AN EFFICIENT PLUGIN FOR REPRESENTING HETEROGENEOUS TRANSLUCENT MATERIALS

Sermet Önel[1] – Murat Kurt[2] – Aydın Öztürk[3]

[1] Department of Computer Engineering, Yaşar University

[2] International Computer Institute, Ege University

[3] Department of Computer Engineering, İzmir University

OUTLINE

- **Introduction to subsurface scattering and translucent material representation**
  - Efficient models on homogeneous translucent materials
  - Efficient models on heterogeneous translucent materials

- A better representation for heterogeneous translucent materials
  - Singular Value Decomposition (SVD) approach

- The integration plugin
  - Details about the software
  - Details about importing the scene parameters

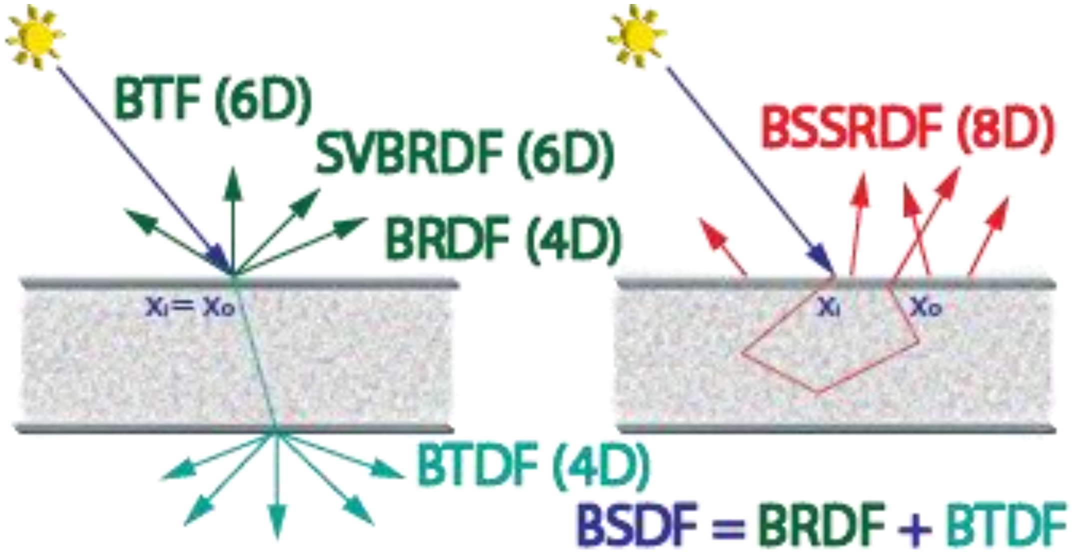
- Results and discussion on the future work

INTRODUCTION TO SUBSURFACE SCATTERING

- Subsurface scattering: a more realistic rendering mechanism
- Bidirectional Surface Scattering Reflectance Distribution Function (BSSRDF) [Nicodemus et al., 1977]
- BSSRDF is an 8D function : expensive to compute
- The function relates to the outgoing radiance at one point to the incident flux at another

$$dL_o(x_o, \vec{\omega}_o) = S(x_i, \vec{\omega}_i; x_o, \vec{\omega}_o) d\varphi(x_i, \vec{\omega}_i)$$

INTRODUCTION TO SUBSURFACE SCATTERING



The functions that define the interaction between light and materials [Kurt, 2014]

SUBSURFACE SCATTERING MODELS

- **Homogeneous Translucent Materials:**
  - Jensen (2001) proposed a dipole diffusion approximation model for homogeneous translucent material representation
  - Mertens (2005) modelled human skin using an interactive model to achieve local subsurface scattering
  - Donner and Jensen (2005) proposed multiple dipoles for multi-layered translucent materials
  - Jakob et al. (2010) extended dipole model using an anisotropic approach
  - Jimenez et al. (2010a; 2010b) modelled human skin and added representation of psychological and emotional states

SUBSURFACE SCATTERING MODELS

- **Heterogeneous Translucent Materials:**
  - Goesele et al. (2004) offered a compact model depending on underlying geometry
  - Tong et al. (2005) modelled quasi-homogeneous materials
  - Peers et al.'s (2006) employed Non-Negative Matrix Factorization (NMF) algorithm to represent heterogeneous translucent materials
  - Song et al.'s (2009) SubEdit representation allowed interactive editing and rendering
  - Kurt et al. (2013) and Kurt (2014) replaced NMF with Tucker Factorization and Singular Value Decomposition (SVD), respectively.

