コンピューターグラフィックスS

第12回　シェーディング、マッピング

システム創成情報工学科　尾下 真樹

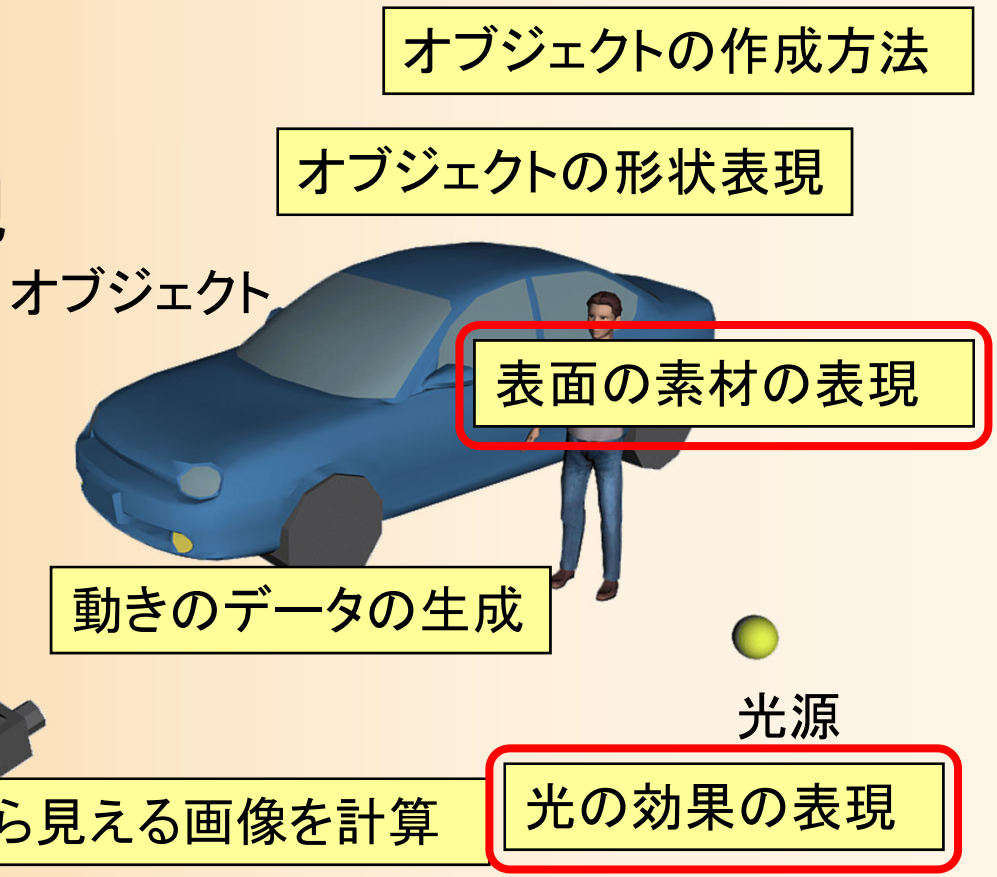
2019年度 Q2

今回の内容

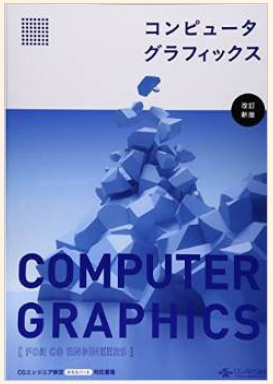
- 前回の復習
  - シェーディング
- シェーディング（前回の続き）
  - OpenGLでの光源情報の設定
  - ラジオシティ
  - 影の表現
  - BRDF
- マッピング

今回の内容

- シェーディング
  - 光の効果の表現
- マッピング
  - 表面の素材の表現



教科書(参考書)



- 「コンピュータグラフィックス」
  CG-ARTS協会 編集・出版
  - 4章 4-4、4-7(詳しい)
  - 4章 4-5
- 「ビジュアル情報処理 ―CG・画像処理入門―」
  CG-ARTS協会 編集・出版
  - 4章 4-4～4-5
  - 4章 4-6

参考書

- 「コンピュータグラフィックス」
  CG-ARTS協会 編集・出版（3,200円）
  - 4章
- 「3DCGアニメーション」
  栗原恒弥 安生健一 著、技術評論社 出版
  - 第2章（68～108ページ）
- 「3次元CGの基礎と応用」
  千葉則茂 土井章男 著、サイエンス社 出版
  - 第2章（23～28ページ）、第4章（35～39ページ）、
    第5章（40～49ページ）、第8章（73～75ページ）、
    第9章（79～90ページ）

前回の復習

光のモデル

- 輝度の計算式
  - 全ての光による影響を足し合わせることで、物体上の点の輝度（RGBの値）が求まる
    - 各Iは光の明るさ（RGB）
    - 各kは物体の反射特性（RGB）

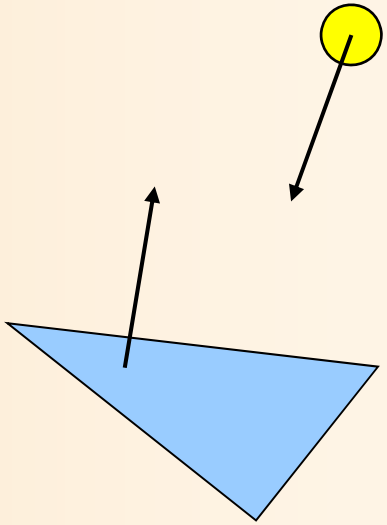
$$I = I_a k_a + \sum_{i=1}^{n_L} I_i \left[k_d \left(N \cdot L \right) + k_s \left(R \cdot V \right)^n \right] + k_r I_r + k_t I_t$$

環境光　拡散反射光　鏡面反射光（局所照明）　鏡面反射光（大域照明）　透過光

それぞれの光源からの光（局所照明）　大域照明

光のモデル

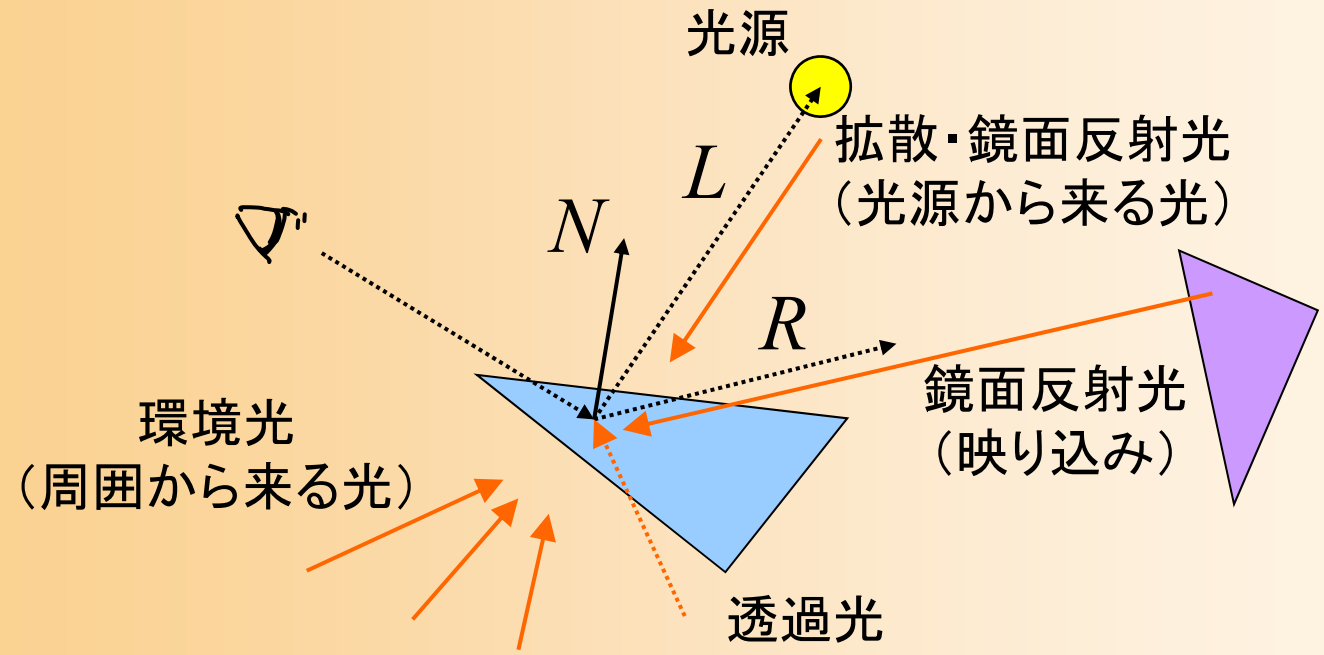
- 局所照明モデル
  - 光源と一枚の面の関係のみを考慮したモデル
    - 環境光、拡散反射光、鏡面反射光



- 大域照明モデル
  - 周囲の物体の影響も考慮したモデル
    - 環境光、鏡面反射光、透過光

  - 同じ種類の光でも考慮する範囲に応じて局所モデルと大域モデルがあるので注意

光のモデルのまとめ



$$I = I_a k_a + \sum_{i=1}^{n_L} I_i \left[k_d \left(N \cdot L \right) + k_s \left(R \cdot V \right)^n \right] + k_r I_r + k_t I_t$$

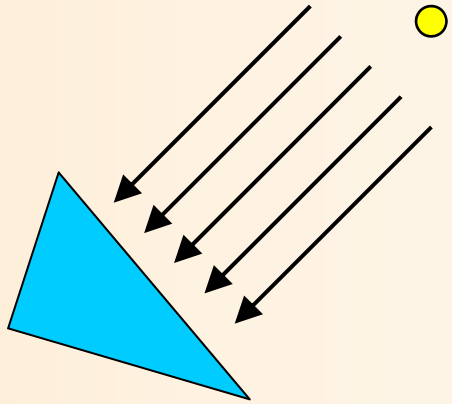
環境光　拡散反射光　鏡面反射光（局所照明）　鏡面反射光（大域照明）　透過光

それぞれの光源からの光（局所照明）　大域照明

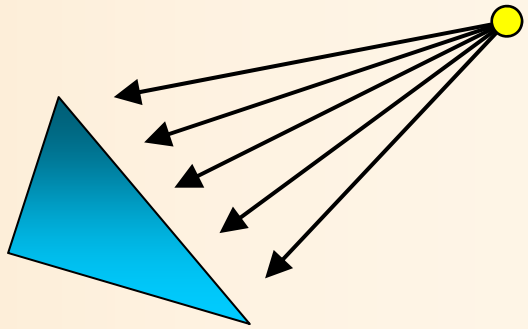
光源の種類

- 平行光源
  - 一定方向からの光源
  - 計算量が最も少ない
  - 太陽などの遠くにある光源の表現に適している

無限遠に光源があると見なす



- 点光源
  - 位置の決まった光源
  - ライトなどの表現に適している
  - 光の方向は点光源と面の位置関係により決まる
  - 光の減衰も考慮できる



光のモデルの計算

- OpenGLによる光の効果の計算
  - 設定された光源情報、及び、法線に従い、局所照明モデルのみを計算
    - 環境光には、一定の明るさを指定可能
    - 光源の種類や位置を指定可能（複数指定可能）

$$I = I_a k_a + \sum_{i=1}^{n_L} I_i \left[k_d \left(N \cdot L \right) + k_s \left(R \cdot V \right)^n \right] + k_r I_r + k_t I_t$$

