FAST AND HIGH RESOLUTION 3D FACE SCANNING

Philipp Fechteler, Peter Eisert and Jürgen Rurainsky

Fraunhofer Institute, Heinrich-Hertz Institute
Image Processing Department,
Einsteinufer 37, D-10587 Berlin, Germany
Email: {philipp.fechteler, eisert, rurainsky}@hhi.fraunhofer.de
URL: http://iphome.hhi.de/fechteler

ABSTRACT

In this work, we present a framework to capture 3D models of faces in high resolutions with low computational load. The system captures only two pictures of the face, one illuminated with a colored stripe pattern and one with regular white light. The former is needed for the depth calculation, the latter is used as texture. Having these two images a combination of specialized algorithms is applied to generate a 3D model. The results are shown in different views: simple surface, wire grid respective polygon mesh or textured 3D surface.

Index Terms— Stereo image processing, 3D modelling & synthesis

1. INTRODUCTION

The construction of 3D models out of 2D views on a scene is a field of ongoing research for some decades now. One common way of approaching this problem is the Stereo Vision approach. In this case the information of two or more different views are triangulated to achieve a 3D model. A good overview and evaluation on such algorithms is given in [1].

Very similar to the Stereo Vision approach is the structured light method which is used in this work. Here normally only one camera and one projector is used. There are various different structured light approaches, for example: in [2] a real-time system is proposed which is running on specialized hardware; in [3] a method was presented for generating high resolution depth maps of complex scenes using multiple projectors, cameras and camera snapshots per camera; in [4] a method is shown which uses just one projector and one camera without any modifications running on a typical PC. This last mentioned work is the starting point of this work.

2. OVERVIEW OF THE FRAMEWORK

A 3D model of a face is computed by projecting a simple colored stripe pattern onto the face. The depth information is then calculated by taking into account the distortion of the stripes in the face caused by its shape. To measure the degree of distortion the projected stripes are compared with the detected stripes in order to find corresponding stripes respective to find corresponding pixels per vertical scan line.The depth is evaluated for all correspondences with respect to the focal points of the camera. After having a cloud of 3D points it is converted into a mesh of triangles. This mesh constitutes the surface of the 3D model. In the last step a picture taken with regular white light is put onto this surface as texture information.

3. THE ARCHITECTURE

The hardware used by our framework consists of regular devices: a camera and a projector (see figure 1). In our experiments we used a DLP projector ”Projection Design F1+” with a resolution of 1400 x 1050 pixels and a camera ”Canon EOS 20D” with a resolution of 8.2 megapixel. Both devices are controlled by a typical PC running the framework. The devices are mounted so that their image centers are one upon the other.

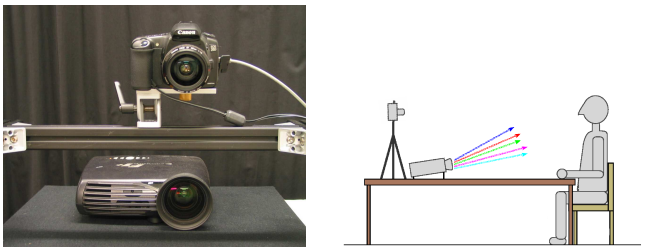

Fig. 1. The devices and setting used in this framework

In order to generate a 3D model of a face the system does the following:

1. take pictures $P_{colored}$ of the face illuminated with a color stripe pattern and P_{clean} with clean/white light
2. extract stripes and corresponding colors in $P_{colored}$
3. match the detected stripes with the projected ones
4. calculate the 3D coordinates of the correspondences

5. create a triangle mesh from the 3D point cloud

6. project P_{clean} onto the surface as texture

4. OFFLINE CREATION OF PATTERN

The pattern projected onto faces should allow an easy assignment of imaged parts to parts of the pattern. Therefore we chose a stripe pattern with horizontal lines having fully saturated colors. This reduces the search for corresponding pixels to a 1D search along the corresponding scan lines. The colors in the resultant pattern image $P_{pattern}$ are (see figure 2): red, green, blue, white, cyan, magenta and yellow. Because we have only 7 of such colors they will be repeated rather often. To ease the unique assignment of detected stripes to projected ones we chose a series of stripe colors with a big period. This is achieved by using de Bruijn sequences [5].