OUTLINE

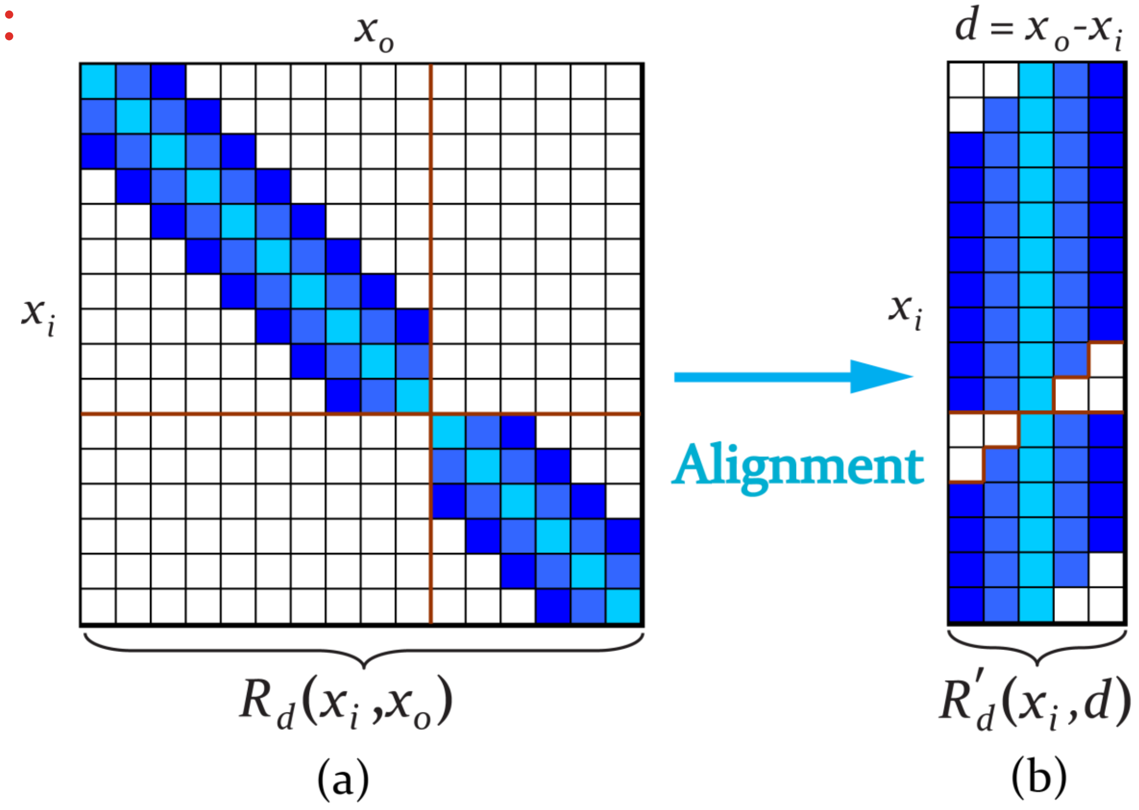
- Introduction to subsurface scattering and translucent material representation
  - Efficient models on homogeneous translucent materials
  - Efficient models on heterogeneous translucent materials
- **A better representation for heterogeneous translucent materials**
  - Singular Value Decomposition (SVD) approach
- The integration plugin
  - Details about the software
  - Details about importing the scene parameters
- Results and discussion on the future work

SVD APPROACH

- **Pre-processing:**
  - Similar to Peers et al.'s (2006) NMF factorization and Kurt et al.'s (2013) Tucker factorization methods
  - Therefore, subsurface scattering data can be represented through a tensor model. 2D matrix is considered as a second degree tensor in this approach
  - In SVD approach, an *MxN* matrix is defined as the product of a *U* matrix with dimensions *MxK* and a matrix *V* with dimensions *KxN* and a core tensor with dimensions *KxK*.
  - However, in this approach K is chosen to be 1, so the matrices *U* and *V* become *Mx1* and *1xN*, respectively, the tensor becomes a scalar

