Robust Segmentation and Decoding of a Grid Pattern for Structured Light

Jordi Pagès, Joaquim Salvi, and Carles Matabosch

University of Girona, Institute of Informatics and Applications
av. Lluís Santaló s/n, 17071 Girona, Spain

Abstract. This paper describes the implementation details of a coded structured light system useful for one-shot measurements of a surface. Since a unique pattern is projected, the technique is useful for measuring moving surfaces. A pattern based on grid structure is used. The main advantage of such structure is that redundant codification is applied to the cross-points. Since both pattern axis are coded, decoding errors can be corrected thanks to the proposed algorithm. Moreover, not only the cross-points of the grid can be reconstructed but also the pixels belonging to vertical and horizontal slits. A description of the segmentation and decoding stage is given in order to take profit of the advantages of the pattern codification.

1 Introduction

Structured light systems appeared in order to ease the correspondence problem of stereovision systems. Such techniques are based on replacing one of the cameras by a light source. Then, projecting a set of known patterns onto the measuring scene and grabbing images with the remaining camera(s), the correspondence problem is solved by all those points where the patterns have been projected on. The former systems projected patterns consisting of simple geometric primitives like points or lines. Later, more complex patterns were developed, including some kind of codification in order to distinguish different parts of the pattern and increasing the number of correspondences in every projection. Such approach has been known as coded structured light.

A large number of 3D points are recovered if multiple patterns can be projected. However, when measuring dynamic surfaces, techniques based on a unique pattern must be used. Nevertheless, the resolution decreases since the codification must be condensed in a single pattern [1].

The most used coding strategy to generate such patterns is based on spatial neighborhood codification. This approach consists of identifying a set of points of the pattern with the information contained in a small neighborhood nearby. Then, a set of well distributed pixels on the image can be reconstructed with a single projection.

This paper presents the implementation of a low-cost coded structured light technique based on spatial neighborhood codification. The technique permits

F.J. Perales et al. (Eds.): IbPRIA 2003, LNCS 2652, pp. 689–697, 2003.

to obtain the shape of an unknown surface even if it is moving. Besides, since the proposed pattern has a grid structure, both row and column codification is included into the crossing points. With such redundancy some interesting features are obtained: high accuracy in the reconstruction of the cross points, possibility to reconstruct both vertical and horizontal slits, and error detection and correction when decoding the cross points.

The main drawback of most part of techniques based on a unique pattern is their inability to reconstruct surfaces containing discontinuities correctly. In this paper it is demonstrated that encoding both pattern axis such limitation can be eliminated.

The paper is structured as follows: first, the codification principle of the pattern is detailed in section 2. The image processing involved in the segmentation of the pattern and the proposed algorithm for robust decodification are presented in section 3. Afterwards, in section 4 some examples of surface reconstruction using the implemented technique are shown. The paper ends with conclusions.

2 Pattern Design

The proposed coded structured light has been designed in order to use low-cost devices such a LCD projector to project the pattern, a color video camera, a frame-grabber and a standard PC.

The codification principle of the pattern used for the current implementation was proposed by Salvi et al. [2]. The structure of the projected pattern is defined as a grid composed of vertical and horizontal colored slits of a certain thickness over black background. In order to choose the sequence of colors that is assigned to the horizontal and vertical slits, a De Bruijn sequence is used. De Bruijn sequences have been used by several authors with the aim of defining patterns without periodicity. Concretely, the most exploited patterns coded with De Bruijn sequences consists of parallel slits, i.e. the one proposed by Monks et al. [3].

A De Bruijn sequence of order m over an alphabet of n symbols is a circular string of length n^m that contains each substring of length m exactly once. Such characteristic is called window property. For example, given the De Bruijn sequence of eq. 1, if every element of the alphabet $\{0, 1, 2\}$ is mapped to a certain color, a total of $3^3 = 27$ parallel slits of a pattern can be colored by mapping such sequence. The given sequence has length 29 due to its circular property.

$$22021020012011010002212111222 \tag{1}$$

Since the given De Bruijn sequence has a window property of 3, every three consecutive slits in the pattern will be uniquely identified by the codeword formed by their three colors. Another author who took profit of this coding strategy was Zhang et al. [4] proposing a coded pattern containing 125 vertical slits with a De Bruijn sequence of order 3 and 5 colors.

