Performance Considerations for a Real-Time Integral Image Camera in Ray Tracing Environments

N.P. Sgouros, D. Chaikalis, S.S. Athineos, D. Maroulis, and N. Theofanous

Department of Informatics and Telecommunications, University of Athens, 15784, Ilisia, Athens, Greece
{nsg,sathin,rtsimage}@di.uoa.gr

Abstract. Integral Imaging is a highly promising technique for delivering full parallax autostereoscopic images. A straight-forward approach for producing high quality photorealistic Integral Images or Integral Image sequences is the use of Ray-Tracing techniques. However, Ray-Tracing tasks are time consuming and in most cases scene renderings greatly deviate from performing in real time. In this work, we describe an Integral Image specific benchmarking procedure that allows accurate rendering performance evaluation of different parts of the Ray-Tracing process. A correlation based method is used to characterize the Integral Image complexity and finally calculate its actual complexity. Moreover, a number of issues are exposed that should be taken into account in real-time Integral Imaging applications.

Keywords: Three-dimensional, Image acquisition, Ray Tracing, Integral Imaging.

1 Introduction

The rapid increase in processing power and graphic card acceleration, combined with improvements in high fidelity optical systems, over the past few years, revived the interest for three-dimensional (3D) applications. Many promising technologies evolved, ranging from the classic stereoscopic ones, like polarizing glasses, mostly used at the early stages of 3D cinema, and eye shuttering glasses [1], to most sophisticated techniques like autostereoscopic displays [2,3].

Autostereoscopic display devices provide 3D stereoscopic view without the need of additional glasses, as all optical components are integrated in the display, reducing eye fatigue. Most of the currently existing autostereoscopic displays are characterized by increased spatial resolution, the reproduction of vivid colors and the ability to support multiple simultaneous users.

A special category of autostereoscopic displays, functions on the principles of Integral Photography (IP) first introduced by Lippman [4] back in 1908. As digital means for capture and display are used, the term Integral Imaging (InIm) is widely used to characterize modern digital IP systems.
A simple InIm capturing setup is built using a CCD sensor and a lens array as shown in Fig 1. The object is projected through the lens array on the CCD surface forming a number of different projections equal to the number of the lenses in the lens array that

are usually called Elemental Images (EIs). An InIm display setup uses a high resolution LCD display in conjunction with an appropriate lens array to produce high quality full parallax stereoscopic images. As InIm is probably a near ideal multiview system a number of applications have already been developed targeting medical [5], educational and entertainment [6] fields.

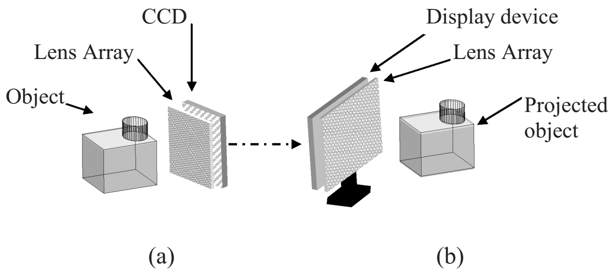

Fig. 1. A typical InIm (a) capturing and (b) display setup

Computer generated InIms provide the necessary 3D output required from a variety of applications in the aforementioned fields. These representations can be used in mixed reality environments and reduce the need for bulky or increased cost 3D cameras in a great number of cases. Moreover, computer generated InIm scenes provide enhanced user control and alleviate most of the fidelity considerations imposed from complex optical systems used in an acquisition setup.

Scanline-based techniques can be used for fast generation of 3D image data but they cannot offer the increased photorealism that ray-based approaches demonstrate [7]. The use of Ray-Tracing (RT) engines [8] for computer generated InIms [9-11] is a highly promising technique, as RT engines provide scene renderings characterized by increased realism. All capturing optics can be modeled within the ray tracer as ordinary scene objects further simplifying the architecture of the virtual InIm capturing setup [9,10]. However, the high complexity introduced by modeling the acquisition optics in the RT prohibits their use in real-time applications. The lens array introduces a specific type of complexity in an RT scene due to the repeatability of the lens objects. This fact is not taken into account by generic hardware acceleration methods for RT tasks [12,13].

