Adaptive LightSlice for Virtual Ray Lights

R. Frederickx[†], P. Bartels, and Ph. Dutré[‡]

Department of Computer Science, KU Leuven, Belgium

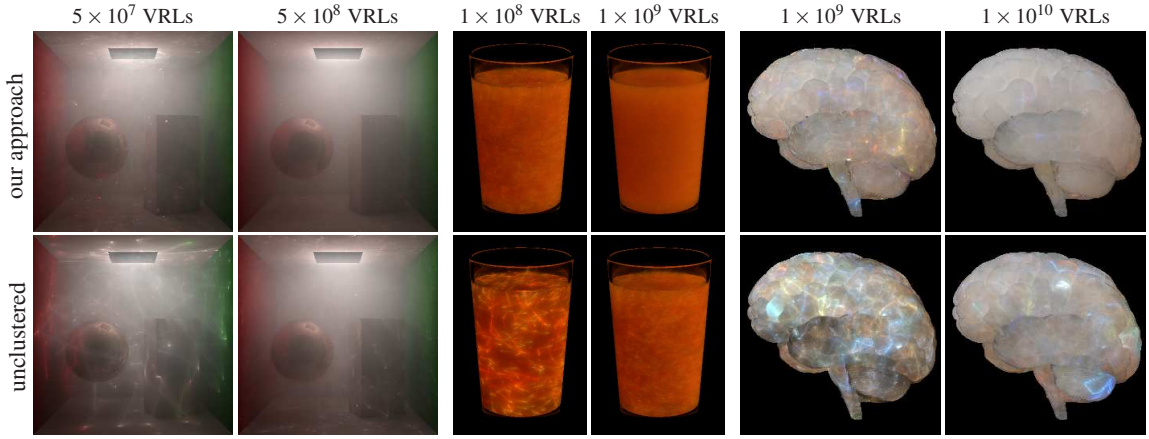

Figure 1: *Comparison of Virtual Ray Light (VRL) rendering with our clustering (above) and without (below) for three scenes. Each column of images is rendered with the same total number of evaluated VRLs, including preprocessing. The top right image of each scene is rendered till near convergence, the left images of each set are rendered with 1/10th the number of VRLs.*

Abstract

We speed up the rendering of participating media with Virtual Ray Lights (VRLs) by clustering them in a preprocessing step. A subset of representative VRLs is then sampled from the clustering, which is used for the final rendering. By performing a full variance analysis, we can explicitly estimate the convergence rate of the rendering process and automatically find the locally ideal number of clusters to maximize efficiency. Overall, we report speed-up factors ranging from 13 to 16 compared to unclustered rendering.

Categories and Subject Descriptors (according to ACM CCS): I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism—Color, shading, shadowing, and texture

1. Introduction

Rendering multiple scattering in participating media is an inherently difficult sampling problem due to the combinatorial explosion of light paths. The Virtual Ray Light (VRL) method [NNDJ12] solves the problem of volumetric light transport by storing virtual emitting line segments that act as secondary light sources. We show how the LightSlice algorithm [OP11] can be be adapted to efficiently cluster these VRLs. Moreover, by estimating the resulting convergence rate, we can automatically determine the locally ideal number of clusters to maximize efficiency.

2. Background and Related Work

The most general techniques to render participating media typically rely on a form of Monte Carlo integration over the scattered light paths in the volume. Methods that are based

† roald.frederickx@cs.kuleuven.be
‡ phil.dutre@cs.kuleuven.be

on density estimation have been shown to robustly solve the scattering problem, albeit by introducing some bias in the result [JNSJ11, KGH*14]. Unbiased methods include bidirectional path tracing [LW96, GKH*13], Metropolis light transport [PKK00] and the Virtual Ray Light (VRL) algorithm [NNDJ12]. This last algorithm employs many virtual light (VL) sources in order to capture indirect volumetric lighting. A drawback of such many-light algorithms is that they can spend a lot of work calculating the contributions of VLs that either only add an insignificant contribution (e.g. lights that are very far away or occluded) or a highly similar contribution (e.g. lights emanating from a similar position in the scene) to the final result. To circumvent this, methods have been proposed that cluster similar VLs into roughly equally important sets and only use a single representative VL to capture the aggregate effect of the cluster [WABG06, HPB07, OP11]. As of yet, these methods have only been used for Virtual Point Lights (VPLs). We will adapt the most recent method, LightSlice [OP11], to handle VRLs. Additionally, we automatically determine the locally ideal number of clusters in order to maximize convergence.

