CG ソフトを利用した初心者向けプログラミング教育の試み

倉田 和夫 Kazuo KURATA

島森 功 Isao SHIMAMORI

概要: 本稿は、CG、Web デザイン、映像編集などコンピュータを使うデジタルデザイン分野において共通するスキルとしてプログラミング能力が要求されるようになってきていることから、情報系の短期大学でデザインを専攻する学生へのプログラミング入門教育の新しい試みとして CG ソフト POV-Ray を使ってプログラミングの思考法を理解させる教育事例を報告する。

キーワード: CG／造形教育

1. はじめに

あらゆるデザインの分野において制作プロセスにコンピュータが使われるようになると、デザイナーにもプログラミングセンスが求められるようになってきている。筆者の所属する短期大学部は情報・ネットワーク学科の中に4つの系を設けそれぞれ特色ある専門教育を行っているが、とりわけメディア系におけるプログラミング教育を担当する筆者らにとって年々プログラミングに対する学生たちの関心が薄くなる現状をなんとか打開したいと考えてきた。本学は企業の OA 化に対応できる簿記会計などの事務処理技術者を養成するマネジメント系 150 名、プログラマを養成する情報サービス系 100 名、CG や DTP、Web デザイン分野のデザイナーやプログラマを養成するメディア系 100 名、ハードウェア技術者を養成するエンジニアリング系 50 名の 4 つ系約 400 名の学生が在籍している。1999 年のカリキュラム改訂時、すべての学生にコンピュータの基礎知識とプログラミングの基礎を習得させ卒業させることを旗印としてプログラミング教育を必修科目としてきた。特にプログラミング入門として、プログラミング基礎教育の目的はプログラミング文法よりその論理的思考力、思考法に重点を置いている。プログラミング入門の科目は言語 PASCAL を使った授業を展開してきている。一方、プログラミングに対する学生の学習意欲は年々低下傾向にある。この原因のひとつとしてプログラミングの授業が文字の表示や入力といったテキスト操作を中心としていることから面白さが感じられないのではと考えている。特にメディア系の学生は将来プログラマを目指す者も少ないために最初のプログラム経験がその後のプログラムへの興味の持ち方に大きな影響を与えられと考える。本稿の試みはこのプログラミングへの興味を失わせないための新しいプログラミング教育のひとつである。

2. 教育の内容

2. 1. CG ソフト POV-Ray について

2. 1. 1. POV-Ray を使う理由

CG ソフトといえば一般的にマウスやタブレットといったポインティングデバイスを使って作業を進めるイメージが強い。このため、CG ソフトをプログラミング教育に使うというと奇異に感じられるかもしれない。しかし、今回採用した POV-Ray という CG ソフトはプログラミングのような手続きでシーンを記述し、CG 画像を作成するソフトである。しかも、C 言語の表記によく似た表現で用いているために後で使うことになる C 言語でプログラムを書く感覚でシーンの記述を行うことができる。このことが POV-Ray を採用した主たる理由である。また、フリーソフトであることから学生が自宅で勉強することができる点も大きい理由となっている。学生はこの授業をきっかけとして CG をより深く勉強したいと考えてくれることを望んでいる。

2. 1. 2. POV-Ray の動作環境

POV-Ray はフリーソフトである。インターネットから誰でも無料でダウンロードできる。また、いろいろな OS で動かすことができるソフトなのでさまざまな環境で使うことができる。著者等は教室のパソコンが Windows 環境であるので Windows の下で動かしている。本学では 60 台のパソコンが設置されたコンピュータ室が 6 教室あり、1 教室をメディア系の専用教室として使っている。その教室のすべてのパソコンに POV-Ray をインストールしている。最新バージョンは v3.6 であるが v3.5 を採用している。

2. 2. カリキュラム

本稿でとりあげた科目は 2006 年度の情報・ネットワーク学科のカリキュラムのプログラミング基礎科目群のひとつで、「CG プログラミング思考法」という科目である。プログラミング基礎科目群は以下の 4 科目である。

- ① プログラミング基礎・実習 (1, 2 セメスタ開講)
PASCAL によるプログラミング入門 (全系共通)
- ② システム系プログラミング・実習 (2, 3 セメスタ開講)
C 言語によるプログラミング (情報サービス系)
- ③ CG プログラミング思考法 (1 セメスタ開講)
POV-Ray によるプログラミング入門 (メディア系)
- ④ 制御系プログラミング (1 セメスタ開講)
レゴロボットによるプログラミング入門 (エンジニアリング系)

