Graphics with Processing

2010-11 シェーディングとマッピング

http://vilab.org

塩澤秀和

11.1 シェーディング

シェーディング

- シェーディングモデル
  - 光の反射・材質のモデル（前回）
  - ポリゴンの陰影計算モデル
  - shade vs shadow

- 法線ベクトル（p.101）

  平面の方程式から

  $$ax + by + cz + d = 0$$

  $$\vec{N} = (a, b, c)$$

  ポリゴンの辺（ベクトル）から

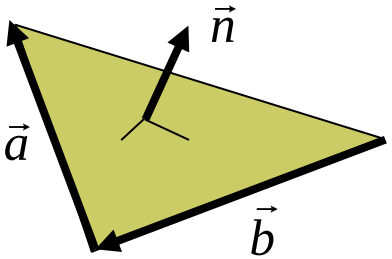
  $$\vec{N} = \vec{a} \times \vec{b}$$ （外積）

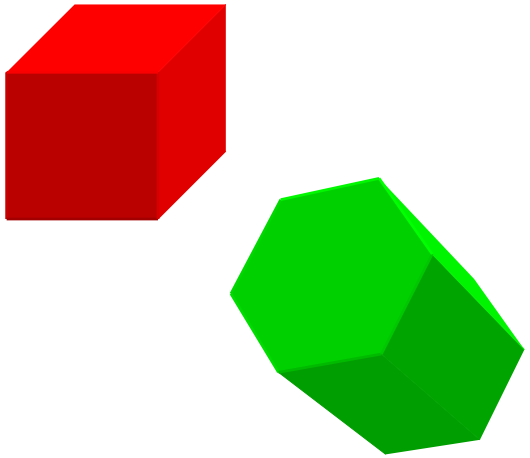
  単位法線ベクトル（大きさ1）

  $$\vec{n} = \vec{N} / |\vec{N}|$$

フラットシェーディング（p.133）

- 各ポリゴンを単一色で描画
  - コンスタントシェーディング
  - ポリゴンの代表点（例：重心）の法線ベクトルを面の向きとする
  - 代表点での光の反射を計算し，面全体の描画色を決定する
  - 単純な図＋平行光線に適する

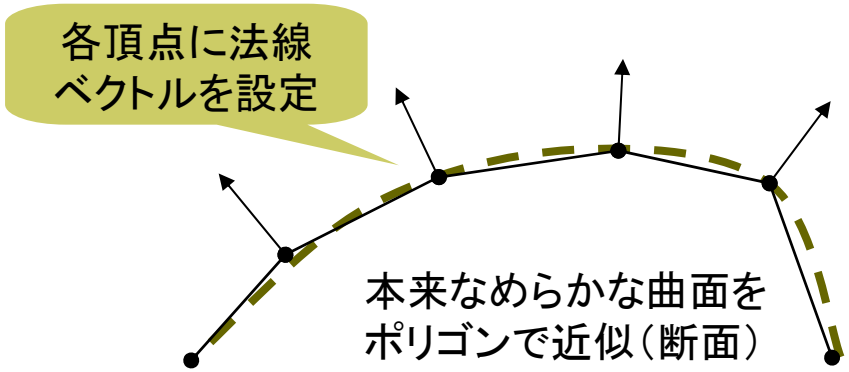




11.2 ポリゴン曲面

ポリゴン曲面（p.78）

- ポリゴンの集合で曲面を近似
  - 三角形を使うことが多い（頂点が必ず同一平面上にあるから）
  - 元の曲面の法線ベクトルを設定することで，曲面に見せかける



- スムーズシェーディング
  - ポリゴン内の色を滑らかに描画
  - グローシェーディング
  - フォンシェーディング

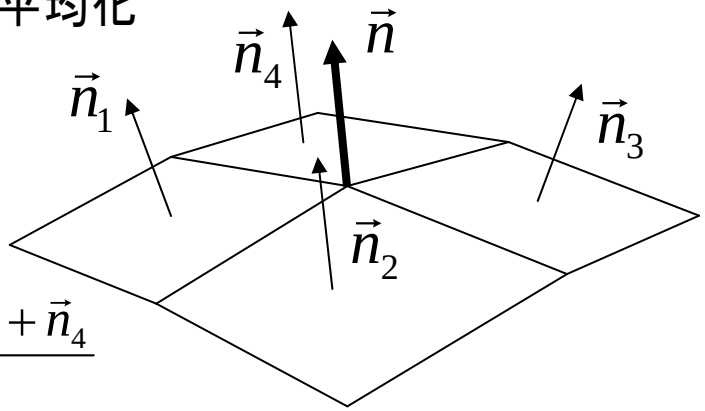
```
void draw() {
  background(0);
  directionalLight(255,255,255,1,0,-1);
  noStroke();
  translate(width/2, height/2, 0);

  beginShape(QUAD_STRIP);
  for (int a = 0; a <= 360; a += 20) {
    float x = 100 * cos(radians(a));
    float z = 100 * sin(radians(a));
    if (mousePressed) normal(x, 0, z);
    vertex(x, -100, z);
    if (mousePressed) normal(x, 0, z);
    vertex(x, 100, z);
  }
  endShape();
}
```

