Photon: A Modular, Research-Oriented Rendering System

Tzu-Chieh Chang, Ming Ouhyoung
National Taiwan University

thrive SIGGRAPH2019
LOS ANGELES • 28 JULY – 1 AUGUST

Introduction

To develop a graphics project with ease and confidence, the reliability and extensibility of the underlying framework are essential. We would like to present a solution named Photon, which is an open-source and cross-platform rendering system written in C++. The goal of our system is to provide a set of building blocks to facilitate the implementation of new rendering algorithms, as well as a unified foundation for comparing the performance of different methods.

Related Work

Existing options such as pbrt-v3 and Mitsuba exist. However, they either focus on education or not being updated for quite some time.
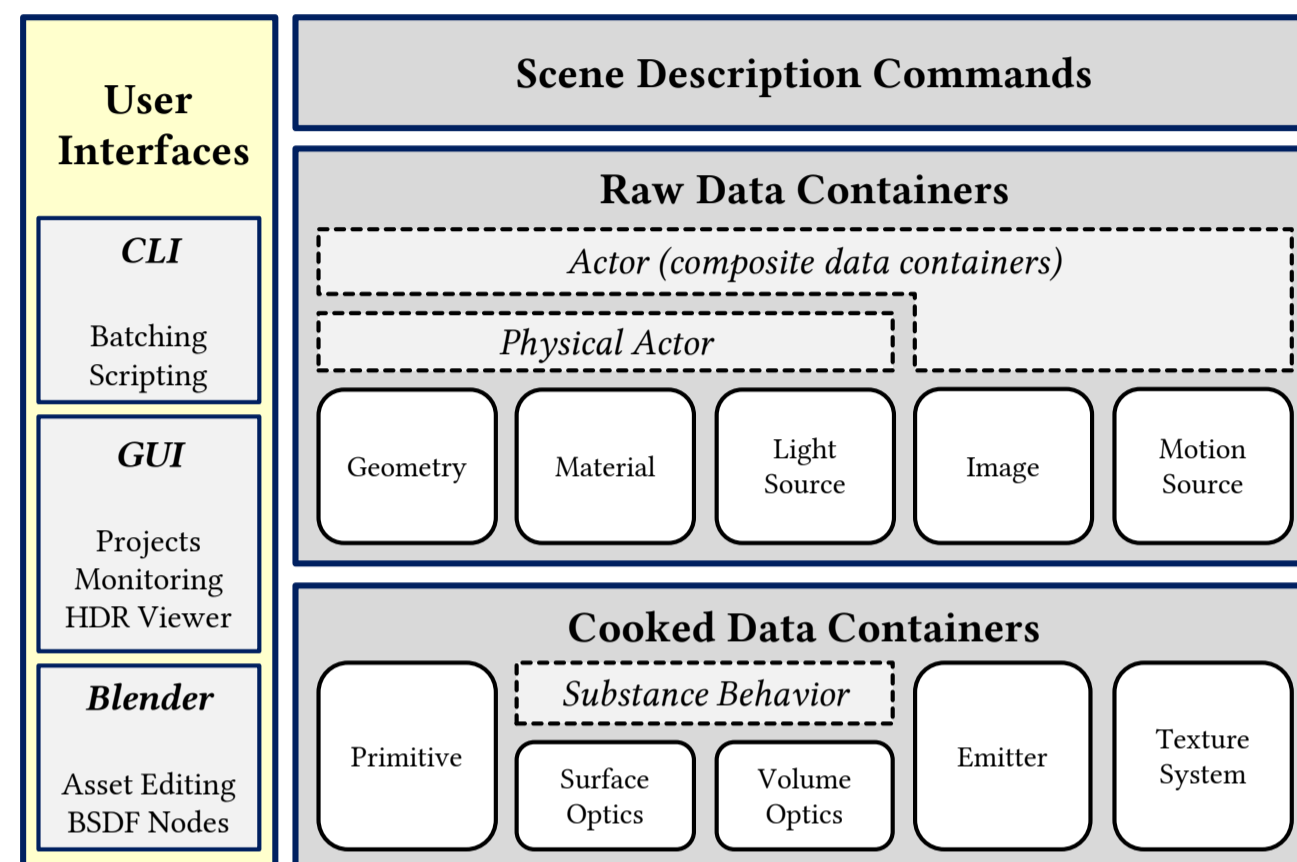
Other projects such as Tungsten and SORT are similar to ours but are weaker in unit test coverage.

Our design is inspired by Microsoft's Component Object Model (COM) and the concept of *content cooking* in Unreal Engine 4.

Our Approach

Our tri-layer architecture helps to simplify logics, and opens an opportunity for project developers to natively alter scene data without intervening lower level modules.

The Rendering System

User Interfaces	Scene Description Commands				
CLI — Batching Scripting	**Raw Data Containers**				
	Actor (composite data containers)				
GUI — Projects Monitoring HDR Viewer	*Physical Actor*				
	Geometry	Material	Light Source	Image	Motion Source
Blender — Asset Editing BSDF Nodes	**Cooked Data Containers**				
	Substance Behavior		Emitter	Texture System	
	Primitive	Surface Optics / Volume Optics			

◻ Architectural Layer   ⬚ Aggregate Module   ▢ Standalone Module

We consider a rendering system as an assembly line that gradually transforms scene description data into 2-D imageries. As a result, each layer in our architecture corresponds to data in a specific state.