3. Overview of our Approach

The LightSlice algorithm [OP11] computes a single Virtual Light (VL) clustering for each slice (a set of 'geometrically similar' pixels) by assessing the contributions of the full set of VLs in nearby pixels. Because LightSlice operates on pixel values, it is agnostic to the type of VL used, and hence ideally suited to adapt to VRLs [NNDJ12] (in contrast, the Lightcuts algorithm [WABG06] requires tight analytical error bounds on the VL contributions, which is much harder to come by for VRLs than for VPLs). Moreover, we extend the LightSlice algorithm to adaptively determine the ideal number of representative VLs per slice.

We start by formalizing the 'ideal' number of representatives for a given set of VRLs. We can define the convergence constant c of a Monte Carlo rendering process through the relation $V = c/n$, with V the variance of the resulting estimator and n the number of samples. In a slice with P pixels we aim to minimize the convergence constant, $c = (N^{\text{prep}} + N^{\text{rend}}) \sum_i^P V_i$, with V_i the variance of pixel i, and N^{prep} and N^{rend} the number of evaluated VRLs during preprocessing and rendering, respectively.

We further formalize the VRL tracer as a stochastic process $\tilde{X}$ from which N samples (VRLs) $X_1, \dots X_N$ are taken. Let $C(X_j, i)$ denote the (exact) contribution of a VRL X_j to a pixel i. Then, each VRL sample X_j itself induces a set of stochastic processes $\tilde{X}_{ij}$: the Monte Carlo integration to compute the contribution of X_j to the eye ray associated with pixel i. We model these processes as $\tilde{X}_{ij} = C(X_j, i)/N + \tilde{\xi}_{ij}$, where $\tilde{\xi}_{ij}$ is stochastic noise with expectation value $\langle \tilde{\xi}_{ij} \rangle = 0$ and the $1/N$ factor is to keep with the conventions of [OP11]. Lastly, locally similar VRLs are combined in a cluster from which an additional discrete stochastic process selects one representative. Let $j_1, \dots, j_n$ be the indices of the VRLs in a cluster under consideration, then sampling from $\sum_l \tilde{X}_{ij_l}$ is replaced by sampling from $\tilde{X}_{ij_m}/p_m$ with some probability p_m. These three types of stochastic processes (the VRL tracer $\tilde{X}$, the VRL integration $\tilde{X}_{ij}$ and the selection of the representatives) give rise to the final pixel variance V_i.

We now find an estimate of the variance V_i (see the supplementary material for a detailed derivation and assumptions). Let $\mathbf{A}_{ij}$ be the transfer matrix element which is a sample from $\tilde{X}_{ij}$, making $\sum_j^N \mathbf{A}_{ij}$ an estimate of the true pixel value $\langle C(\tilde{X}, i) \rangle$. The variance introduced by the VRL tracer $\tilde{X}$ on the observed pixel value $\sum_j \mathbf{A}_{ij}$ is given by $\text{Var}(C(\tilde{X}, i))/N$, which is estimated by $V_i^{\text{trace}} = N/(N-1) \sum_j (\mathbf{A}_{ij} - \sum_k \mathbf{A}_{ik}/N)^2 - \sum_j \xi_{ij}^2$. Here, ξ_{ij}^2 estimates the variance $\langle \tilde{\xi}_{ij}^2 \rangle$, which can be readily obtained during the VRL – eye ray integration. Intuitively, to estimate the bare VRL tracer variance, the observed variance in $\sum_j \mathbf{A}_{ij}$ is corrected for the VRL integration variance ξ_{ij}^2 in the $\mathbf{A}_{ij}$ samples. Moreover, the undersampling from the clustering induces extra variance, which combines with the VRL integration variance in the form $V_i^{\text{und}} = \sum_l (\mathbf{A}_{ij_l}^2 + \xi_{ij_l}^2)/p_l - (\sum_l \mathbf{A}_{ij_l})^2$, where $j_1, \dots, j_n$ are the indices of VRLs in the cluster under consideration. This variance V_i^{und} is minimized over the P pixels of the slice by choosing the probabilities as $p_l \propto \sqrt{\sum_i^P (\mathbf{A}_{ij_l}^2 + \xi_{ij_l}^2)}$. Hence, highly contributing VRLs or VRLs with a high integration variance have a high probability of getting exactly resolved by being chosen as representatives. We further split the combined effect of the cluster undersampling variance contributions V_i^{und} in V_i^{clust} and V_i^{int}, where the first is the sum of $\sum_l \mathbf{A}_{ij_l}^2/p_l - (\sum_l \mathbf{A}_{ij_l})^2$ over all clusters, which gauges the local variability in the VRL contributions within a slice, and the latter is the sum of $\sum_l \xi_{ij_l}^2/p_l$ over the clusters, which accounts for the inherent variance of the VRL integration. The final variance estimate V_i is thus $V_i = V_i^{\text{trace}} + V_i^{\text{clust}} + V_i^{\text{int}}$.