SVD APPROACH

- **Pre-processing:**



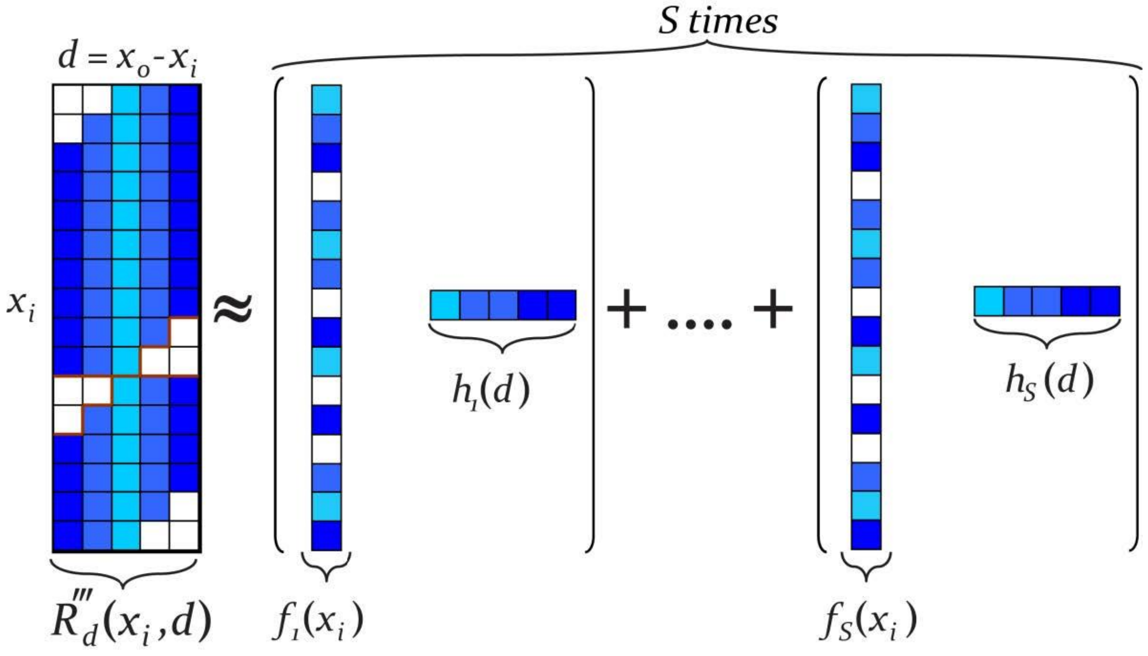
[Kurt, 2014]

SVD APPROACH

- **Pre-processing:**
  - Another consideration in SVD approach is making the data stay at positive values which leads to physically correct values
  - Such an approximation is done using another transformation on R'_d
  - Then we get $R'''_d(x_i, d) = \ln(\frac{R'_d(x_i,d)}{A} + B)$
  - R'''_d is factorized using SVD approach and error terms are modelled using Bilgili et al.'s approach (2011)

SVD APPROACH

- **Resulting Subsurface Scattering Model:**



[Kurt, 2014]

SVD APPROACH

- **Subsurface Scattering Representation:**
  - After interpolation:

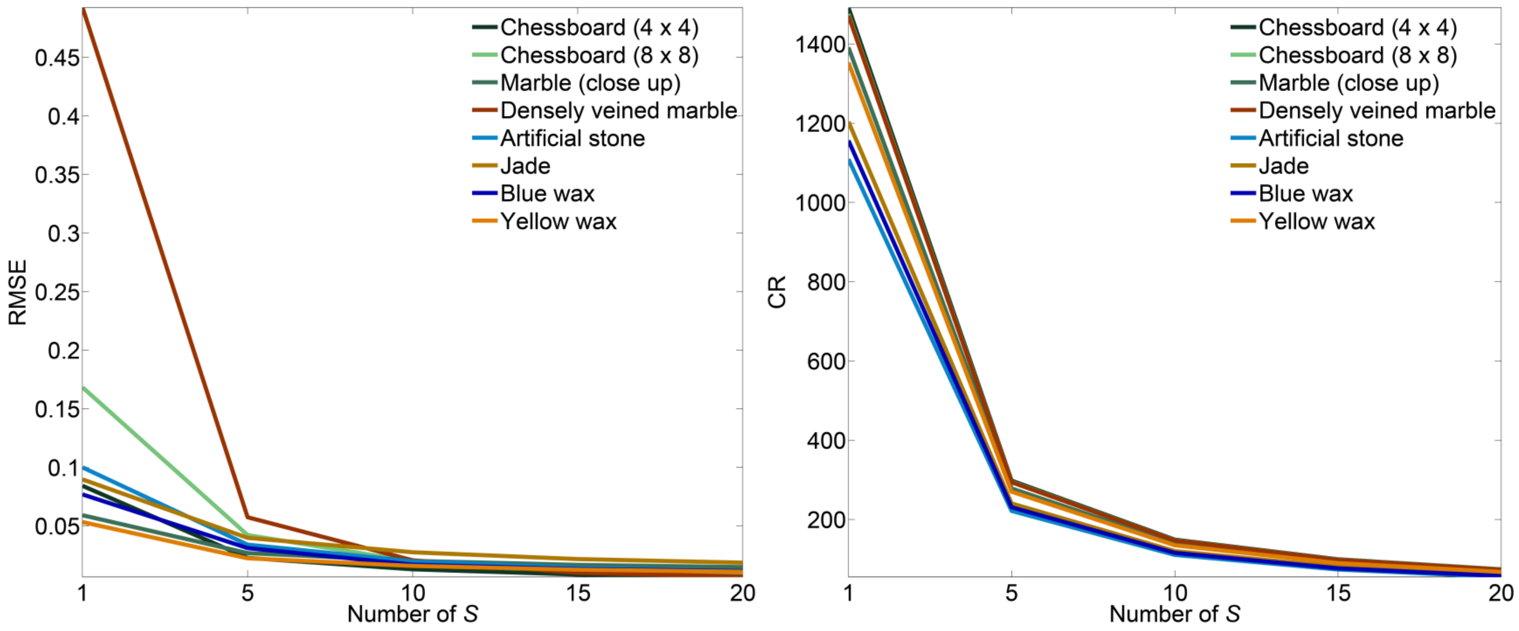
$$R_{dr}(x_i, x_o) \approx A * \exp\left(R'''_{dr}(x_i, d)\right) - A * B$$
$$R_{dg}(x_i, x_o) \approx A * \exp\left(R'''_{dg}(x_i, d)\right) - A * B$$
$$R_{db}(x_i, x_o) \approx A * \exp\left(R'''_{db}(x_i, d)\right) - A * B$$

