

# Graphics with Processing



2007-10 隠面処理と照明

<http://vilab.org>

塩澤秀和

# 10.1 隠面処理(1)

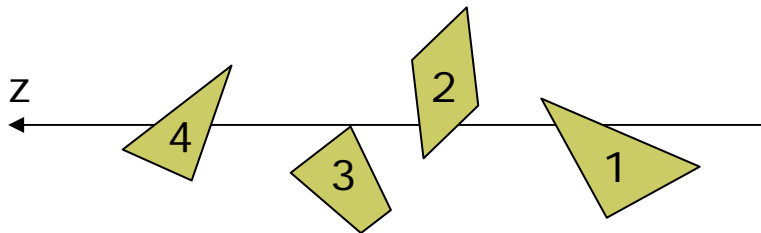
## 隠面処理(隠線・隠面消去)

### □ 隠面処理とは

- 他のオブジェクトの裏側に隠れて見えないオブジェクト(の全部または一部)を描画しない処理
- 視体積のz座標(=視点からの距離)を利用

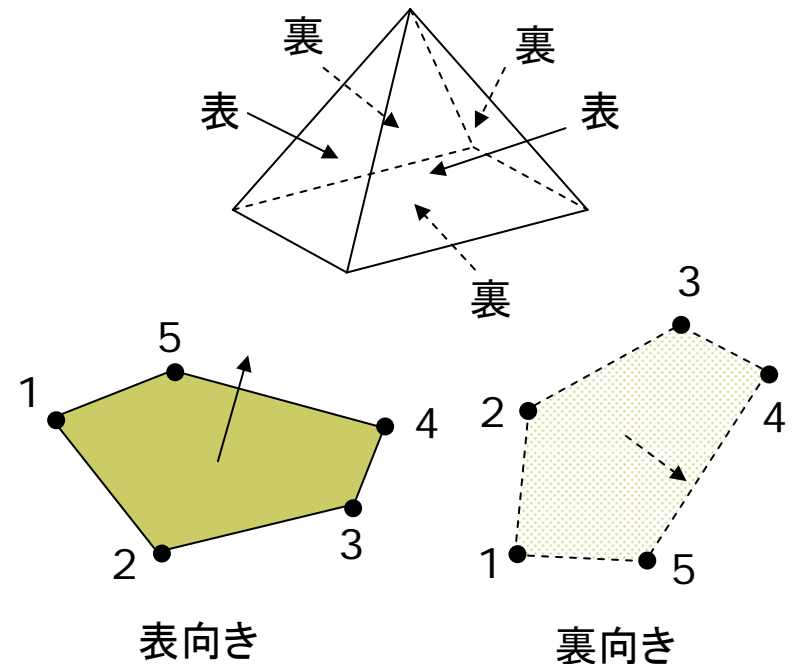
### □ Zソート法(デプスソート法)