Of course, all of this this assumes knowledge of the full matrix $\mathbf{A}$, which is exactly what we want to calculate. Therefore, the average over all P pixels in the definition of c above is replaced with a sparse average over a random subset of P^{repr} representative pixels of the slice

$$c = (N^{\text{prep}} + N^{\text{rend}})(P/P^{\text{repr}}) \sum_i^{P^{\text{repr}}} V_i. \tag{1}$$

We set $P^{\text{repr}} = \max(2, P/\alpha)$, where $\alpha > 1$ denotes the target pixel undersampling factor and the lower bound of two is needed for the variance estimators. Note that this parameter replaces the number of neighbours of the original LightSlice algorithm. In this case, $N^{\text{prep}} = NP^{\text{repr}}$ and $N^{\text{rend}} = N^{\text{repr}}P$, with N^{repr} the number of representative VRLs sampled from the clustering, i.e. the number of clusters.

For each slice, we progressively refine an initial cluster that contains all VRLs. The cluster with the highest contribution to $\sum_i^{P^{\text{repr}}} V_i$ is split, thereby increasing N^{repr} by one

and decreasing the V^{clust} and V^{int} components. We follow the splitting procedure of [OP11] by projecting the VRLs on a P^{repr}-dimensional line; $\sum_i V_i$ can be calculated in an on-line manner and the best split can be found in linear time in the cluster size. Splitting continues until the minimum of the estimated convergence constant c is found: initially, c behaves as $1/N^{\text{repr}}$ as the effect of the $\sum_i V_i$ factor dominates, eventually V_i converges to $V_i^{\text{trace}} + \sum_j \xi_{ij}^2$ and we have $c \sim N^{\text{repr}}$, the optimum is found in between the two regimes.

Compared to the original LightSlice algorithm, we need an extra term containing ξ_{ij}^2 in the cluster cost and in the probabilities p_l to account for the variance of the VRL integral samples. Furthermore, we also take into account the variance due to tracing, V^{trace}, allowing us to estimate and optimize the convergence constant c. To this end, we need accurate estimates of the variance in a slice, achieved by changing from the number of neighbours parameter to the pixel undersampling α and sampling only within the slice.

4. Scenes and Parameter Selection

Implementation We implemented Virtual Ray Lights and our adaptive LightSlice variant in the Mitsuba [Jak10] CPU ray tracer. All reported results are generated with isotropically scattering media. We did not implement the anisotropic importance sampling routines of [NNDJ12], but we verified that anisotropic media give similar results with our isotropic sampling combined with a higher sample rate to make up for the imperfect sampling.

Scenes We test the algorithm on the three different scenes shown in Fig. 1. The first scene is a typical unit-length Cornell box containing a reflecting sphere and a white diffuse box. The space is filled with a thin fog with scattering coefficient $\sigma_s = 2$ and absorption coefficient $\sigma_a = 0.2$. The second scene is a glass of grapefruit juice with RGB parameters taken from [NNDJ12]: $\sigma_s = (0.45, 0.32, 0.23)\,\text{cm}^{-1}$ and $\sigma_a = (0.41, 0.95, 4.73)\,\text{cm}^{-1}$. Lastly, we have a unit-diameter model of a brain with a marble-like high albedo material of $\sigma_s = (10.1, 13.1, 15.0)$ and $\sigma_a = (10.5, 20.5, 35.5) \times 10^{-3}$ with index of refraction 1.5. All media are isotropically scattering and the image resolutions are 256×256 for the box and brain scenes and 240×360 for the glass.