環境光　拡散反射光　鏡面反射光（局所照明）　鏡面反射光（大域照明）　透過光

それぞれの光源からの光（局所照明）　大域照明

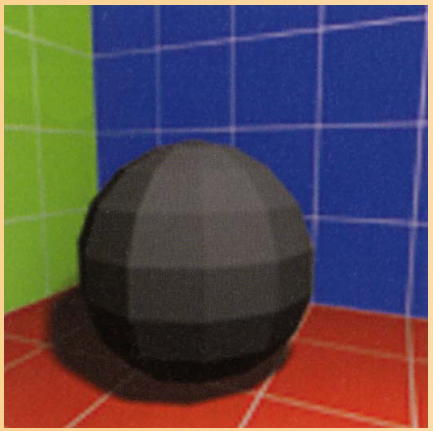
プログラムの例

- 光源の位置や色の設定（詳細は後日の演習）
  - 以下の例では、環境光と、一つの点光源を設定

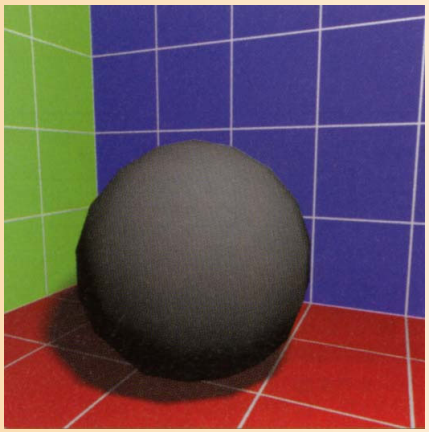
```
float  light0_position[] = { 10.0, 10.0, 10.0, 1.0 };
float  light0_diffuse[] = { 0.8, 0.8, 0.8, 1.0 };
float  light0_specular[] = { 1.0, 1.0, 1.0, 1.0 };
float  light0_ambient[] = { 0.1, 0.1, 0.1, 1.0 };
glLightfv( GL_LIGHT0, GL_POSITION, light0_position );
glLightfv( GL_LIGHT0, GL_DIFFUSE, light0_diffuse );
glLightfv( GL_LIGHT0, GL_SPECULAR, light0_specular );
glLightfv( GL_LIGHT0, GL_AMBIENT, light0_ambient );
glEnable( GL_LIGHT0 );
glEnalbe( GL_LIGHTING );
```

シェーディングの方法

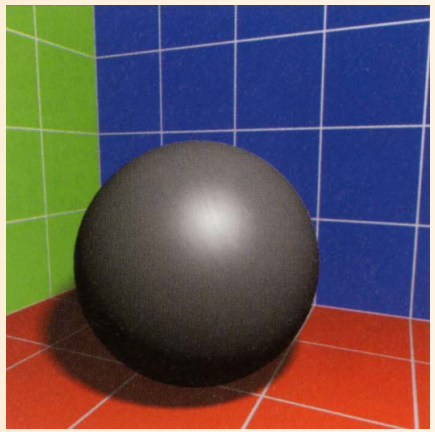
- フラットシェーディング
- スムーズシェーディング
  - グローシェーディング
  - フォンシェーディング



フラットシェーディング



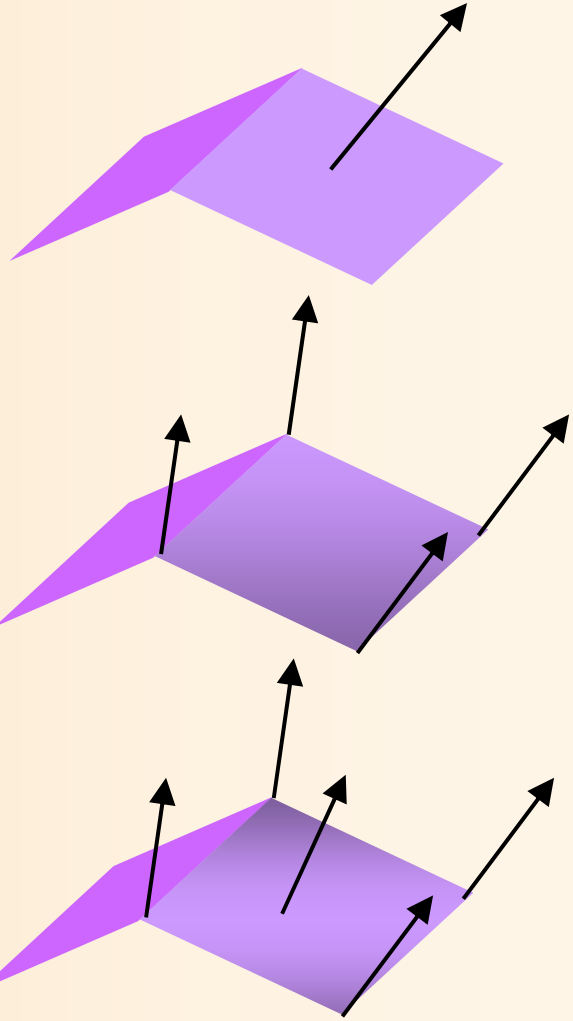
グローシェーディング



フォンシェーディング

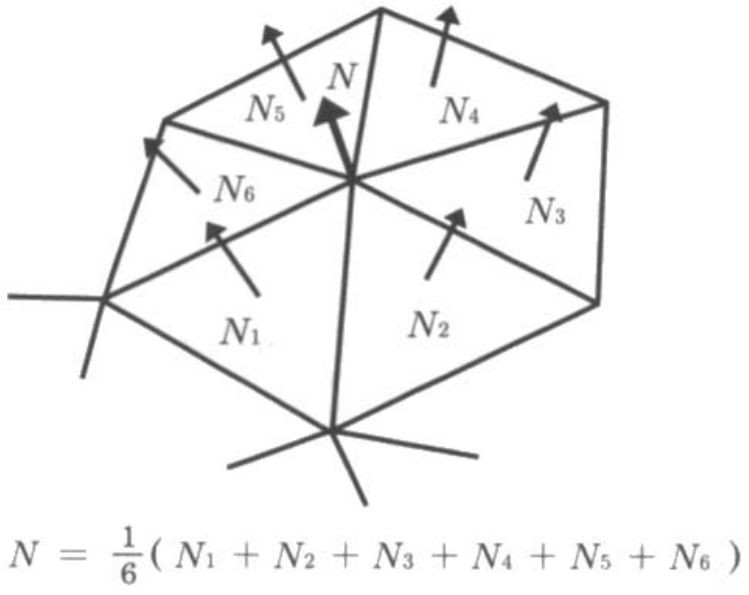
シェーディングの処理のまとめ

- フラットシェーディング
  - 面の法線から面の色を計算
- グローシェーディング
  - 頂点の法線から頂点の色を計算
  - 頂点の色から、各点の色を決定
- フォンシェーディング
  - 頂点の法線から、面内の各点（ピクセル）の法線を計算
  - 各点の法線から、色を計算



頂点の法線

- 頂点の法線
  - もともと頂点には法線という概念はない
  - シェーディングを計算するために、頂点の法線を利用
- 計算方法
  - 頂点に隣接する全ての面の法線を平均
    - 面の面積に応じて加重平均する方法もある



基礎と応用 図4.2

レポート課題

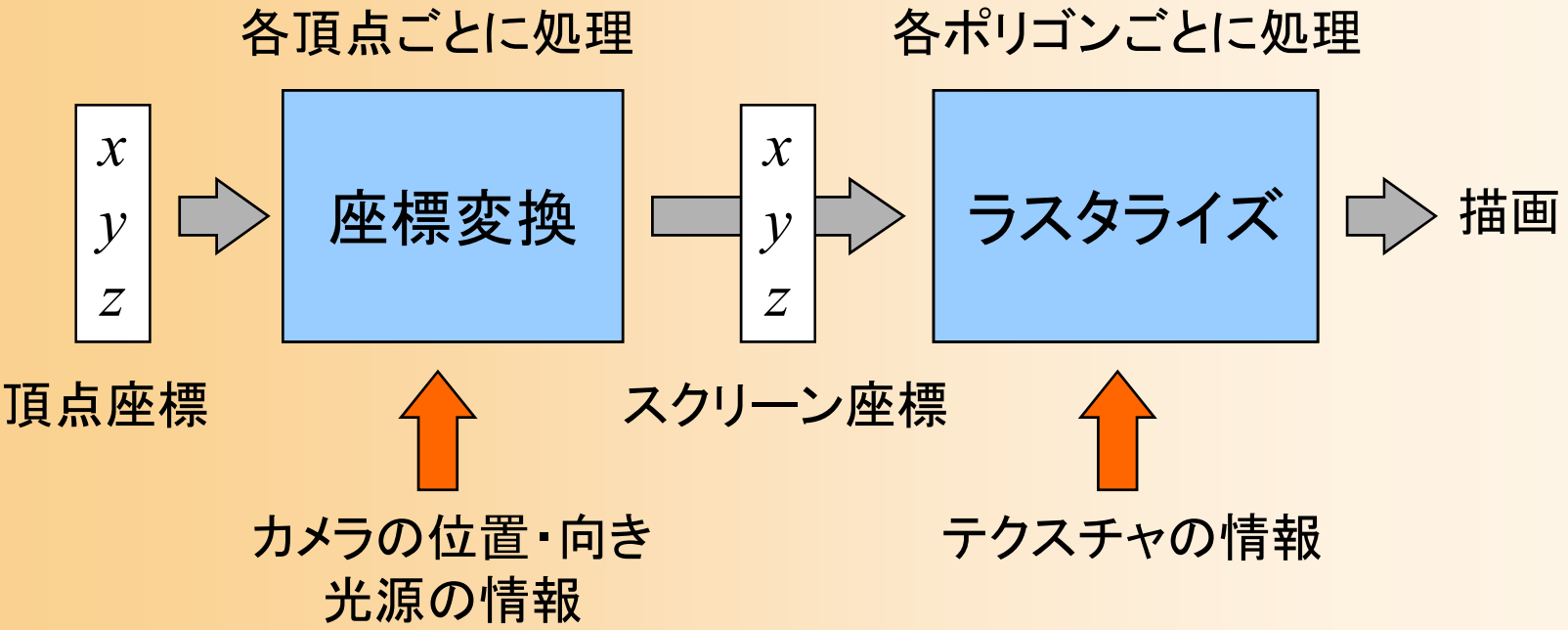
- OpenGLを使った課題プログラムの作成
  - 各自、自分に与えられた課題を実現するプログラムを作成する
    - ポリゴンモデルの描画
    - 視点操作インターフェースの拡張
    - アニメーション
- Moodleから提出
  - レポート、作成したプログラム一式を提出
- レポートの締め切りは後日連絡
  - 8月上旬（期末試験後）の締め切りを予定

OpenGLでの光源情報の設定

OpenGLの光源処理の概要

- 光源と物体の素材（頂点の色）・法線によって、描画される頂点（ポリゴン）の色が決まる
- OpenGLの光源処理
  - OpenGLの関数を使って、光源や物体の素材・法線の情報を指定
  - OpenGLは、各頂点ごとに、自動的に光源処理を行い、各頂点の色を決定
    グローシェーディングにより、各頂点の色をもとに、ポリゴンが描画される

レンダリング・パイプライン



- 光源の情報、物体の素材、法線を指定
- 各頂点ごとに、自動的に光源処理を行い、各頂点の色を決定（座標変換）
- グローシェーディングにより、各頂点の色をもとに、ポリゴンが描画される（ラスタライズ）

光のモデル（復習）

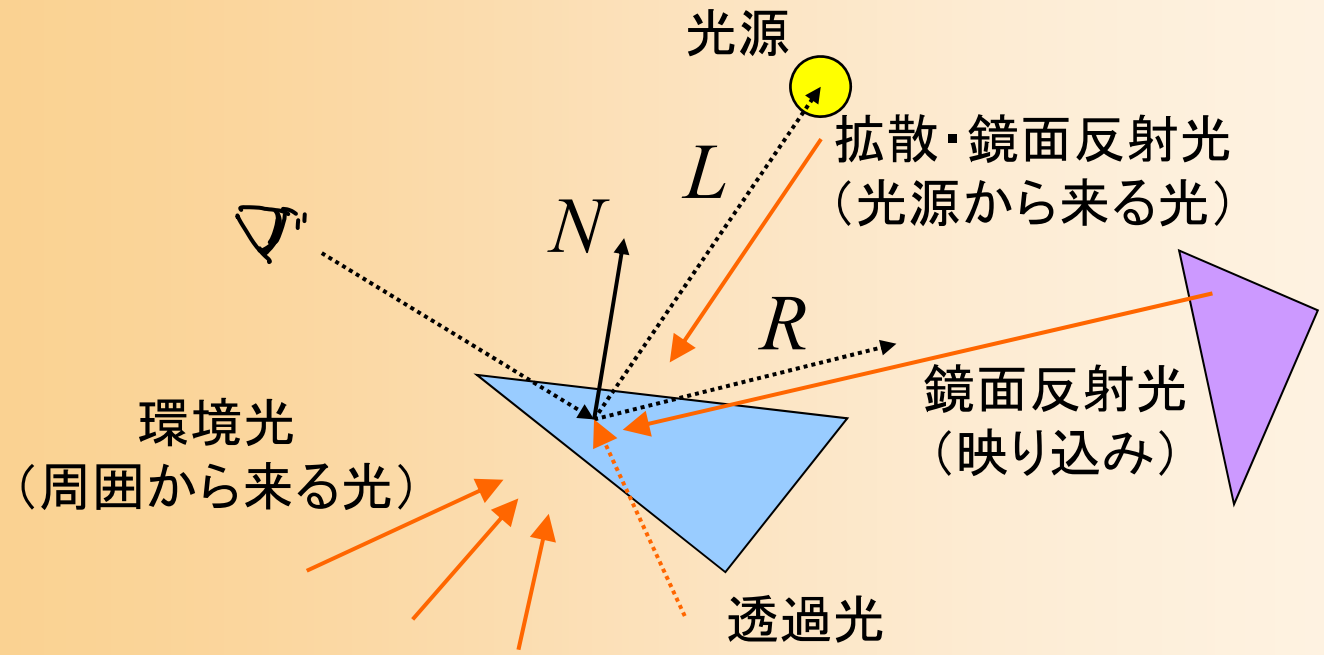
- 輝度の計算式
  - 全ての光による影響を足し合わせることで、物体上の点の輝度（RGBの値）が求まる
    - 各Iは光の明るさ（RGB）
    - 各kは物体の反射特性（RGB）

$$I = I_a k_a + \sum_{i=1}^{n_L} I_i \left[k_d \left(N \cdot L \right) + k_s \left(R \cdot V \right)^n \right] + k_r I_r + k_t I_t$$

