RenderMan Interface を用いた高画質レンダリング環境の構築

2C－8

土井章男、伊藤貴之
日本アイ・ビー・エム株式会社 東京基礎研究所

はじめに

現在、コンピュータグラフィックス（CG）におけるアプリケーション・プログラム・インターフェース（API）は、OpenGL, GL, PEX, PHIGS 等が提案され、アプリケーション・プログラマに提供されている。しかしながら、レイトレーシングやラジオシティ法といった高品質な画像を生成するためのアプリケーション・インターフェースは、十分な標準化が進んでいないと思われる。我々は、レンダリングに適した API として、Pixar 社により提案されている RenderMan[1] Interface [1, 2] を取り上げ、そのインターフェースを拡張して、レイトレーシングおよびラジオシティ法をインプリメントした。本論文では、そのインプリメント方式、拡張性、汎用性および使い易さについて報告する。

RenderMan Interface

RenderMan Interface とは、物体やシーン、光源やカメラなどを記述する方法を定義したものであり、モデリングプログラムとレンダリングプログラムを橋渡しする標準インターフェースである。RenderMan Interface を用いることにより、ユーザはモデリングシステムから生成された幾何データから、容易に高品質な画像を生成することが可能になる。RenderMan Interface には、この目的のため、各 API に対応した出力ファイルフォーマットとして、RIB(RenderMan Interface Bytestream) を定義しており、幾何データおよび属性データのデータベース化も容易である。

IBM/RenderMan ツールキット

東京基礎研究所と IBM T. J. Watson Research で共同研究している汎用レンダリングツールキットである。このツールキットで使用できるレンダラは、

- ソフトウェア Z バッファ法
- ハードウェア Z バッファ法 (GL/OpenGL 版)
- レイトレーシング法
- ラジオシティ法
- 対話型レンダリング環境（DesignScene）

である。各レンダラは、すべて同じ RenderMan Interface API から構成されており、ユーザは効率、速度、画質などから最適のレンダラを選ぶことが出来る。対応している OS は、UNIX および Windows/NT である。使用方法として、モデリングプログラムやユーザプログラムから本ライブラリーを直接呼び出す方式と作成された RIB（RenderMan Interface Bytestream, or RIB プロトコール）ファイルを直接、レンダリングする方式がある。よって、RIB フォーマットで幾何データを出力できるモデラーと本システムを組み合わせたり、ユーザ自身のモデリングシステムに直接組み込んで、高品質なレンダリング機能を追加することが出来る。

各種モデリングシステムや作成された RIB ファイル、および各種データファイルから変換された RIB ファイルは、一般に幾何データおよび色属性である。より高品質な画像を生成するためには、対話的に RIB ファイルを編集出来る環境が必要不可欠である。このため、我々は、DesignScene と呼ぶ対話処理システムを構築して、プログラムを書かなくても画像を作成できる環境をユーザに提供している。図 1に全体のシステム構成を示す。

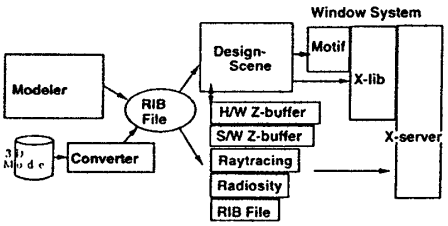

図 1: システム構成

実装方法

RenderMan Interface に定義されていない各種レンダラに必要なパラメータ（たとえば、ラジオシティ法の場合、初期メッシュサイズ、収束回数、適合型メッシュ生成のための閾値等）は、RiAttribute() と RiOption() を用いて、すべて、トークンおよびパラメータ対として、レンダラに受け渡される。これらの初期値は、あらかじめレンダラに定義している２つのテーブル（Attribute Table と Option Table) に記述している。この方式を用いるとそのトークンをサポートしていないレンダラが適用された場合は、そのトークンのみを無視するので、

A High-Quality Rendering System Using RenderMan Interface
Akio Doi and Takayuki Itoh
Tokyo Research Laboratory, IBM Japan, Ltd.
Tel: 0462(73)4925, E-mail: itot@trl.ibm.co.jp
1) RenderMan is a registered trademark of Pixar