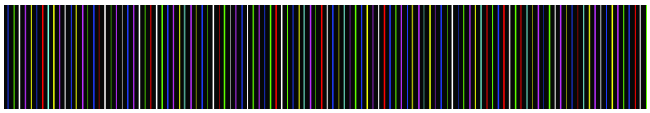

Fig. 2. A cut-out of the used pattern rotated by $90°$

5. ONLINE MODEL CREATION

In this section we present the framework for the model creation with all its methods.

5.1. Extraction of colored stripes

The extraction of colored stripes out of $P_{colored}$ can be subdivided into four parts: broad region selection, image preprocessing, detection of stripe centers and assignment of colors.

5.1.1. Broad region selection

The selection of a region is simple. The region of interest is defined by using a face outlined shape model. All the pixels outside will be set to black. In this way the search space of all the following steps is reduced significantly.

5.1.2. Image Preprocessing

The properties of $P_{colored}$ are very important for the analysis and 3D model reconstruction. An important role plays the contrast of the colored stripes to the facial skin. A higher contrast leads to better reconstruction results. In order to allow brighter ambient light during the capturing a few preprocessing steps are used. A sharpening filter applied vertically reduces the influence from the skin to the colored stripes. Parallel an edge detection based on quantization extracts the available lines. The results of this preprocessing step are shown in figure 4.

5.1.3. Detection of stripe centers

To ease the comparison of stripes they are reduced to their one pixel wide centers. For every scan line in $P_{colored}$ a list of the centers of stripes is collected in the following way: choose an interval around a local maximum brightness about the size of a stripe plus surrounding, fit a parabola to the distribution of brightness in the least mean squares sense and take the extrema of this parabola as center. With this procedure the location of the stripe centers are detected with sub-pixel resolution.

5.1.4. Extraction of color

The next step is to assign every stripe the color it was radiated with. Our experiments have shown that this is not a trivial task. In figure 3 the intensity distribution is shown of the red, green and blue channel in $P_{colored}$ along one scan line. Big distortions are introduced because the skin has its own behaviour of reflecting colored light. Other sources of trouble are the ambient light and distortions by the lenses of the projector and camera.

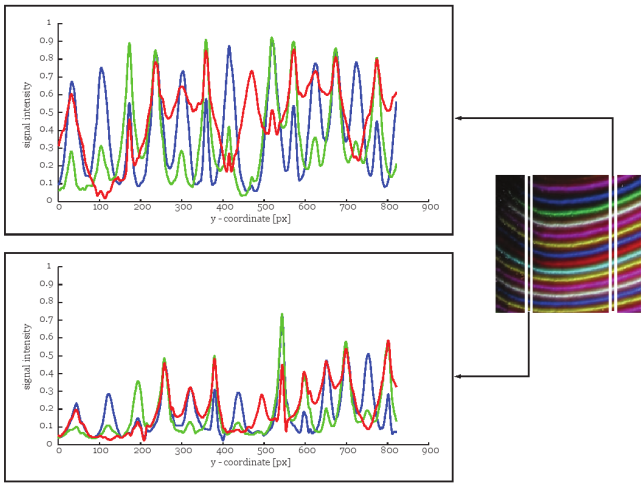

Fig. 3. Profiles of color channels along scan lines in $P_{colored}$

A simple strategy was developed to address this problem. To estimate the color of a stripe pixel the ratios between the channels are examined: $\frac{r}{g}$, $\frac{r}{b}$, $\frac{g}{r}$, $\frac{g}{b}$, $\frac{b}{r}$, $\frac{b}{g}$. In this way the choice of color assignment is independent of brightness. The ratios will be sorted in a list according to their values. If all this ratios have a value around one, then the color white is chosen. Otherwise, the red channel of the resultant color will be set on, if $\frac{r}{g}$ or $\frac{r}{b}$ is among the two biggest ratios in the sorted list. The same rule holds for the green and blue channel.

After this step for every scan line in $P_{colored}$ a list exists which contains the coordinates and estimated color for every detected stripe.

5.2. Matching the stripes

To calculate the depth of a stripe in $P_{colored}$ the corresponding stripe in P_{clean} has to be found. A point in $P_{colored}$ can only correspond to a point in a line (the epipolar line) in P_{clean}. Because of the rectified setting this line corresponds directly to the corresponding scan line. In order to keep the task simple we solve each scan line assignment problem independently of the others.

This type of task can be formulated as a typical combinatorial optimization problem (COP): Find a combination of stripe assignments that fits best. A common way to solve a COP is performed by setting up a cost function which represents the quality of the whole assignment of all stripes of one scan line.