Number of slices The number of slices is chosen such that the local lighting within a slice is similar and can be captured with a single, compact clustering. Nonetheless, slices should be big enough to allow a sufficient pixel undersampling factor α without degenerating to the $P^{\text{repr}} = 2$ lower bound. Moreover, unlike the original LightSlice setting, there should now be sufficient variation in the local lighting to overcome the VRL integration noise, or no informed clustering decision can be made. The V^{clust} and V^{int} values can be used as a guide. We found that for for all three scenes, choosing 100 slices satisfied these criteria.

Pixel undersampling The pixel undersampling factor is a critical parameter that should be chosen as high as possible, but not so high that the sparse sampling starts to miss important VRLs: this would introduce isolated areas where the VRLs are only resolved near the representative pixels. The other pixels would be substantially darker and occasionally receive a very bright contribution when an important VRL that was not properly detected does get selected. Visual inspection or monitoring the variance while gradually increasing α can detect these cases. For the glass and brain scenes, an undersampling factor of $\alpha = 50$ was found to be a good balance between efficiency and safety. The thinner medium in the box scene has a higher transmission coefficient, causing VRLs to have a bigger volume of influence. This allows the pixel undersampling to safely go up to $\alpha = 100$.

Number of VRLs We progressively trace batches of VRLs which are clustered and rendered. How many VRLs should we trace per batch? In contrast to the original LightSlice, we now have to take into account the extra variance on the VRL integral samples; more is not always better. We found the optimal values to lie in the range of 100 to 1000 VRLs for our scenes. Within this range, for a cluster that has been split to minimize the estimated c, V^{clust} is of the order of V^{int}. Indeed, if V^{int} is much larger than V^{clust} after splitting, then the noise on the representative samples was too high to discover any structure and the cluster was essentially split without guided information. If, on the other hand, V^{int} is much lower than V^{clust}, then we are better off with more VRLs so we can split further to exploit more structure. For our shown results, we chose a batch size of 300 VRLs for all scenes.

5. Results and Discussion

We record the convergence constant as the product of the total number of integrated VRLs with the mean squared pixel deviations compared to a reference image. The overhead of our adaptive clustering algorithm is on the order of 5% of the total CPU time, hence the total number of integrated VRLs is a good indicator of the total rendering time. We compare our convergence constants with those obtained for rendering the full set of VRLs and report the corresponding speed-up.

Compared to no clustering Fig. 1 shows equal-time renders with and without our adaptive clustering. The speed-ups are 16.45 ± 0.06, 13.42 ± 0.03 and 15.0 ± 0.1 for the box, glass and brain scenes, respectively. For each scene, the clustered result in the top left has roughly the same (or slightly higher) visual quality as the unclustered result in the bottom right after a factor 10 less work, which is consistent with our reported speed-up. The results in the top right are rendered with 980, 1210 and 7000 passes for the respective scenes.

Compared to nonadaptive clustering Fig. 2 compares our adaptive method to a fixed VRL undersampling factor, i.e. always choosing $N^{\text{repr}} = N/\beta$ for some global β like the

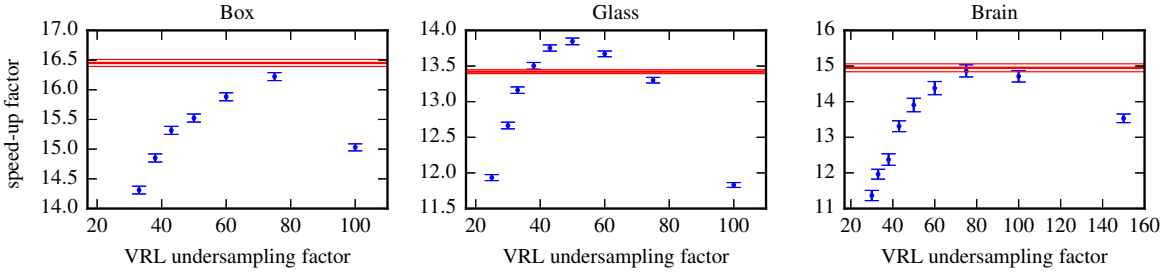

Figure 2: *Speed-ups compared to unclustered rendering for the scenes of Fig. 1. Individual data points correspond to fixed, global VRL undersampling factors* β. *The horizontal line indicates the speed-up and its error bar of our adaptive algorithm.*

original LightSlice algorithm. For the glass scene, we do slightly worse than the optimal undersampling. This is not surprising, as we are trying to determine the optimum from a noisy and sparse sampling. Nonetheless, we reach a speed-up that is within 3% of the optimum for fixed β. More interestingly, we are on par for the brain scene and even outperform a global fixed VRL undersampling factor in the box scene. Indeed, our undersampling factor is chosen for each slice separately, which allows us to adapt to the local lighting context, allocating VRLs only where needed. We refer to the supplementary material for false color images and a more in-depth discussion.

