Graphics with Processing

2022-10 照明と材質のモデル

http://vilab.org

塩澤秀和

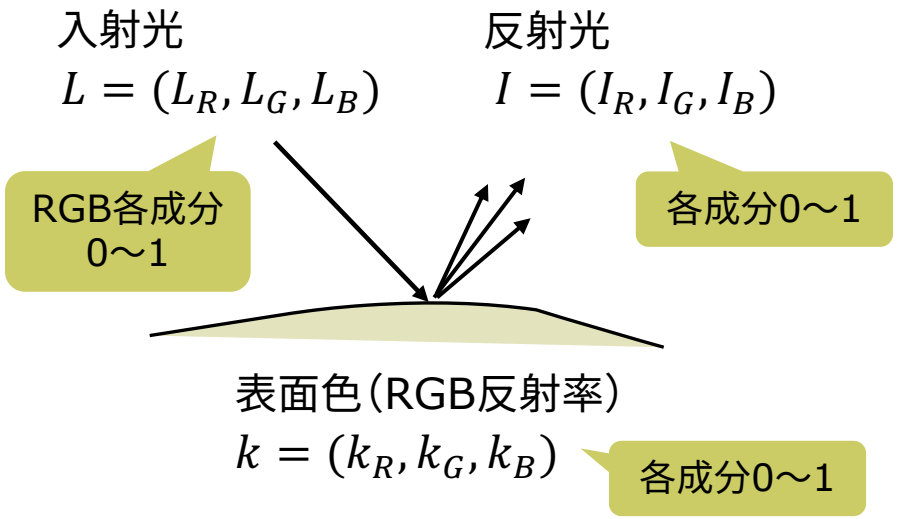
10.1* レンダリング

レンダリング(p.122)

- レンダリングとは
  - 座標変換後の画像生成処理
  - 照明の効果をシミュレーションし，色，陰影，質感などを表現する
- レンダリング関連技術
  - 隠面消去(前回説明)
  - 照明と光の反射の効果(今回)
  - 表面の陰影と模様・質感(次回)
  - オブジェクト同士の影の処理
- 高速 vs 高品質
  - 用途に合わせた手法を選ぶ
  - 単純な拡散反射＋鏡面反射
  - 物理ベースレンダリング
  - レイトレーシング＋大域照明
    (第13回で概要を説明)

レンダリングの基本

- 光の反射と色の関係



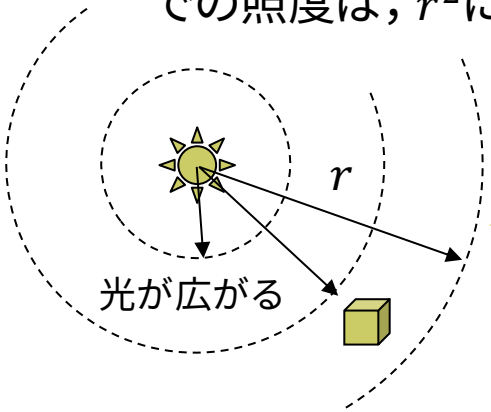
- 反射光(色)＝入射光×表面色
  - 例)赤い照明×白い壁⇒赤く見える
  - 光のRGB成分ごとに反射を計算
    ⇒ ベクトルの「要素ごとの積」

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

10.2* 照明と光源の種類

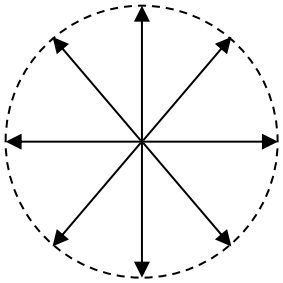
照明の光の性質

- 照明(光源)の色
  - 照明にも色(RGB成分)がある
  - 太陽光～蛍光灯 ⇒ 白～灰色
- 照明の明るさ
  - 光束: 光源が発する“光の総量”(単位ルーメン lm)
  - 照度: 単位面積に当たる光の量(単位ルクス lx=lm/m^2)
  - 点光源から距離 r 離れた位置での照度は, r^2に反比例する

