Graphics with Processing

2016-10 照明と材質のモデル

http://vilab.org

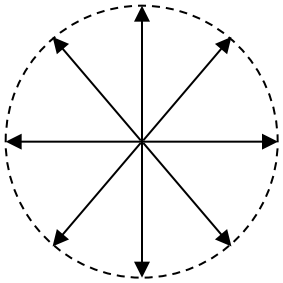
塩澤秀和

10.1 レンダリングと光源

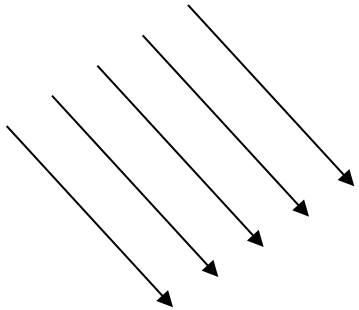
レンダリング（p.122）

- 座標変換後の画像生成
  - 3次元シーン → 2次元画像
  - 色，陰影，質感などの表現
  - 光学現象のシミュレーション
  - 高品質 vs リアルタイム
- レンダリング関連技術
  - 隠面消去
  - ライティングとシェーディング
  - マッピング（テクスチャ・法線等）
  - 影付け，など…
- 高品質CGのレンダリング
  - レイトレーシング
  - 大域照明
  - ボリュームレンダリング，など…

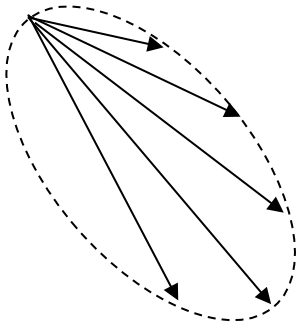
光源の種類（p.144）



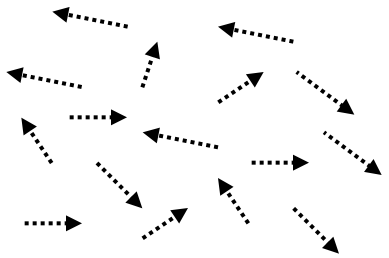
点光源
（電球など）



方向光
（太陽光など）



スポットライト

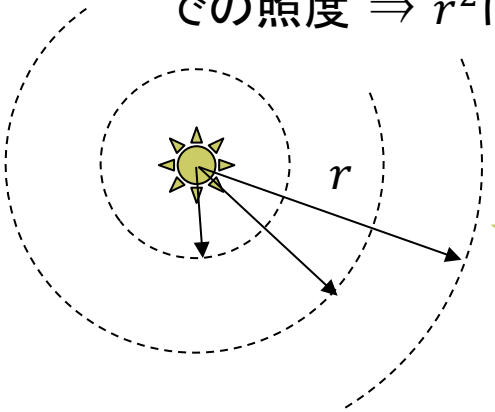


環境光（壁などに
何回も反射した
間接光のモデル化）

10.2 照明の効果

光源からの照明

- 照明の色
  - 太陽光・蛍光灯 ⇒ 白
  - 白熱電球 ⇒ 白っぽいオレンジ
- 照明の明るさ
  - 光束＝光源から発する光の量（単位ルーメン lm）
  - 照度＝単位面積あたりに当たる光の量（単位ルクス lx）
  - 点光源から距離 r 離れた場所での照度 ⇒ r^2に反比例



