ZÁPADOČESKÁ
UNIVERZITA
V PLZNI

University of West Bohemia in Pilsen
Department of Computer Science and Engineering
Univerzitni 8
30614 Pilsen
Czech Republic

Hologram Synthesis by use of Patterns

Technical Report

Martin Janda, Ivo Hanák, Levent Onural (Bilkent University)

Technical Report No. DCSE/TR-2006-09
June, 2006

Distribution: confidental

Technical Report No. DCSE/TR-2006-09
June 2006

Hologram Synthesis by use of Patterns

Martin Janda, Ivo Hanák, Levent Onural (Bilkent University)

Abstract

The report describes a simple method for synthesising a hologram of line segments. The method is based on the diffraction pattern splatting. Optical verification of the results is included.

This work has been partialy supported by the Ministry of Education, Youth and Sports of the Czech Republic under the research program LC-06008 (Center for Computer Graphics). This work has been partialy supported by the EU project EU within FP6 under Grant 511568 with the acronym 3DTV.

Copies of this report are available on
`http://www.kiv.zcu.cz/publications/`
or by surface mail on request sent to the following address:

University of West Bohemia in Pilsen
Department of Computer Science and Engineering
Univerzitni 8
30614 Pilsen
Czech Republic

Contents

Chapter 1

Method's Description

The method proposed by Levent Onural is based on the observation of diffraction patterns generated by in infante vertical line parallel to a hologram plane. If the line is rotated α degrees around Z-axis, which is the one pointing from a hologram plane towards a scene, the diffraction pattern remains the same and it is also rotated by α degrees.

If the line is moved along Z-axis the diffraction pattern changes in a way similar to the stretching in X-axis direction if the line is oriented along Y-axis. Based on this observation, a scene consisting of infinitely long vertical lines can be computed by splatting a rotated and stretched pattern of a single vertical line.

If the line is not infinite but finite with endpoints covered by the holograms frame, the situation becomes a little bit more complicated. However, even this case can be done via the previously mentioned splatting of prepared patterns. The only problematic areas are the endpoints.

Endpoints of a line segment generate a patten that resembles the one of a single point radiator; the remaining part of the segment generates a pattern similar to the one of an infante line described above. The important observation is that the endpoints pattern have very low contrast in comparison to the rest of the pattern. This observation makes possible to employ an approximation of a line segments pattern with the one of an infinite line.

Unfortunately, the approximation is a cause of blur near the line's endpoints. It also places a restriction on a minimal line segment's length because an intensity of the segment's end points and intensity of the inner segment are comparable for a short enough line segment. A reconstruction of such short line segment's fringe pattern then produces too blurred image.

Based on assumptions and approximations mentioned above, it is possible to synthesise a hologram of a smooth curve by approximating it as a

