A Matrix Sampling-and-Recovery Approach for Many-Lights Rendering Supplmentary Document

Yuchi Huo Rui Wang* Shihao Jin Xinguo Liu Hujun Bao*
State Key Lab of CAD&CG, Zhejiang University

1 Reduced Lighting Matrix Construction

1.1 Search Nearby Light Nodes in Light Cut

While we estimate the maximum contribution of one light node, we improve the reliability of estimation by borrowing material and geometry terms from nearby cut nodes. We use bidirectional list to store the nodes on the cut. When a cut node is split, we remove it from the list and insert its two children at the same position in the cut node list. Therefore, the range search to find nearby nodes is simply to traverse in the list. The nearby indices of light node j is computed as:

$$\mathcal{N}(j) = \left\{k | \sum_{u=k}^{j-1} I_u \le 2I_j\right\} \cup \{j\} \cup \left\{k | \sum_{u=j+1}^{k} I_u \le 2I_j\right\} \quad (1)$$

where twice of current intensity of light node j is used as a bound to find nearby light nodes. A simple illustration is shown in Fig. 1, where we assume each pair of children nodes averagely share their ancestor node's intensity.

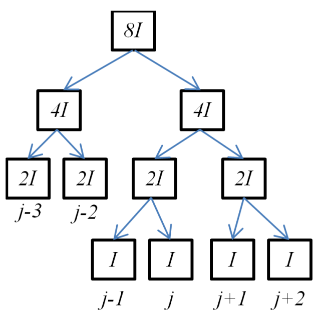


Figure 1: *Illustration of finding the similar VPLs (neighboring columns). The squares denotes the VPL node with intensity in it. The cut nodes in the current lightcut are $j-3$, $j-2$, $j-1$, j, $j+1$, $j+2$. By the definition in Eq. 1, we have $\mathcal{N}(j-2) = \{j-3, j-2, j-1, j, j+1, j+2\}$, $\mathcal{N}(j-1) = \{j-2, j-1, j, j+1\}$, $\mathcal{N}(j) = \{j-1, j, j+1, j+2\}$, $\mathcal{N}(j+1) = \{j-1, j, j+1, j+2\}$.*

1.2 Splitting Threshold in Light Clustering

The the splitting threshold, Γ, is computed as

$$\Gamma = \alpha \sum_j \Gamma_j \quad (2)$$

where α is a parameter to control the splitting of light node. Larger α gives higher splitting threshold, leading to less columns in the reduced lighting matrix, which may result in more illumination error. Fig. 2 shows a CornellBox rendered with increasing α values. In this scene, the illumination results under $\alpha = 0.01$ and $\alpha = 0.02$ are both very smooth. $\alpha = 0.04$ shows some blocky artifacts at shadow boundary. For $\alpha = 0.08$ and $\alpha = 0.16$, significant visual artifacts can be noticed near the light, at the top of the sphere and the red wall.

*Corresponding authors: Rui Wang, rwang@cad.zju.edu.cn, Hujun Bao, bao@cad.zju.edu.cn

2 Matrix Separation

The optimization function is

$$\min_{\mathbf{X},\mathbf{Y},\mathbf{Z}} \quad \|P_\Omega(\mathbf{Z})\|_1 \quad (3)$$

$$s.t. \quad P_\Omega(\mathbf{XY} + \mathbf{Z}) = P_\Omega(\mathbf{D}) \quad (4)$$

$$\mathbf{H} = 2\mathbf{Q} \circ \mathbf{Z} - \mathbf{1}, \quad \|\mathbf{H}\|_F^2 = c. \quad (5)$$

By relaxing the first and second constraints, its Lagrangian equation is [Shen et al. 2014]:

$$\begin{aligned} L(\mathbf{X}, \mathbf{Y}, \mathbf{Z}, \mathbf{\Lambda}, \mathbf{\Pi}) = \|\mathbf{Z}\|_1 + \\ \langle \mathbf{\Lambda}, \mathbf{XY} + \mathbf{Z} - \mathbf{D} \rangle + \langle \mathbf{\Pi}, \mathbf{Z} - \mathbf{H} \rangle + \\ \frac{\beta}{2}\|\mathbf{XY} + \mathbf{Z} - \mathbf{D}\|_F^2 + \frac{\beta}{2}\|\mathbf{Z} - \mathbf{H}\|_F^2. \end{aligned} \quad (6)$$

where β is a parameter to weight the constraints.

