COMPLEXITY REDUCTION OF REAL-TIME DEPTH SCANNING ON GRAPHICS HARDWARE

Sammy Rogmans[1,2], Maarten Dumont[1], Tom Cuypers[1], Gauthier Lafruit[2] and Philippe Bekaert[1]
[1]*Hasselt University – tUL – IBBT, Expertise centre for Digital Media, Wetenschapspark 2, 3590 Diepenbeek, Belgium*
[2]*Multimedia Group, IMEC, Kapeldreef 75, 3001 Leuven, Belgium*

Keywords: Complexity reduction, Real-time, Depth scan, Graphics hardware.

Abstract: This paper presents an intelligent control loop add-on to reduce the total amount of hardware operations – and therefore the resulting execution speed – of a real-time depth scanning algorithm. The analysis module of the control loop predicts redundant brute-force operations, and dynamically adjusts the input parameters of the algorithm, to avoid scanning in a space that lacks the presence of objects. Therefore, this approach reduces the algorithmic complexity in proportion with the amount of void within the scanned volume, while remaining fully compliant with stream-centric paradigms such as CUDA and Brook+.

1 INTRODUCTION

For many years, depth map estimation has been an active research topic to progressively drive a vast amount of applications such as robot navigation, immersive teleconferencing and 3DTV with autostereoscopic displays. Although depth information can be derived in a variety of ways, when real-time performance is considered, many researchers still tend to prefer local algorithms instead of global optimization schemes. The reason is that local algorithms are data-parallel and can therefore be highly accelerated by *many-core* platforms such as the Graphics Processing Unit (GPU). However, these algorithms often resort to brute-force approaches, resulting in a large amount of redundant operations. Although real-time performance is still achieved, the redundant brute-force operations significantly suppress the high-speed and low-power potential of these local algorithms.

This paper presents an intelligent control loop add-on to reduce the intrinsic complexity of real-time depth scanning algorithms, overcoming their 'unintelligent' brute-force approach. The algorithms theirselves remain completely unaltered by predicting the redundant operations in advance, while dynamically changing the input parameters similar to a regulator in control systems. The proposed approach is therefore compliant with stream-centric paradigms such as CUDA and Brook+, but also with fixed-functionality hardware (e.g. ASICs).

The layout of the paper is as follows: Section 2 explains the principles of depth scanning, and presents the reader with a complexity analysis of one of the currently most promising state-of-the-art local algorithms. Consequently, Section 3 describes the control loop add-on to reduce the total algorithmic complexity, while Section 4 exposes optimization schemes that can be applied specifically on the GPU. The results are discussed in Section 5, and the paper is concluded in Section 6.

2 DEPTH SCANNING

In the most general case, local algorithms determine depth information by sweeping a plane through the 3D volume (Yang et al., 2002), checking each voxel (i.e. pixel-sized 3D elementary volume) for objects in a brute-force manner. This general approach can be applied to multi-camera setups, but a rectified two-fold camera setup is often used to simplify the 3D scan, which is known as *stereo matching*. In a stereo camera setup, the depth of objects is determined by estimating the amount of pixels they have shifted from the left to the right image. As depicted in Fig. 1, this shift is consistently known as the motion paral-

Rogmans S., Dumont M., Cuypers T., Lafruit G. and Bekaert P. (2009).
COMPLEXITY REDUCTION OF REAL-TIME DEPTH SCANNING ON GRAPHICS HARDWARE.
In *Proceedings of the Fourth International Conference on Computer Vision Theory and Applications*, pages 547-550
DOI: 10.5220/0001806505470550
Copyright © SciTePress

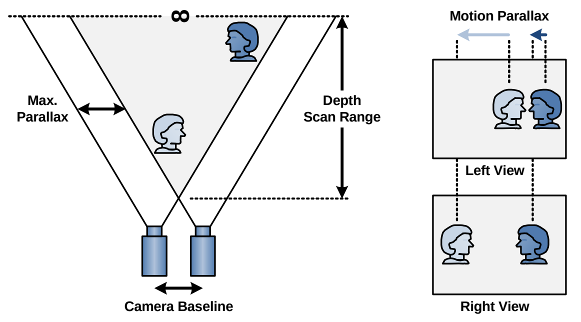

Figure 1: Basic principles of stereo matching, that can be seen as a simplified 3D depth scan.

lax or disparity, and directly correlates to the objects' depth. Objects close to the camera exhibit a large parallax and vice versa, with the maximum parallax determined by the baseline distance between the two cameras. Looking for matches in the stereo pair – using a systematically decreasing disparity, starting from the maximum value – can therefore be seen as a simplified front-to-back depth scan.