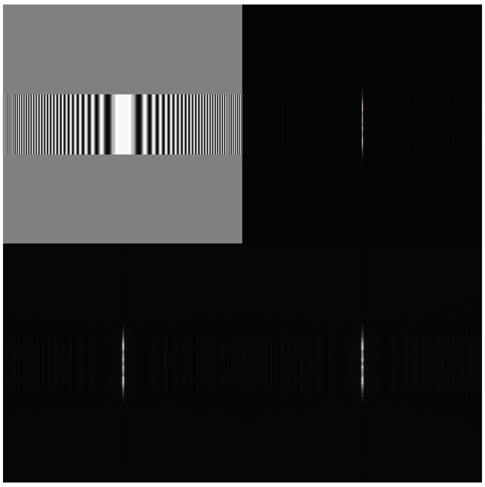

Figure 1.1: Line fringe pattern and its reconstruction

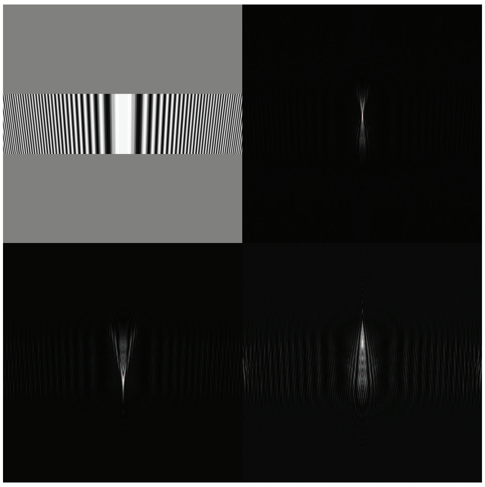

Figure 1.2: Slanted line fringe pattern and its reconstruction

set of line segments and accumulating fringe pattern, rotated and stretched accordingly, of each line segment. The method should work well for loose and smooth curves. Sharp corners are assumed to be blurred too much and therefore are marked as unwanted. It is also assumed that this method is faster then a brute force approach thanks to a lack of complicated operation such as evaluating sinus function, cosine function, and square root function. It may also benefit from a computational power of current graphics hardware.

Chapter 2

Method's Implementation

The method was implemented to verify the assumptions noted in the Section 1. Because only technologically old graphic hardware was available for testing, the method was implemented in an inefficient way. With some more advanced hardware shorter times could be achieved.

The basis fringe pattern is computed as an optical field of an infinite vertical line positioned at a given distance from a hologram. Height of the field is one sample. The computed optical field is stored in a texture that contains both real and imaginary components obtained from the expression $\exp(-ikr)$, where r is Euclidean distance of a sample to a vertical line. The amplitude modificator is omitted due to a limited number precision that we had available. In a case of a proper implementation the modificator can be stored separately and applied later in form of an approximation $1/z$. The texture is computed only once and can be stored and used for later renderings.

The basis fringe is then applied to a polygonal patch that is tapered, rotated, scaled, and translated to a proper form according to a line segment. The polygonal patch is a quad, a square, centred around the origin. While majority of transformations are straightforward, the tapering transformation is more complicated. The only approach that can create required transformation is a perspective projection because it enables well known perspective correct texture mapping. Simple scaling of quad's vertices forms a rather disturbing effect over the diagonal edge that prevents generation of proper results.

In order to keep the solution simple, the implementation utilises vertex shader to apply tapering transformation. Input coefficients for this operation, i.e. edge scaling along local X-axis, are computed as a ratio of desired instance r and distance r_b utilised during computation of the basis fringe. In an ideal case this ratio shall be computed for each basis fringe sample, but in our approximation we have computed it just for sample on a side of the basis fringe pattern. The computation of the coefficient is based on an assumption that all distances are floored to nearest wavelength multiple. Thus, a phase can be expressed as $\exp\left(-ikr\right) = \exp\left[-i2\pi n - ik\Delta_z\right]$ where $n \in \mathbb{Z}$ and hence $-i2\pi n$ can be omitted from the expression because it has no effect on phase itself.

This leaves $k\delta_z$ as an expression that defines the phase, where $r = z + \Delta_z$. In order to obtain a valid coefficient a modification is required. This modification creates an equality between Δ_z and Δ_{zb} where zb is the distance used for computing the basis fringes. Let us assume that the approximation $r \approx z + x^2/\left(2r\right)$ is valid. Then, $x^2 = z$ resembles Δ_z expression and thus the expression Δ_{zb}/Δ_z defines a coefficient that has to modify the x^2 component of the square root approximation. Hence, the final expression for tapering transformation coefficients:

$$c = \sqrt{\frac{\Delta_{zb}}{\Delta z}}. \tag{2.1}$$

As an input a list of points is required. Each pair of succussing points determines the line segment. Each line segment requires the polygonal patch to be transformed into a proper position. The patch is then drawn and the resulting image is retrieved from the GPU. Values stored in the retrieved image are then accumulated into a buffer.

Once all segments are processed the accumulation buffer contains the final hologram. The hologram can be saved on a disk or reconstructed. Because the method was implemented as a module of the MVE2 system the already existing modules for a numerical reconstruction can be immediately used.

Chapter 3

Results

The first implementation of the method gave us results similar to the expected ones. For planar curves approximated by a reasonable number of lines it creates a hologram that reconstructs very well, see the Figure 3.1.