- ポリゴンをz座標で並び替えて、遠くのものから順に描画する
- 別名, ペインターズアルゴリズム



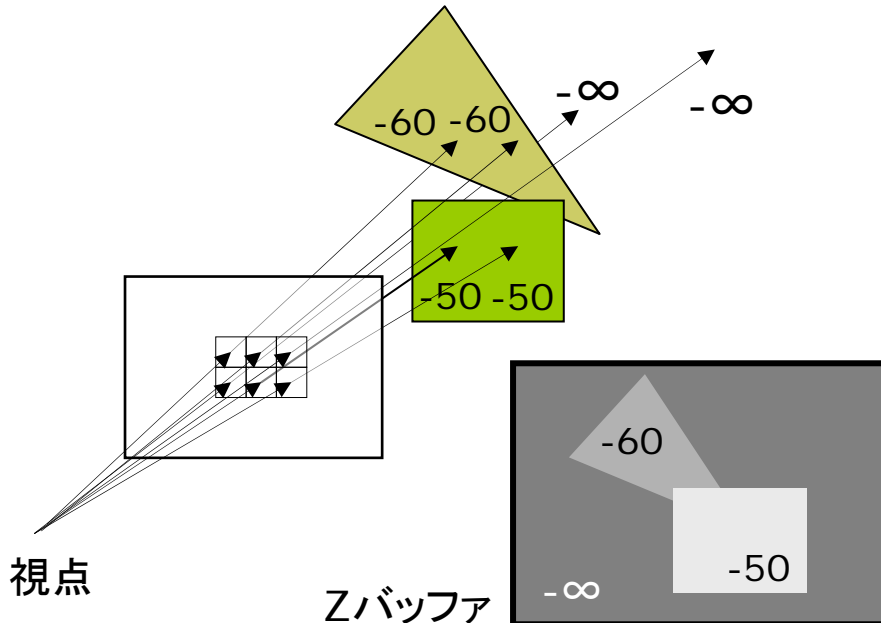
### □ 法線ベクトル法

- ポリゴンに表裏を設定し, 裏側を向いているポリゴンを描画しない
- ポリゴン作成時の頂点の順序(右回り・左回り)で表裏を指定

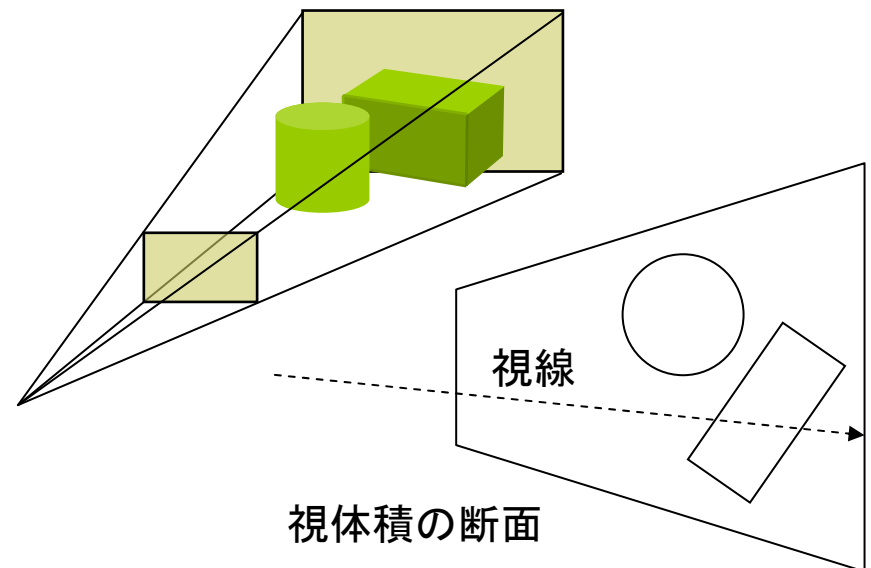


## 10.2 隠面処理(2)

- Zバッファ法(デプスバッファ法)
  - 画面上の全ピクセルに z座標を持たせ, ピクセル描画時に常に遠近関係をチェックする
  - 単純なのでハードウェア化に向く
  - リアルタイムグラフィックスで広く利用(OpenGL, DirectXなど)



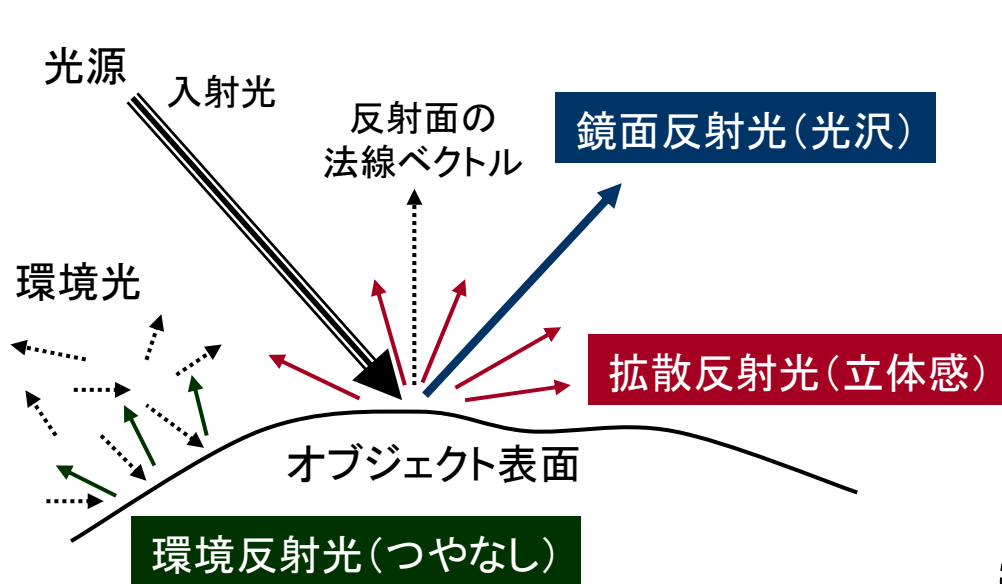
- スキャンライン法
  - ピクセル横1行(スキャンライン)ごとにポリゴンの断面の重なりを幾何的に計算し, 描画する