2005 年度までは①のみであったが 2006 年度より②、③、④を追加した。学生は上記の 4 つの基礎科目群から必ず 1 科目はとらなければならない。メディア系の学生はほとんどが①と③を履修する。メディア系の専門科目でプ

プログラミングを扱う科目は以下のようなものがあり、これらの科目を履修するためには③を履修させるようにしている。

CG プログラミング・実習 (C 言語)

マルチメディア系プログラミング実習 (Java 言語)

グラフィックスプログラミング (C 言語による 3D 表現)

デザインワーク (C 言語による 2D 表現)

総合実習 (C 言語による高度なアニメーション等の作成)

2. 3. 講義内容とテキスト

表 1 は「CG プログラミング思考法」の講義内容である。最初の 4 週を使って POV-Ray の基本を学習する。ここでは CG に関する基本的な概念 (座標系、カメラ、光源、照明モデル) ついても必要最小限説明を行っている。使用するテキストはすべて筆者らが独自に作成したものを PDF 化して配布している。表 2 に配布テキストの一部を示す。これらのテキストは授業の進行に伴い順次イントラネット上に置いて学生がいつでも参照できるようにしている。また、「POV-Ray V3.5 日本語マニュアル」^[1]もネット上に置いているので必要に応じて参照させている。

表 1 講義スケジュール

週	講義内容
1週	プログラミングとCG表現 (ガイダンス:POV-Ray の紹介)
2週	基本となるかたち (球、ボックス、円錐)
3週	基本立体の組み合わせによる表現 (集合演算)
4週	2次元をもとにした立体表現 (回転体、スイープ)
5週	記憶するしくみ (変数、配列、代入)
6週	物体を規則的に並べる。 (繰り返しの考え方)
7週	物体を交互に並べてみる (条件分岐の考え方)
8週	より本物らしくする (質感表現と各種係数の設定)
9週	高度な配置 (数学関数を使う)
10週	他のデータを使う (ファイル入出力)
11週	かたちを自分で定義する (マクロ定義)
12週	物体が動くアニメーションを作る (clock変数、フレームの考え方、動画作成)

13週	なんでもアニメーション (色、透明度、屈折率)
14週	アニメーション制作(自由課題制作)
15週	作品講評会

表 2 配布テキスト

タイトル	ページ数
00_PDF マニュアル使用法	1
01_インストールと設定	6
02_はじめの一步	7
03_基本操作	9
04_雪だるま	12
04_補足 union&declare	3
05_コーヒーマップ	10
06_集合演算	3
07_繰り返し	12
08_条件判断	6
09_アニメーション 1	7
関数	15
形状見本	2
材質見本	2
色見本	1
動画作成	5

2. 4. POV-Ray のシーン記述と実行

POV-Ray はシーンを記述し、シーンファイルを作成し、開始ボタンをクリックするとすぐに CG 画像が生成される非常に扱いやすいソフトである。ここではシーン記述に必要な最小限のものを以下に示しておく。

2. 4. 1. インクルードの記述

POV-Ray では、C 言語とよく似た方法で色や基本形状などの定義情報をファイルに書いておきインクルードで読み出すことができる。インストール時に基本的な色や形状についての多くのインクルードファイルが提供されるのですぐに高品位の画像の生成が可能である。ここでは物体の形状を定義した”shapes.inc”と色を定義した”colors.inc”を記述した例を示す。

記述例

```
#include "shapes.inc"
#include "colors.inc"
```

2. 4. 2. カメラ(camera)の設定

シーンを見ているカメラ(視点)を設定する。視点の位置(location)と 注視点(look_at:見ている場所、ここが画面の中心になるように調整される。)と画角(angle:この角度

が大きくとると魚眼レンズのように視野が広くなり、小さくとると望遠レンズのように視野が狭くなる)を設定する。ここではカメラを(0,5,-15)に置き、原点(0, 0, 0)に向けている視野角 30 度のカメラを設定している。

記述例

```
camera {  
    location <0, 5, -15>  
    look_at <0, 0, 0>  
    angle 30  
}
```