Other pattern structures coded with De Bruijn sequences have been proposed. For example, Griffin et al. used an array of colored dots such that every

window of 3×3 dots in the pattern was unique [5]. The dot representation requires to locate the mass centers of all the imaged spots in order to triangulate them. Nevertheless, all those spots that appear partially occluded in the camera image must be discarded since their mass center might not be well segmented.

The work by Salvi et al. [2] proposed to design a grid pattern instead of a single sequence of parallel slits or a dot array. By selecting the colors of both vertical and horizontal slits with the same De Bruijn sequence, both axis are coded. The set of colors used to encode vertical and horizontal slits is different in order to differentiate both kind of slits. All the pixels belonging to a vertical or horizontal slit have a codeword which indicates the position of the slit in the pattern. All the imaged pixels belonging to the slits can be reconstructed by intersecting the equation of the camera 3D line which contains the image pixel and the equation of the 3D plain corresponding to the slit. Moreover, the cross points of slits in the grid have two codewords. Therefore, cross points can be reconstructed more accurately by intersecting the equations of two 3D lines. Another advantage of this pattern structure is that redundancy in the coding is included since two codewords are defined for every cross point. This fact permits to detect and even correct errors in the decoding stage of the imaged pattern.

The colors used for the horizontal slits are red, green and blue, while yellow, cyan and magenta are used for the vertical slits. The resulting pattern can be observed in figure 1a. Lavoie et al., a year later, proposed a similar pattern [6].

3 Pattern Segmentation and Decodification

Once the pattern is projected onto the measuring surface, an image must be grabbed with the camera. Then, for all those cross points of the grid that can be identified and decoded, the correspondence problem between camera and projector can be solved and, therefore, the corresponding 3D points can be triangulated.

In order to segment the grid in the image, a stage of image processing must be fulfilled. Then, the decoding stage must obtain the codewords of every visible cross point of the grid. This stage must be robust against errors since some parts of the projected grid can be occluded from the camera point of view.

The implemented algorithm has been structured in the following steps: first the segmentation of the grid, then, the cross points detection, and finally, the decodification of the detected cross points. Hereafter, all three steps are detailed.

3.1 Segmentation of the Grid

For correctly segment the projected grid it is necessary to clearly distinguish the 6 primary colors used. In order to success, a color calibration procedure is made only once, when installing the system in the working area, by projecting the grid onto a color-neutral surface and calibrating the gains for every projected color. The camera iris is also adjusted to perceive basically only the projected grid.

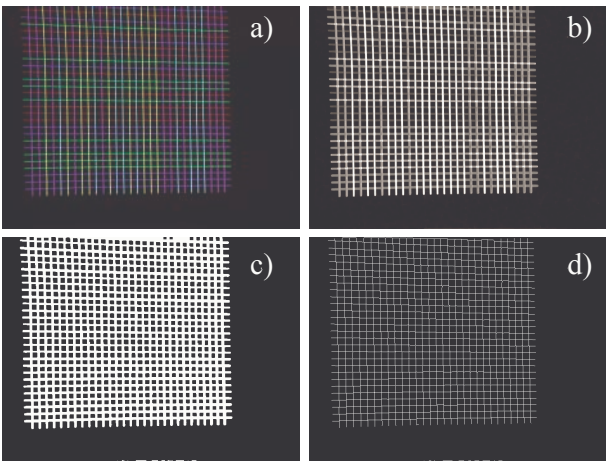

Fig. 1. Pattern segmentation. a) original 24-bit. b) image after edge detection, conversion to 8-bit and 3 Close iterations. c) binarization. d) thinning until skeleton

The first step of the segmentation algorithm consists of applying a Sobel operator to the camera image in order to detect the edges of the projected grid. The resulting image is converted to 8-bit greyscale. Afterwards, three Close morphological operators are applied in order to merge the parallel edges that appear using Sobel filter. Then, the obtained image is binarized with a low threshold in order to get thick slits. Finally, a thinning algorithm must be applied until the skeleton of the image is obtained. The sequence of operations can be observed in figure 1.