- その他のアルゴリズム
  - レンダリングの回に概論を説明

# 10.3 光と反射のモデル

## 反射のモデル



環境反射光 = 環境光

× オブジェクトの環境反射色

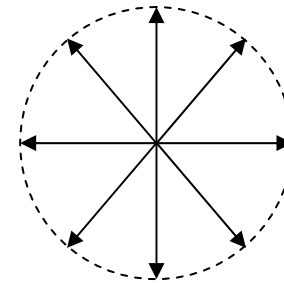
拡散反射光 = 入射光の拡散反射成分

× オブジェクトの拡散反射色

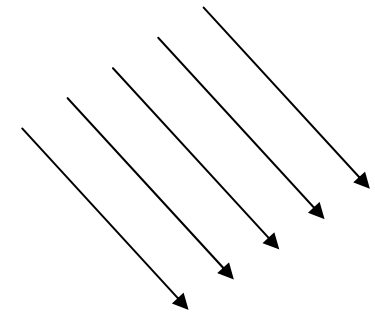
鏡面反射光 = 入射光の鏡面反射成分

× オブジェクトの鏡面反射色

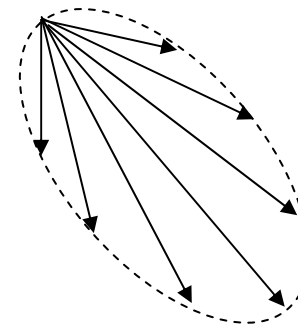
## 光源のモデル



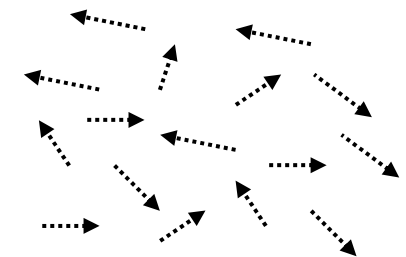
点光源  
(電球など)



方向光  
(太陽光など)



スポットライト



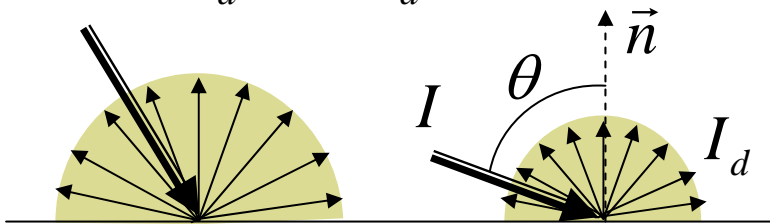
環境光 (壁などに  
何回も反射した  
間接光のモデル化)