2. 4. 3. 光源(light_source)の設定

光源は最初の<>で光源の位置を指定し、colorで光源の色を指定する。colorの後のWhiteはインクルードファイル”colors.inc”に定義済みの色を指定している。(RedやYellowなどの先頭が大文字の色の名前はすべて定義済みのもの)光源の種類は何も指定しなければ点光源となる。必要ならスポットライトなどの光源を設定することもできる。また、光源は複数個指定することもできる。

記述例

```
light_source {  
    < 10, 10, -10 >  
    color White  
}
```

2. 4. 4. 物体(object)の設定

ここでは、シーンの中に物体(object)を配置する。以下の例では定義済みの Plane_XZ(XZ 平面)を指定している。translate<0,-1,0>は物体を Y 軸方向に-1 平行移動している。(つまり下に下げている) pigment は物体の色を指定している。checker は市松模様を表し、白(White)と灰色(Gray)で作るように指定している。

記述例

```
object {  
    Plane_XZ  
    translate < 0, -1, 0 >  
    pigment { checker color White Color Gray }  
}
```

以上の設定をしたシーンの画像を生成したものが図1である。学生たちにはこのシーンファイルを保存させ、新しい課題を作成するときにこのファイルを呼び出し加工するようにさせている。

図1 基本画像(市松模様の床のみ)

2. 5. プログラミング思考法と POV-Ray

プログラミング思考法では以下のようなプログラミング特有の考え方を理解させるための課題を考えている。

変数の概念の理解
繰り返し処理の理解
条件分岐の理解
論理的思考の訓練

2. 5. 1 変数(のようなもの)を使う

POV-Ray では識別子の宣言を使ってプログラミング言語の変数に近いものを実現する。識別子の宣言は本来 POV-Ray のステートメントや特定の値をキーワードとして定義するための機能である。この機能は複雑な形状を作成する場合には欠かせないものである。

以下に使用例を示す。識別子を宣言をすることで K を変数のように値をセットしたり、参照したりすることができる。

使用例

```
#declare K=0;  
#declare K=K+1;
```

2. 5. 2 繰り返し処理をする。

POV-Ray でステートメントを指定回数繰り返しを行う場合 #while 指示を使う。繰り返し処理の中でオブジェクトを定義すると繰り返した回数分のオブジェクトが生成されることになる。

以下に 5 つの赤い球を配置するシーンを示す。

```
#declare K=0;  
#while (K<5)  
    object {  
        Sphere  
        scale 0.3  
        pigment{color Red}  
        translate<K*2, 0, 0>  
    }  
    #declare K=K+1;  
#end
```

図2 繰り返し処理実行画面

学生には以上のサンプルを参考にして以下のような多重繰り返しの課題を考えさせる。

図3 多重繰り返し課題画像

さらに回転して配置することで次のようなシーンを考えさせている。

図4 繰り返しの応用課題画像

繰り返しの最終課題は、図 5 の堀内正和作「円筒の二等分(1969・国立近代美術館)」を再現した画像を手がかりにプログラミングを考えさせている。この課題ではねじれた円筒をどのように作るかに創意工夫が必要である。半円の薄い板を回転させながら重ねていく方法に気がつくかどうか鍵となる。

図5 繰り返し処理の最終課題画像

2. 5. 3 条件分岐処理をする。

POV-Ray で条件分岐をするには条件指示を使う。最大 200 までのネストが可能である。条件指示を使うことで条件によりシーンの定義を変えることができる。以下に条件によって球の色を変える例題を示す。

```
#declare K=-4;
#while (K<=4)
  object{
    Sphere
    scale 0.5
    translate<K,0,0>
    #if (K<0)
      pigment {color Red}
    #else
      pigment{color Blue}
    #end
  }
  #declare K=K+1;
#end
declare K=K+1;
#end
```

図6 条件分岐画像(左が赤、右が青)

さらに論理演算子を使った多重条件の課題として以下のような問題を考えさせている。

```
#if (mod(K,2)=0 & K<=0)
  pigment {color Yellow}
```



```
#else
    pigment {color Green}
#end
```

図7 多重条件分岐画像(緑と黄色の球を配置)