環境光　拡散反射光　鏡面反射光（局所照明）　鏡面反射光（大域照明）　透過光

それぞれの光源からの光（局所照明）　大域照明

光のモデル（復習）



$$I = I_a k_a + \sum_{i=1}^{n_L} I_i \left[k_d \left(N \cdot L \right) + k_s \left(R \cdot V \right)^n \right] + k_r I_r + k_t I_t$$

環境光　拡散反射光　鏡面反射光（局所照明）　鏡面反射光（大域照明）　透過光

それぞれの光源からの光（局所照明）　大域照明

OpenGLの光源処理

- 光のモデルにもとづき、各光源による輝度を、RGBごとに次式で計算して加算

$$Color = L_{\text{ambient}} \cdot M_{\text{ambient}} + \max\{\ \mathbf{l}\cdot\mathbf{n},\ 0\ \}\ L_{\text{diffuse}} \cdot M_{\text{diffuse}}$$
$$+ \max\{\ \mathbf{s}\cdot\mathbf{n},\ 0\ \}^{M_{\text{specular\_factor}}}\ L_{\text{specular}} \cdot M_{\text{specular}}$$

  - max{A, B}は、A, B のうち大きい値を使用
    内積が負の場合は、その項は0になる
  - 全ての値を足し合わせた結果は、0.0～1.0の範囲に丸められる

- $L_{\text{ambient}}, L_{\text{diffuse}}, L_{\text{specular}}$ は光の輝度
- $M_{\text{ambient}}, M_{\text{diffuse}}, M_{\text{specular}}, M_{\text{specular\_factor}}$ は素材の特性

光源情報の設定

- 光源情報の設定
  - glLight(), glLightv() 関数 を使用
    - 光源番号、設定パラメタの種類、設定する値、を指定
    - glLight() 関数はスカラ値を設定
    - glLightv() 関数はベクトル値を設定
- 光源処理を有効にする
  - 光源処理を有効にする glEnable(GL_LIGHTING)
  - 各光源の影響を有効にする glEnable(GL_LIGHT0)

光源情報の設定の例(1)

- 初期化処理での設定

```
float  light0_position[] = { 10.0, 10.0, 10.0, 1.0 };
float  light0_diffuse[] = { 0.8, 0.8, 0.8, 1.0 };
float  light0_specular[] = { 1.0, 1.0, 1.0, 1.0 };
float  light0_ambient[] = { 0.1, 0.1, 0.1, 1.0 };

glLightfv( GL_LIGHT0, GL_POSITION, light0_position );
glLightfv( GL_LIGHT0, GL_DIFFUSE, light0_diffuse );
glLightfv( GL_LIGHT0, GL_SPECULAR, light0_specular );
glLightfv( GL_LIGHT0, GL_AMBIENT, light0_ambient );

glEnable( GL_LIGHT0 );
glEnalbe( GL_LIGHTING );
```

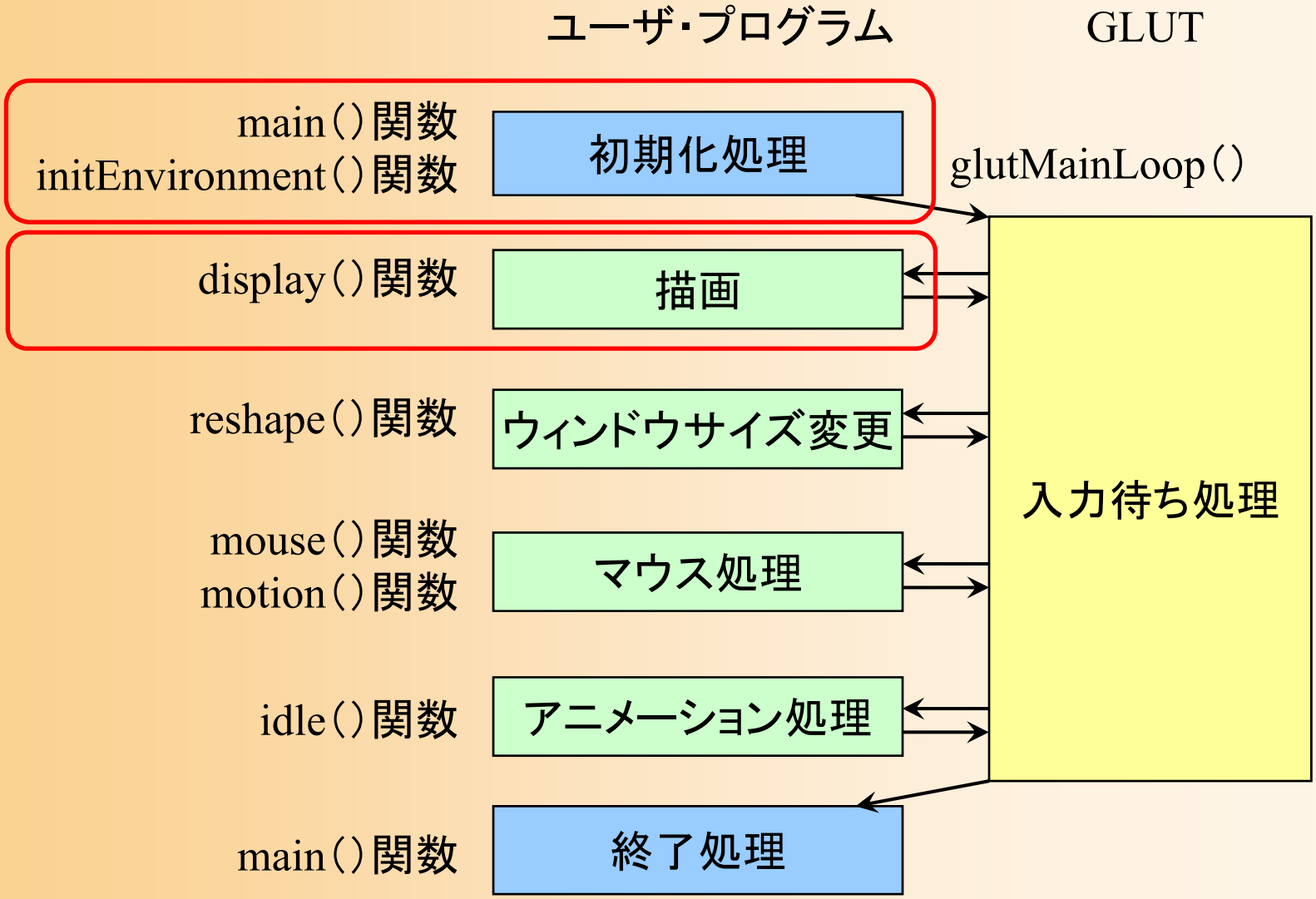
詳細は、後ほど説明

光源情報の設定の例(2)

- 変換行列の変更後に、光源位置を再設定
  - 光源計算は、カメラ座標系で適用されるため

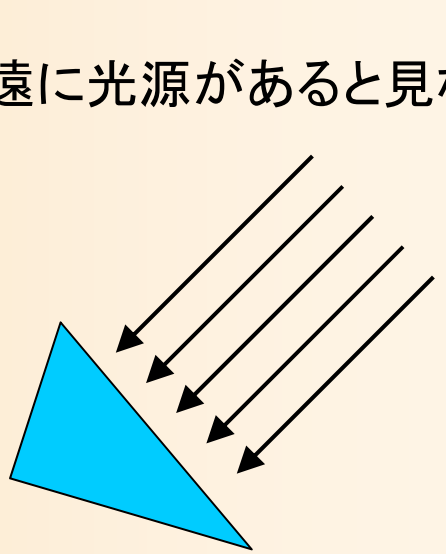
```
void  display( void )
{
        ・・・・・・
        // 変換行列を設定(ワールド座標系→カメラ座標系)
        glMatrixMode( GL_MODELVIEW );
        ・・・・・・

        // 光源位置を設定(変換行列の変更にあわせて再設定)
        float  light0_position[] = { 10.0, 10.0, 10.0, 1.0 };
        glLightfv( GL_LIGHT0, GL_POSITION, light0_position );
        ・・・・・・
```

サンプルプログラムの構成（確認）

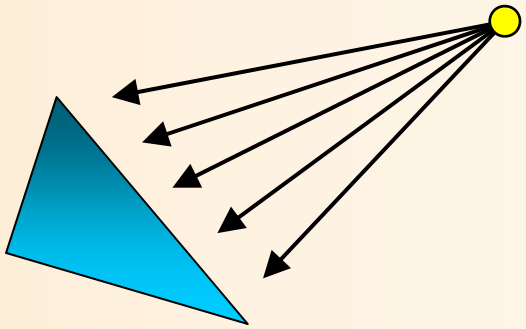


光源の種類と設定方法(1)

- 平行光源
  - (x,y,z)の方向から平行に光が来る
  - 光源位置のw座標を0.0に設定



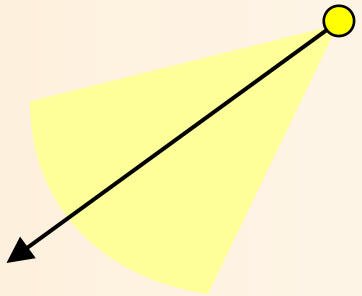
- 点光源
  - (x,y,z)の位置に光源がある
  - 光源位置のw座標を1.0に設定

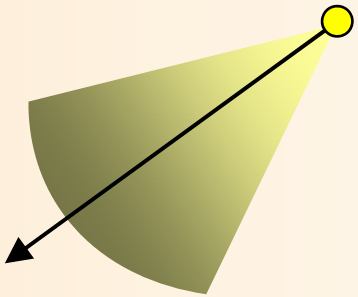


光源の種類と設定方法(2)

- スポットライト光源
  - 点光源にさらに、スポットライトの向き・角度範囲などの情報を設定したもの
- 光源の減衰も設定可能
  - 点光源・スポットライト光源から距離が離れるほど暗くなるような効果を加える
- 設定方法の説明は省略

指定した方向・角度にのみ有効な点光源





光源情報の設定の例

- サンプルプログラムの例

```
float  light0_position[] = { 10.0, 10.0, 10.0, 1.0 };
float  light0_diffuse[] = { 0.8, 0.8, 0.8, 1.0 };
float  light0_specular[] = { 1.0, 1.0, 1.0, 1.0 };
float  light0_ambient[] = { 0.1, 0.1, 0.1, 1.0 };

glLightfv( GL_LIGHT0, GL_POSITION, light0_position );
glLightfv( GL_LIGHT0, GL_DIFFUSE, light0_diffuse );
glLightfv( GL_LIGHT0, GL_SPECULAR, light0_specular );
glLightfv( GL_LIGHT0, GL_AMBIENT, light0_ambient );

glEnable( GL_LIGHT0 );
glEnalbe( GL_LIGHTING );
```