[Kurt, 2014]

  - As the study shows, SVD approach gives better performance on 2D matrices where Tucker factorization is better on higher order tensors. Therefore, SVD approach is implemented in this thesis study.

SVD APPROACH

- **Analysis:**



[Kurt, 2014]

OUTLINE

- Introduction to subsurface scattering and translucent material representation
  - Efficient models on homogeneous translucent materials
  - Efficient models on heterogeneous translucent materials
- A better representation for heterogeneous translucent materials
  - Singular Value Decomposition (SVD) approach
- **The integration plugin**
  - Details about the software
  - Details about importing the scene parameters
- Results and discussion on the future work

THE INTEGRATION PLUGIN

- The integration plugin is the script that is used to transport the parameters of the scene that is created in Blender 3D modelling tool to the Mitsuba Renderer

- The Blender 3D Modelling Tool is an open-source SW, which is widely used. It is also important that Blender includes lots of rendering features

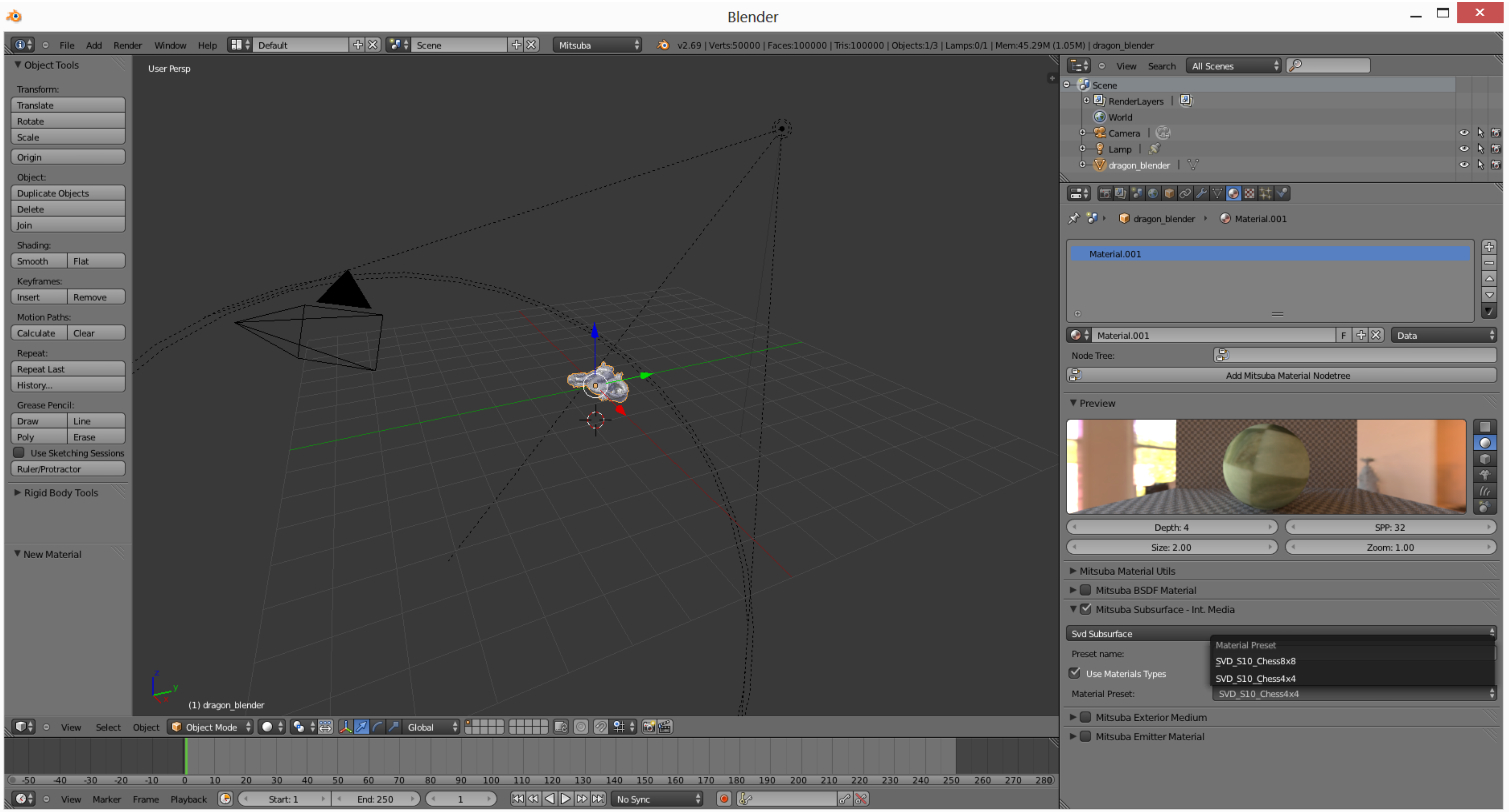
- Mitsuba renderer is also an open-source project. The renderer's main characteristics are its efficiency, robustness, scalability, usability and consisting of heavy optimizations targeted on CPU