Consistently surveyed by (Scharstein and Szeliski, 2002), stereo matching algorithms can be divided into a *cost computation*, *cost aggregation*, *disparity selection* and *disparity refinement* processing module. Local algorithms tend to put their focus on the aggregation, and generally discard the refinement. As recently indicated by a performance study of (Gong et al., 2007), one of the most promising local algorithms is the *adaptive weights* approach. Therefore, this approach is used as a case-study. For the interested reader, the complete algorithmic description can be found in the paper of (Yoon and Kweon, 2006). However, our paper only presents a high-level description of the adaptive weights algorithm, to obtain a basic analysis of its current complexity.

The GPU implementation of the adaptive weights is composed out of three different types of floating point-based operations. These are the *texture* (i.e. memory read), *render* (i.e. memory write), and *computational* operations. The matching cost C_{AD} computation inputs an RGB-colored left/right image pair $\mathbf{I}_0^*(x,y)$ and $\mathbf{I}_1^*(x,y)$, disparity hypothesis d, and integrates the absolute differences of each color channel according to the dot product

$$C_{AD}(x,y,d) = \mathbf{g} \cdot \left|\mathbf{I}_0^*(x,y) - \mathbf{I}_1^*(x-d,y)\right| \; , \qquad (1)$$

with a grey-scaling vector $\mathbf{g} = \langle 0.299,\ 0.587,\ 0.114\rangle$. The matching cost is computed for each hypothesis d from the search range S, regardless of the presence of objects at the examined depth. Consequently, the cost is truncated to a given maximum τ and normalized (toward 255 when considering 24-bit color images), following

$$C_{TAD}(x,y,d) = \frac{255}{\tau} \cdot \min\left(C_{AD}(x,y,d), \tau\right) \; , \qquad (2)$$

to reduce the impact of noise inside the source images. The cost is then aggregated by a 33×1 horizontal and 1×33 vertical separated 1D convolution kernel $w_H(x,y,u)$, respectively $w_V(x,y,v)$. The composition of these kernels are based on the Gestalt principles of similarity and proximity, and are therefore computed in a preprocessing step according to

$$w_H(x,y,u) = e^{-\frac{1}{\gamma_s}|\Delta c_{xu}|} + e^{-\frac{1}{\gamma_p}|u|} \; , \qquad (3)$$

with $|\Delta c_{xu}|$ being the Euclidean color distance between $\mathbf{I}_0^*(x,y)$ and $\mathbf{I}_0^*(x+u,y)$, and $\gamma_s = 17.6$, $\gamma_p = 40.0$ being a fall-off rate for the similarity, respectively proximity term. The vertical kernel w_V is computed in a similar manner. Consequently, the aggregated cost A_{WE} is defined as

$$A'_{WE}(x,y,d) = \frac{w_H(x,y,u) \ast_u C_{TAD}(x+u,y,d)}{\mu_{H_{xy}}} \; , \qquad (4)$$

$$A_{WE}(x,y,d) = \frac{w_V(x,y,v) \ast_v A'_{WE}(x,y+v,d)}{\mu_{V_{xy}}} \; , \qquad (5)$$

where $\mu_{H_{xy}}$, $\mu_{V_{xy}}$ are normalization constants, and A'_{WE} stores the intermediate result of the horizontal aggregation. Finally, the best matching cost is selected out of all disparity hypotheses conceived in S, on a winner-takes-all (WTA) basis. The resulting depth map $D(x,y)$ is therefore composed according to

$$D(x,y) = \arg\min_{d \in S} A_{WE}(x,y,d) \; . \qquad (6)$$

Table 1 reflects our analysis of the type and amount of operations that are involved to compute a depth map with the adaptive weights approach using a 33×33 convolution kernel, where p defines the amount of pixels in the image, and h the amount of disparity hypotheses, i.e. the cardinality of S. Hence the total amount of operations O to generate a depth map can be described as

$$O = (1027 + 23 \cdot h) \cdot p \; , \qquad (7)$$

resulting in a little over 17.6 Gops for the 1800×1500 *Teddy* scene (Middlebury, 2003).

3 COMPLEXITY REDUCTION

The proposed method can reduce the total complexity of the adaptive weights algorithm, or any other local stereo matching algorithm for that matter. To achieve this, the algorithm is executed twice, but with different inputs. In the first pass, the complete volume

Table 1: Amount of operations needed to generate a depth map with the adaptive weights algorithm using a 33×33 convolution kernel, in function of the amount of image pixels p and disparity hypotheses h.