To solve this optimization, we start from $\mathbf{\Lambda_0} = 0, \mathbf{\Pi_0} = 0$. Usually, the classical alternating direction method doesn't need specific initialization of $\mathbf{X}$ and $\mathbf{Y}$. But, in our practice, we found if we initialize $\mathbf{Y} = \mathbf{V}^T$ from the SVD decomposition in our rank estimation, it will speed up the optimization.

At k-th iteration, we first minimize $\mathbf{X}, \mathbf{Y}, \mathbf{Z}$ in $L(\mathbf{X}, \mathbf{Y}, \mathbf{Z}, \mathbf{\Lambda}, \mathbf{\Pi})$ as:

$$\begin{aligned} \mathbf{X}_{k+1} &= (\mathbf{D} - \mathbf{Z}_k - \mathbf{\Lambda}_k/\beta)\mathbf{Y}_k^\dagger \\ \mathbf{Y}_{k+1} &= \mathbf{X}_{k+1}^\dagger(\mathbf{D} - \mathbf{Z}_k - \mathbf{\Lambda}_k/\beta) \\ \mathbf{Z}_{k+1} &= \text{Shrink}\left(\frac{1}{2}(\mathbf{D} + \mathbf{H}_k - \mathbf{X}_{k+1}\mathbf{Y}_{k+1} - \frac{\mathbf{\Lambda}_k + \mathbf{\Pi}_k}{\beta}), \frac{1}{\beta}\right) \end{aligned}$$

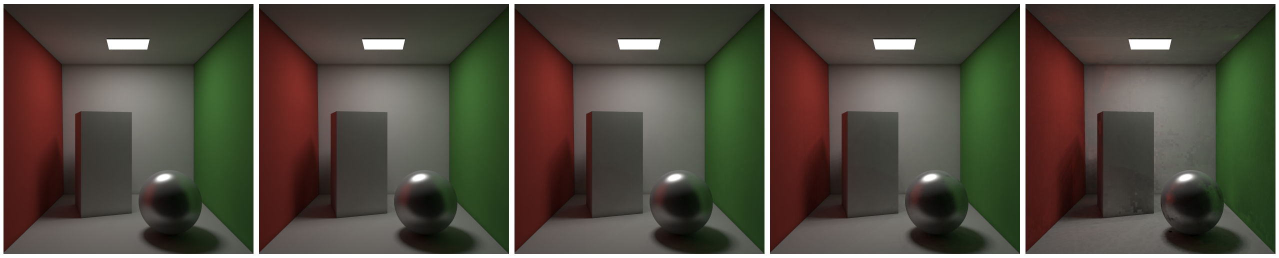
where $\mathbf{Y}_k^\dagger$ represent pseudo-inverse of $\mathbf{Y}$, and Shrink$(\cdot,\cdot)$ is the shrinkage operator defined as Shrink$(x, \gamma) \triangleq \text{sign}(x)\max(|x| - \gamma, 0)$ for a scalar variable x. It also can be applied component-wise to vector or matrix [Hale et al. 2008]. Then we compute $\mathbf{H}$ as:

$$\begin{aligned} \mathbf{H}_{k+1} &= \mathbf{Z}_{k+1} + \mathbf{\Pi}_k/\beta \\ P_\Omega(\mathbf{H}_{k+1}) &= 0, \quad \gamma = \sqrt{\frac{c}{\|2\mathbf{Q} \circ \mathbf{H}_{k+1} - \mathbf{1}\|_F^2}} \\ \mathbf{H}_{k+1,ij} &= \gamma(\mathbf{H}_{k+1,ij} - \frac{1}{2\mathbf{Q}_{ij}}) + \frac{1}{2\mathbf{Q}_{ij}} \end{aligned}$$

Once having new $\mathbf{X}, \mathbf{Y}, \mathbf{Z}, \mathbf{H}$, we compute $\mathbf{\Lambda}, \mathbf{\Pi}$ to finish this iteration as:

$$\begin{aligned} \mathbf{\Lambda}_{k+1} &= \mathbf{\Lambda}_k + \beta(\mathbf{Z}_{k+1} + \mathbf{X}_{k+1}\mathbf{Y}_{k+1} - \mathbf{D}) \\ \mathbf{\Pi}_{k+1} &= \mathbf{\Pi}_k + \beta(\mathbf{Z}_{k+1} - \mathbf{H}_{k+1}) \end{aligned}$$

Here $\beta = 500/\|D\|_F$ is the constraint weight.



(a) $\alpha = 0.01$ **(b)** $\alpha = 0.02$ **(c)** $\alpha = 0.04$ **(d)** $\alpha = 0.08$ **(e)** $\alpha = 0.16$

Figure 2: *CornellBox rendered with different* α.