光源位置のw座標が1.0なので、点光源となる

LIGHT0の
・光源の位置・種類
・拡散反射成分の色
・鏡面反射成分の色
を設定

LIGHT0の
・環境光成分の色
を設定

光源位置・向きの設定（注意）

- 光源の位置・向きは、描画関数内で毎回更新
  - 光源の位置・向きは、現在のカメラ座標系にもとづいて設定されるため、カメラが移動・回転する度に設定し直す必要がある

```
void  display( void )
{
        ・・・・・・
        // 光源位置を設定（変換行列の変更にあわせて再設定）
        float  light0_position[] = { 10.0, 10.0, 10.0, 1.0 };
        glLightfv( GL_LIGHT0, GL_POSITION, light0_position );
        ・・・・・・
```

一般的な光源の設定方針

- LIGHT0を使って環境の主な光源を設定
  - その環境の明るさに応じて環境光を設定
  - 全体の明るさを決めるような、平行光源or点光源を設定
- LIGHT1以降を使って追加の光を設定
  - 電灯や車など、空間中にあるオブジェクトが周囲のオブジェクトを照らすような場合に、点光源やスポットライトを追加する
  - ２番目以降の光源では、環境光はあまり大きくしないことが多い

素材の設定

- 頂点の色の設定
  - glColor()関数
    - デフォルトでは、頂点の環境特性と拡散反射特性を同時に設定（個別に設定することも可能）
- その他の素材特性を個別に設定（詳細は省略）
  - glMaterial()関数
  - 環境特性、拡散反射特性、鏡面反射特性、鏡面反射係数など

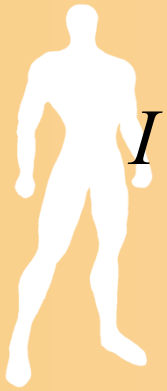
$$Color = L_{\text{ambient}} \cdot M_{\text{ambient}} + \max\{\ \mathbf{l}\cdot\mathbf{n},\ 0\ \}\ L_{\text{diffuse}} \cdot M_{\text{diffuse}}$$

$$+ \max\{\ \mathbf{s}\cdot\mathbf{n},\ 0\ \}^{M_{\text{specular\_factor}}}\ L_{\text{specular}} \cdot M_{\text{specular}}$$

ラジオシティ

ラジオシティ

- 環境光をより正しく計算するための方法
  - 面同士の相互反射を考慮
  - 各面ごとの環境光を計算する方法
    - これまでのモデルでは、環境光は一定と仮定していたため、周囲の明るさによる影響を表現できなかった
  - どのレンダリング手法とも組み合わせが可能
    - ただし非常に計算時間がかかる

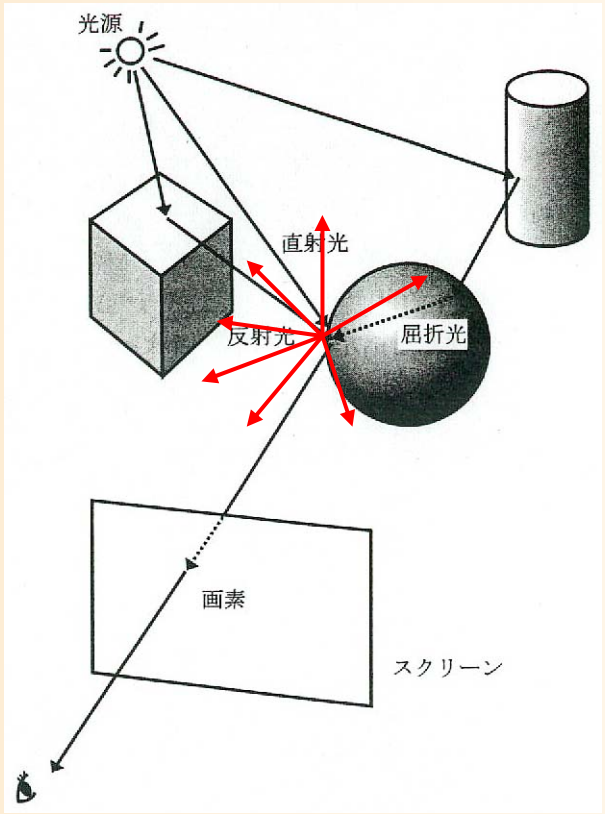


$$I = I_a k_a + \sum_{i=1}^{n_L} I_i \left[k_d \left(N \cdot L \right) + k_s \left(R \cdot V \right)^n \right] + k_r I_r + k_t I_t$$

環境光　　拡散反射光　鏡面反射光（局所照明）　鏡面反射光（大域照明）　透過光

レイトレーシングの限界

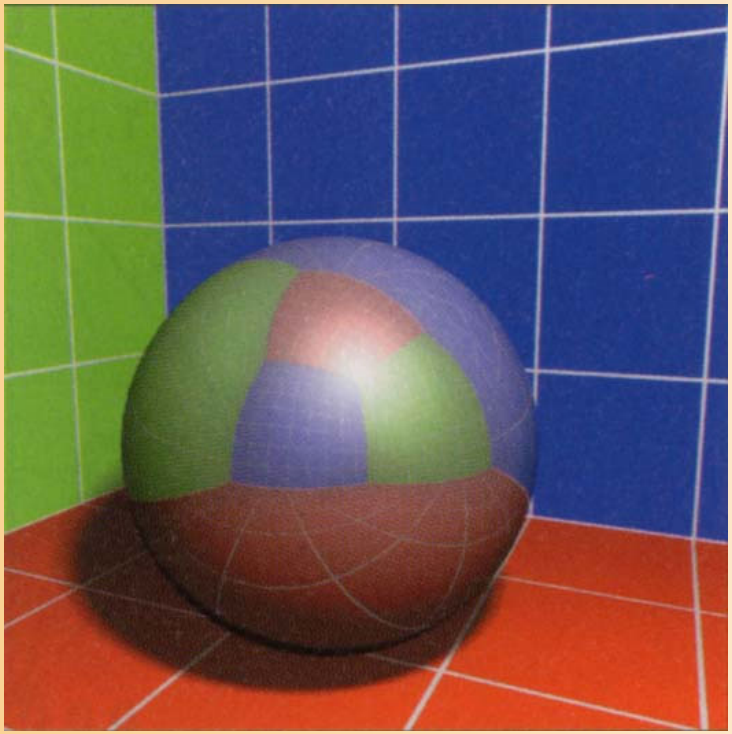
- 反射光や透過光はほぼ正しく計算できる
- 環境光を正しく計算するのは難しい
  - 環境光を計算しようとすると、レイと物体の衝突点から、無数の方向にレイを飛ばす必要があるため

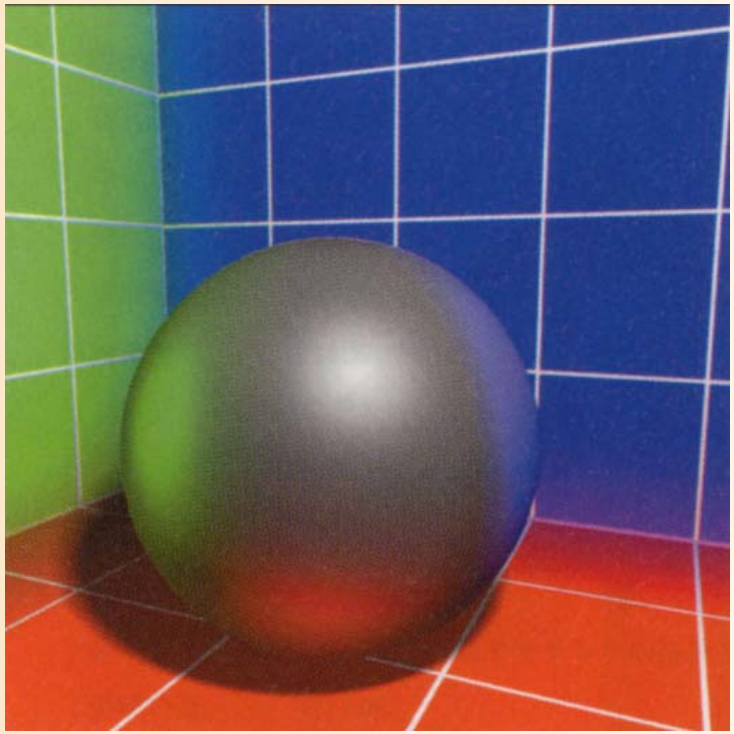


基礎と応用 図8.2

ラジオシティの具体例

- レイトレーシングとラジオシティの比較
  - 鏡面反射による映り込み（レイトレーシング）と、拡散反射による色の影響（ラジオシティ）

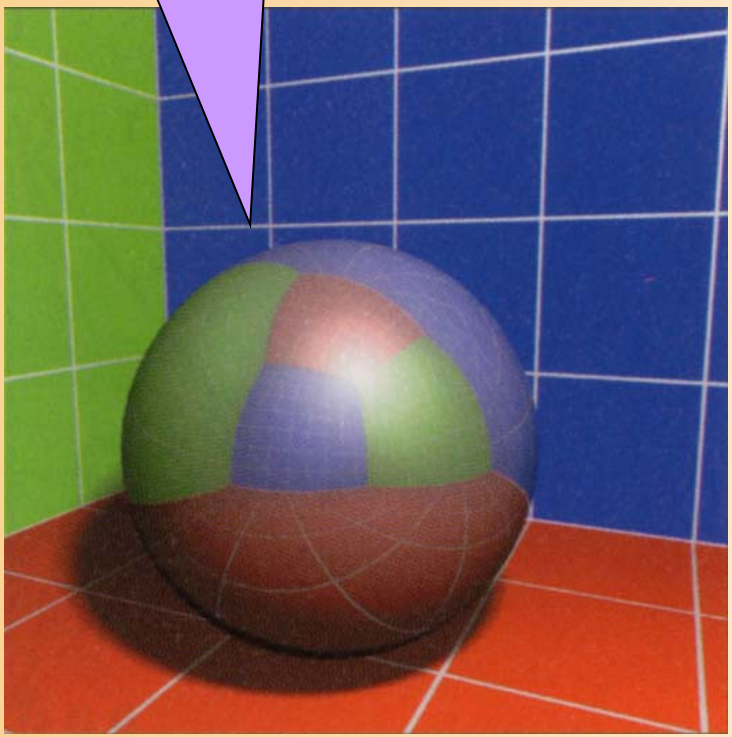


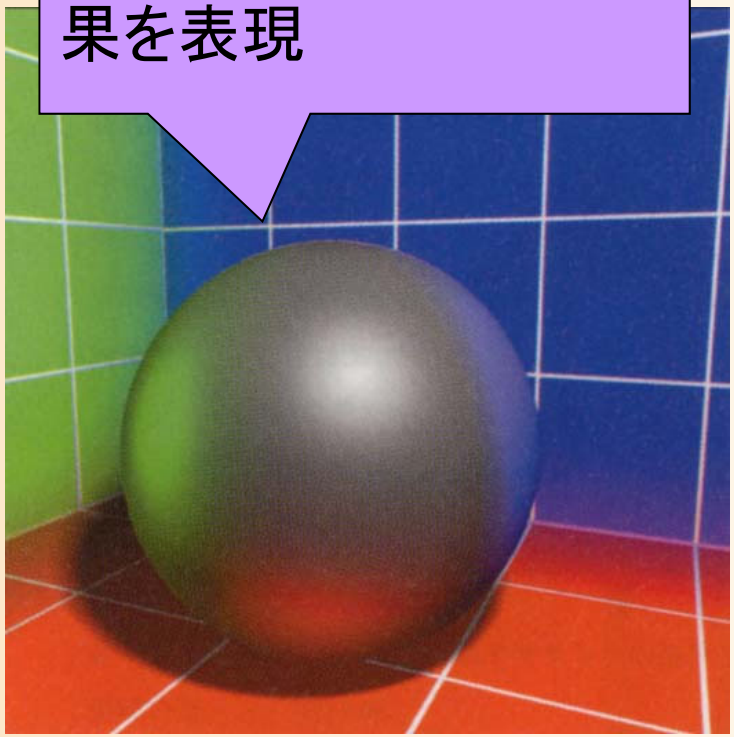


ラジオシティの具体例

レイトレーシング法
鏡面反射により、周囲の一点の色が映り込む

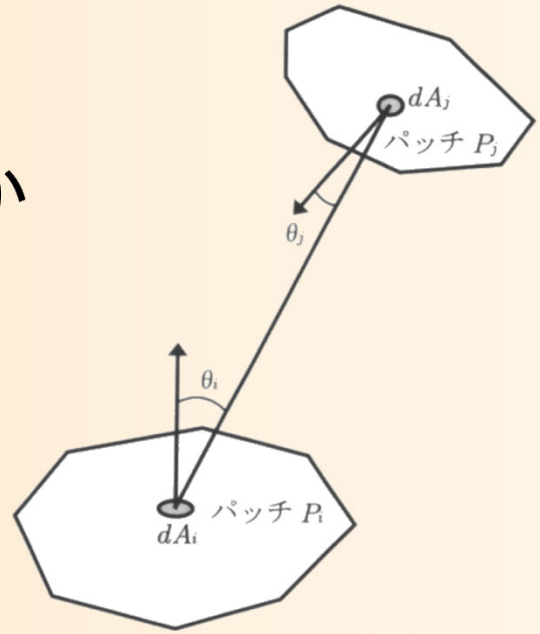
ラジオシティ法
周囲の面との明るさの相互影響を計算することで、周囲の面からぼんやりと照らされる効果を表現