光源を中心とする球の表面積は
$S = 4\pi r^2$

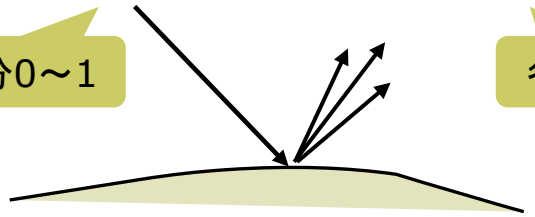
反射と色の関係

入射光
$L = (L_R, L_G, L_B)$

反射光
$I = (I_R, I_G, I_B)$

各成分0～1

各成分0～1



表面色（反射率）
$k = (k_R, k_G, k_B)$

物体の色＝白色光の反射率（各成分0～1）

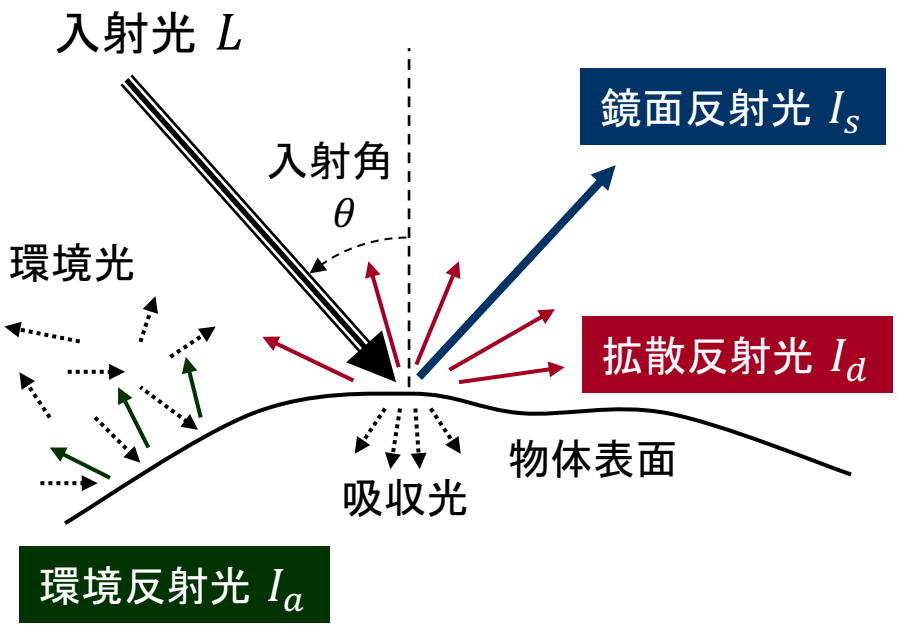
- 反射光＝表面色×入射光
  - 色の見え方は照明にも影響される
  - 光のスペクトル（RGB）ごとに計算

⇒ ベクトルの「要素ごとの積」

$$(I_R, I_G, I_B) = (k_R L_R, k_G L_G, k_B L_B)$$

10.3 反射光のモデル

反射光のモデル(p.141)



- 観測される色
  - 各反射光(+放射光)の総和

$$I = I_d + I_s + I_a + I_e$$

※ それぞれにRGB成分があることに注意

反射光の種類

- 拡散反射光(I_d:diffuse)
  - 物体表層で複雑に反射・透過・屈折することで拡散した光
  - 光の入射角に依存 ⇒ 立体感
- 鏡面反射光(I_s:specular)
  - 物体表面に並んだ分子で鏡のようにきれいに反射した光
  - 見る角度に依存 ⇒ “光沢”
- 環境反射光(I_a:ambient)
  - 特定の光源ではなく，空間全体の間接光に対する反射光
  - シーン全体が一様に照らされる
- 放射光(I_e:emissive)
  - 物体自体からの発光
  - 周囲に関係なく一定の明るさ

10.4 材質属性のモデル

材質（マテリアル）属性

- “色”の設定
  - 反射・吸収される光の波長は，物体表面の材質によって違う
  - 白色光に対する反射スペクトル（分光分布）で材質をモデル化
- 拡散反射色（k_d）
  - 拡散反射の反射率（RGB）
  - 表面が粗いほど，特定の波長が吸収され，残りの光が拡散する
  - 通常の意味での物体の色
- 鏡面反射色（k_s）
  - 鏡面反射の反射率（RGB）
  - 表面が滑らかだと，すぐに反射して着色の少ない光が増える
  - 金属光沢，ハイライト，つや
- 環境反射色（k_a）
  - 環境光に対する反射率（RGB）
  - 通常は拡散反射色と同じ色
- 放射光（k_e）
  - 電球などの発光スペクトル（RGB）
  - 周囲に関係なく一定の色になる