- normal(nx, ny, nz)
  - 曲面近似等のために，頂点位置の法線ベクトルを明示的に設定
  - 対応するvertexの直前で使う

11.3 グローシェーディング (p.134)

グローシェーディング

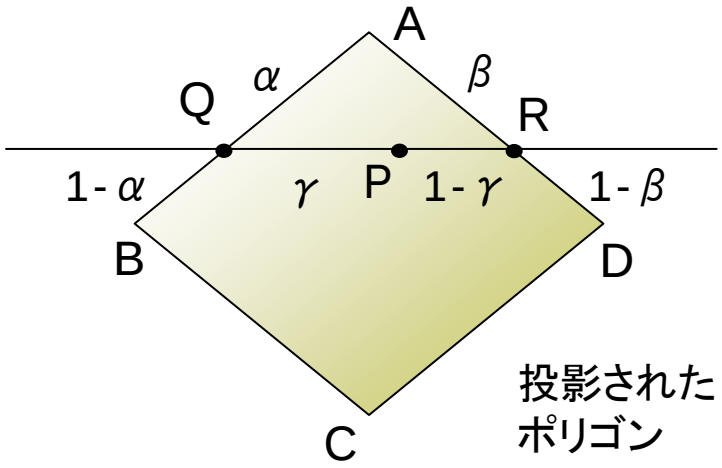
- 頂点間の描画色を補間
  - 各頂点の法線ベクトルによってその位置での光の反射を計算し，物体の色を決定
  - 頂点の間の色を線形補間して，ポリゴン内を滑らかに描画する
- 頂点の法線ベクトル
  - 曲面の方程式などから計算できない場合，周囲の面の法線ベクトルを平均化



$$\vec{n} = \frac{\vec{n}_1 + \vec{n}_2 + \vec{n}_3 + \vec{n}_4}{4}$$

バイリニア補間

- 描画色の高速な補間手法
  - 投影変換後，画面描画する時に各ピクセルの色を線形補間する

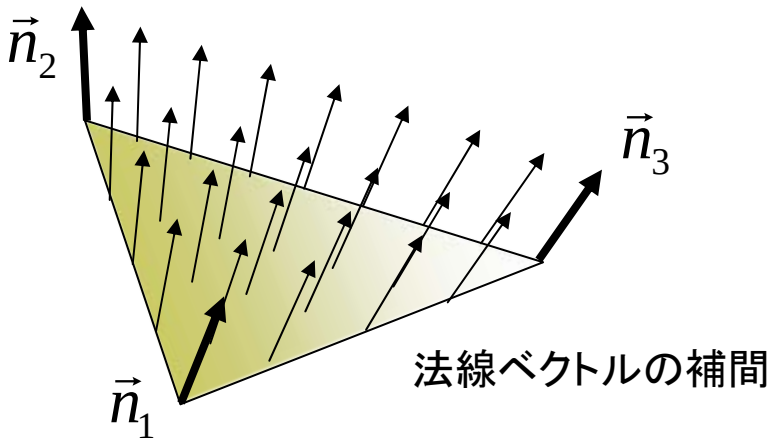


$$C_Q = (1-\alpha)C_A + \alpha C_B$$
$$C_R = (1-\beta)C_A + \beta C_D$$
$$C_P = (1-\gamma)C_Q + \gamma C_R$$