The cost we define for each stripe assignment contains two terms:

- $\texttt{colorDiff}(i)$ returning the difference between the color channels of the ith stripe center of the current scan line in $P_{colored}$ and its assignment in $P_{pattern}$
- $\texttt{jumpWeight}(i, i-1)$ returning a penalty value for gaps in the sequence of the assigned stripes in $P_{pattern}$

The full cost function C is just the sum of these two terms over all correspondences of one scan line:

$$C = \texttt{colorDiff}(0) + \sum_{i=1}^{m} \texttt{colorDiff}(i) + \texttt{jumpWeight}(i, i-1)$$

with m being the number of detected stripes in the particular scan line.

To solve this COP we use Dynamic Programming (DP) [6]. DP is a method to solve problems by subdividing problems into smaller subproblems, solving them, and combining them to the optimal overall solution. It is used very often in Stereo Vision, because of its low computational complexity. A drawback of DP is that it just solves the assignments per scan line independent of the other scan lines. There are many approaches to insert 2D information in this search, e.g. [7]. Our experiments have shown that in our case the results are sufficient with the simple 1D search (see figure 4). But our investigations concerning the introduction of 2D information into DP are still going on.

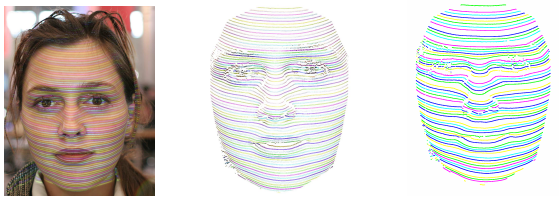

Fig. 4. Results of image preprocessing and DP

Evaluating the properties of the reconstructed model shows sometimes reconstructed points, which do not really fit to the surrounding points. Therefore, an analysis step for the output of the DP result is used, which examines the line colors in the sense of similarity and corrects such misassignments.

5.3. Calculation of depth information

Next, the depth of every point correspondence will be evaluated by triangulating the point cloud. For this purpose, the projection matrix of the camera and projector are needed. They are initially defined via camera calibration [8]. The distance between the focal points of the camera and projector as well as the angles between their associated coordinate systems are the so called extrinsic parameters. Together with the intrinsic parameters of the camera (focal length, pixel width and height and lens shift) the transformation and projection matrix can be set up which maps 3D points in world coordinates to 2D picture coordinates.

The depth calculation is done by calculating the intersection of the two lines of sight through the focal points and the image points of the camera respective projector (see figure 5). For this purpose we used the method proposed in [9].

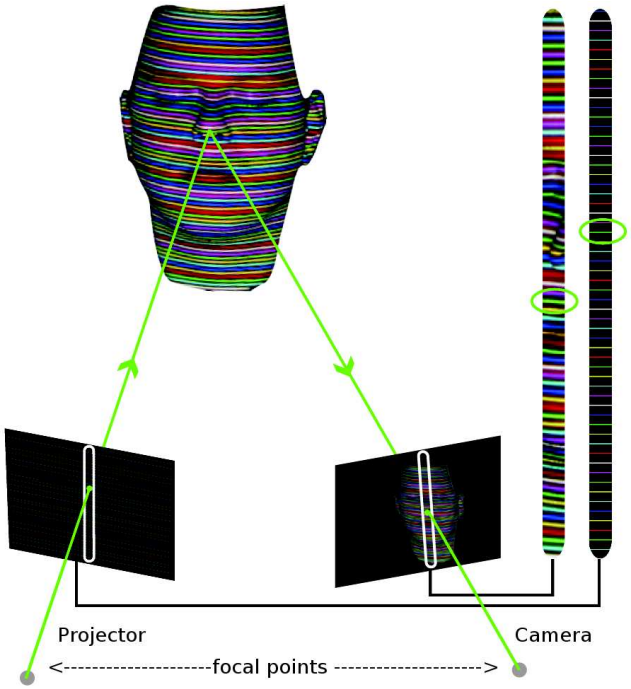

Fig. 5. Visualization of projected point

5.4. Calculation of triangle mesh model and textured surface

The general task of computing a triangle mesh from a 3D point cloud to build a polygon mesh can be simplified here, since we can exploit existing ordering constraints of the regular pixel grid: Points of one stripe have the same y coordinate

in $P_{pattern}$ and their horizontal distances are uniformly distributed due to the column-wise correspondence analysis.