$$I_e = k_e \quad \text{（常に一定の色）}$$

材質による特徴

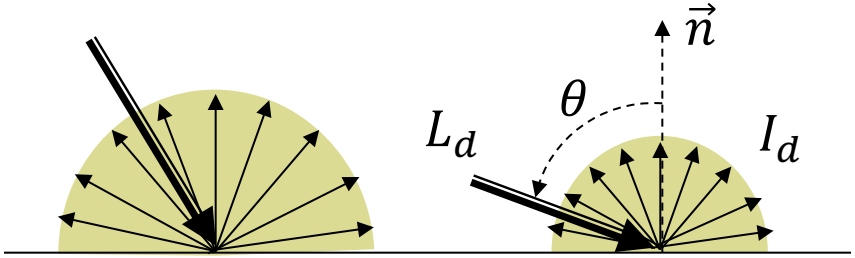
- 紙・木など
  - 鏡面反射（光沢）がほとんどない
- プラスチックなど
  - 若干の鏡面反射によるつやがある
- 金属など
  - 強く白っぽい鏡面反射（$k_s \neq k_d$）

10.5 反射光の計算モデル

拡散反射光（p.144）

- ランベルト（Lambert）の余弦則
  - 光がどの方向から入射しても，全方向に均等に拡散する場合
  - 入射角余弦の法則より，表面の明るさは入射角のcosに比例

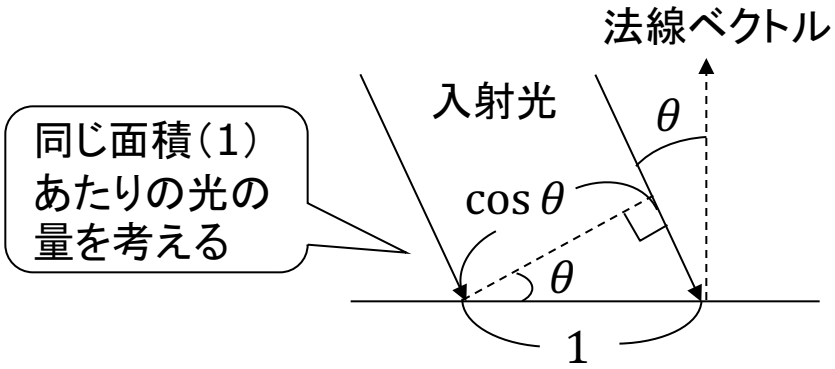
$$I_d = k_d L_d \cos\theta$$



上から照らされると明るくなる　　横から照らされると暗くなる

L_d：入射光（拡散光用成分）　I_d：反射光
k_d：物体表面の拡散反射率　θ：入射角

- 入射角余弦の法則
  - 単位面積あたりに当たる入射光の量は入射角のcosに比例



環境反射光（p.144）

- 環境光による拡散反射光
  - 環境光（L_a）は，四方八方から均等に当たるので方向がない
  - どこからでも同じ色に見える

$$I_a = k_a L_a$$ （k_a：環境光の反射率）

10.6 照明と材質の関数

基本的な光源

- 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
  - 3D (Basics) → Lights
  - **物体をおく前に，光源をおくこと**

標準の光源

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

基本的な材質特性

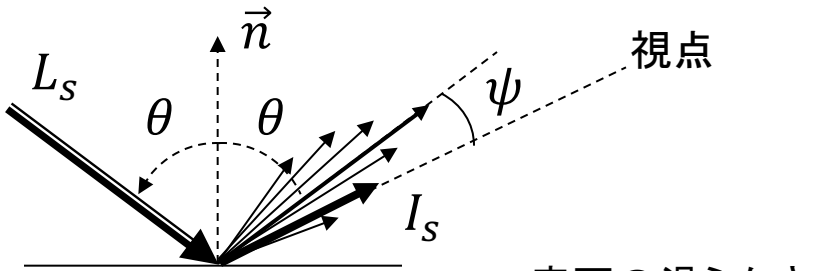
- fill(色)
  - 通常の色＝拡散反射率（k_d）
- ambient(色)
  - 環境反射率（k_a）の設定
  - 無指定時にはfillと同じ色で計算
- emissive(色)
  - 放射光（k_e）の設定（蛍光面）

10.7 光沢の表現

鏡面反射光(p.148)