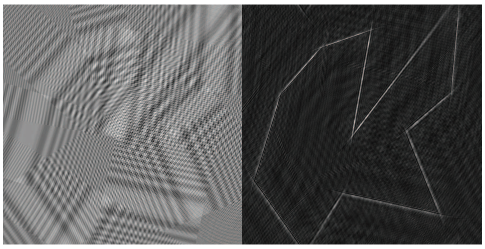

Figure 3.1: Hologram and reconstruction of a coarse planar curve. Notice the blurred endpoints.

For non-planar curve the reconstructed image is clear if fucus distance is equal or very close to a distance for which a splatting texture were computed originally, see Figure 3.2 and Figure 3.3. If a viewer is focused farther from that distance, the image becomes blurry due to non-linearities of stretching utilised for distance emulation. Still, the image does not exhibit any additional noise that may destroy the image completely.

Rendering times were not measured because the implementation was rather experimental containing several inefficient parts. The implementation was also not able to benefit from capabilities of current graphic hardware as there was no access to it.

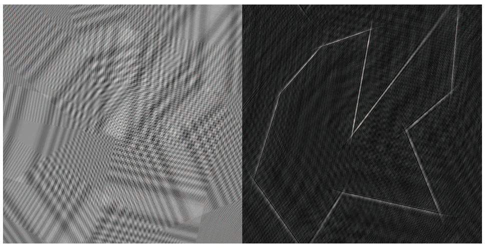

Figure 3.2: Hologram and reconstruction of a coarse nonplanar curve. Some part of the curve are out of focus.

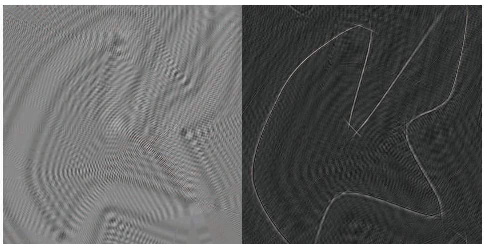

Figure 3.3: Hologram and reconstruction of a refined nonplanar curve. The approximation is still too coarse at some parts.

Chapter 4

Conclusion

The method seems to work well for curves. It is able to emulate pattern change by use of linear stretching in a direction of X-axis. However, due to a linear nature of this operation the resulting image contains non-linear effects that deform line segments to slightly curved segments, where only the endpoints are both in correct Z-axis distance. This is crucial for line segments with significant difference in Z-axis between endpoints.

Noise caused by the non-linearity of the stretch operation can be corrected by refining the stretch operation similar to an approximation of a smooth function by linear segments. This linear approximation takes places in both local X-axis and local Y-axis of a line segment. It is assumed that the refinement in the local Y-axis is more important than the local X-axis one because the correction can be applied by use of precomputed patterns for various Z-axis distances in the case of the local X-axis.

Unfortunately, the method cannot handle line segments that are perpendicular to the hologram properly. Even line segments close the perpendicularity may not be rendered correctly because their orthogonal projection to the hologram is too short and thus it violates restrictions of the pattern approximation. For the current state of the numerical reconstruction process, such segments may be omitted because the viewing plane is usually parallel to the hologram. Nevertheless, for the optical reconstruction these line segments has to be handled properly by use of a complete pattern without approximation.

Computation complexity of proposed method is $O(MN^2)$, where N is number of samples in a hologram and M is number of primitives in a scene. As the number of the primitives increases, the performance of the method decreases significantly. In a worst case it became close to a complexity of the brute force approach that is $O(KN^2)$, where K is a total number of points in the scene. This is caused by possible interpretation of the brute force approach as a splatting of individual point source patterns.

Besides the already mentioned features, the method described here cannot handle local intensity variation over a single line segment. It can modify the intensity of the whole line. This allows a local intensity variation to be approximated by geometry and/or primitive. Yet, this solution limits the size of a single detail in the local intensity variation as the orthogonal projection size of a single line segment is limited as well. Also, it increases number of primitives.

The method is also prone to effects caused by a lack of occlusion solution because it does not handle it at all. This is not a problem in the case of thin curves because curve is thin enough so it does not disturb viewer during focusing. It is a serious problem in the case of planar objects such as triangles for which a noise generated by parts that should be occluded may prevent viewer from recognising and/or focusing upon given point of interest.