- The script is implemented in Python. Since Python has bindings for C++, the transportation of the parameters need no extra work

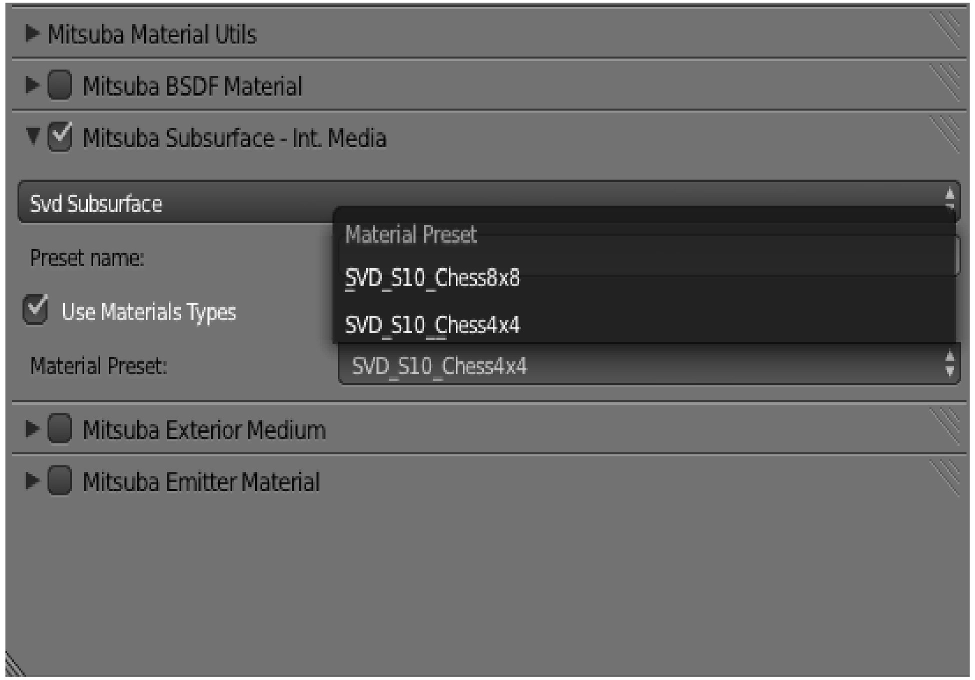
THE INTEGRATION PLUGIN

- The plugin is implemented by modifying the already available integration plugin of Styperek and Juhe’s (2011)
- The implementation can be classified into two groups:
  - Firstly, the dipole approach is improved by adding the default material types with appropriate absorption and scattering coefficient values
  - Secondly, SVD approach is implemented in C++. Since the EXE file is defined on the plugin, the methods of SVD became available. (mitsuba_sss_svd) class is added to the script for making the SVD approach available for heterogeneous materials on Blender project
- The script stores the parameters in an XML file. The renderer reads this file to apply subsurface scattering on the sample material