- フォン(Phong)の反射モデル
  - 光がごく表層でほぼ完全に反射
    ⇒ 反射光が正反射方向に集中
  - 従来，フォンのモデルで近似

$$I_s = k_s L_s \cos^{\alpha} \psi$$



α ： 表面の滑らかさ
k_s： 鏡面反射率

- より正確なモデル(p.149)
  - ブリンの反射モデル
  - クック･トランスの反射モデル

鏡面反射の材質特性

- specular(色)
  - 鏡面反射率(k_s)
- shininess(輝き)
  - 鏡面反射光の集中度(α)
  - 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光用の成分(L_s)を追加
  - 通常は光源と同じ色でよい
- lightFallOff(c1, c2, c3)
  - 光の減衰のしかたを変更する
  - 距離dとして $\dfrac{1}{c_1 + c_2 d + c_3 d^2}$

10.8 照明と材質の設定例

```
void draw() {
  float a = radians(frameCount);
  background(0);
  perspective();
  camera(0, -100, 200, 0,0,0, 0,1,0);

  // 環境光
  ambientLight(50, 50, 50);

  // 回転する点光源を設置する
  // ボタンを押すと鏡面反射成分をつける
  float lx = 100 * cos(a);
  float ly = -100;
  float lz = 100 * sin(a);
  if (mousePressed)
      lightSpecular(128, 128, 128);
  pointLight(128, 128, 128, lx, ly, lz);

  stroke(128);
  line(lx, 0, lz, lx, ly, lz);
  noStroke();
  pushMatrix();
    rotateX(PI/2);
    fill(100);  ellipse(0, 0, 200, 200);
  popMatrix();

  rotateY(a / 2);
  pushMatrix();
    translate(60, -20, 0);
    fill(250, 200, 10);  // 拡散反射色
    specular(100, 100, 100);  //鏡面反射色
    shininess(20);  // 輝きの集中度
    sphere(20);
    specular(0);  // ゼロに戻す
  popMatrix();

  pushMatrix();
    translate(70, -20, 50);
    fill(40, 40, 230); // 拡散反射色
    box(20, 40, 20);
  popMatrix();
}
```

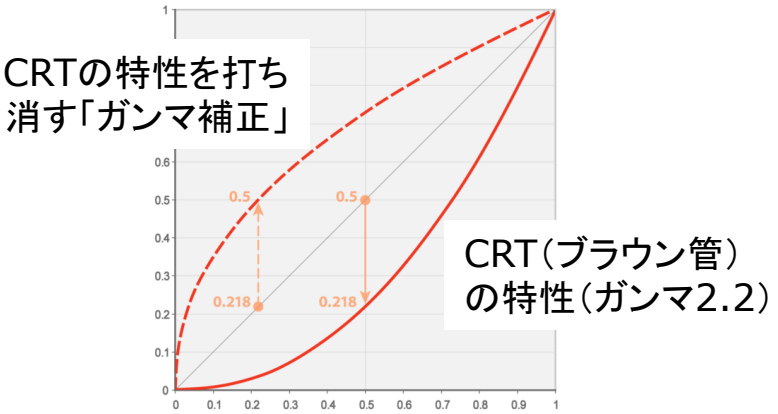
10.9 物理ベースレンダリング

Physically based rendering

- 光学現象をより忠実に再現
  - 最近ゲームなどで急速に採用
  - 厳密な定義はない言葉？

主な使用技術

- リニア（線形）色空間の利用
  - ブラウン管由来の伝統的な階調補正（ガンマ補正）を逆補正し，物理的光量に比例する値で計算

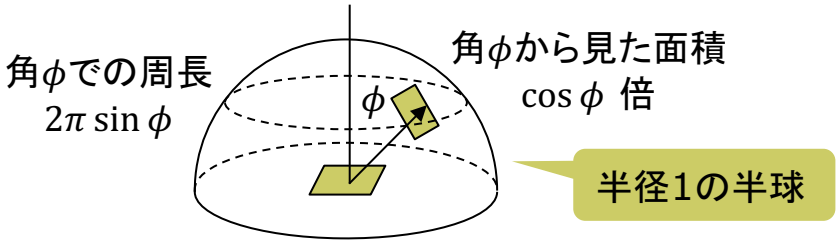


- 光エネルギー保存則
  - 任意の点で単位面積当たり
    入射光の総量≧反射光の総量
  - 入射/反射光の関係を表すBRDF（双方向反射率分布関数）を考慮