The second layer translates raw data (scene description) into cooked data, which is the target format that actually participate in the rendering process.

Results

Three flavors of photon mapping (PM [Jensen 1996], PPM [Hachisuka et al. 2008], SPPM [Hachisuka and Jensen 2009]) are demonstrated.
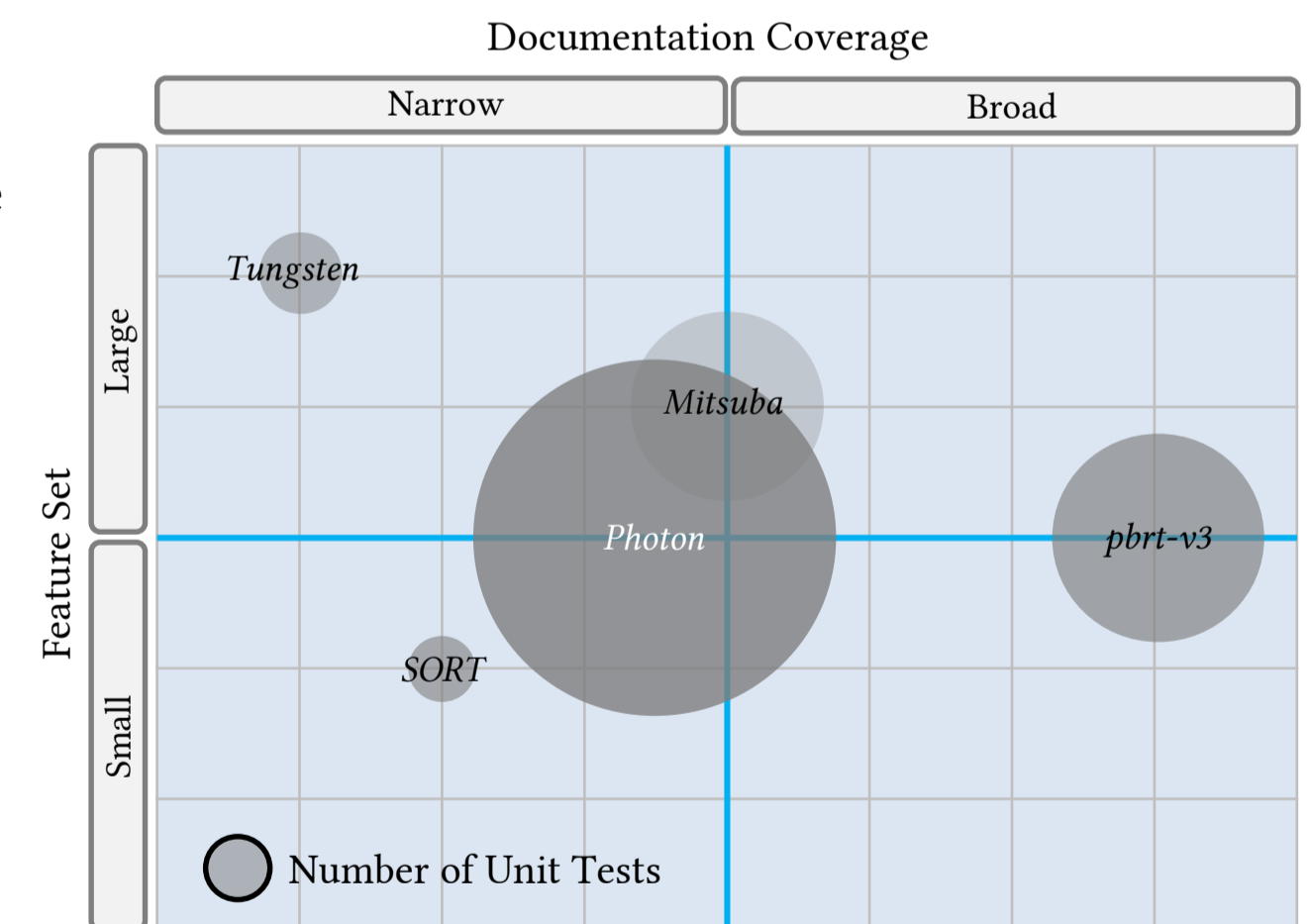
In the comparison, although our system is weaker in document and features, we have a relatively young codebase and is quickly catching up its predecessors.

PM   PPM   SPPM

A general comparison between similar systems. Darker shade indicates higher development activity.

References

Benedikt Bitterli. 2016. Tungsten renderer. https://benedikt-bitterli.me/tungsten.html.
Toshiya Hachisuka and Henrik Wann Jensen. 2009. Stochastic Progressive Photon Mapping. ACM Trans. Graph. 28, 5, Article 141 (Dec. 2009).
Toshiya Hachisuka, Shinji Ogaki, and Henrik Wann Jensen. 2008. Progressive Photon Mapping. ACM Trans. Graph. 27, 5, Article 130 (Dec. 2008).
Henrik Wann Jensen. 1996. Global Illumination Using Photon Maps. In Proceedings of the Eurographics Workshop on Rendering Techniques '96.
Cao Jiayin. 2016. SORT. https://jerrycao1985.github.io/SORT/.
Matt Pharr, Wenzel Jakob, and Greg Humphreys. 2016. Physically Based Rendering: From Theory to Implementation (3rd ed.).
Wenzel Jakob. 2010. Mitsuba renderer. http://www.mitsuba-renderer.org.