# 10.4 反射光の計算モデル

## □ 拡散反射光

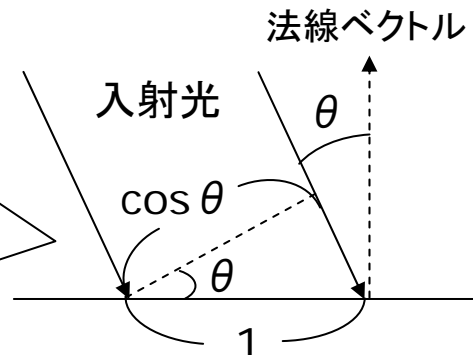
- 光が表面の分子で反射・屈折・透過を繰り返し、色として観測

$$I_d = I K_d \cos \theta$$



$I$  : 入射光の強度       $\theta$  : 入射光の角度  
 $I_d$  : 反射光の強度       $K_d$  : 拡散反射係数

反射面の明るさは  
 単位面積あたりに  
 当たる入射光の量  
 (図より $\cos \theta$ に比  
 例)に比例する



## □ 環境反射光

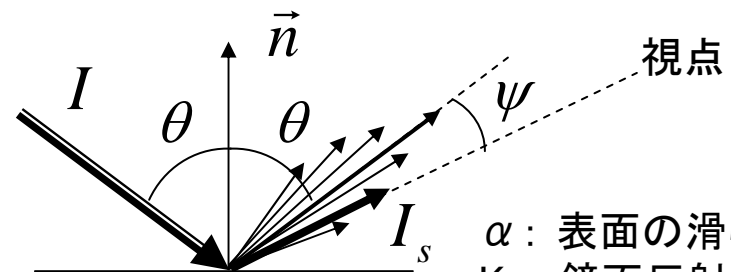
- 環境光に対する拡散反射
- 入射光=環境光はどこでも同じ

$$I_a = I K_a \quad (K_a: \text{環境反射係数})$$

## □ 鏡面反射光

- 表面の分子でほぼ完全反射  
(物体の色の影響が少ない)
- 近似法(フォンモデル)で計算

$$I_s = I K_s \cos^\alpha \psi$$



$\alpha$  : 表面の滑らかさ  
 $K_s$  : 鏡面反射係数

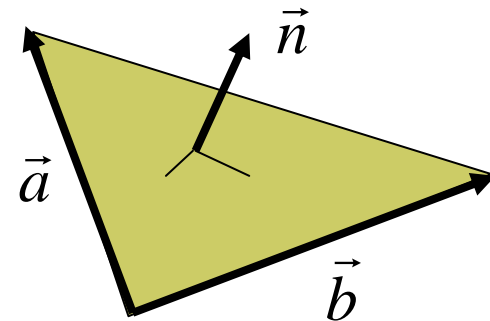
# 10.5 物体表面のモデル

## オブジェクトの材質

- 色と反射係数
  - 色 = 環境反射光 + 拡散反射光 + 鏡面反射光 + 放射光
- 紙・木など
  - 鏡面反射(光沢)がほとんどない
- プラスチックなど
  - 若干の鏡面反射による光沢
- 金属など
  - 強く白っぽい鏡面反射 ( $K_s \neq K_d$ )
- 蛍光面
  - 放射光
  - 照明装置本体などの描画

$$I_e = K_e \text{ (一定)}$$

## 法線ベクトル



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

$$\vec{N} = \vec{a} \times \vec{b} \text{ (ベクトルの外積)}$$

単位法線ベクトル(大きさ1)

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

## 10.6 照明と材質

### 光源の設定

- `pointLight(r, g, b, x, y, z)`
  - 点光源(例:電球)
  - `r, g, b`: 光の色(HSBモードの場合は, 色相, 彩度, 明度)
  - `x, y, z`: 光源の座標
- `directionalLight(r, g, b, nx, ny, nz)`
  - 方向光(例:太陽光, 天井照明)
  - `nx, ny, nz`: 光の方向ベクトル
- `ambientLight(r, g, b)`
  - 環境光(間接光のモデル化)
  - 全方向から均等にあたる光
- サンプル
  - **物体をおく前に, 光源をおくこと**
  - 例: 3D and OpenGL  
→ Lights → Directional

### 標準の照明

- `lights()`
  - 下記の光源を設定
  - `ambientLight(128, 128, 128)`
  - `directionalLight(128, 128, 128, 0, 0, -1)`

### オブジェクトの材質特性

- `fill(色)`
  - オブジェクトの色=拡散光の色
- `ambient(色)`
  - 環境光に対する反射色を別に指定する場合
- `emissive(色)`
  - 発光するオブジェクト(蛍光体)

# 10.7 光沢の表現

## オブジェクトの材質特性

- specular(色)
  - 光源の鏡面反射成分に対する反射色
- shininess(輝き)
  - 鏡面反射光の集中度 ( $\alpha$ )
  - 輝き: 10~50~500(金属)

## 光源のパラメータ

- lightSpecular(r, g, b)
  - 後に設置する光源に鏡面反射成分を追加
- lightFallOff(c1, c2, c3)
  - 同様に光の減衰率を設定
  - 距離dとして

$$\frac{1}{c_1 + c_2 d + c_3 d^2}$$

## □ サンプル

```
void setup () {
  size(200, 200, P3D);
}

void draw() {
  background(0);
  ambientLight(50, 50, 50);
  if (mousePressed)
    lightSpecular(128, 128, 128);
  directionalLight(128, 128, 128,
                  -1, 1, -1);

  noStroke();
  fill(255, 220, 0);
  shininess(10);
  specular(255, 255, 100);
  translate(width/2, height/2, 0);
  sphere(50);
}
```



# 10.8 演習課題

## 課題

- 床にスポットライトをあてる照明効果表現してみなさい
  - シェーディングはポリゴン単位で行われるので、床は右のようにタイルを敷き詰めるようにする
  - スポットライトの設置例
  - `spotLight(255, 0, 0, 50, -50, -50, -1, 1, 1, PI/2, 100)`
  - `spotLight(0, 255, 255, -50, -50, -50, 1, 0.6, 1, PI/2, 70)`

## スポットライト関数

- `spotLight(r, g, b, x, y, z, nx, ny, nz, 角度, 集中度)`
  - 角度: 光の範囲 ( $\sim \pi/2$ 程度)
  - 集中度: 1~100~それ以上

## □ 床の描画

```
noStroke();
for (x=-100; x<100; x+=10) {
  for (z=-100; z<100; z+=10) {
    beginShape(QUADS);
    vertex(x, 0, z);
    vertex(x, 0, z+10);
    vertex(x+10, 0, z+10);
    vertex(x+10, 0, z);
    endShape();
  }
}
```