RIB ファイルの重複保持やプログラムの修正を最小限に抑えることが出来る。

```
RiBegin();
RiOption("radiosity",RI_RCYCLE,50,
                     RI_RTOLE,0.005);
      /* 収束回数の定義、閾値              */
....
RiWorldBegin();
RiAttribute("radiosity",RI_PMESH,0.5,
                        RI_EMESH,0.25);
      /* 初期パッチおよびエレメントの設定 */
RiPolygon(4,RI_P,(RtPointer)face,RI_NULL);
      /* ポリゴンの定義                   */
....
WorldEnd();
```

図 2: プログラム例

レンダラの実行結果は、RiFrameEnd() あるいは、RiEnd() の呼びだし後、ウィンドウ画面に表示されるようになっている。図2は、トークンによるパラメータの受渡しのサンプルプログラムであり、図2は、そのプログラムに対応する出力 RIB ファイルである。

```
##RenderMan RIB-Structure 1.1
##Scene Untitled
##creator IBM DesignScene 1.0
Option "radiosity" "cycle"[50] "tole"[0.03]
....
WorldBegin
Attribute "radiosity" "pmesh"[5.0] "emesh"[1.5]
Polygon "P" [ 2.0 2.0 0.0 2.0 -2.0 0...]
....
WorldEnd
```

図 3: RIB ファイル

評価

RIB ファイルをサポートしているモデラより形状を入力して、そのRIB ファイルから画像生成を行なった。各種レンダラ間のパラメータの受渡しは、トークンの受渡しで行なうため、RIB ファイルの重複を避けることができた。また、各レンダラの API を統一したので、アプリケーションからダイナミック・ローディング (Dynamic Loading) により、各レンダラを呼ぶことが出来、アプリケーションプログラムを書く負荷を軽減出来た。図4 は、本ライブリによる出力例である。図5は、DesignScene による対話編集画面である。

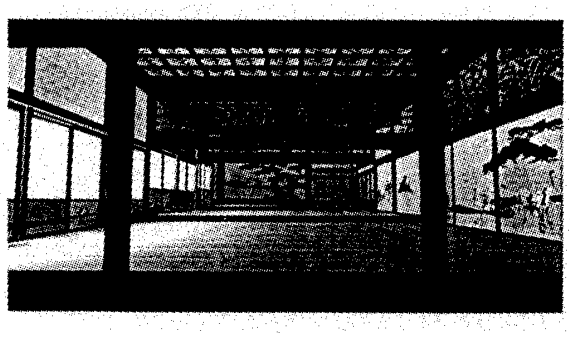

図 4: ラジオシティ法によるレンダリング

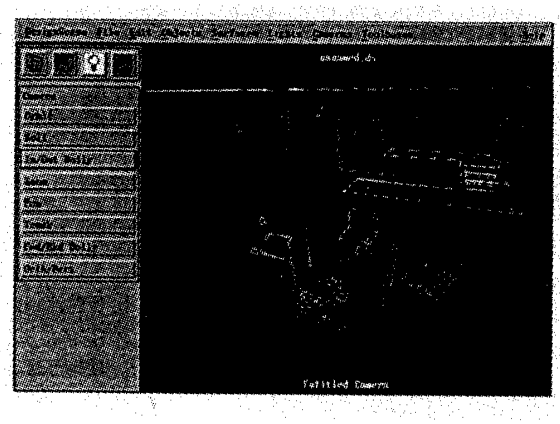

図 5: DesignScene 対話編集画面

おわりに

従来、個別に行なわれてきた各種レンダラの開発を RenderMan Interface を用いて、各レンダラを統一的に開発することができた。RenderMan Interface を用いるユーザ間で、広範囲のデータの受渡しが可能になった。また、アプリケーションプログラムとレンダラを分離することにより、本ライブラリを用いると高画質のレンダリングシステムの容易な構築が可能になった。今後、RenderMan Interface がより標準化された API にするためには、レンダラがサポートするトークンの名前や型の統一、対話機能の拡張、RenderMan Interface を使用するユーザの拡大等を今後、十分行なっていく必要があると思われる。

参考文献

[1] The RenderMan Companion A Programmer's Guide to Realistic Computer Graphics, Steve Upstill, Addison-Wesley.

[2] The RenderMan Interface, Version 3.1, Pixar