Graphics with Processing

2014-14 モデリング

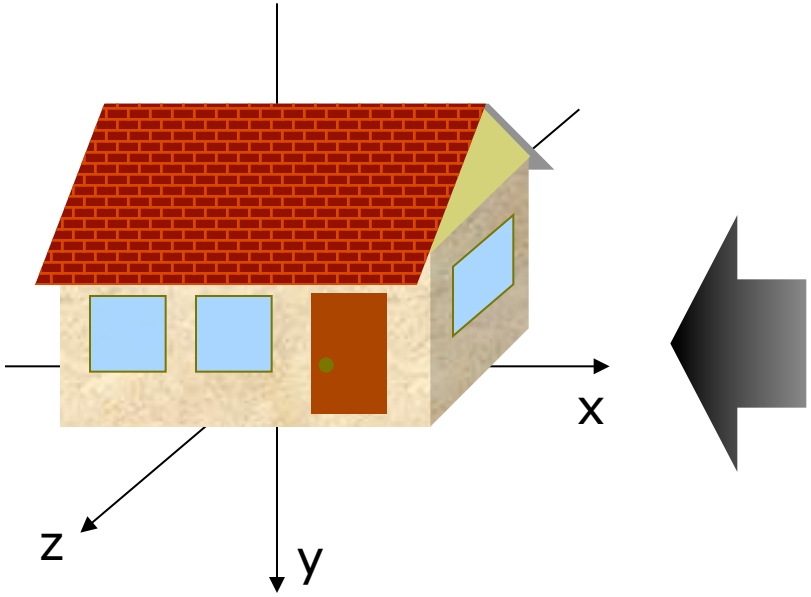
http://vilab.org

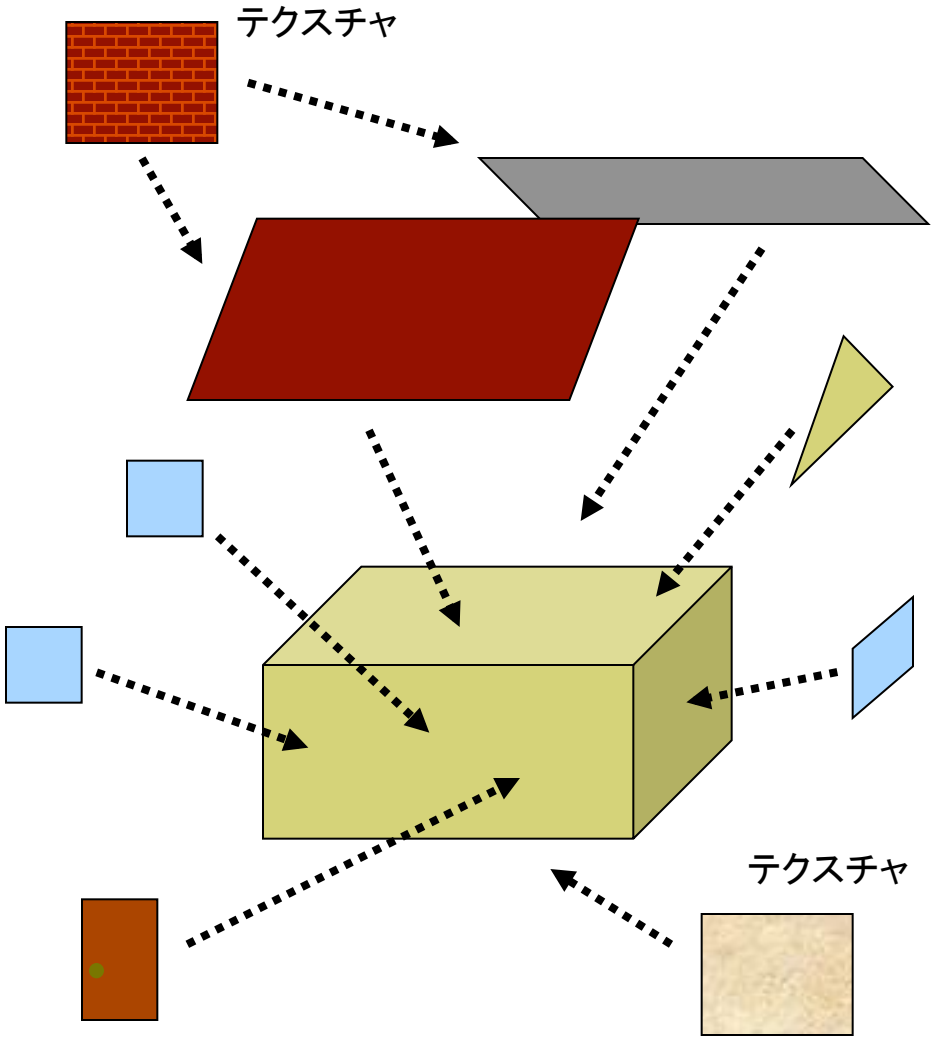
塩澤秀和

14.1 3Dモデリング

モデリング

- 3Dオブジェクト（物体）の形状を数値データの集合で表すこと
- オブジェクト座標系で基本図形やポリゴンを組み合わせる

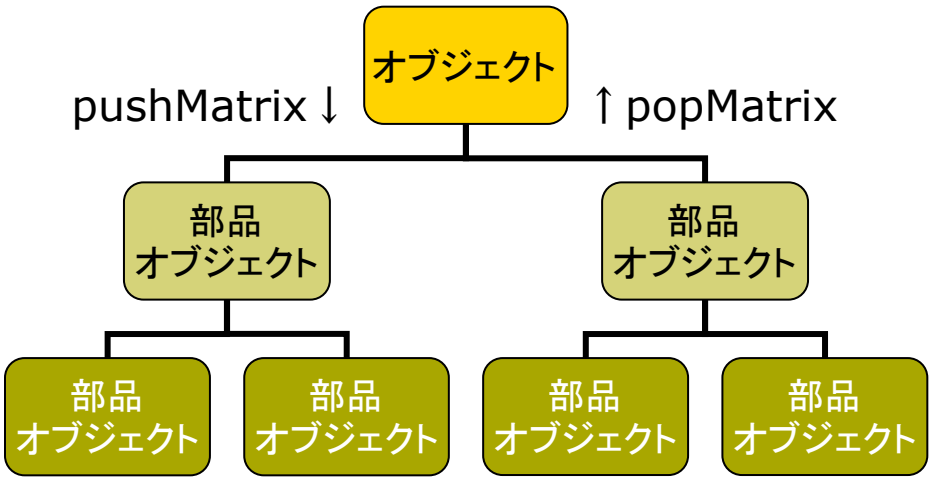




14.2 階層モデリング

階層モデリング（p.45）

- ローカル座標系の階層化
  - 部品はそれぞれの座標系で作り，階層的に大きな部品に組み立てていくようにモデリングする
  - 可動部は，動きの基準点（関節など）を原点として部品化
  - 描画では行列スタックを使う（pushMatrix / popMatrix）




```
void cone() {  // 円錐
  pushMatrix();
  beginShape(TRIANGLE_FAN);
  vertex(0, -1, 0);
  for (int a = 0; a <= 360; a += 10) {
    float x = cos(radians(a));
    float z = sin(radians(a));
    vertex(x, 0, z);
  }
  endShape();
  popMatrix();
}

void tree() {  // 円錐を組み合わせた木
  pushMatrix();
    translate(0, -0.3, 0);
    scale(0.2, 0.7, 0.2);
    fill(0, 255, 0);  cone();  // 円錐1
  popMatrix();
  pushMatrix();
    scale(0.1, 1, 0.1);
    fill(100, 0, 0);  cone();  // 円錐2
  popMatrix();
}
```