3 More Results

3.1 Matrix Prediction and Separation

More results generated by partial steps of our matrix recovery algorithm are shown in Fig. 3. We show the result directly generated by matrix prediction, i.e. skipping the matrix separation, and a final result generated by all steps. Comparing the rendered image and the error image, it can be seen that the separation step further improve the rendering quality by separating errors and compensating corrupted values.

3.2 Comparison of Equal Quality

We compare our method with multidimensional Lightcut [Walter et al. 2006] and Lightslice [Ou and Pellacini 2011] to produce equal quality results, shown in Fig. 4. Besides the rendered images, we also show the actual required visibility samples per pixel. In multidimensional Lightcut and Lightslice, the required visibility samples is the final cut size. However, in our method, we only require a small number of visibility samples. Note that Lightslice [Ou and Pellacini 2011] uses a fixed number of columns, thus the actual visibility sample number is approximately a constant for all pixel. They are 1145, 1953, 1088 and 1164 for four test scenes. We only visualize the per pixel sampling number of multidimensional Lightcut and ours. As can been seen, the visibility samples required by our method is greatly smaller than those in multidimensional Lightcut.

References

HALE, E. T., YIN, W., AND ZHANG, Y. 2008. Fixed-point continuation for \ell_1-minimization: Methodology and convergence. *SIAM Journal on Optimization 19*, 3, 1107–1130.

OU, J., AND PELLACINI, F. 2011. Lightslice: matrix slice sampling for the many-lights problem. *ACM Trans. Graph. 30*, 6 (Dec.), 179:1–179:8.

SHEN, Y., WEN, Z., AND ZHANG, Y. 2014. Augmented lagrangian alternating direction method for matrix separation based on low-rank factorization. *Optimization Methods and Software 29*, 2, 239–263.

WALTER, B., ARBREE, A., BALA, K., AND GREENBERG, D. P. 2006. Multidimensional lightcuts. In *ACM Transactions on Graphics (TOG)*, vol. 25, ACM, 1081–1088.