	Preprocessing	Matching Cost	Cost Aggregation	Disparity Selection
Texture Ops	$198 \cdot p$	$6 \cdot p \cdot h$	$132 \cdot p \cdot h$	$1 \cdot p \cdot h$
Computational Ops	$764 \cdot p$	$13 \cdot p \cdot h$	$256 \cdot p \cdot h$	$1 \cdot p \cdot h$
Render Ops	$64 \cdot p$	$1 \cdot p \cdot h$	$1 \cdot p \cdot h$	$1 \cdot p$
TOTAL OPS	$1026 \cdot p$	$20 \cdot p \cdot h$	$1 \cdot p \cdot h$	$(2 \cdot h + 1) \cdot p$

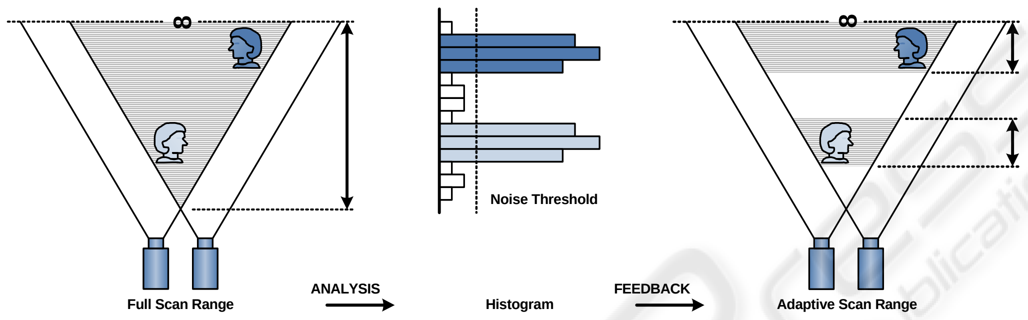

Figure 2: The principle of our proposed complexity reduction. The full 3D volume is scanned with a low-complexity approach, and analyzed through a histogram. Consequently, an accurate high-complexity scan is performed on an adaptive range.

is scanned over the entire search range S, but with a smaller input image resolution. Consequently, an analysis on the depth map D is performed by computing the histogram $H(d)$. As shown in Fig. 2, the histogram indicates the depths where objects are actually located. In (Gallup et al., 2007), the histogram has already been used to find the best orientation of the depth sweep, by performing the sweep for multiple orientations and selecting the orientation that leads to the minimum data-entropy of the histogram. Furthermore, (Geys et al., 2004) also used histogram information, but similar to (Gallup et al., 2007), this is mainly to lever the resulting quality of the algorithm. In our proposed approach, the histogram information acquired from a coarse-grained full range scan leads to an accurate fine-grained scan over an adaptive range. As the complexity of the histogram computation is fairly low itself, the potential to accelerate a local stereo matching algorithm becomes very high.

In general, the less pronounced *foreground fattening* (Scharstein and Szeliski, 2002) an algorithm exhibits, the more the input image resolution can be reduced in the first pass, while still maintaining a representative histogram. In case of the adaptive weights approach, the input image dimensions can be reduced up to four times, resulting in a 16-fold reduction of the original complexity.

To determine which subset $S_{sub} \subseteq S$ to take, the values of the histogram are compared with a given threshold δ. Instead of fixing the treshold δ, it can be made dynamic by setting the level proportional to the data-entropy of the histogram. Therefore, the threshold will be set low for complex scenes with fine-grained geometry, and high when little geometry is present. Hence, this efficiently filters out noise due to mismatches inherent to the local algorithm.

4 OPTIMIZATIONS

The aforementioned algorithmic complexity is only valid when considering the use of conventional General Purpose GPU (GPGPU) computing (Owens et al., 2007). However, several optimization schemes can be applied to further reduce the complexity of local stereo algorithms. For the cost aggregation module, the amount of texture operations can be reduced by half, when bilinear sampling is used to aggregate two values with a single memory lookup. Furthermore, in the next-generation GPGPU paradigm CUDA, 16 memory operations can be coalesced into a single memory read operation. For the disparity selection, the depth test of the GPU can be exploited to avoid all explicit operations (Lu et al., 2007). However, the depth test can only be exploited through the traditional GPGPU paradigm using Direct3D or OpenGL. Nevertheless, thanks to the recent introduction of CUDA v2.0 and the complete support of interoperability between CUDA and Direct3D, both paradigms can be combined to form a highly optimized implementation.