14.3 少し複雑なモデリング例

```
// 推奨モード
// バージョン2,3 ⇒ P3D
// バージョン1 ⇒ OPENGL

void house()
{
  // 壁
  pushMatrix();
  translate(0, -0.5, 0);
  fill(#ffffaa);
  box(2, 1, 1.4);
  popMatrix();
  // 屋根の下
  beginShape(TRIANGLES);
  vertex(1, -1, 0.7);
  vertex(1, -1.7, 0);
  vertex(1, -1, -0.7);
  vertex(-1, -1, 0.7);
  vertex(-1, -1.7, 0);
  vertex(-1, -1, -0.7);
  endShape();
  // 屋根
  beginShape(QUAD_STRIP);
  fill(#ffffff);
  // テクスチャはsetup()の中で
  // roof = loadImage("roof.jpg");
  // として読み込んでおく
  texture(roof);
  textureMode(NORMALIZED);
  vertex(-1.1, -0.8, 0.9, 0, 1);
  vertex(1.1, -0.8, 0.9, 1, 1);
  vertex(-1.1, -1.7, 0, 0, 0);
  vertex(1.1, -1.7, 0, 1, 0);
  vertex(-1.1, -0.8, -0.9, 0, 1);
  vertex(1.1, -0.8, -0.9, 1, 1);
  endShape();
  // 煙突
  fill(#880000);
  pushMatrix();
  translate(-0.5, -1.4, -0.5);
  box(0.2, 1, 0.2);
  popMatrix();
  beginShape(QUADS);
  // 窓
  fill(#4444ff);
  float z = 0.701;
  vertex(-0.8, -0.7, z);
  vertex(-0.8, -0.3, z);
  vertex(-0.4, -0.3, z);
  vertex(-0.4, -0.7, z);
  vertex(-0.2, -0.7, z);
  vertex(-0.2, -0.3, z);
  vertex(0.2, -0.3, z);
  vertex(0.2, -0.7, z);
  // ドア
  fill(#883333);
  vertex(0.4, -0.8, z);
  vertex(0.4, -0.1, z);
  vertex(0.8, -0.1, z);
  vertex(0.8, -0.8, z);
  endShape();
}
```

14.4 複雑な形状の表現

曲面や自然形状

- パラメトリック曲面（p.73）
  - パラメータ方程式による曲面
  - ベジエ曲面やNURBS曲面など
  - レンダリング時にポリゴンに変換する方式としない方式がある
- ポリゴン曲面の操作（p.78）
  - 細分割曲面：ポリゴンを再帰的に分割し，滑らかな面を生成
  - 詳細度制御：視点から遠い曲面のポリゴン数を削減して簡略化
- フラクタル（p.86）
  - 自然界によく見られる再帰的な形状（※）のモデリングに適する

※ 海岸線や木の枝など，一部分が全体の縮小のような形状のもの

```
// フラクタルによる地形生成の例（14.5へ続く）
final int N = 256;
float [][] h = new float[N+1][N+1];
Int w = N;  // wは計算済みの要素の間隔

public void setup() {
  size(800, 800, P3D);
  frameRate(30);
  randomSeed(millis());
  // 計算の起点になる4隅の高度を0とする
  h[0][0] = h[0][N] = h[N][N] = h[N][0] = 0.0;
}

// クリックで1段階ずつ細かくなる
public void mouseClicked() {
  generate();
}

// 補間点の高度に加えるランダム量
public float rnd() {
  return random(-0.2, +0.2) * w;
}
```

14.5 地形生成の例（続き）

```
public void draw() {
  background(50, 50, 150);
  lights();
  translate(width/2, height/2);
  rotateX(PI/4);
  rotateZ(radians(frameCount));
  noStroke(); fill(180, 150, 50);

  // 間隔wの要素を使って地形を描画
  scale(2.0);  translate(-N/2, -N/2, 0);
  beginShape(QUADS);
  for (int x = 0; x < N; x += w) {
    for (int y = 0; y < N; y += w) {
      vertex(x, y, h[x][y]);
      vertex(x, y+w, h[x][y+w]);
      vertex(x+w, y+w, h[x+w][y+w]);
      vertex(x+w, y, h[x+w][y]);
    }
  }
  endShape();
}
```

```
// 地形を1段階細かくする
public void generate() {
  if (w == 1) return;

  for (int x = 0; x < N; x += w) {
    for (int y = 0; y < N; y += w) {
      // 中点の高度を補間し，適当な乱数を加える
      h[x+w/2][y] = (h[x][y] + h[x+w][y]) / 2 + rnd();
      h[x][y+w/2] = (h[x][y] + h[x][y+w]) / 2 + rnd();
      // 4点の中央の高度も同様の計算で求める
      h[x+w/2][y+w/2] = (h[x][y] + h[x+w][y]
          + h[x+w][y+w] + h[x][y+w]) / 4 + rnd();
    }
  }
  for (int i = 0; i < n; i += w) {
    h[i+w/2][N] = (h[i][N] + h[i+w][N]) / 2 + rnd();
    h[N][i+w/2] = (h[N][i] + h[N][i+w]) / 2 + rnd();
  }
  // 計算済みの要素の間隔は1/2になる
  w /= 2;
}
```