ラジオシティの考え方(1)

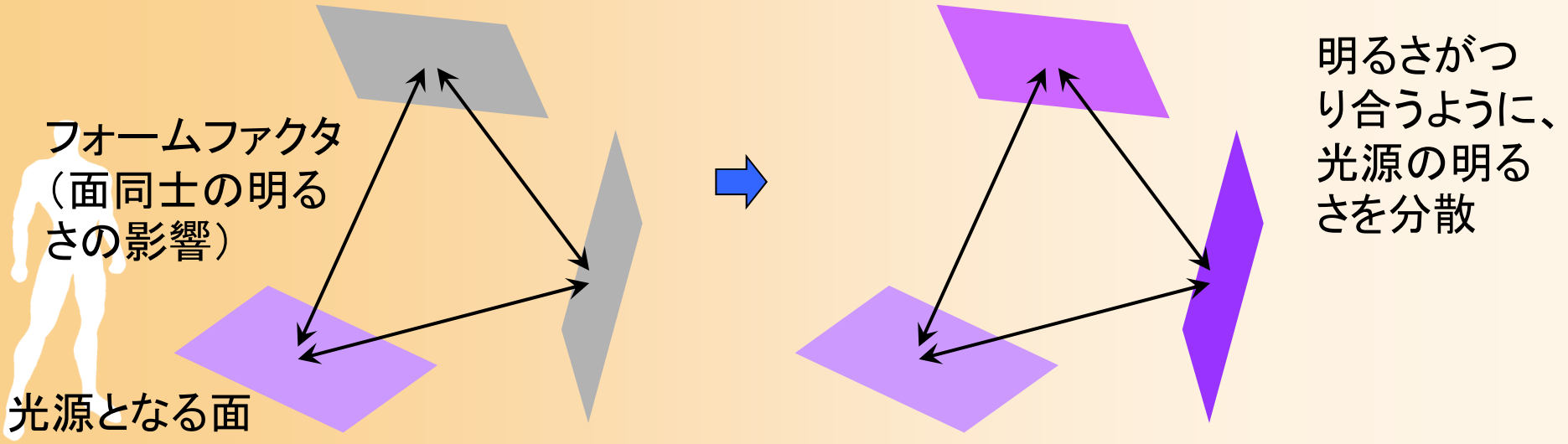
- ラジオシティ(熱の放射エネルギー)の考え方にもとづく、面ごとの環境光の計算
- 面同士の相互反射モデル
  - ある面の明るさが、別の面の明るさに、どの程度影響を与えるか(フォームファクタ)を計算
    - 面同士が向き合っており、障害物がなければ、大きな影響を与える(一方が明るくなると、もう一方も照らされて明るくなる)



基礎と応用 図9.3

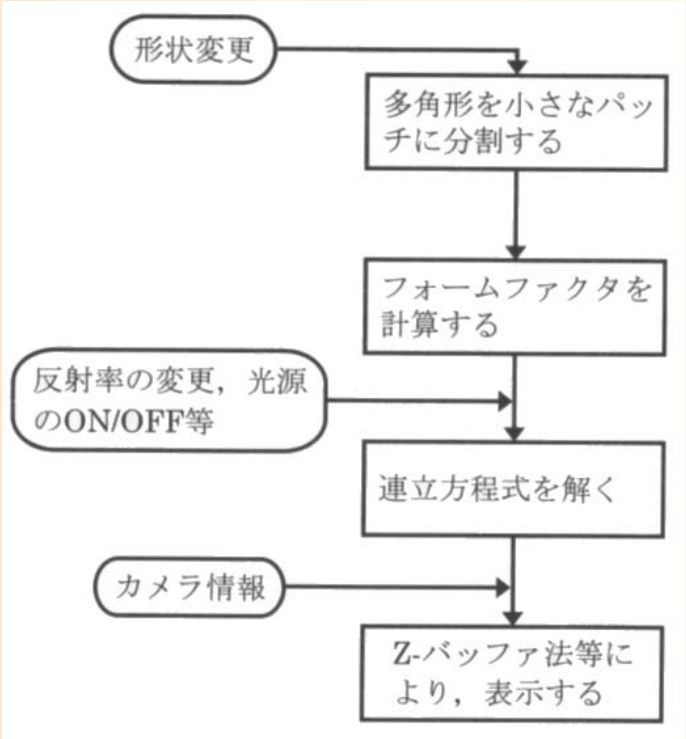
ラジオシティの考え方(2)

- フォームファクタをもとに、全ての面同士の明るさがつり合うような各面の明るさを計算
  - エネルギーの分散と同様の考え方にもとづいて、光源となる面の明るさを、フォームファクタの大きさに応じて、他の面に分散させていく



ラジオシティの計算手順

- ポリゴンを細かい面に分割
- フォームファクタを計算
- 光源の情報などを設定
- 連立方程式を解く
- 計算結果の明るさを使ってレンダリング



基礎と応用 図9.4

ラジオシティ計算のための分割

- もとのポリゴンモデルを細かく分割する
  - それぞれの分割ポリゴンごとに、フォームファクタによる明るさを計算
  - ある程度細かく分割する必要がある

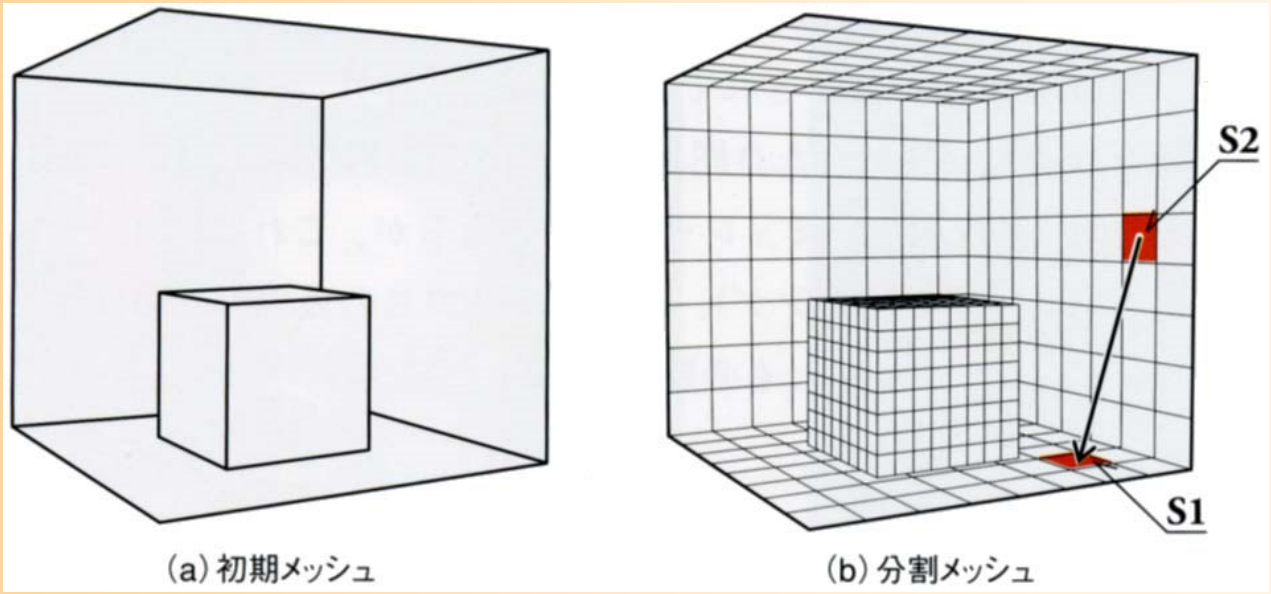
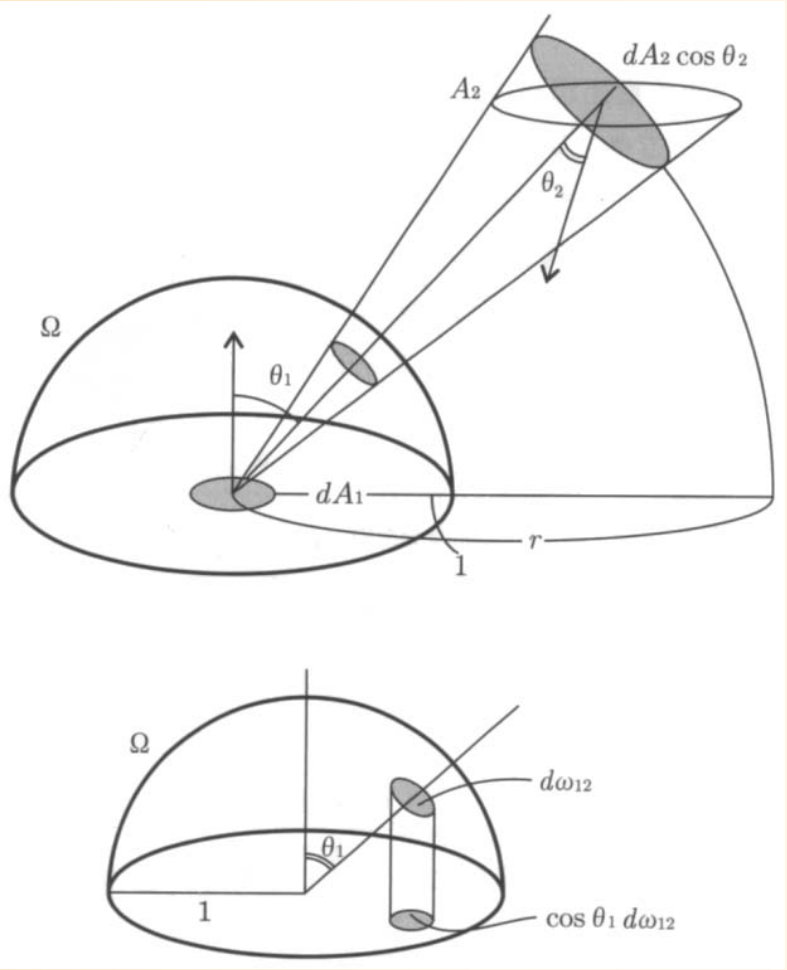


図2.38

フォームファクタの計算

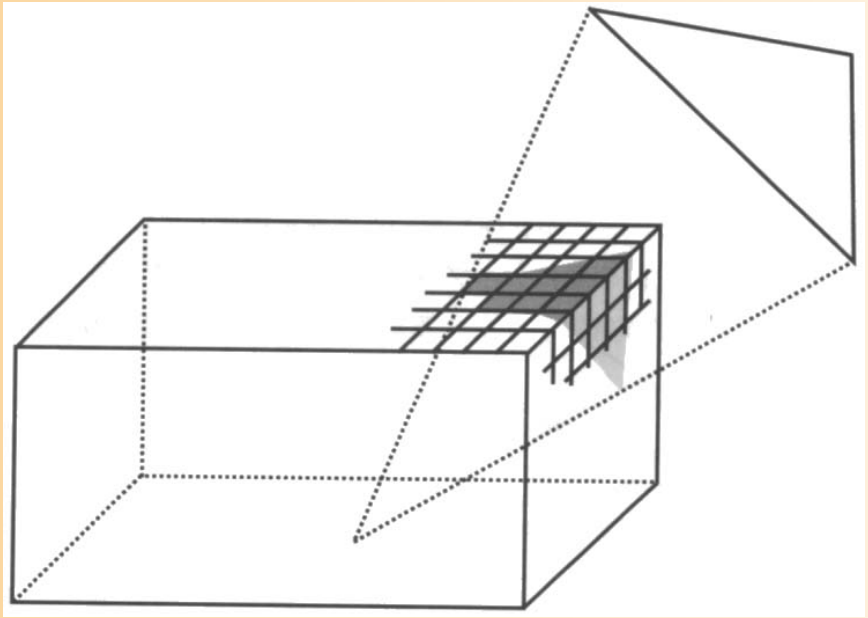
- フォームファクタの計算モデル
  - 球面に投影した面積を水平面に射影し、その面積の広さによりフォームファクタを計算