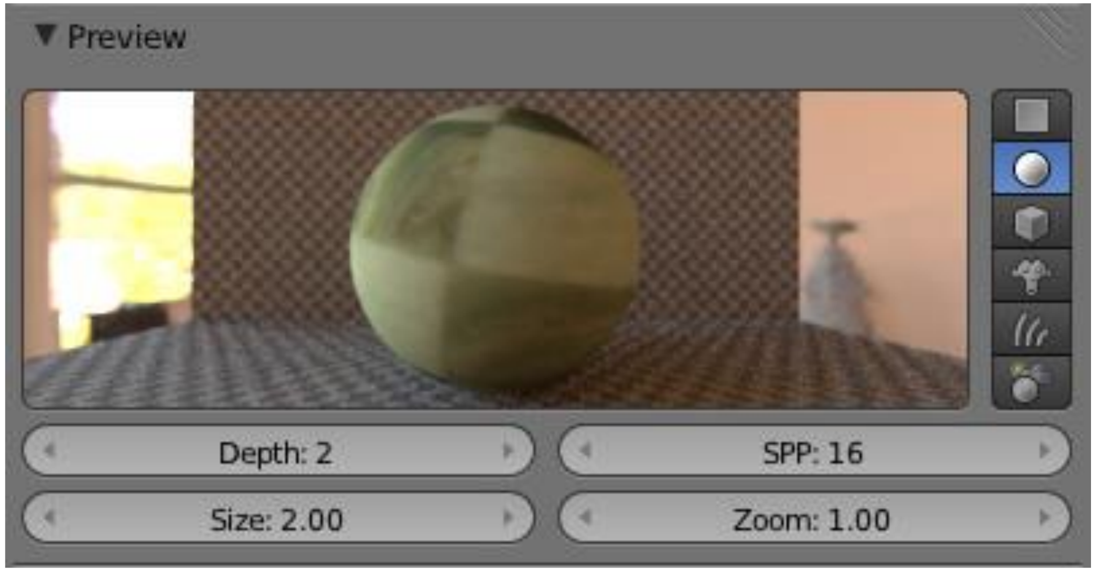
THE INTEGRATION PLUGIN



THE INTEGRATION PLUGIN



The GUI of the SVD Subsurface class on the integration plugin



Preview scene of the plugin. Chessboard 4x4 material sample is applied on the object

THE INTEGRATION PLUGIN

```
                <rfilter type="gaussian"/>
            </film>
        </sensor>

        <emitter type="spot">
            <transform name="toWorld">
                <!-- Orient the light so that points from (1, 1, 1) towards (1, 2, 1) -->

                <lookat target="0, 0, 0" origin="0, 0, 7" up="0.0336821, 0.00834246, 0.999398"/>
            </transform>
            <float name="cutoffAngle" value="200"/>
            <spectrum name="intensity" value="100"/>
        </emitter>

        <shape type="obj">
            <string name="filename" value="dragon_blender.obj"/>
            <transform name="toWorld">
                <scale value="4"/>
                <rotate y="1" angle="0"/>
                <rotate z="1" angle="45"/>
                <translate z="1"/>
            </transform>

            <subsurface type="svd">
                <string name="material" value="SVD_S10_Div100_Add1_for_Rd_Chess4x4_v1(277x277x39x39)"/>
            </subsurface>
        </shape>

        <shape type="obj">
            <!-- Shiny floor -->
            <string name="filename" value="plane.obj"/>

            <bsdf type="diffuse">
                <rgb name="diffuseReflectance" value="100, 100, 100"/>
            </bsdf>
            <transform name="toWorld">
                <translate y="-1.8"/>
                <rotate x="1" angle="90"/>
                <rotate z="1" angle="45"/>
            </transform>
        </shape>
</scene>
```