3.2 Cross Points Detection

Once the camera image has been processed in order to enhance and to segment the grid skeleton, the cross points between horizontal and vertical slits can be located. The set of masks shown in figure 2 are convolved with the image containing the skeleton of the grid.

$$\begin{pmatrix} 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \end{pmatrix} \begin{pmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \end{pmatrix} \begin{pmatrix} 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \end{pmatrix} \begin{pmatrix} 0 & 0 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \end{pmatrix} \begin{pmatrix} 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \end{pmatrix} \cdots$$

Fig. 2. Samples of the binary masks used to detect grid cross points

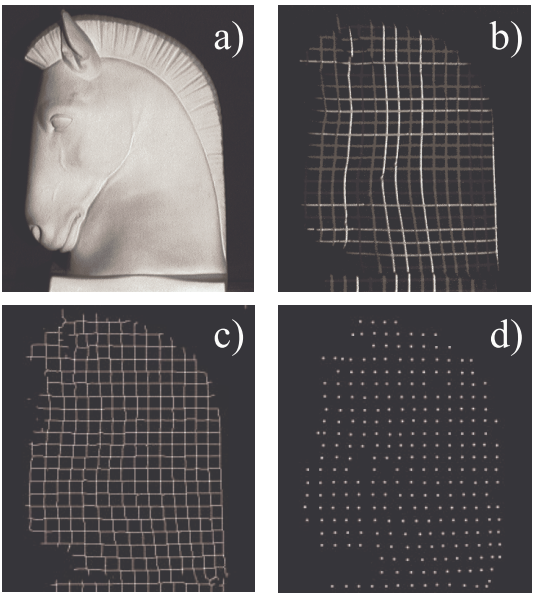

Fig. 3. Cross points detection. a) A horse statue. b) the pattern projected. c) the extracted skeleton of the grid. d) the detected cross points

For every position of the image where the convolution result is greater than 6 for any of the masks, the pixel is considered to be a cross point of the grid. All these masks must be used since the thinning operation to obtain the grid skeleton not always leads to perfect intersections.

An example of the cross-point detection process can be seen in figure 3, where the pattern has been projected onto a horse statue.

3.3 Decoding the Detected Cross Points

The projected grid is colored so that every cross point has two codewords. The first codeword is formed by the colors of the vertical slit containing the cross point and the colors of both adjacent vertical slits. The second is generated in the same way but with the horizontal slits. Both codewords are unique for every cross point (window property), so there is a direct mapping from both codewords to the cross point coordinates in the pattern.

In the previous subsection the process to detect cross points in the camera image has been explained. In order to decode the maximum number of cross points the following algorithm is applied:

- Generation of the cross points adjacency graph
- Obtaining the horizontal and vertical slits colors corresponding to every cross point

- Decoding all those cross-points whose both codewords have been found by using the adjacency graph
- Transfer of codewords to neighbors not decoded
- Correction of inconsistent codewords

To generate the adjacency graph is necessary to start from the coordinates of every cross point and track the edges of the grid skeleton towards four directions in order to find the neighbors. Since the skeleton has thickness of 1 pixel, windows of dimensions 1×3 and 3×1 are enough for tracking vertical and horizontal edges respectively.

The colors of the slits intersecting in every cross point are recovered during the tracking process. The mean Hue of every tracked edge is used to identify the original color projected. Let define the neighborhood of a cross-point as the grouping of its two codewords. Therefore, the neighborhood of a cross-point whose position in the grid is the y row the x column is defined as a vector like

$$neighborhood(crosspoint(y,x)) = [c_x \quad r_y \quad c_{x-1} \quad r_{y-1} \quad c_{x+1} \quad r_{y+1}] \tag{2}$$

Where c_i and r_j are the colors of column number i and row number j respectively. The graphic interpretation of the neighborhood of a cross-point can be seen in figure 4. Note that thanks to the De Bruijn codification, the neighborhood of a cross-point uniquely identifies its position in the projected grid.

Once the adjacency graph is constructed, the neighborhoods of all the detected cross-points in the camera image can be obtained. Firstly, only the cross-points whose complete neighborhood has been recovered are decoded. Otherwise, the cross point remains undecoded. The decodification of a detected cross-point consists of searching in the original pattern in which coordinates there is a cross-point with the same neighborhood. Then, the correspondence between both camera pixel and projector pixel is solved.