In this work we initially describe a detailed evaluation procedure for the additional overhead that is introduced, by including a lens array containing a large number of lens objects in the scene. Instead of relying on a generic array description model, we focus on the accurate simulation of a physical InIm camera by creating a precise optics simulation of the lens array [11]. As the number of lenses that contribute to the InIm generation greatly affects performance, and there is a different behaviour if the lens is hit by light rays emanating from scene objects or not, we introduce a sliding window cross correlation technique to provide an accurate estimation of the active lenses of the lens array. As the number of lenses hit by light rays can be calculated we finally derive a relation between the number of lenses and rendering time.

In the second part of this work, we perform an experimental study using a number of different scene types and respective complexities and show that the introduction of the lens array causes a rapid increase in processing time regardless of a scene's initial complexity.

2 Capturing Setup and Performance Evaluation for Synthetic Integral Images

A virtual InIm camera is assembled by constructing a lens array from individual lenses using Constructive Solid Geometry (CSG) principles. The simulation of the capturing optics is realized by modeling the lens array as an ordinary object of the 3D scene [9] using the ray-tracer's Scene Description Language (SDL). This approach takes advantage of the optimized algorithms implemented in POV-Ray in order to produce high quality photorealistic InIms.

Two types of POV-Ray scenes are constructed, containing objects described using CSG or triangle primitives, in order to evaluate the effect of the lens array in different scene types. Figure 2(a) presents four representative objects and Fig. 2(b) illustrates the corresponding objects when the lens array is inserted in the scene creating the corresponding InIms. In detail, the sphere and vase are CSG objects combining a set of ray-tracer primitives while the car and teapot objects are entirely described using triangle primitives. For each of the scenes six different renderings were generated by scaling the objects behind the lens array, varying the number of EIs that depict part of an object in the scene and thus characterizing the corresponding lenses as active. The rest of the lens array whose lenses don't correspond to parts of the scene objects produces uncorrelated noise as a result of the rendering process. For example, for the four representative scenes, 24 (4x6) images of the initial objects and their corresponding InIms are created in total. The rendering time for each one is calculated using a system-level benchmarking procedure.

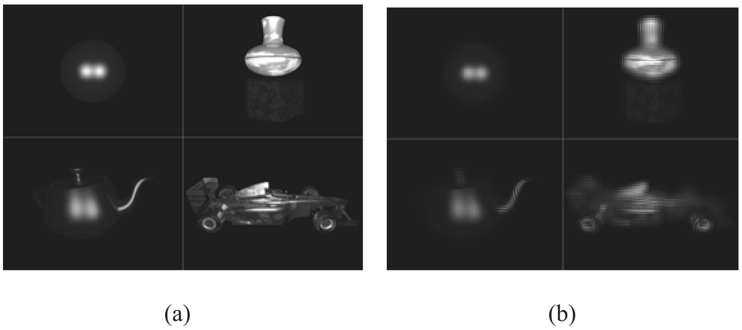

Fig. 2. Four reference images rendered using POV-Ray. From top left to bottom right: the sphere, vase, teapot and car objects (a) without lens array and (b) with lens array.

In order to evaluate which of the lenses project part of the object or correspond to noise, the cross correlation is calculated between rectangular image parts of adjacent EIs. The mean value of the correlation coefficient between an EI and its immediate neighbors determine if its corresponding lens is active or not. A schematic representation of the areas used in the correlation process is depicted in Fig. 3. The use of arectangular window is based on the symmetries of an InIm structure and increases the pixel count in each correlation window leading to more accurate determination of the Number of Active Lenses (NAL) even in cases where EIs exhibit low pixel counts.

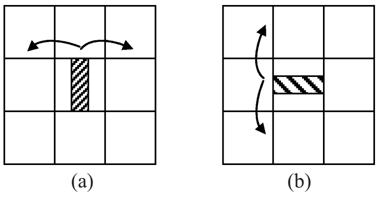

Fig. 3. The correlation operation between adjacent EIs. (a) Horizontal and (b) vertical direction. The shaded parts represent the rectangular window used in the correlation process.