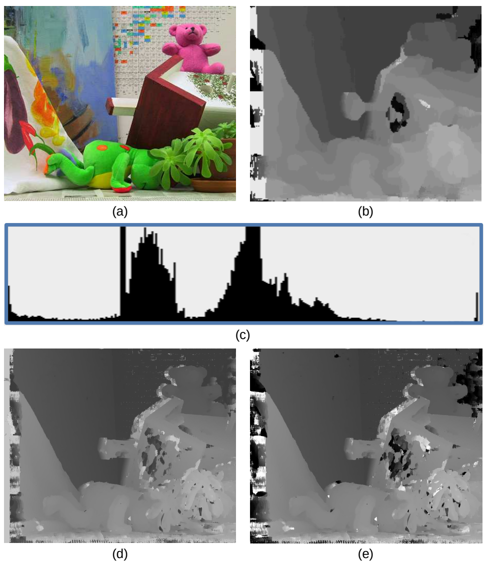

Figure 3: (a) The *Teddy* scene, (b) low-complexity depth map, (c) joint histogram, (d) the resulting high-quality depth map with reduced operations, and (e) the original result.

5 EXPERIMENTAL RESULTS

The proposed method was tested on the 1800×1500 *Teddy* scene (see Fig. 3a) of the Middlebury dataset (Middlebury, 2003), with disparity search range $S = \{0,\ldots,240\}$. The image resolution was lowered 16 times to 450×375, which yields the disparity map depicted in Fig. 3b. After the histogram (see Fig. 3c) analysis, the detected subset $S_{sub} = \{60,\ldots,96,108,\ldots,180\}$ is only 45% of the orginal search range S. As the first scan is only $1/16^{\text{th}}$ (or 6.25%) of the original complexity, a total complexity reduction of about 48% is harnessed. The disparity map in Fig. 3d is therefore generated in about 8.5 Gops, instead of the original result in Fig. 3e, which takes 17.6 Gops. Our control-loop scheme therefore yields a two-fold complexity reduction, and is able to double the execution speed of the algorithm.

Considering that the complexity of the first pass is almost negligible, the proposed control loop add-on will allow for a speedup, proportional to the amount of void within the scanned volume. This is particularly useful in eye-gaze corrected video chat (Dumont et al., 2008), as only the chat participant needs to be scanned in a rather large office space.

6 CONCLUSIONS

We have proposed a control loop add-on to reduce the complexity of local real-time stereo matching algorithms. The histogram of a coarse-grained depth scan is analyzed to adjust the input parameters of a consequent fine-grained accurate scan, causing the algorithm to avoid scanning in spaces that lack the presence of objects. Hence, the orignal brute-force algorithmic complexity can be reduced in proportion with the amount of void within the volume, while still remaining fully compliant with stream-centric paradigms such as CUDA or Brook+.

ACKNOWLEDGEMENTS

Sammy Rogmans would like to thank the IWT for the financial support under grant number SB071150. Tom Cuypers and Philippe Bekaert acknowledge the financial support by the European Commission (FP7 IP 2̈020 3D media)̈. Furthermore, all authors recognize the financial support from the IBBT.

REFERENCES

Dumont, M., Maesen, S., Rogmans, S., and Bekaert, P. (2008). A prototype for practical eye-gaze corrected video chat on graphics hardware. In *Proc. of SIGMAP*.

Gallup, D., Frahmand, J., Mordohai, P., Yang, Q., and Pollefeys, M. (2007). Real-time plane-sweeping stereo with multiple sweeping directions. In *Proc. of CVPR*.

Geys, I., Koninckx, T. P., and Gool, L. V. (2004). Fast interpolated cameras by combining a GPU based plane sweep with a max-flow regularisation algorithm. In *Proc. of 3DPVT*.

Gong, M., Yang, R., Wang, L., and Gong, M. (2007). A performance study on different cost aggregation approaches used in real-time stereo matching. *IJCV*, 75(2):283–296.

Lu, J., Rogmans, S., Lafruit, G., and Catthoor, F. (2007). High-speed dense stereo via directional center-biased support windows on programmable graphics hardware. In *Proc. of 3DTV-CON*.

Middlebury (2003). http://vision.middlebury.edu/stereo.

Owens, J., Luebke, D., Govindaraju, N., Harris, M., Kruger, J., Lefohn, A., and Purcell, T. (2007). A survey of general-purpose computation on graphics hardware. *CG Forum*, 26(1):80–113.

Scharstein, D. and Szeliski, R. (2002). A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. *IJCV*, 47(1-3):7–42.

Yang, R., Welch, G., and Bishop, G. (2002). Real-time consensus-based scene reconstruction using commodity graphics hardware. In *Proc. PG*.

Yoon, K.-J. and Kweon, I.-S. (2006). Adaptive support-weight approach for correspondence search. *IEEE PAMI*, 28:650–656.