11.4 フォンシェーディングとバンプマッピング

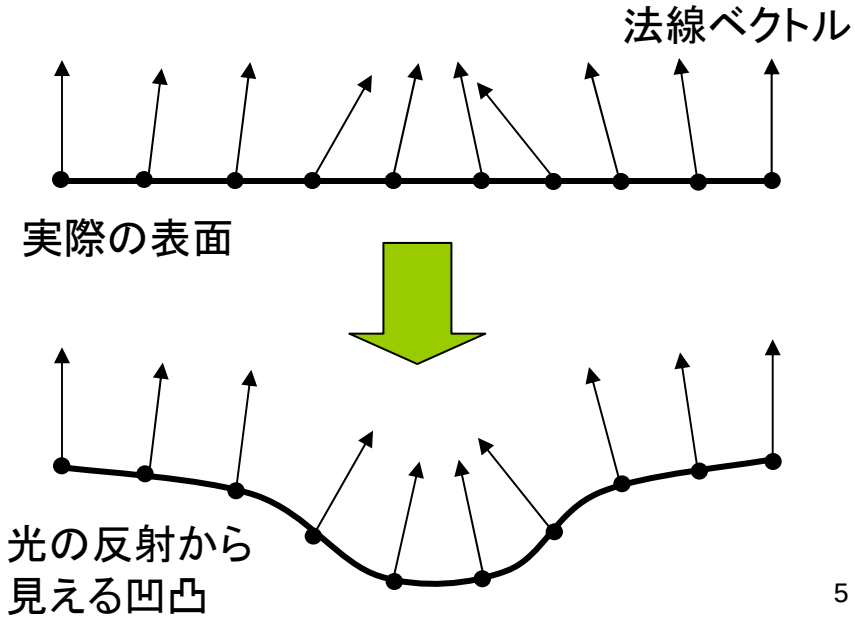
フォンシェーディング（p.135）

- 法線ベクトル自体を補間
  - 色を補間するのでなく，面全体の法線ベクトルを補間する
  - 描画時に各ピクセルの法線ベクトルを計算し，光の反射からピクセルごとの描画色を決定する
  - グローシェーディングより，光沢（つや）のある反射がリアル



バンプマッピング（p.151）

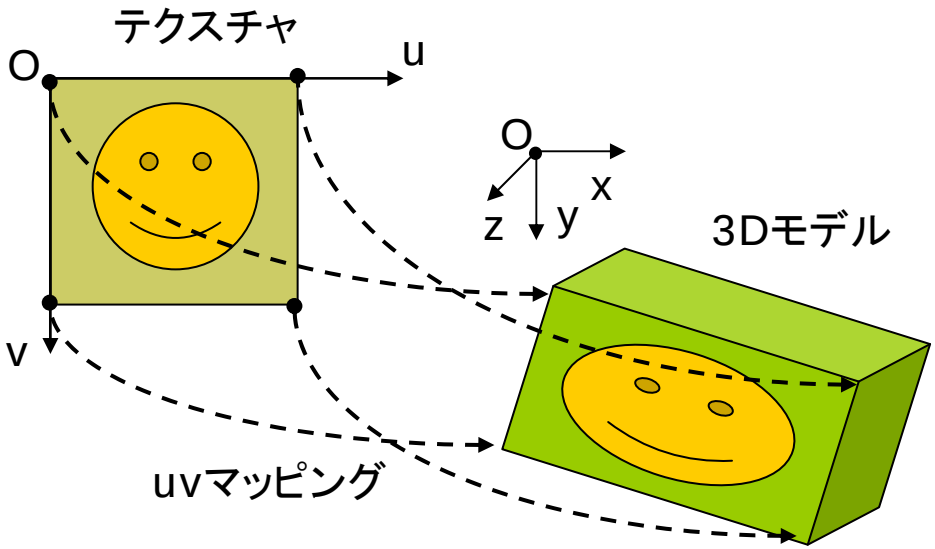
- 凹凸を表面にマッピング
  - 平らな面で法線ベクトルだけを変化させることで，まるで凹凸があるかのように見せる
  - 表面の細かい凹凸を簡単かつ少ない計算量で表現できる



11.5 テクスチャマッピング (p.146)

テクスチャマッピング

- texture＝布目・模様
  - 立体の表面に画像をシールのように貼りつける
  - 質感を表すのに効果てきめん
  - テクスチャ画像を構成する画素をテクセル(texel)という
  - 例）球に世界地図を貼りつける
- uv座標(テクスチャ座標)
  - テクスチャ画像の2次元座標
  - モデリング座標と区別するため，(u，v)（またはs,t)で表す
- uvマッピング
  - 2次元のテクスチャ座標を3次元座標に対応づけること
  - 画像(u，v) → 空間(x，y，z)