The mean correlation coefficient values for the central EI of each 3x3 EI area is calculated using the aforementioned procedure. The result for the object presented in Fig. 4(a) is depicted in Fig. 4(b). Based on the values of the correlation coefficient a binary map is generated, shown in Fig. 4(c), where white pixels correspond to active lenses and black pixels to inactive ones. The NAL value is derived directly from this map and used in the benchmarking part of this work. The accuracy of the NAL value determination method was verified through visual inspection for a number of representative cases.

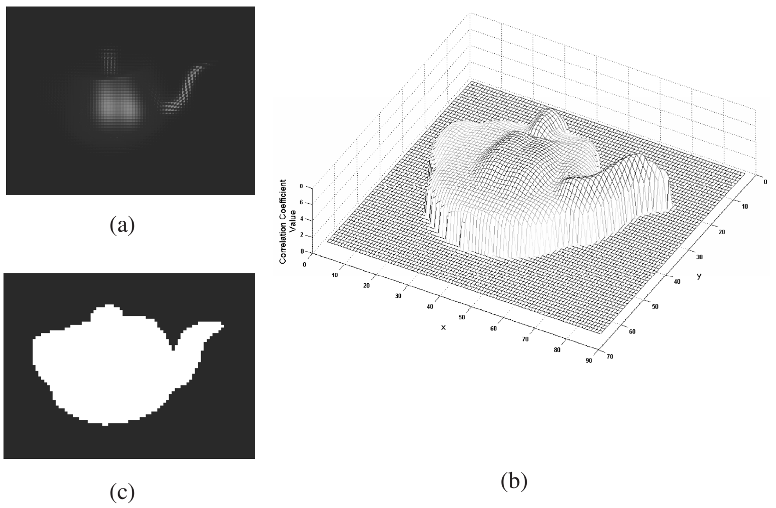

Fig. 4. (a) Original InIm, (b) Correlation coefficient values, (c) Active Lens map (magnified). White pixels indicate active lenses, black pixels indicate inactive lenses.

3 Experimental Results

In order to evaluate the complexity introduced by the existence of a lens array in a scene, we measure t_l and t_{nl} which are the rendering times with and without the lens array respectively. Next we calculate the rendering time ratio $a = t_l / t_{nl}$ for each of the generated scenes. The results versus the NAL values are plotted in Fig 5. As shown in the figure, the scenes that are solely assembled of CSG objects are affected in a greater degree by the number of active lenses in regard to the scenes generated using triangle primitives. Specifically, the rendering time for the CSG object scenes when a lens array is present is at least 5 times greater than the respective scenes without the lens array. This is due to the fact that a lens array introduces a large number of CSG objects with a similar effect on scene complexity. The rendering time for the triangle primitive object scenes is at least double when the lens array is present during the rendering process. However in these cases the increased complexity of the scenes reduced the effect of the lens array on the rendering times.

Additionally, a general increase in the rendering time ratio occurs in all scene types as the NAL value increases, regardless of the objects' description. The additional calculations introduced by each active lens have a high impact on rendering time as more lenses participate in the rendering process.

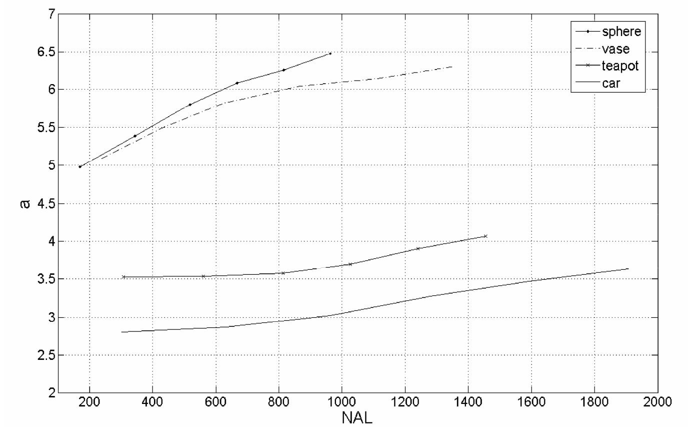

Fig. 5. The rendering time ratio for the 4 POV-Ray InIm scenes

Significant rendering information was obtained during POV-Ray's scene rendering by using AMD's CodeAnalyst software [14] as a system-level benchmark. This CPU-specific performance analyzer tool was used to measure the performance of the software's internal functions in great detail. The benchmarking procedure provided a significant amount of data which can determine the functions that greatly affect the performance of the rendering process in the relevant scenes. These functions were derived by specific measurements such as the frequency of calls, the total active time percentage and the comparison in number of calls for different scenes. These benchmarking procedure results pinpoint the functions that are candidates for hardware optimization, in order to achieve real-time performance.