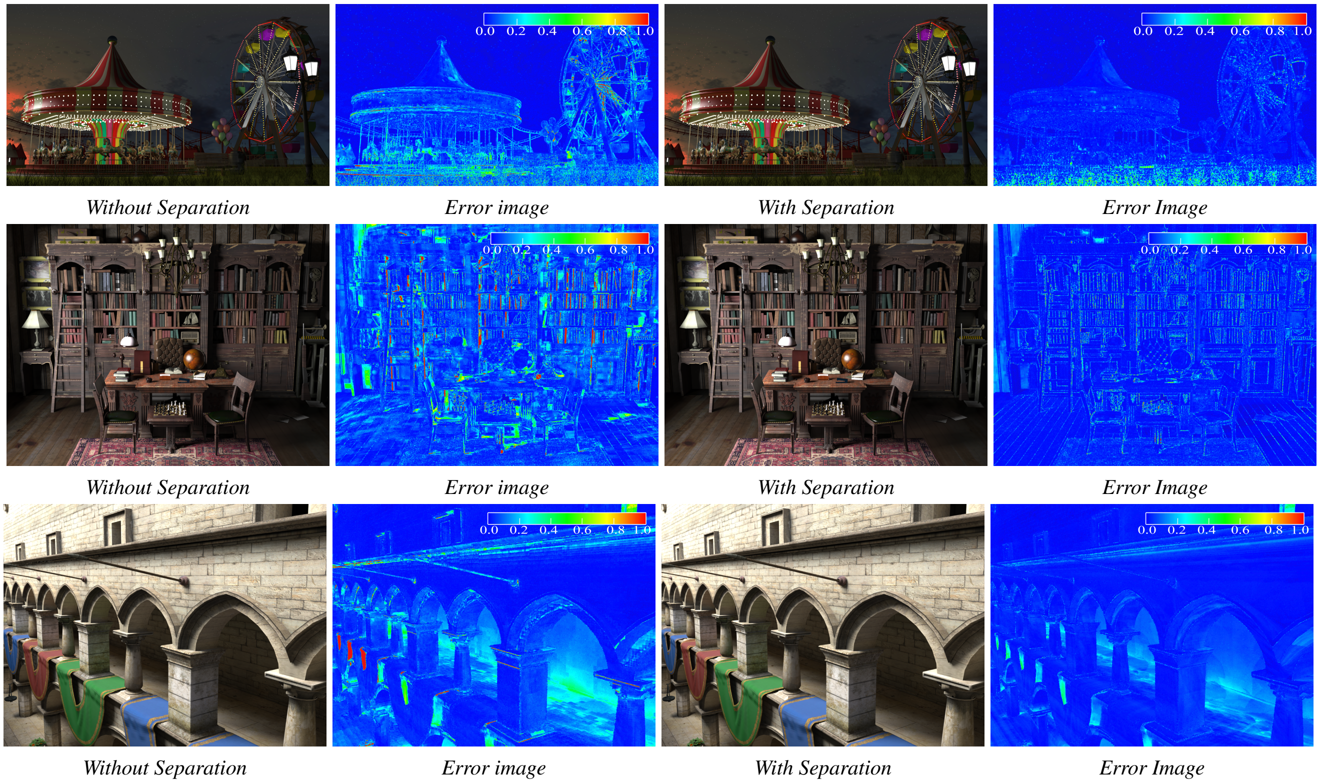


Figure 3: *Results with and without matrix separation.*

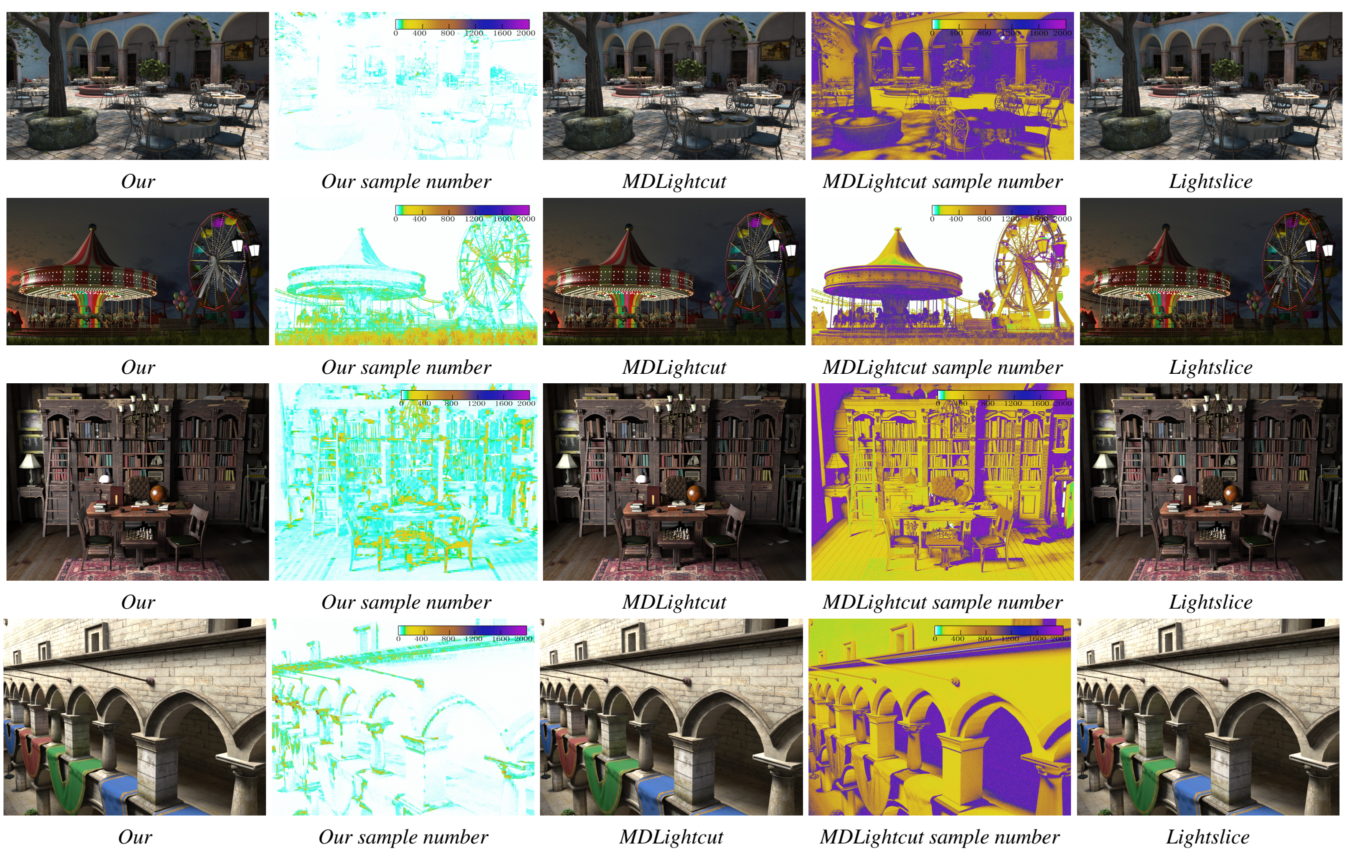


Figure 4: *Results comparison of equal quality.*