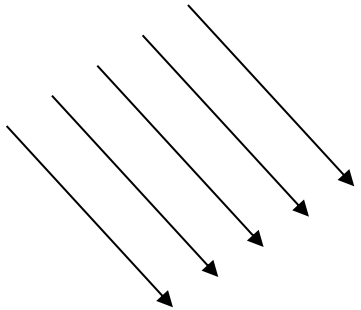


光源を中心とする球の表面積は
$S = 4\pi r^2$

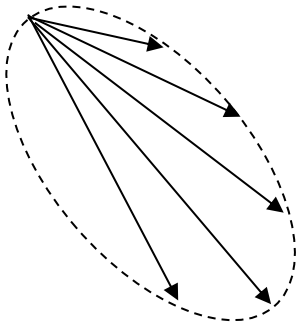
一般的なCGの光源(p.144)



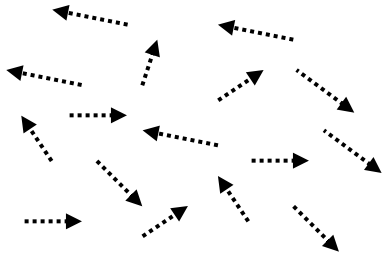
点光源
(電球など)



平行光線(方向光)
(太陽光など)



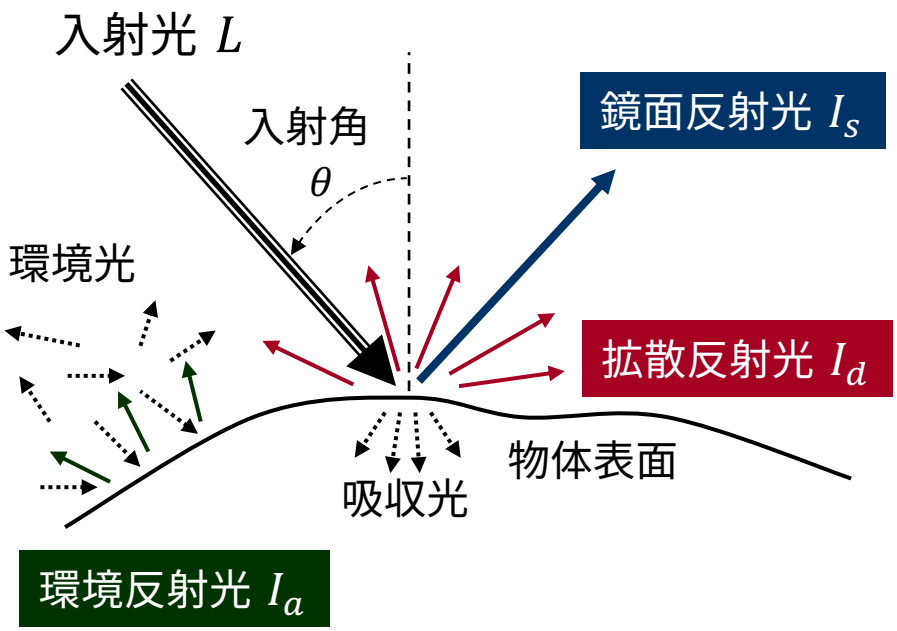
スポットライト



環境光(壁などに何回も反射した間接光の単純化)

10.3* 反射光の種類

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 ⇒ つやと質感
- 環境反射光（I_a：ambient）
  - 間接光を単純化した空間全体の環境光による反射光
  - 位置や角度に関係なく一様
- 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