14.5 モデルデータの利用

3Dモデル表示

- PShape型
  - P3DではOBJデータが利用可能（2DのPShapeは第3回資料参照）
- 読み込みと表示
  - loadShape("ファイル名")
  - shape(図形)
  - shape(図形, x, y, z)
- その他の操作
  - PShapeのメソッドで拡大，回転，頂点の座標・法線ベクトル・色の編集，図形の追加などができる
  - scale, rotate, getVertex 等
- OBJ Loader（ver.1でも対応）
  - https://code.google.com/p/saitoobjloader/

```
// 準備: beethoven.zip をダウンロードし，
// 中身の3ファイルをdataフォルダに入れる

PShape model;

void setup() {
  size(400, 400, P3D);
  model = loadShape("beethoven.obj");
  model.scale(200);
}

void draw() {
  background(0, 0, 100);
  lights();
  pushMatrix();
    translate(width/2, height/2, 0);
    rotateX(PI);
    rotateY(radians(frameCount));
    shape(model);
  popMatrix();
}
```

14.7 3DCGソフトウェア(1)

Art of Illusion

- 3DCGフリーソフトウェア
  - 基本機能をサポート(モデリング，レンダリング，アニメーション)
  - http://www.artofillusion.org
  - Processingで使えるOBJ形式の3Dモデルを作成可能
- インストールと実行
  - ArtOfIllusion???-Windows.exe
  - (英語で)ライセンスへの承諾を求められるので，[Yes]を選択
  - スタートメニューの[Start Art of Illusion]から起動
- 使い方の参考(日本語)
  - http://ei-www.hyogo-dai.ac.jp/~masahiko/moin.cgi/AOI

使い方のポイント

- 基本描画
  - 左のツールボタンから選択
  - 図形の配置，移動，回転など…
  - [シーン]→[レンダー]でレイトレーシングのCGも生成できる
- 色とテクスチャ
  - 単色：タイプ[Uniform]
  - 画像：タイプ[Image Mapped]
- OBJ形式への変換
  - [ファイル]→[データ書き出し]→[Wavefront(.obj)]
  - [テクスチャをmtlで書き出し]
- OBJ変換での注意点
  - AoIの発光色(Ke)は，OBJでは環境反射色(Ka)に変換される

14.8 3DCGソフトウェア(2)

SketchUp

- 概要
  - 人工物のモデリングに適する
  - Google Earthに建物のモデルをアップロードして設置できる
  - http://www.sketchup.com
- OBJ形式への変換
  - 商品版(Pro)だけの機能だが…
  - フリーのプラグイン(拡張機能)を使えば，無料版でも変換可能
  - http://sketchup-onigiri.jimdo.com/plugin-su2objmtl/
- 参考サイト
  - http://www.atmarkit.co.jp/fwcr/rensai2/3dcurl01/01.html
  - http://www.sketchup.com/learn/videos?playlist=58

演習室で使えるソフトウェア

- LightWave と Shade
  - 総合3DCGソフトウェア
  - http://www.dstorm.co.jp
  - http://shade.e-frontier.co.jp
- Terragen
  - 3D自然景観生成ソフトウェア
  - 映画，CMなどでも利用
  - http://www.planetside.co.uk

プロ向けのハイエンド製品

- 3大CGソフト(Autodesk社)
  - 3ds Max, Maya, Softimage
  - 学生なら無料で個人利用可能
  - http://www.autodesk.co.jp
  - http://students.autodesk.com