6. Conclusion and Future Work

We have adapted the LightSlice algorithm to work with Virtual Ray Lights. Such VRLs inherently introduce extra variance, which needs to be taken into account during the clustering and when selecting the parameters of the algorithm. By extending LightSlice to adaptively find the optimum number of clusters, we effectively remove one of these parameters and moreover allow for potential higher convergence rates by adapting to local lighting conditions. This extension is orthogonal to the use of VRLs and can also be applied for VPLs. Overall, we report speed-up factors ranging from 13 to 16 compared to unclustered rendering of VRLs.

Nonetheless, even though we have eliminated one parameter, there are still three left that need to be chosen. Failure cases arise when the pixel undersampling is chosen too high such that important VRLs don't get sampled, leading to high variance. Choosing it too low hampers efficiency. The number of initial virtual lights is also more important in the VRL case. We report how monitoring V^{clust} and V^{int} can help in finding the sweet spot of these parameters. It remains for future work to do this automatically in the hopes of eliminating another parameter altogether.

7. Acknowledgements

Roald Frederickx is a predoctoral fellow of the Fund for Scientific Research (FWO) of Flanders.

References

[GKH*13] GEORGIEV I., KŘIVÁNEK J., HACHISUKA T., NOWROUZEZAHRAI D., JAROSZ W.: Joint importance sampling of low-order volumetric scattering. *ACM Transactions on Graphics (TOG) 32*, 6 (2013), 164. 2

[HPB07] HAŠAN M., PELLACINI F., BALA K.: Matrix row-column sampling for the many-light problem. In *ACM SIGGRAPH 2007 Papers* (New York, NY, USA, 2007), SIGGRAPH '07, ACM. doi:10.1145/1275808.1276410. 2

[Jak10] JAKOB W.: Mitsuba renderer, 2010. URL: http://www.mitsuba-renderer.org. 3

[JNSJ11] JAROSZ W., NOWROUZEZAHRAI D., SADEGHI I., JENSEN H. W.: A comprehensive theory of volumetric radiance estimation using photon points and beams. *ACM Transactions on Graphics (TOG) 30*, 1 (2011), 5. 2

[KGH*14] KŘIVÁNEK J., GEORGIEV I., HACHISUKA T., VÃČÂĽVODA P., IK M. Ã., NOWROUZEZAHRAI D., JAROSZ W.: Unifying points, beams, and paths in volumetric light transport simulation. *ACM Transactions on Graphics (Proceedings of SIGGRAPH) 33*, 4 (July 2014). doi:10.1145/2601097.2601219. 2

[LW96] LAFORTUNE E. P., WILLEMS Y. D.: Rendering participating media with bidirectional path tracing. In *Proceedings of the eurographics workshop on Rendering techniques' 96* (1996), Springer-Verlag, pp. 91–100. 2

[NNDJ12] NOVÁK J., NOWROUZEZAHRAI D., DACHSBACHER C., JAROSZ W.: Virtual ray lights for rendering scenes with participating media. *ACM Transactions on Graphics (Proceedings of ACM SIGGRAPH 2012) 31*, 4 (July 2012). doi:10.1145/2185520.2185556. 1, 2, 3

[OP11] OU J., PELLACINI F.: Lightslice: Matrix slice sampling for the many-lights problem. In *Proceedings of the 2011 SIGGRAPH Asia Conference* (New York, NY, USA, 2011), SA '11, ACM, pp. 179:1–179:8. doi:10.1145/2024156.2024213. 1, 2, 3

[PKK00] PAULY M., KOLLIG T., KELLER A.: Metropolis light transport for participating media. In *Proceedings of the Eurographics Workshop on Rendering Techniques 2000* (2000), Springer-Verlag, pp. 11–22. 2

[WABG06] WALTER B., ARBREE A., BALA K., GREENBERG D. P.: Multidimensional lightcuts. *ACM Trans. Graph. 25*, 3 (July 2006), 1081–1088. doi:10.1145/1141911.1141997. 2