- オブジェクトの“色”
  - 表面の材質によって反射・吸収される光の波長が違う
  - 白色光に対するRGB各成分の反射特性で材質をモデル化
- 拡散反射色（k_d）
  - ＝ 拡散反射率 (k_{dR}, k_{dG}, k_{dB})
  - 粗い表面では，一部の波長の光が吸収され，残りは広く拡散する
  - 通常の意味での物体の色
- 鏡面反射色（k_s）
  - ＝ 鏡面反射率 (k_{sR}, k_{sG}, k_{sB})
  - 滑らかな表面では，鏡面反射が増え，反射光の色は白に近づく
  - 金属光沢，ハイライト，つや
- 環境反射色（k_a）
  - 環境光に対する拡散反射率
  - 通常は拡散反射色と同じ色になる
- 放射光（k_e）
  - 発光している物の表面色（RGB）
  - 環境や角度に関係なく一定の色

$$I_e = k_e \quad \text{（常に一定の色）}$$

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 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, vx, vy, vz)
  - 平行光線・方向光(例:太陽光)
  - vx, vy, vz: 光の方向ベクトル
- ambientLight(r, g, b)
  - 環境光(間接光の単純化)
  - 全方向から均等にあたる光
- サンプル
  - [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)

基本的な材質属性(色の設定)

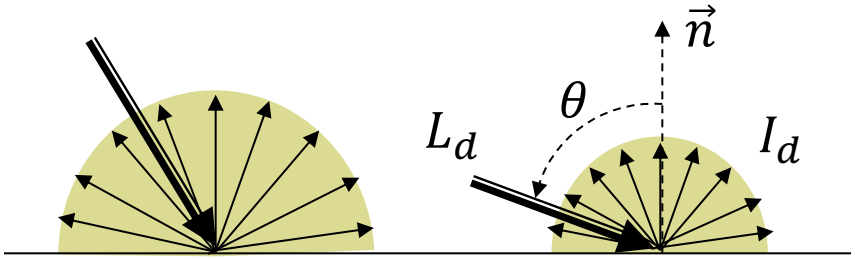
- fill(色)
  - 通常の色=拡散反射率(k_d)
- ambient(色)
  - 環境反射率(k_a)の設定
  - 無指定時にはfillと同じ色で計算
- emissive(色)
  - 放射光(k_e)の設定(蛍光面)

10.6* 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
  - どの方向から入射しても全方向に均等に拡散反射する光
  - 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

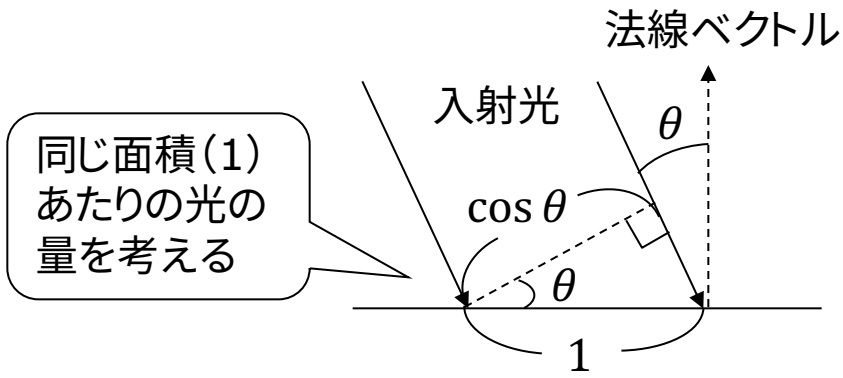
横から照らされると暗くなる

L_d: 入射光(照明の色)　　I_d: 反射光

k_d: 物体表面の拡散反射率　　θ : 入射角

※ それぞれにRGB成分があることに注意

- 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
  - どの方向からも同じ色に見える

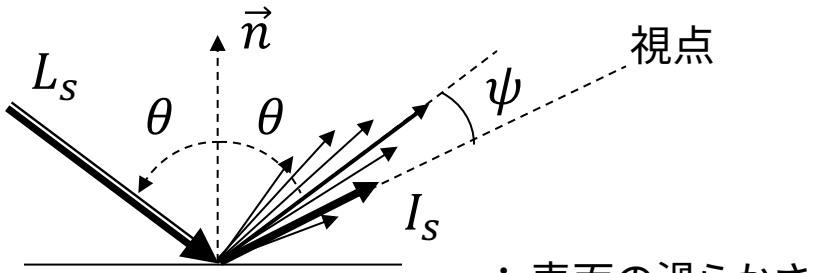
$$I_a = k_a L_a$$ (k_a:環境光の反射率)