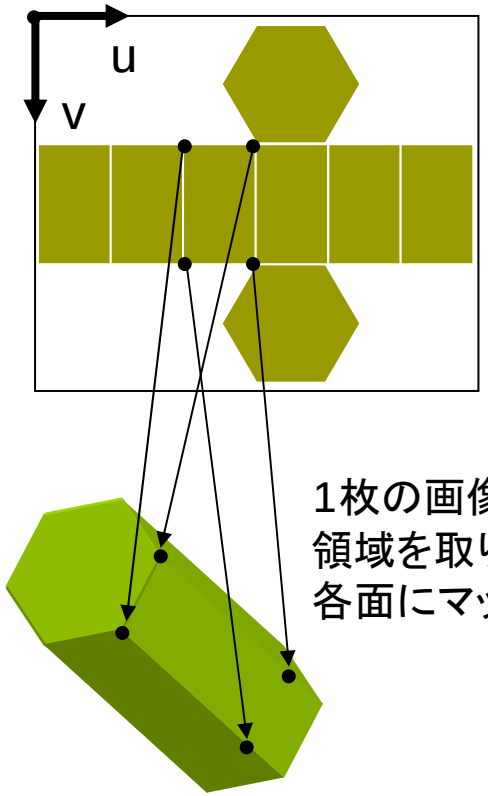
- 描画処理
  - 画面描画時に各ピクセルに対応するuv座標を，頂点のuv座標から逆算(バイリニア補間など)
  - 逆算したuv座標の近傍テクセルから色を補間し，ピクセルを描画
  - 透視投影でテクスチャが歪む
    ⇒“パースペクティブ補正”

11.6 テクスチャマッピング関数

テクスチャマッピング関数

- texture(画像)
  - 画像: PImage型(4.5参照)
  - テクスチャ画像の設定
  - beginShape(), endShape()の中で指定する
- vertex(x, y, z, u, v)
  - 通常のvertex(x, y, z)の処理に加え，その点をテクスチャ座標(u, v)に対応づける
  - 2次元での画像変形にも使えるvertex(x, y, u, v)
- textureMode(座標モード)
  - uv座標の指定モード
  - IMAGE: 実際の画像の座標
  - NORMALIZED: 0.0～1.0

展開図画像などの利用



1枚の画像ファイルから領域を取り出して立体の各面にマッピング可能

1枚のテクスチャから別々のオブジェクトに貼り付けることも可能 ⇒ “テクスチャアトラス”

11.7 テクスチャマッピングの使用例

```
// 準備：画像ファイル（kouji50m.jpg）を
// あらかじめ講義ページからダウンロードして
// スケッチのdataフォルダに入れておく
// （メニューで Sketch → Add File...）
import processing.opengl.*;

PImage tex;

void setup() {
  size(300, 300, OPENGL);
  tex = loadImage("kouji50m.jpg");
}

void draw() {
  background(0);
  translate(width/2, height/2, 0);
  rotateY(-radians(frameCount));
  noStroke();
  beginShape(QUADS);
  texture(tex);
  textureMode(NORMALIZED);
  vertex(-20,-50,  0, 0, 0);
  vertex( 20,-50,  0, 1, 0);
  vertex( 20, 50, 15, 1, 1);
  vertex(-20, 50, 15, 0, 1);
  endShape();

  fill(#ffffff, 128);
  stroke(#555555);
  beginShape(QUADS);
  vertex(-20,-50,   0);
  vertex( 20,-50,   0);
  vertex( 20, 50, -15);
  vertex(-20, 50, -15);
  endShape();
}
```

画像はグローバル変数で定義する

P3DとOPENGLを比較してみよ

画像はsetupの中で一度だけ読み込む

長方形に画像texをテクスチャマッピング

uv座標は0〜1モード

半透明は必ず最後に描画（理由は次回）

11.8 演習課題

課題

- 11.2のサンプルコードをもとに円筒（円柱）の表面にテクスチャ画像をぐるりと貼り付けるプログラムを作成しなさい
  - アニメーションなどによって円筒のすべての表面が確認できるようにすること
  - テクスチャマッピングした場合のスムーズシェーディングの効果を確認しなさい
- 発展
  - 缶詰のラベルやお菓子の容器を展開して，デジカメで撮った写真をテクスチャ画像にしてみる
  - PNG形式などで，透過色（透明部分）が設定されている画像をテクスチャとして貼ってみる

- 提出方法
  - テクスチャ画像も一緒に提出する必要がある
  - プログラムを保存したら，Tools → Archive Sketch で，**画像もまとめたZIPファイルを作る**
  - Processingフォルダにできた「プログラム名（スケッチ名）.zip」というファイルを提出する
  - アップロード時に，種類で**「フォルダ圧縮ZIPファイル」**を選ぶこと

- サンプルプログラム
  - Examples → 3D → Textures → 4つのプログラム
  - Examples → Libraries → OpenGL → TexturedSphere