基礎と応用 図9.7

フォームファクタの計算

- 障害物を考慮した計算（ヘミキューブ法）
  - ある面の周囲にZバッファを持った面を置く
  - 全ての面を周囲の面に描画、それぞれの面の最終的な面積によってフォームファクタを決める



基礎と応用 図9.8

連立方程式の計算

- 面の光の計算式

$$B_i = E_i + \rho_i \sum_{j=1}^{n} B_j F_{ij}$$

B_i 光の強さ　E_i 放射光の強さ

ρ_i 反射率　F_{ij} フォームファクタ

- 連立方程式

$$\begin{bmatrix} 1-\rho_1 F_{11} & -\rho_1 F_{12} & \cdots & -\rho_1 F_{1n} \\ -\rho_2 F_{21} & 1-\rho_2 F_{22} & \cdots & -\rho_2 F_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ -\rho_n F_{n1} & -\rho_n F_{n2} & \cdots & 1-\rho_n F_{nn} \end{bmatrix} \begin{bmatrix} B_1 \\ B_2 \\ \vdots \\ B_n \end{bmatrix} = \begin{bmatrix} E_1 \\ E_2 \\ \vdots \\ E_n \end{bmatrix} E_i + \rho_i \sum_{j=1}^{n} B_j F_{ij}$$

– ガウス-ザイデル法などを使って、解が収束するまで繰り返し計算

ラジオシティの具体例

基礎と応用 図9.1, 9.2

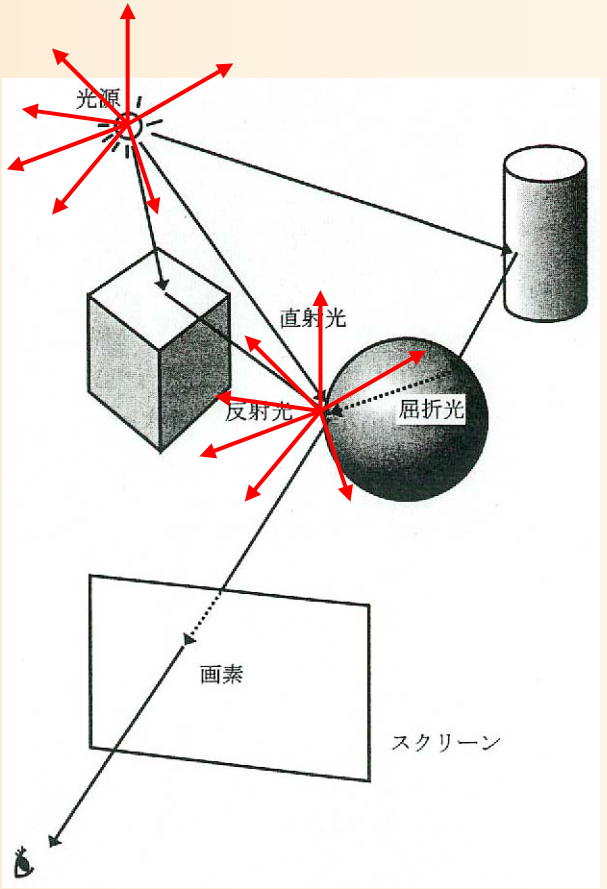
基礎と応用 図5.4

ラジオシティの特徴

- どのレンダリング手法とも組み合わせ可能
  - レンダリングの前処理として計算される
  - 光源や物体が動かなければ再計算の必要がないので、リアルタイム処理でも利用可能
    - 前もってラジオシティを計算した結果を保存しておく
    - ウォークスルーなどでの利用
- 自然な照明の効果を計算するためには必須の技術

他の大域照明計算手法

- モンテカルロ法
  - 1本のレイを追跡するレイトレーシングとは異なり、複数の経路を追跡する手法
  - 乱数にもとづいて、反射面から複数の経路を生成する
- フォトンマッピング法
  - 光源から放射される光を追跡することで、各点における明るさを計算する手法

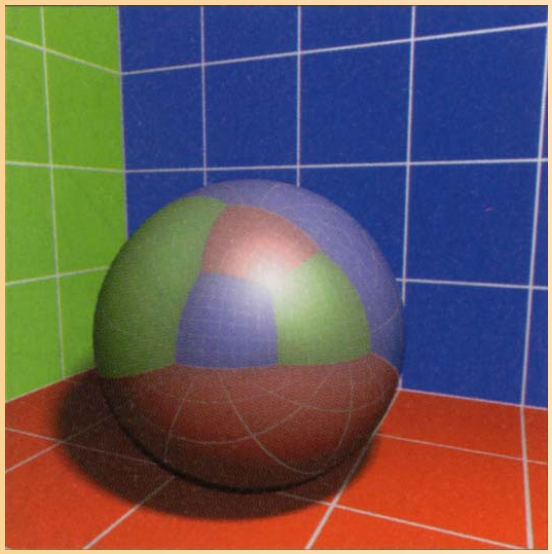


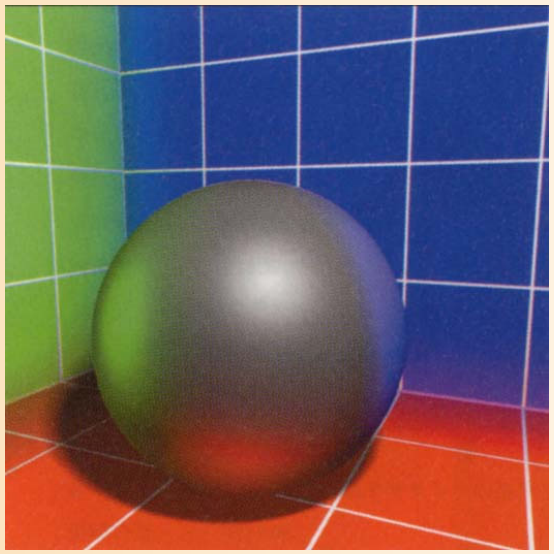
基礎と応用 図8.2

影の表現

影の表現方法

- レイトレーシングやラジオシティなどを用いることで、影も自動的に表現できる





CG制作独習事典 p.5

- コンピュータゲームなどで使用するのは難しい
- 影は非常に重要なので、他の手段で特別に実現

影の擬似的な表現方法

- テクスチャマッピング
- 投影ポリゴン
- シャドウ・ヴォリューム、シャドウ・マップ、など
  - ここでは各手法の説明は省略


GLUT sample program
No Shadow
影の描画
Shadow

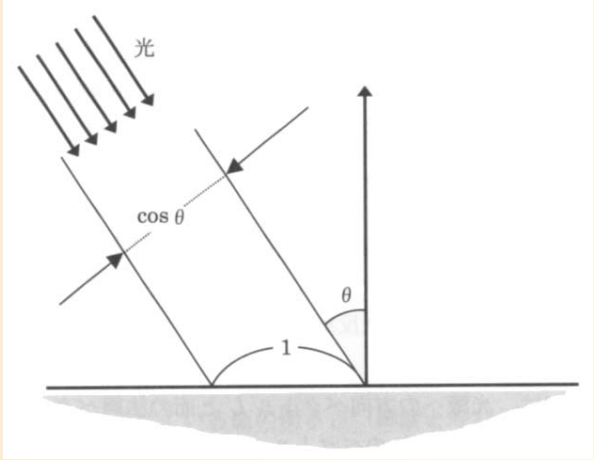
BRDF

BRDF

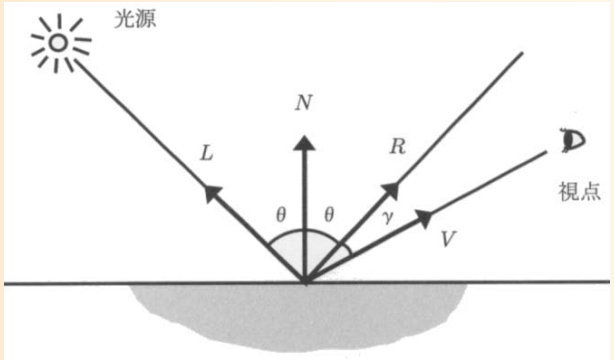
- BRDF
  - Bi-directional Reflectance Distribution Function
  - 双方向反射分布
- 現実世界の物質の反射特性を正確に再現するための技術
- イメージベースドレンダリングの考え方を発展させたもの

反射特性のモデル

- フォンのモデル
  - 拡散反射光 $I_d = k_d I_l (N \cdot L)$
  - 鏡面反射光 $I_d = k_s I_l (R \cdot V)^n$
- 現実の物体
  - 表面は平らではなく、乱反射などが起こる
  - モデルとのずれが生じる