$BRDF = (1-m)\times$拡散反射 $+ m\times$鏡面反射

同じ入射光から拡散反射と鏡面反射に分配する（mは金属性を表すパラメータ）

  - 反射光の正規化（拡散反射の例）



従来手法では $\int_0^{\pi/2} \cos\phi \cdot 2\pi \sin\phi \, d\phi = \pi$ 倍に増幅

πで割って正規化 $\Rightarrow$ $I_d = \frac{1}{\pi} k_d L \cos\theta$

10.10 物理ベースレンダリング（続き）

- 高度な鏡面反射モデル
  - クック・トランスのモデルなど

$$\text{反射率}\quad R_s = \frac{F}{\pi}\frac{DG}{(\vec{n}\cdot\vec{L})(\vec{n}\cdot\vec{V})}$$

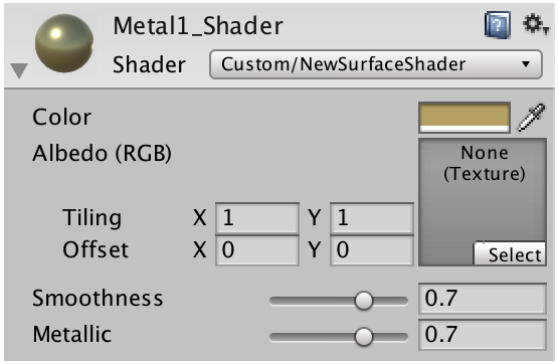
F：フレネル項　D：微小面分布関数　G：幾何減衰係数
$\vec{n}$：法線ベクトル　$\vec{L},\vec{V}$：光源および視点へのベクトル
（F,D,Gは精度等によっていくつかの式がある）

  - フレネル反射（F）：入射角が大きいと反射率が高くなる現象
  - 微小面の分布（D,G）によって，表面の粗さの影響をモデル化

- 物理的な材質モデル
  - 従来：拡散反射色＋鏡面反射色
  - ⇒ Albedo（基本色）＋Metallic（金属性）＋Roughness（粗さ）という物理的なパラメータで指定

- 画像による照明と材質
  - IBL: 光源画像による照明（Image Based Lighting）
  - 表面色の他にも各種の材質属性をテクスチャのようにマッピング

- HDRレンダリング
  - 人間の視覚の広範囲な輝度の認知に対応するため，RGB値に実数（16ビット0～10^{12}）を採用
  - 薄明かりやブルームの表現



Unityでは Roughness の代わりに Smoothness

10.11 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();
}
```

10.12 演習課題

課題

問1) ピンク色の紙に斜め45度から青白い光を照らすとどのような色に見えるか，拡散反射光のRGB値を計算して考察せよ

- 色のRGB値は適当に設定せよ

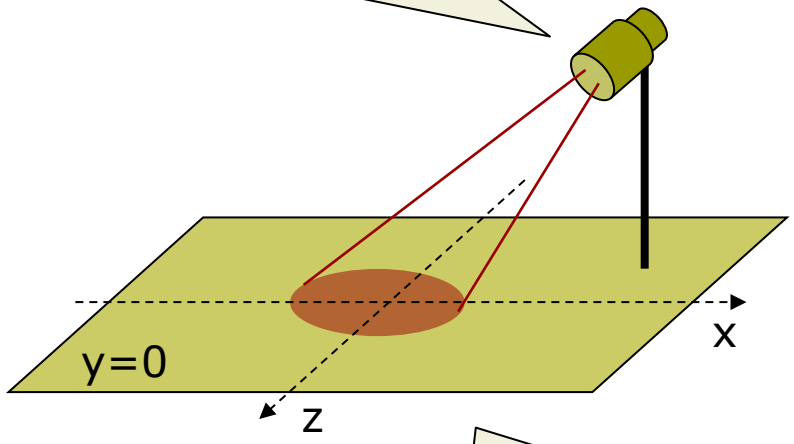
問2) スポットライト（下記）を使用したシーンを工夫して作成せよ

- 床など広い面は，タイルを敷き詰めるようにする（理由は次回）
- 環境光も少し照らすとよい

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（～π/2程度）
  - 集中度：1～100～それ以上

```
// スポットライトの設置例
spotLight(255, 0, 0, 50, -50, -50,
-1, 1, 1, PI/2, 100)
```




```
// 床の描画例
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();
    }
}
```