The following step tries to decode such cross points not decoded in the previous step by choosing the proposed pattern coordinates most voted by all the available decoded neighbors. For example, the cross-point occupying the x column and the y row in fig. 4, will propose the following coordinates to its four

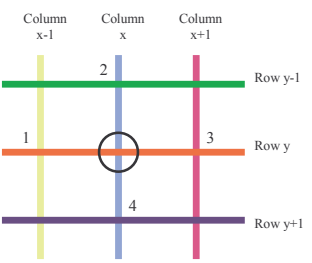

Fig. 4. Neighborhood of a cross-point

Fig. 5. Side view of the horse reconstruction

neighbors 1, 2, 3 and 4 respectively: $(x-1, y)$, $(x, y-1)$, $(x+1, y)$, $(x, y+1)$. The transfer step is repeated until no changes occur.

The final step of the algorithm consists of comparing the decoded pattern coordinates of every cross points with the coordinates that its neighbors would propose. Then, if the decoded pattern coordinates of a cross point do not coincide with the most voted coordinates proposed by its neighbors, such most voted coordinates are accepted as the correct ones. The last step of the algorithm is also repeated until no changes in the decoded coordinates occur.

4 Experimental Results

The hardware setup used for the experiments consists of a Mitsubishi XL1 video projector, a Sony 3 CCD camera, a Matrox Meteor-II frame grabber and a standard PC. The resolution of the projected pattern is 1024×768 pixels, while the camera images are digitized at 768×576 pixels, width a depth of 24 bits per pixel.

The rendered surface corresponding to the 3D reconstruction of the horse statue from figure 3a can be observed in figure 5 from different points of view.

As a second example, a human hand has been reconstructed to demonstrate the robustness of our technique against surfaces containing discontinuities. The cloud of 3D points obtained after applying the technique is presented in figure 6. The fingers of the hand are difficult to reconstruct since their small surface does not permit to contain large neighborhoods of cross-points, so none can be decoded directly. However, thanks to the transfer and correction steps of the designed decoding stage such constraint is removed.

5 Conclusions

The implementation of a coded structured light technique based on the projection of a unique grid pattern has been detailed. The paper has focused on the design of a robust decoding stage.

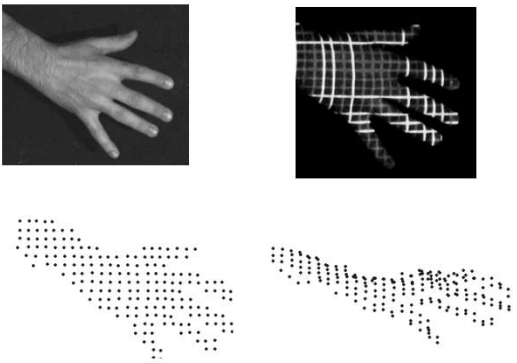

Fig. 6. Human hand reconstruction

The grid structure of the used pattern improves other pattern representations proposed in the bibliography. The different set of colors used for vertical and horizontal slits permit to be distinguished easily. Since both axis of the pattern are coded, cross points of the grid are redundantly coded, leading to a more accurate triangulation. Redundancy also permits to detect and correct errors in the cross points decodification.

The resolution of the technique can be increased by enlarging the number of slits and, therefore, the length of the De Bruijn sequence used to encode the grid. Moreover, since the pixels belonging to the horizontal and vertical slits are also coded with respect to a single axis, they can also be reconstructed producing a denser surface.

References

[1] J. Batlle, E. Mouaddib, J. Salvi, Recent progress in coded structured light as a technique to solve the correspondence problem: a survey, Pattern Recognition 31 (7) (1998) 963–982. 689

[2] J. Salvi, J. Batlle, E. Mouaddib, A robust-coded pattern projection for dynamic 3d scene measurement, Pattern Recognition Letters (19) (1998) 1055–1065. 690, 691

[3] T. P. Monks, J. N. Carter, C. H. Shadle, Colour-encoded structured light for digitisation of real-time 3D data, in: International Conference on Image Processing, 1992, pp. 327–30. 690

[4] L. Zhang, B. Curless, S. M. Seitz, Rapid shape acquisition using color structured light and multi-pass dynamic programming, in: Int. Symposium on 3D Data Processing Visualization and Transmission, Padova, Italy, 2002. 690

[5] P. Griffin, L. Narasimhan, S. Yee, Generation of uniquely encoded light patterns for range data acquisition, Pattern Recognition 25 (6) (1992) 609–616. 691

[6] P. Lavoie, D. Ionescu, E. Petriu, A high precision 3D object reconstruction method using a color coded grid and nurbs, in: Proceedings of the International

Conference on Image Analysis and Processing, Venice, Italy, 1999, pp. 370–375. 691