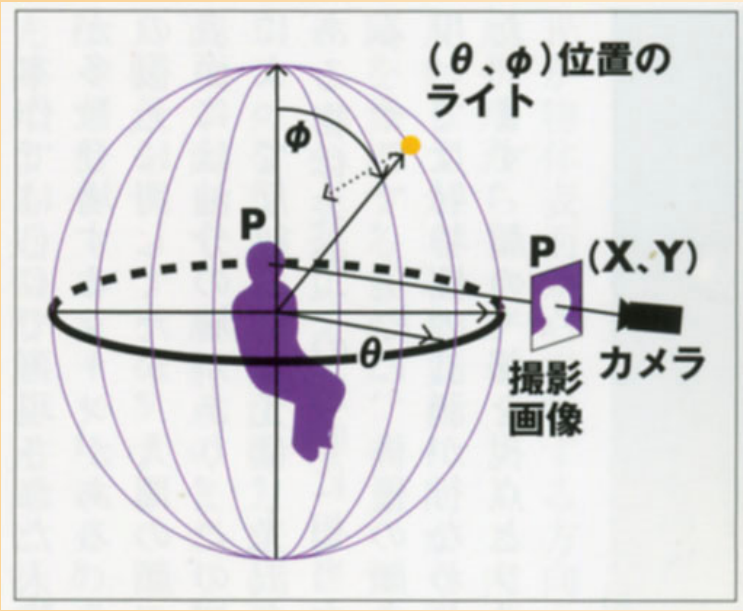
基礎と応用 図2.9

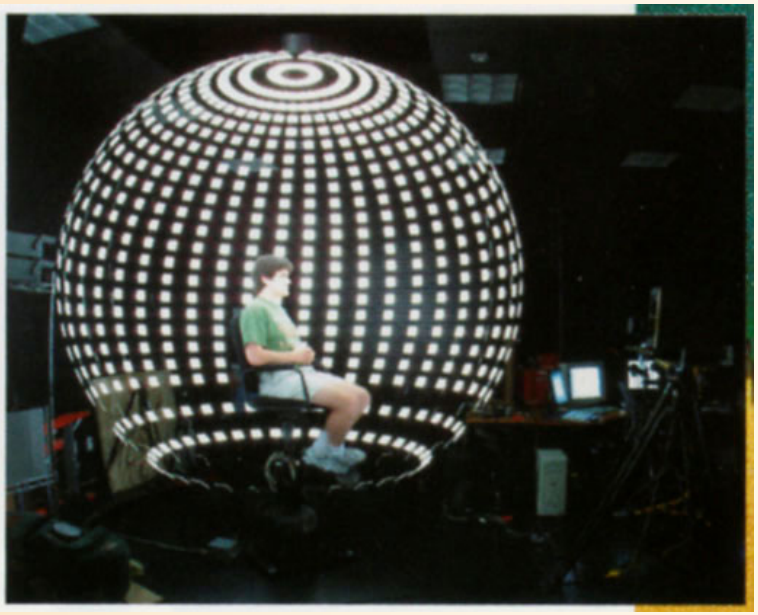


基礎と応用 図2.10

BRDFの考え方

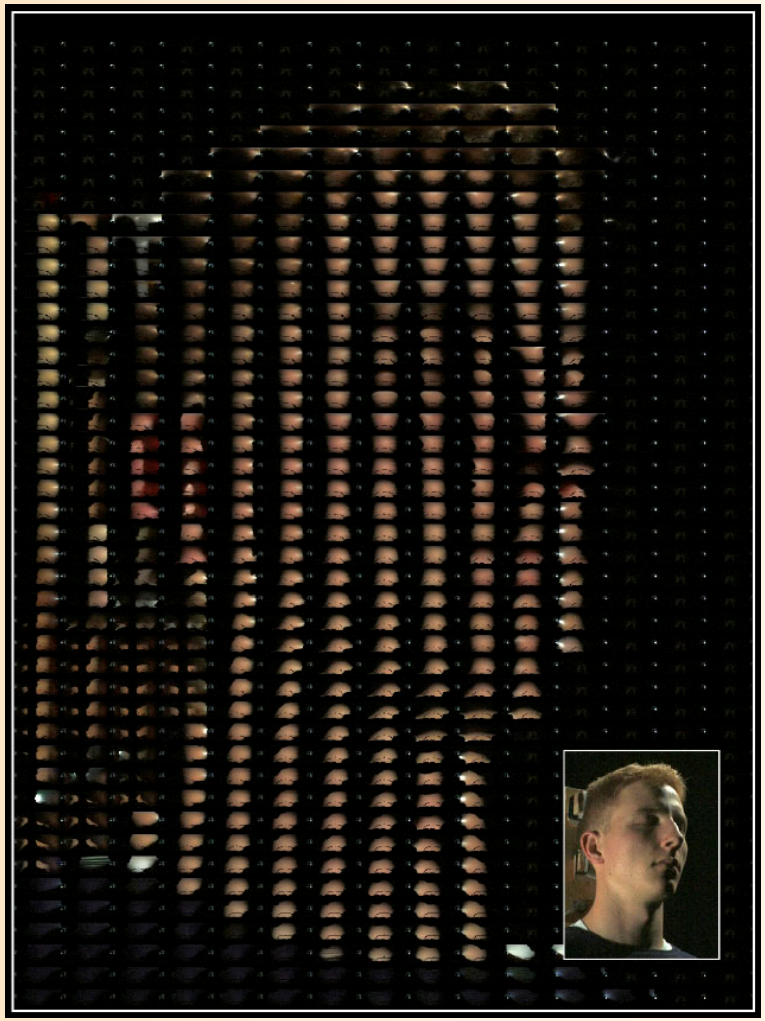
- 反射特性を表現
  - カメラ方向・光源方向の関数によって表される
    - 法線に対するカメラ方向・光源方向
  - 特殊な装置を使って実際の素材から計測





CG WORLD
2004年12月号

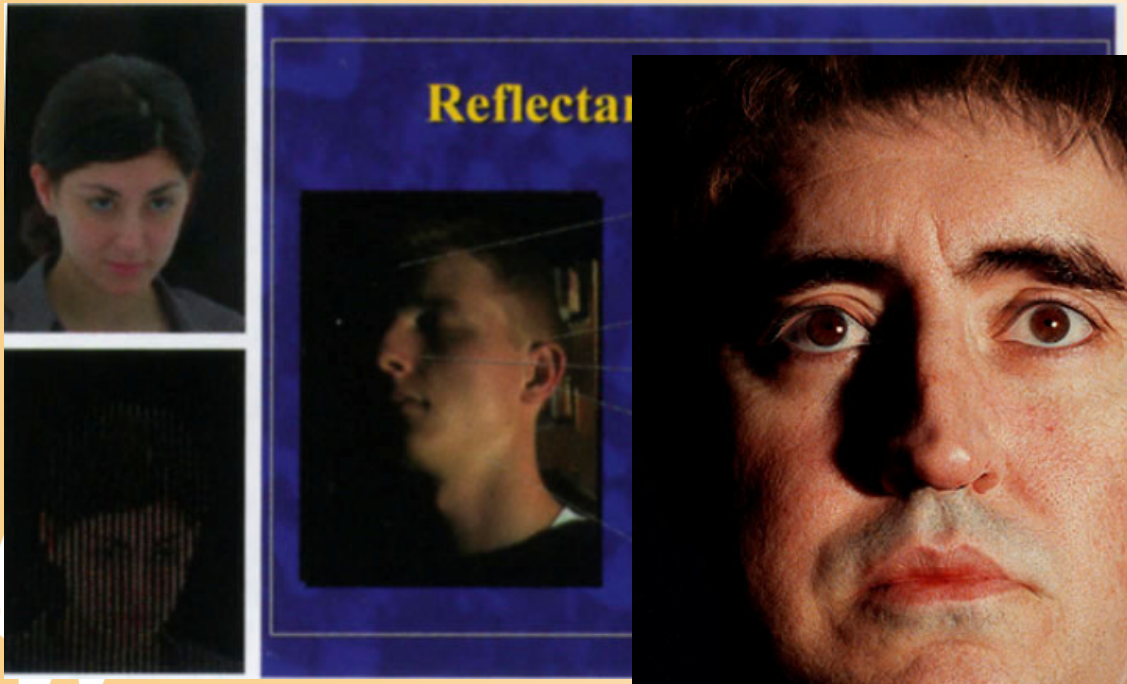
サンプル画像の取得



Debevec
2000

映画への応用

- Spider-Man 2
  - 完全CGのキャラクタの、顔の皮膚の質感を再現



CG WORLD Spider-Mann 2
2004年12月号 Sony Pictures

映画への応用

- Matrix (2, 3)
  - 完全なCGキャラクタにBRDFが使われた最初の例

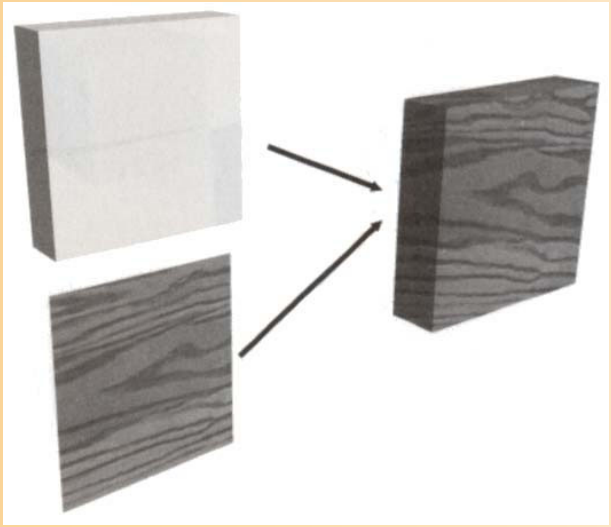
Matrix
Warner Bros.

CG WORLD
2004年12月号

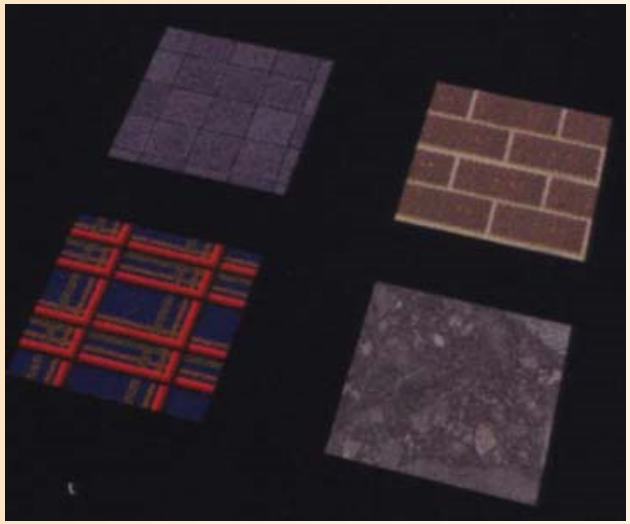
マッピング

マッピング

- マッピング
  - 面を描画する時に、面の表面に画像を貼り付ける技術
  - 複雑なモデリングをすることなく、細かい模様などを表現できる



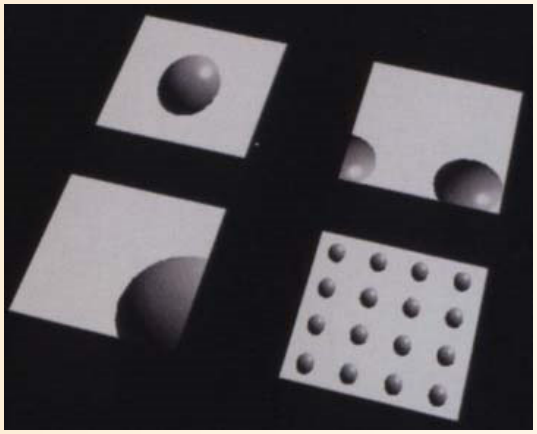
基礎知識 図3-19



基礎と応用
図5.2

マッピングの方法

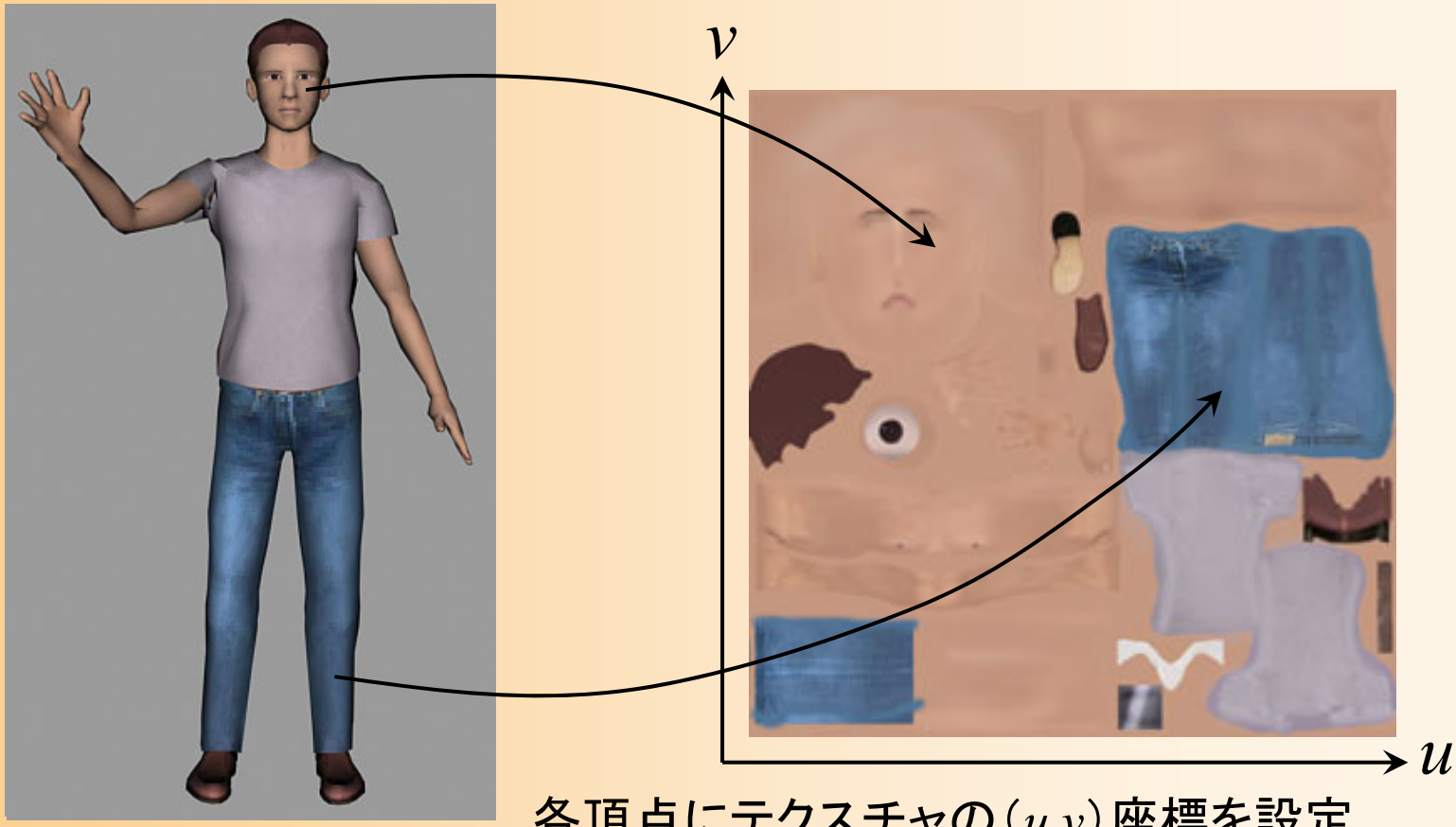
- 物体への画像の貼り付け方
  - マッピングの方向や繰り返しの方法
- uv座標系
  - テクスチャ画像の座標は(u,v)で表せる
  - モデルデータの各頂点(x,y,z)ごとに、対応するテクスチャ画像の(u,v)座標を与えておく