XML tags are used for transporting the effects to the renderer

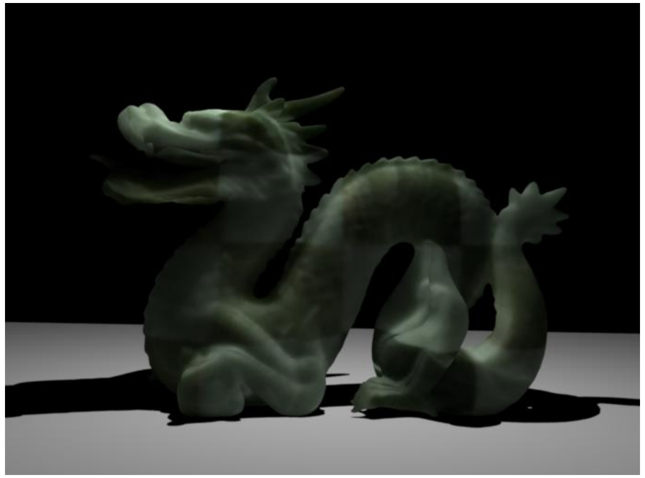
OUTLINE

- Introduction to subsurface scattering and translucent material representation
  - Efficient models on homogeneous translucent materials
  - Efficient models on heterogeneous translucent materials
- A better representation for heterogeneous translucent materials
  - Singular Value Decomposition (SVD) approach
- The integration plugin
  - Details about the software
  - Details about importing the scene parameters
- **Results and discussion on the future work**

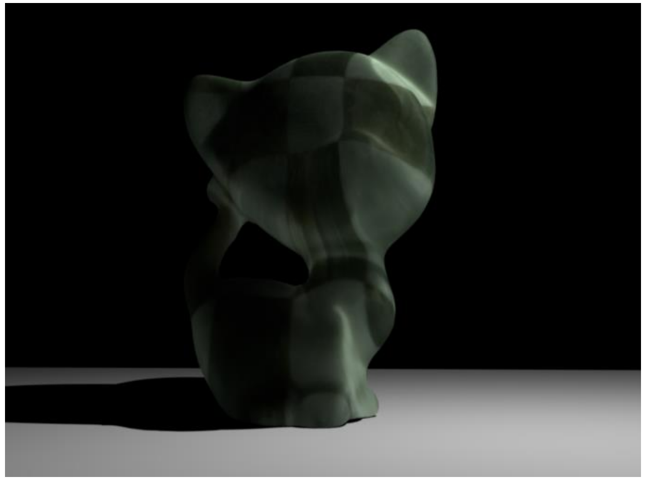
RESULTS

- In order to test the plugin's efficiency, different object samples are rendered and the memory usage and time used for the operation is compared with doing it directly using the renderer
- The Hardware: i7-3630QM processor with 8GB RAM and NVIDIA GTX660M/2GB GDDR5.
- The test objects: Dragon object and kitten object
- Test Material Types: Chessboard (4x4) and Chessboard (8x8)

RESULTS

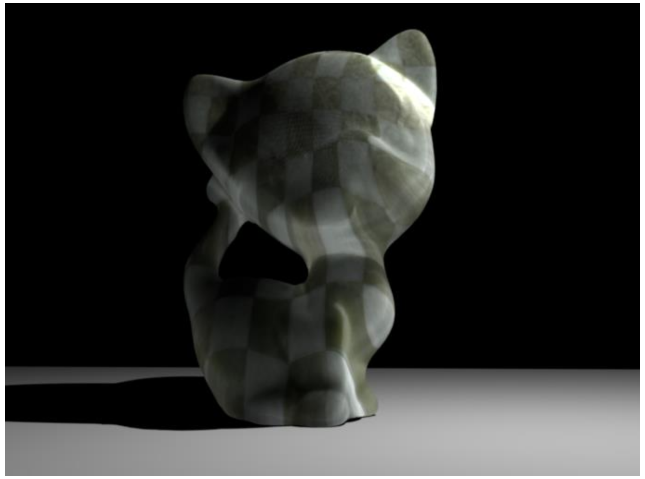


Chessboard (4 x 4), dragon object
Rendering time: 20 min., Storage size: 36.1 MB



Chessboard (4 x 4), kitten object
Rendering time: 16.157 min., Storage size: 32.66 MB

Chessboard (8 x 8),dragon object
Rendering time: 20.9 min., Storage size: 38.42 MB



Chessboard (8 x 8), kitten object
Rendering time: 17.36 min., Storage size: 34.72 MB

FUTURE WORK AND DISCUSSION

- In this study several subsurface scattering models are analyzed and SVD-based representation for heterogeneous translucent materials was offered as a plugin on a software modelling tool
- The experiments on the plugin show that the plugin works efficiently by causing no delay or no excessive memory usage
- Although SVD-based approach is implemented and tested, other factorization methods may be added to the plugin as a future study
- The plugin is dependent to Blender Modelling Tool and Mitsuba Renderer. The availability of the plugin on other platforms would decrease its platform dependence

REFERENCES

[1] H. W. Jensen, S. R. Marschner, M. Levoy, and P. Hanrahan, "A practical model for subsurface light transport," in *Proc. SIGGRAPH '01*, 2001, pp. 511–518.