10.7* 光沢の表現

鏡面反射光(p.148)

- フォン(Phong)の反射モデル
  - 光が表層でほぼ完全に反射
    ⇒ 反射光が正反射方向に集中
  - 見た目による近似モデル

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



α : 表面の滑らかさ
k_s: 鏡面反射率

- より物理的なモデル(p.149)
  - ブリンの反射モデル
  - クック・トランスの反射モデル

鏡面反射の材質属性

- 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用の成分(L_s)を加える
  - 通常は光源と同じ色でよい
- 光源側に成分を加える理由
  - フォンモデルは近似モデル
  - 拡散反射と鏡面反射は無関係に計算される(物理的には不正確)

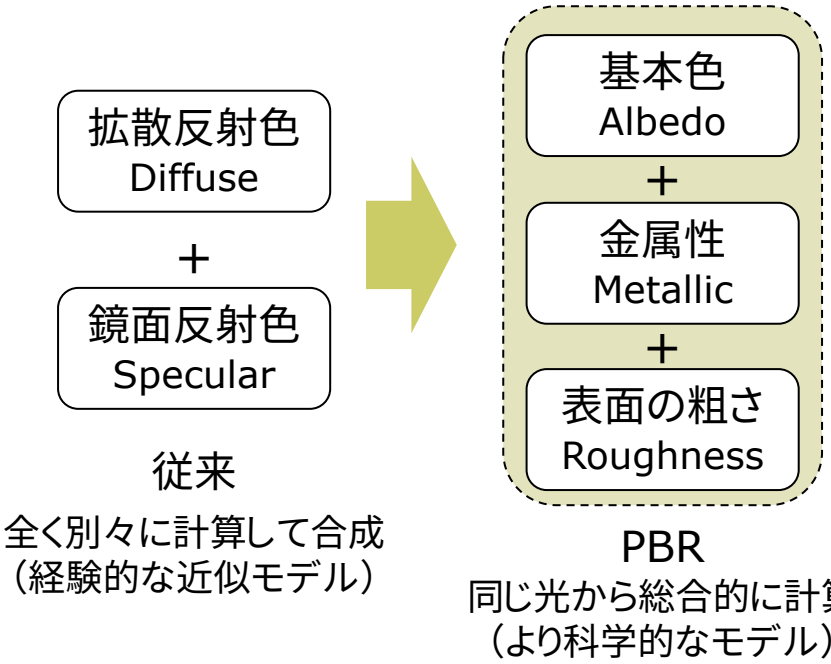
10.8 照明と材質の設定例

```
void setup() {
  size(800, 600, P3D);
}

void draw() {
  background(0);
  float mx = (float) mouseX / width;
  float my = (float) mouseY / height;

  perspective();
  camera(0, -200, 200,  0, 0, 0,
         0, 1, 0);

  // 環境光（なくすとどうなるか?）
  ambientLight(50, 50, 50);

  float lx = 400 * mx - 200;
  float ly = -100;
  float lz = 0;

  // ボタンを押すと鏡面反射光成分を除く
  if (!mousePressed)
    lightSpecular(128, 128, 128);
  pointLight(128, 128, 128, lx, ly, lz);

  for (int i = -10; i <= 10; i++) {
    for (int j = -5; j <= 5; j++) {
      pushMatrix();
      translate(20 * i , 0, 20 * j);
      fill(100, 255, 100);
      // 鏡面反射色と集中度
      specular(100, 100, 100);
      shininess(200 * my);
      noStroke();
      sphere(10);
      popMatrix();
    }
  }
}
```