4 Conclusions and Future Work

The impact of the lens array in the rendering times needed for a computer generated InIm was evaluated using a number of objects with different complexity. All scenes exhibit a constant large increase in the total rendering times which are 3 to 5 times larger than the times needed for scenes rendered without a lens array. It is also shown that as the number of the active lenses increases the rendering time increases. These observations for the rendering time ratio suggest that efficient modifications towards performance boost should be applied to the RT engine before addressing real-time InIm applications. The highly parallel nature of an optical system such as a lens array favours hardware acceleration in order to enhance the engine's performance.

The benchmarking data can be used to port the previous observations of the parallel nature of the optical system to a function level in conjunction with the required number of elemental processing operations. Future work involves the implementation of a number of time-consuming functions in hardware in order to achieve real-time performance that can benefit a large number of InIm applications.

Acknowledgements

This work was realized under the framework 8.3 of the Reinforcement Programme of Human Research Manpower ("PENED 2003" – 03ED656), co-funded 25% by the General Secretariat for Research and Technology, Greece, and 75% by the European Social Fund.

References

1. Real D scientific, http://www.reald-corporate.com/scientific/
2. Halle, M.: Autostereoscopic Displays and Computer Graphics. Computer Graphics. ACM SIGGRAPH 31(2), 58–62 (1997)
3. Dodgson, N.A.: Autostereoscopic 3D Displays. IEEE Computer 38(8), 31–36 (2005)
4. Lippmann, G.: La Photographie Integrale. C. R. Acad. Sci 146, 446–451 (1908)
5. Liao, H., Nakajima, S., Iwahara, M., Kobayashi, E., Sakuma, I., Yahagi, N., Dohi, T.: Intra-operative Real-Time 3-D Information Display System Based on Integral Videography. In: Niessen, W.J., Viergever, M.A. (eds.) MICCAI 2001. LNCS, vol. 2208, pp. 392–400. Springer, Heidelberg (2001)
6. Harman, P.: Home Based 3D Entertainment - an Overview. In: Proc. ICIP, vol. 1, pp. 1–4 (2000)
7. Halle, M.W., Kropp, A.B.: Fast Computer Graphics Rendering for Full Parallax Spatial Displays. In: SPIE, vol. 3011, pp. 105–112 (1997)
8. POV-Ray: http://www.povray.org
9. Athineos, S., Sgouros, N., Papageorgas, P., Maroulis, D., Sangriotis, M., Theofanous, N.: Photorealistic Integral Photography Using a Ray Traced Model of the Capturing Optics. Journal of Electronic Imaging 15(4), 43007–43014 (2006)
10. Milnthorpe, G., McCormick, M., Davies, N.: Computer Modeling of Lens Arrays for Integral Image Rendering. In: Proc. of EGUK 2002, pp. 136–141. IEEE Computer Society, Los Alamitos (2002)
11. Olsson, R., Xu, Y.: An Interactive Ray-Tracing Based Simulation Environment for Generating Integral Imaging Video Sequences. In: Proc. SPIE, vol. 6016, pp. 150–157 (2005)
12. IngoWald: Realtime Ray Tracing and Interactive Global Illumination, PhD thesis, Computer Graphics Group, Saarland University (2004)
13. Carr, N.A., Hoberock, J., Crane, K., Hart, J.C.: Fast GPU Ray Tracing of Dynamic Meshes using Geometry Images. In: Proceedings of the 2006 conference on Graphics Interface, pp. 203–209 (2006)
14. AMD CodeAnalyst performance analyzer tool: `http://developer.amd.com/cawin.jsp`