Appendix A

Optical Reconstruction Results

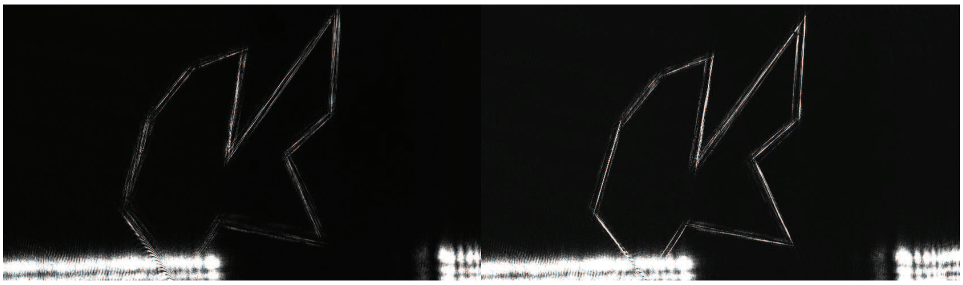

Figure A.1: Planar curve and non planar version.

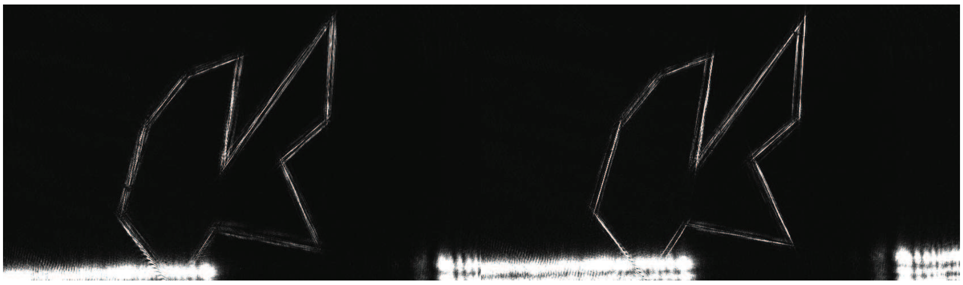

Figure A.2: Non planar curve at different focal planes.

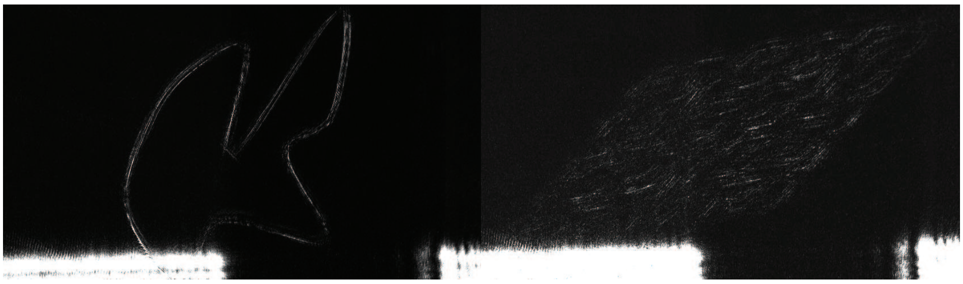

Figure A.3: More complicated curves.

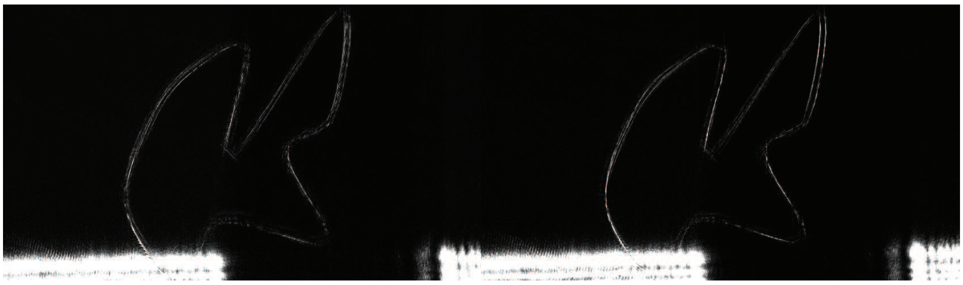

Figure A.4: More complicated curve at different focal planes.

List of Figures