitized. Tiny or huge objects cannot be lit by structured light from physical reasons. The object with deep relief and too curved surface cause that projected grid cannot be reconstructed.

The single shot covers only small part of the object surface. Therefore it may be necessary to take more shots. Manual arranging these parts to compose entire object is a time consuming task. Here automatic model building can help. Unfortunately direction of single shots is usually not known and cannot be retrieved easily from common images. This fact is limiting the method only to the specific target areas.

Acknowledgements

I would like to thank my supervisor Martin Šperka for an inspiration and help with this paper.

References

[1] C. Bajaj, F. Bernardini, G. Xu: *Automatic reconstruction of surfaces and scalar fields from 3D scans*, Proc. of SIGGRAPH '95, Los Angeles, ACM Press, pp. 109-118, 1995.

[2] B. L. Curless: *New Methods for Surface Reconstruction from Range Images*, Dissertation, Department of Electrical Engineering, Stanford University, 1997.

[3] A. Hilton, A. Toddart, J. Illingworth, T. Windeatt: *Reliable surface reconstruction from multiple range images*, Fourth European Conferece on Computer Vision, vol. 1, pp. 117-126, 1996.

[4] H. Hoppe, T. DeRose, T. Duchamp, J. McDonald, W. Stuetzle: *Surface reconstruction from unorganized points*, Computer Graphics, Proc. of SIGGRAPH '92, vol. 26, pp. 71-78, 1992.

[5] T. Chia, Z. Chen, C. Yueh: *Curved surface reconstruction using a simple structured light method*, Proc. of the International Conference on Pattern Recognition, Vol. A, pp 844-848, 1996.

[6] R. Kapusta: *Program pre digitalizáciu 3D objektov*, Bachelor degree project, FEI STU, Bratislava, 2002.

[7] M. Pollefeys: *Self-calibration and metric 3D reconstruction from uncalibrated image sequences*, Dissertation, Katholieke Universiteit Lueven, 1999.

[8] M. Proesmans, L. Van Gool, A. Oosterlinck: *One-shot active 3D shape acquisition*, Proc. of the International Conference on Pattern Recognition, vol. C, pp. 336-340, 1996.

[9] M. Rutishauser, M. Sticker, M. Trobina: *Merging range images of arbitrarily shaped objects*, Proceedings IEEE Computer Society on Computer Vision and Pattern Recognition, pp. 573-580, 1994.

[10] M. Soucy, D. Laurendeau: *Multi-resolution surface modeling from multiple range views*, Proceedings IEEE Computer Society Conference on Computer Vision and Pattern Recognition, pp. 348-353, 1992.

[11] W. Triggs: *Géométrie d'images multiples*, Thesis, Institut National Polytechnique de Grenoble, 1999.

[12] G. Turk, M. Levoy: *Zippered polygon meshes from range images*, Proceedings of SIGGRAPH '94, Orlando, ACM Press, pp. 311-318, 1994.

[13] P. Vuylsteke, A. Oosterlinck: *Range Image Acquisition with a Single Binary-Encoded Light Pattern*, IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 12, no. 2, pp. 148-164, 1990.