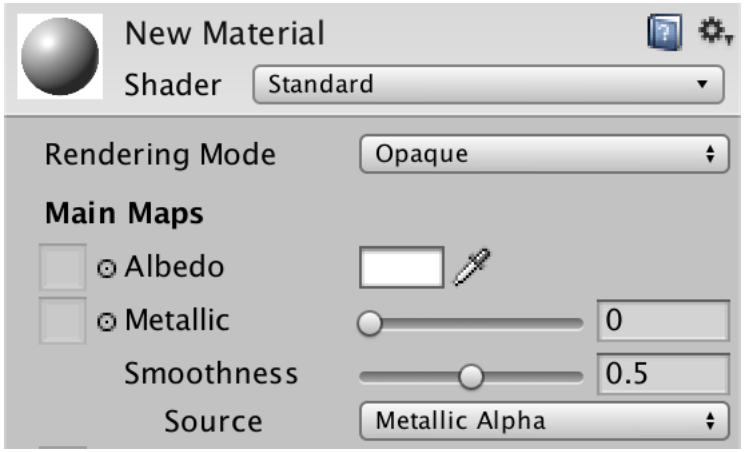
10.9* 物理ベースレンダリング(PBR)

Physically based rendering

- 物理学に基づくレンダリング
  - 従来よりリアルな映像を実現
  - ゲーム等で採用が広がっている
- 物理ベースの材質モデル



- 材質属性のマッピング
  - テクスチャマッピング:
    表面に模様画像を貼り付ける
    (基本色の分布のマッピング)
  - 物理パラメータもマッピング:
    金属性や粗さの分布を貼り付ける
  - その他,透過性や微細な凹凸も
- 例) Unityの材質設定(一部)

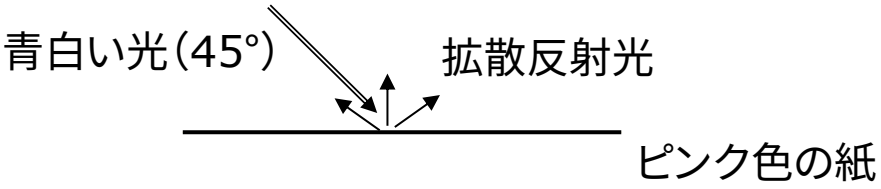


10.10 演習課題

課題

問1) ピンク色の紙に斜め45度から青白い色の光を照らした場合の拡散反射光の色を計算しなさい

- 色のRGB値はぞれぞれ0～1の範囲とし，適当に設定してよい



スポットライト関数

- spotLight(r, g, b, x, y, z, vx, vy, vz, 角度, 集中度)
  - r,g,b：照明の色
  - vx, vy, vz：光の方向
  - 角度：光の範囲（～$\pi/2$程度）
  - 集中度：1～100～それ以上

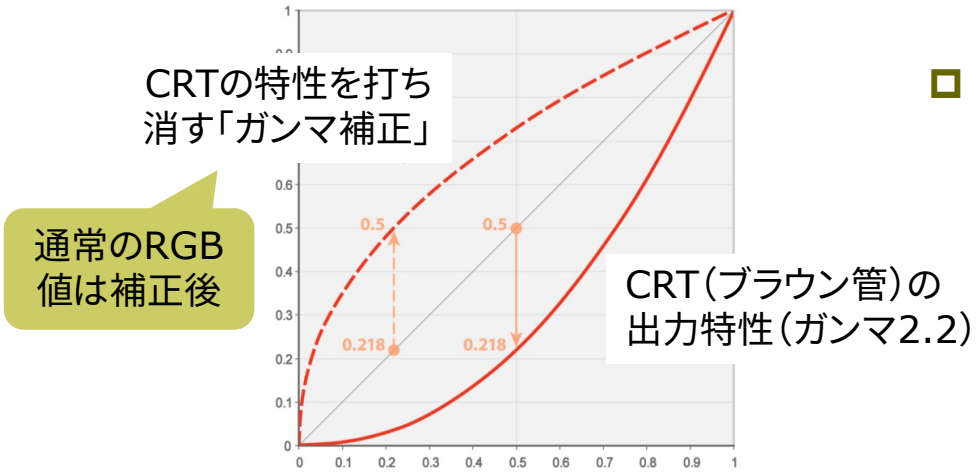
問2) 前回の紙飛行機のプログラム（9.10）を以下のように改造し，照明による効果を加えなさい