Care has to be taken in the order the points are fused to triangles so that all their normal vectors point into the same direction (see figure 6)

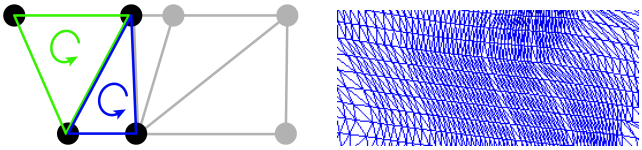

Fig. 6. Creation of wire frame model

In order to eliminate border errors as well as errors, which can occur at light absorption areas (hair and eyes) a triangle area filter is applied. This filter evaluates the triangle areas and removes triangles, which does not fit to the defined specifications, e.g. size.

The covering of the 3D wire grid with the texture is quite easy. The $P_{colored}$ coordinates of the points are interpreted in P_{clean}. From there the texture information is taken.

6. EXPERIMENTAL RESULTS

Many experiments have been performed with the 3D face scanner. Figure 7 shows the results of a typical scenario. Two pictures are taken, one with regular white light and one with the structured light pattern shown in figure 2. After selecting the interesting region, the system calculates the 3D model and optionally presents the wire frame model, the surface or the textured 3D face. Note that the system performs well even in non-labor environments, as it is visible in figure 4, taken during the CeBIT 2006 in Hanover, Germany.

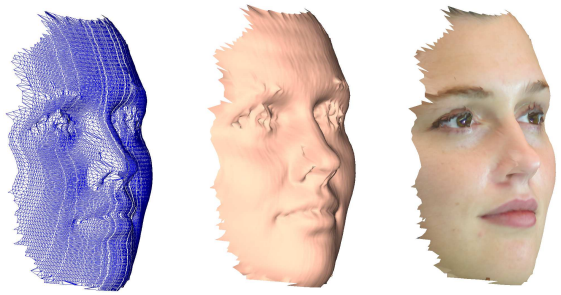

Fig. 7. Resultant 3D models as wire frame model, surface and textured surface

On a typical PC with a 3 GHz Pentium 4 processor the process takes up to a minute.

7. CONCLUSION AND FUTURE WORK

We have presented a system for high resolution 3D face scanning with low computational complexity. Experiments have shown that the system accuracy still remains acceptable even under perturbing ambient light conditions.

The simple setup and its easy usage make the presented system ideal suited for various 3D model creation scenarios, e.g. virtual environments like 3D games or human machine interfaces.

Improvements of the framework could be an enhancement of the color estimation, presented in section 5.1.4, or the stripe matching algorithm in section 5.2, e.g. by the introduction of 2D knowledge in the DP algorithm.

8. ACKNOWLEDGEMENT

The work presented in this paper has been developed with the support of the European Network of Excellence VISNET II (Contract IST-1-038398).

9. REFERENCES

[1] D. Scharstein and R. Szeliski, "A taxonomy and evaluation of dense two-frame stereo correspondence algorithms," *IJCV 47(1/2/3)*, pp. 7–42, April-June 2002.

[2] Song Zhang and Peisen Huang, "High-resolution, real-time 3D shape acquisition," in *Conference on Computer Vision and Pattern Recognition Workshop (CVPRW'04) Volume 3*, June 2004, p. 28, Washington, DC.

[3] D. Scharstein and R. Szeliski, "High-accuracy stereo depth maps using structured light," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2003)*, June 2003, vol. 1, pp. 195–202, Madison, WI.

[4] Li Zhang, Brian Curless, and Steven M. Seitz, "Rapid shape acquisition using color structured light and multi-pass dynamic programming," in *The 1st IEEE International Symposium on 3D Data Processing, Visualization, and Transmission*, June 2002, pp. 24–36, Padova, Italy.

[5] H. Fredricksen, "The lexicographically least debruijn cycle," *Journal of Combinatorial Theory*, vol. 9, pp. 509–510, 1970.

[6] David B. Wagner, "Dynamic programming," *The Mathematica Journal*, 1995.

[7] Mikhail Mozerov, "An effective stereo matching algorithm with optimal path cost aggregation," in *DAGM-Symposium*, September 2006, pp. 617–626, Berlin, Germany.

[8] Peter Eisert, "Model-based camera calibration using analysis by synthesis techniques," in *Proc. 7th International Workshop MODELING, AND VISUALIZATION 2002*, November 2002, pp. 307–314, Erlangen, Germany.

[9] Ron Goldman, *Intersection of Two Lines in Three Space*, p. 304, Academic Press, 1990.