

# Cell上のソフトウェアレンダリングの高機能化と高速化

065725F 金城裕 指導教員：河野真治

## 1 Cell上のソフトウェアレンダリング 3 Cerium

我々は、8SPU+2PPUのヘテロCPU構成を持つCell / PS3 上でのソフトウェアレンダリングエンジンを開発中である。これは、Cerium Task Manager 上に実装され、琉球大学情報工学科の3年生の実験で使用されている。PS3のGraphics Engineの仕様は公開されていないために、ソフトウェアレンダリングを実装する必要がある。

ソフトウェアレンダリングは、SceneGraph、Polygon、Span、Z bufferの各段階があり、それぞれの段階で、並列処理を行なう必要がある。現在は、Spanの生成を1SPU上で行ない、Z bufferを複数のSPU上で処理している。さらに、他の段階でも並列処理を行なう必要がある。PPUよりもSPUの方が高速なので、1SPU上で実行するだけでも意味がある。

## 2 ヘテロジニアス・マルチコアプロセッサCell

Cellは、1基の制御系プロセッサコア (PPE:PowerPc Processor Element) と8基の演算系プロセッサコア (SPE:Synergistic Processor Element) で構成される。各プロセッサコアは、EIB (Element Interconnect Bus) と呼ばれる高速なバスで接続されている。また、EIBはメインメモリや外部入出力デバイスとも接続されていて、各プロセッサコアはEIBを経由してデータアクセスをおこなう。

この2種類のCPUをプログラマ自身が用途に合わせて適切に使い分けるように考慮する必要がある。

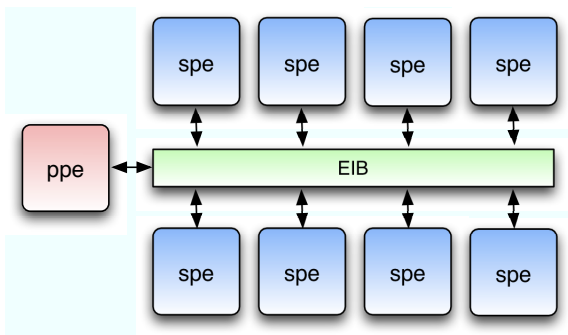


図 1: Cell プロセッサの構成

当研究室ではCeriumと呼ばれるゲーム開発フレームワークがあり、以下の3つの要素から構成されている。

- SceneGraph
- Rendering Engine
- Task Manager

Ceriumは独自にRenderingEngineを持つ。ゲーム中のオブジェクトの振る舞いやルールはSceneGraphで管理し、それらの動きやレンダリングの処理を動的にSPEに割り振るカーネルとして、TaskMnagerが用いられる。TaskManagerは、Taskと呼ばれる、分割された各プログラムを管理する。Taskの単位はサブルーチンまたは関数とし、Task同士の依存関係を考慮しながら実行していく。現在Ceriumはlinux,macosx上で動作し、コンパイル方法によってCellのspeを使うかどうかを選択できる。

## 4 RenderingEngine

RenderingEngineでは、SceneGraphから、実際に表示するポリゴンの抽出、ポリゴンからSpanの生成、SpanにRGBをマッピングし描画する部分と3つに分ける事ができる。ここでいうSpanとは、ポリゴンに対するある特定のY座標に関するデータを抜き出したものである。

### 4.1 光源

RenderingEngineで、未実装だった光源の計算を実装した。各オブジェクトには自身の座標や親子関係などの情報を持っており、その中に法線がある。法線と光のベクトルとの内積をrgbにかけ算することにより光の計算を行っている。以下に光源の計算をした画像を示す。

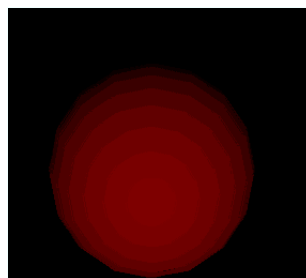


図 2: 光源計算をした描画画像

## 4.2 SceneGraph から Span を作るまで

Cerium では SceneGraph から Rendering に至るまでに以下で示すような流れがある。

1. SceneGraph から、SPE の LS に収まるサイズの SceneGraphPack を生成する。
2. SceneGraph が持つ情報から、Polygon の座標を計算し、PolygonPack を生成する。
3. PolygonPack から、同じ Y 座標を持つ線分の集合 SpanPack を生成する。
4. Span に対応する、描画に必要な Texture を SpanPack に付与する。
5. SpanPack と Texture を Z Buffer を用いて描画する

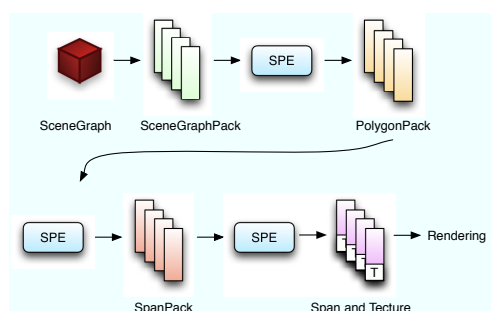


図 3: SceneGraph から Rendering するまでの流れ

以上の工程でそれぞれ SPE に Task を渡し、処理を行っているが、Task は一つにまとめられて一基の SPE で実行されている。

## 5 現状とこれからの課題

### 5.1 TaskManager の例題

Cerium の TaskManager の勉強として TaskManger を用いた WordCount を実装した。input された file を mmap でメモリにマッピングし、その data を分割。各 SPE に、WordCount の Task と分割された data を渡す。SPE は渡された data を wordcount し、指定された output 領域に書き込む。全ての wordcount が実行し終わるとその結果を ppe 側で集計し、結果を出力する。

### 5.2 Cell 上での光源の実装

RenderingEngine の機能として光源の計算を実装した。現在、光源は ppe だけを使った fifo 形式のみに実装されているので、これから Cell の spe 上での実装をしなければならない。

## 5.3 Rendering 工程の並列化

SceneGraph から Span を生成するまでの各段階で Task を分割、複数の SPE に割り振り並列に処理を行うように実装する。

- ・ Polygon → Span の並列化

ここで生成された Span は同じ X, Y 領域でまとめて、次の SPU task に渡す必要がある。バラバラな領域の Span を渡すと幾つかの Texture を SPE の LS へローディングしなければならない、DMA 転送待ちが多く発生してしまう。

## 5.4 Clipping

描画対象のオブジェクトが重なり描画する必要のない部分が生じることがある。描画する必要のない部分を描画の早い段階で明確にし、描画対象から削除する必要がある。

## 5.5 背景の特別扱い

オブジェクトは 1dot ずつ計算し描画しているが、背景はあらかじめ描画するものが決まっているので 1dot ずつ計算する必要がないので、背景専用のルーチンを設ける。

## 5.6 描画対象がない空間の扱い

Cerium の RenderingEngine は描画する対象がない部分も毎フレーム描画計算をしている。それでは計算する必要のない部分も計算してしまい、無駄な時間が生じる。そこで、描画する対象がない部分の判定を行い、zero clear DMA などを用いて計算の手間を省く必要がある。

## 参考文献

- [1] 宮國渡 ”Implementation of Fine-grain Task Manager for Cell” 平成 20 年度 学位論文 (修士)
- [2] 多賀野海人 ”並列プログラミングを用いたゲームフレームワークの設計と実装” 2008 年 卒業研究中間報告資料
- [3] fixstars: <http://cell.fixstars.com/ps3linux/index.php/> メインページ
- [4] OpenCL: <http://www.khronos.org/opencl/>