- デフォルトの照明（lights()）は削除する
- 暗緑色の地面（床）を明るい色に変える（例えば白や薄黄色）
- 点光源とスポットライトを，ぞれぞれ1個以上設置する（位置，色，個数等は自由）
- どれか1つ以上の光源の位置や向きが動くようにする
- 必要なら，環境光や平行光線でシーン全体の明るさを調整する
- その他，壁や天井を設置するなど，特に照明が効果的に働くように自由に工夫してよい

10.11 参考：PBRの使用技術

主な使用技術

- リニア（線形）色空間の利用
  - 標準的なRGBの値は，昔のCRT（ブラウン管）の表示特性に対応して暗い色ほど増幅するガンマ補正が適用されている
  - 光学的により正しい結果を得るためには，これを逆補正してから計算し，表示時に再補正する

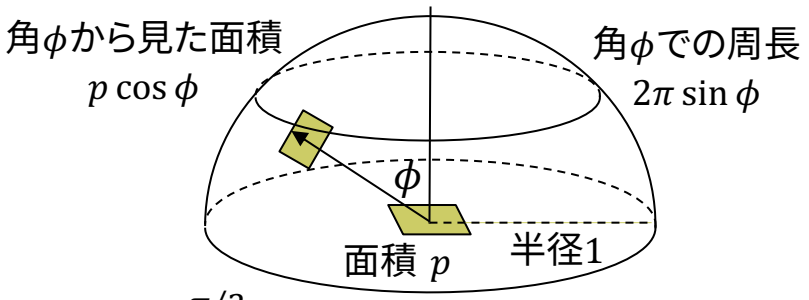


- HDRレンダリング
  - HDR=High Dynamic Range
  - 人間の広範囲な輝度の知覚に対応するため，RGBの値を実数（有効数字7桁）で計算
  - 薄明かりでも細部が表現できる
  - 高輝度領域に対して光がにじむような処理もできる（ブルーム/グレア/グロー/光芒/閃光）
- IBL: 光源画像による照明
  - IBL=Image Based Lighting
  - 空，景色，窓などの実写画像に基づいて照明効果を計算する
    ⇒ 金属等への映り込みも表現
  - 画像から光源の位置等を推定

10.12 参考：PBRの使用技術

- 光エネルギー保存則
  - 物体表面上の任意の点で
    反射光の総量 ≦ 入射光の総量
  - 拡散反射率の正規化にも適用

表面の面積pを，基準の半径(1)だけ離れた半球を通して見た面積の総和を求めると



$$\int_0^{\pi/2} p\cos\phi\cdot 2\pi\sin\phi\,d\phi = \pi p$$

拡散反射光はπ倍の面積に広がるのと同等なので，
視点方向への反射光の量は入射光の$1/\pi$を超えない

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

- BRDF(双方向反射率分布関数)
  - 各種の反射率を統合するモデル
  - 任意方向からの入射光に関する
    任意方向への反射の割合の分布

mは「金属性」

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

- 物理学に基づく鏡面反射モデル
  - クック・トランスのモデルなど

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

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

  - フレネル反射：水平に近い入射光に対して反射率が高くなる現象
  - 「表面の粗さ」の影響もモデル化