2.5.4 アニメーションをつくる。

POV-Ray はアニメーションを直接作成する機能は用意されていない。POV-Ray でアニメーションを実現するには実行時のコマンドラインオプションを使う。コマンドラインオプションは POV-Ray 実行画面の上部の入力フィールドに書き込む。いろいろな書き方があるが、+KFF に続けてフレーム数を書く簡単なやり方を使っている。

+KFF30

と書くことにより、30フレームの画像が自動的に生成される。このとき clock 値が 0.0 から 1.0 までリニアに変化することになる。この clock をステートメントで使うことによりフレームごとに clock を時間のように使ったアニメーションが実現できる。以下にアニメーションの例を示す。

```
#include "colors.inc"
#include "shapes.inc"
#include "textures.inc"
#include "metals.inc"
```

```
camera {
    location < 0, 5, -15 >
    look_at < 0, 1, 0 >
    angle 20
}
```

```
light_source {
    < 5, 20, -5 >
    color 1.5 * White
}
```

```
object {
    Plane_XZ
    pigment{ checker color Green color Yellow}
    translate < 0, -1, 0 >
}
```

```
object {
    Sphere
    scale 0.75
    texture { T_Brass_5E }
    translate <clock*5 - 2.5, 0, 3>
}
```

ここでは+KFF5のときの連続画像を以下に示す。

図8.1. 連続画像(0フレーム clock=0. 0)

図8.2. 連続画像(1フレーム clock=0. 25)

図8.3. 連続画像(2フレーム clock=0. 50)

図8.4. 連続画像(3フレーム clock=0. 75)

図8.5. 連続画像(4フレーム clock=1. 0)

このアニメーションでは画面の左から金属の玉が右側に

移動する連続画像が生成されている。

以上の連続画像から Flash アニメーションを作成する。最終課題では自由にアニメーション作品を作って発表する。授業中に作成した物体でも新たに作った物体でも制限なく自由を作ることができる。最後の授業時間は作品の発表会を行う。学生は全員の作品リストが載った相互評価シートに評価を記入する、教員は評価シートの紹介と講師の個々の作品の講評を行う。

図9 アニメーション作品(1年曰居亜理沙)

3. 評価および考察

3.1. 学生の成績評価

プログラミング思考法の成績評価は、出席状況、課題提出状況、アニメーション作品の完成度、プログラミングの発想と創意工夫の有無、最後の理解度判定試験(簡単な POV-Ray のプログラムを書かせる)を総合的に判断する。学生による相互評価の結果は直接成績には反映させていないが結果として学生の評価と教員の評価は一致することが多い。

3.2. 学生による授業評価アンケート結果

学生の授業評価のうち授業内容に関する部分を抜粋してグラフにしたのが表3である。

表3 学生による授業評価アンケート結果

アンケート結果を見ると学生からは一定の評価がされて

いるが、内容が大きく異なるので従来のプログラミング科目と単純に比較することはできない。

3.3. 考察

評価において CG 作品のデザイン的な評価をどうするかという問題があった。現段階ではデザイン的に優れているものはプログラミングの創意工夫に優れていると見做し一定の評価を与えている。CG 作品として結果が現れるためプログラミング能力のみを評価することは難しい。このような授業をどう評価するかについては今後引き続き検討していきたいと考える。

4. まとめ

この試みは、昨年スタートしたばかりで、2 年目を迎えたところである。プログラミングに対する嫌悪感や苦手意識は PASCAL によるプログラミングの授業に比べて格段に改善が見られた。しかしながら、プログラミングの思考法が本当に身についたかどうかの判断は今後のプログラミング関連科目の学生の動向などを慎重に分析し、判断をしなければならない。現時点では分析するサンプルデータが不足しており、あくまでも学生の授業評価の集計結果からの判断に過ぎない。作成した教材もこれから改良をしながらよりプログラミングの面白さを伝えられるような科目を目指して努力していきたい。

参考文献

- [1] POV-Ray V3.5 利用参考マニュアル
大分大学工学部 福祉環境工学科建築コース 真鍋正規研究室提供

著者紹介

倉田 和夫 : 東海大学短期大学部情報・ネットワーク学科, 108-0074, 東京都港区高輪 2-3-23
島森 功 : 東海大学短期大学部非常勤講師
武蔵野美術大学・女子美術大学・日本大学非常勤講師