[2] H. W. Jensen, "Global illumination using photon maps," in *Proceedings of the Eurographics Workshop on Rendering Techniques '96*. London, UK, UK: Springer-Verlag, 1996, pp. 21–30.

[3] M. Kurt, A. Öztürk, and P. Peers, "A compact tucker-based factorization model for heterogeneous subsurface scattering," in *Proceedings of the 11th Theory and Practice of Computer Graphics*, ser. TPCG '13. Bath, United Kingdom: Eurographics Association, 2013, pp. 85–92.

[4] E. d'Eon and G. Irving, "A quantized-diffusion model for rendering translucent materials," vol. 30, no. 4, pp. 56:1–56:14, Jul. 2011, (Proc. SIGGRAPH '11).

[5] M. Goesele, H. P. A. Lensch, J. Lang, C. Fuchs, and H.-P. Seidel, "DISCO: acquisition of translucent objects," vol. 23, no. 3, pp. 835–844, Aug. 2004, (Proc. SIGGRAPH '04).

[6] X. Tong, J. Wang, S. Lin, B. Guo, and H.-Y. Shum, "Modeling and rendering of quasi-homogeneous materials," vol. 24, no. 3, pp. 1054–1061, Jul. 2005, (Proc. SIGGRAPH '05).

[7] Y. Song, X. Tong, F. Pellacini, and P. Peers, "Subedit: a representation for editing measured heterogeneous subsurface scattering," vol. 28, no. 3, pp. 31:1–31:10, Jul. 2009, (Proc. SIGGRAPH '09).

[8] W. Jakob, "Mitsuba renderer," 2013, http://www.mitsuba-renderer.org.

[9] B. Styperek and F. Juhe, "The blender plugin," 2011, https://www.mitsuba-renderer.org/plugins.html.

[10] Blender, "Blender foundation," 2003, http://www.blender.org/.

[11] F. E. Nicodemus, J. C. Richmond, J. J. Hsia, I. W. Ginsberg, and T. Limperis, "Geometrical considerations and nomenclature for reflectance," National Bureau of Standards (US), Monograph, Oct. 1977.

[12] W. Jakob, A. Arbree, J. T. Moon, K. Bala, and S. Marschner, "A radiative transfer framework for rendering materials with anisotropic structure," vol. 29, no. 4, pp. 53:1–53:13, Jul. 2010, (Proc. SIGGRAPH '10).

[13] T. Mertens, J. Kautz, P. Bekaert, F. V. Reeth, and H.-P. Seidel, "Efficient rendering of local subsurface scattering," *Computer Graphics Forum*, vol. 24, no. 1, pp. 41–49, 2005.

[14] C. Donner and H. W. Jensen, "Light diffusion in multi-layered translucent materials," vol. 24, no. 3, pp. 1032–1039, Jul. 2005, (Proc. SIGGRAPH '05).

[15] J. Jimenez, D. Whelan, V. Sundstedt, and D. Gutierrez, "Real-time realistic skin translucency," *IEEE Computer Graphics and Applications*, vol. 30, no. 4, pp. 32–41, 2010.

[16] J. Jimenez, T. Scully, N. Barbosa, C. Donner, X. Alvarez, T. Vieira, P. Matts, V. Orvalho, D. Gutierrez, and T. Weyrich, "A practical appearance model for dynamic facial color," vol. 29, no. 6, pp. 141:1–141:10, Dec. 2010, (Proc. SIGGRAPH Asia '10).

REFERENCES

[17] P. Peers, K. vom Berge, W. Matusik, R. Ramamoorthi, J. Lawrence, S. Rusinkiewicz, and P. Dutré, "A compact factored representation of heterogeneous subsurface scattering," vol. 25, no. 3, pp. 746–753, Jul. 2006, (Proc. SIGGRAPH '06).

[18] M. Kurt, "An efficient model for subsurface scattering in translucent materials," Ph.D. dissertation, International Computer Institute, Ege University, Izmir, Turkey, January 2014, 122 pages.

[19] A. Bilgili, A. Öztürk, and M. Kurt, "A general brdf representation based on tensor decomposition," *Computer Graphics Forum*, vol. 30, no. 8, pp. 2427–2439, December 2011.

[20] G. V. Rossum, "Python," 1989, https://www.python.org/.

THANK YOU FOR LISTENING
ANY QUESTIONS?