基礎と応用 図5.3

マッピングの例

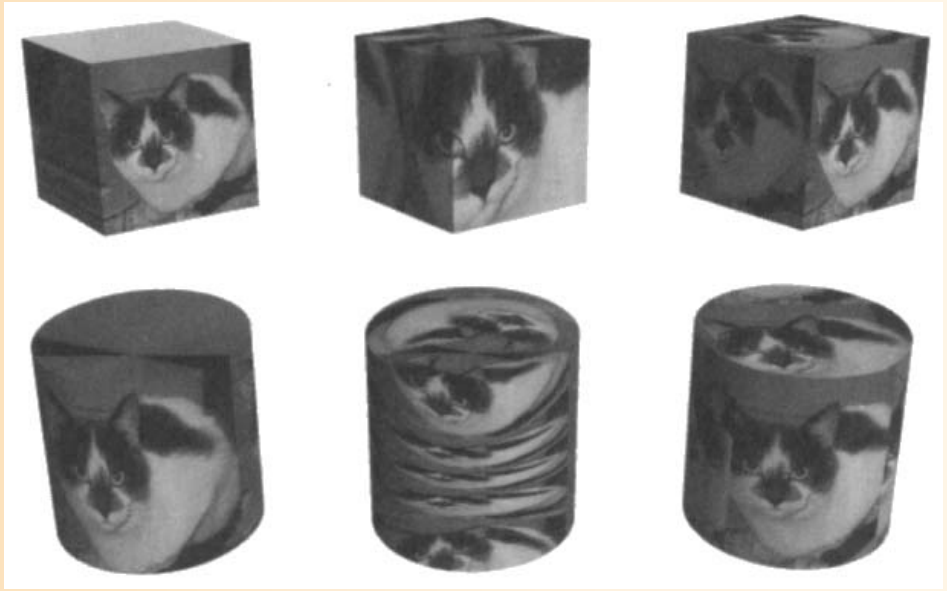
- 人体モデルへのマッピングの例



各頂点にテクスチャの(u,v)座標を設定

自動マッピングの方法

- ラッピング
  - 各頂点の(u,v)座標を自動的に計算する方法
  - 主に単純な物体へのマッピング時に使用
    - 平行ラッピング
    - 曲座標ラッピング



基礎知識 図3-22

手動マッピングの方法

- 手動でのマッピング
  - 複雑な物体へのマッピングをする時は、モデリングをする人が手動で(u,v)座標を設定
- マッピング作成の手順
  1. 最初にまずモデルを作成
  2. モデルの展開図を作り、モデルの各頂点に対応する(u,v)座標を設定(モデリング用ソフトの機能)
  3. テクスチャ画像を描く

  ※ 後からモデルを大きく変更することは難しい

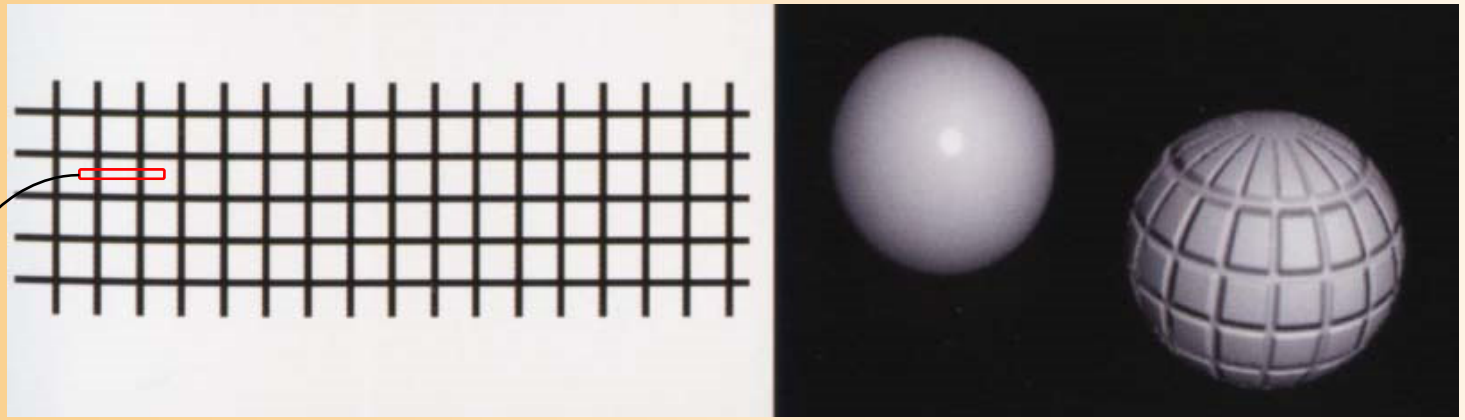
特殊なマッピング

- バンプマッピング
- 透過率や反射率のマッピング
- 環境マッピング

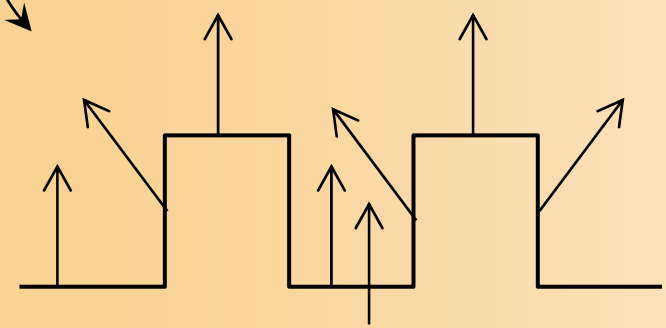
バンプマッピング

- バンプマッピング
  - 面の色ではなく、法線を変化させる方法
  - 法線を変化させることで、見た目の質感を変えることができる
    - ピクセルの周囲との輝度の差に応じて法線を傾ける
    - 形状そのものを変えるわけではないので、輪郭線はそのままであることに注意
  - 各ピクセルごとに法線と色を計算することが必要になる（フォンシェーディングとの組み合わせ）

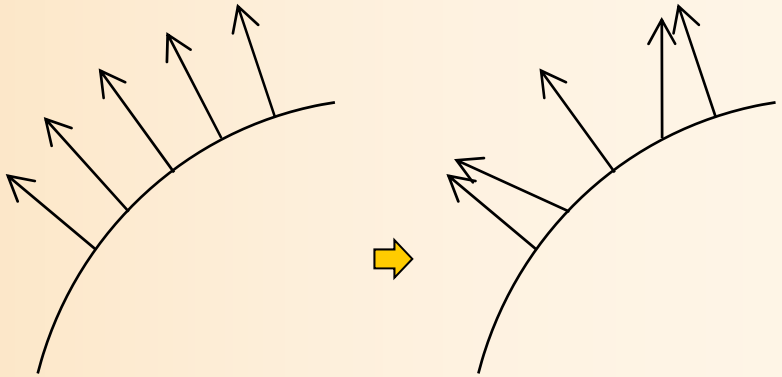
バンプマッピングの原理



基礎と応用 図5.9

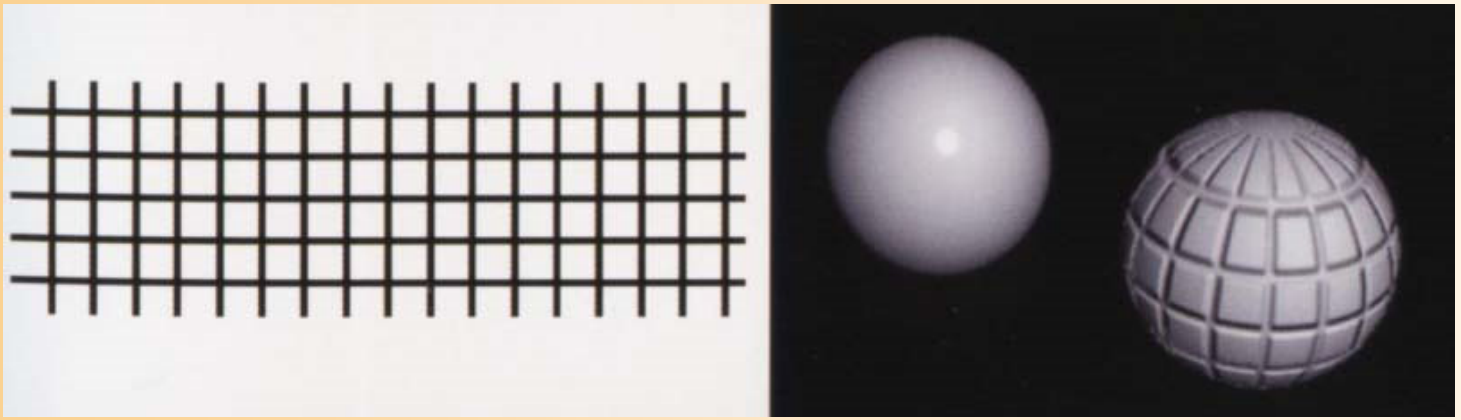


テクスチャに格納された各点の高さから、各点の傾き（法線を計算）

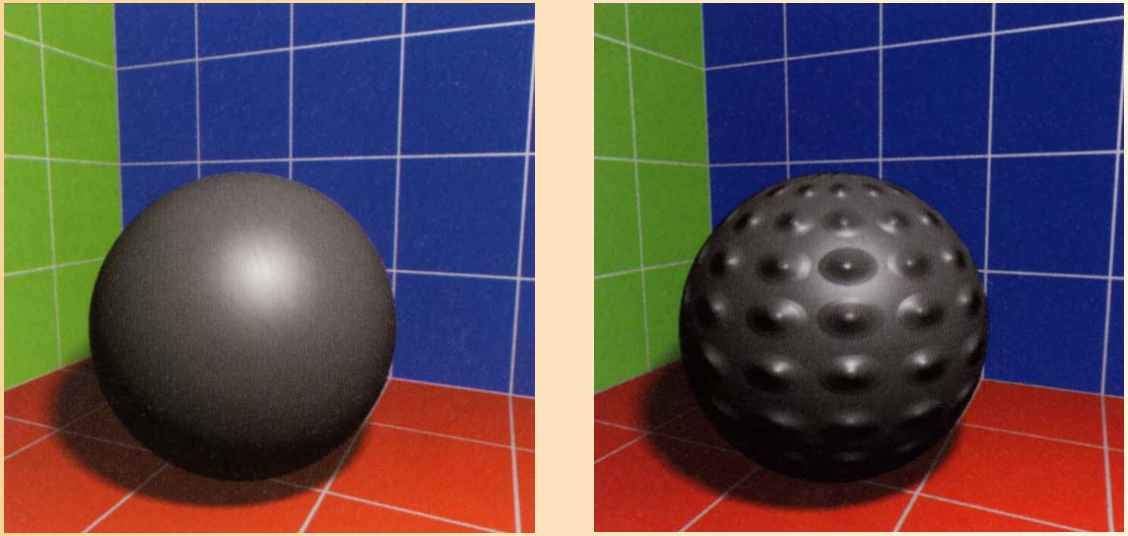


テクスチャから計算された法線に従って各点の法線を変化

バンプマッピングの例



基礎と応用 図5.9



CG制作独習事典 p.13

反射率・透過率のマッピング

- 反射率や透過率をマッピングすることも可能
  - 透過率のマッピングは、髪の毛や破れた着物などを表現する時などに用いられる
    - 細かい形状を一枚の大きなポリゴンで表現できる

[Koh and Huang, CAS 2001]

環境マッピング

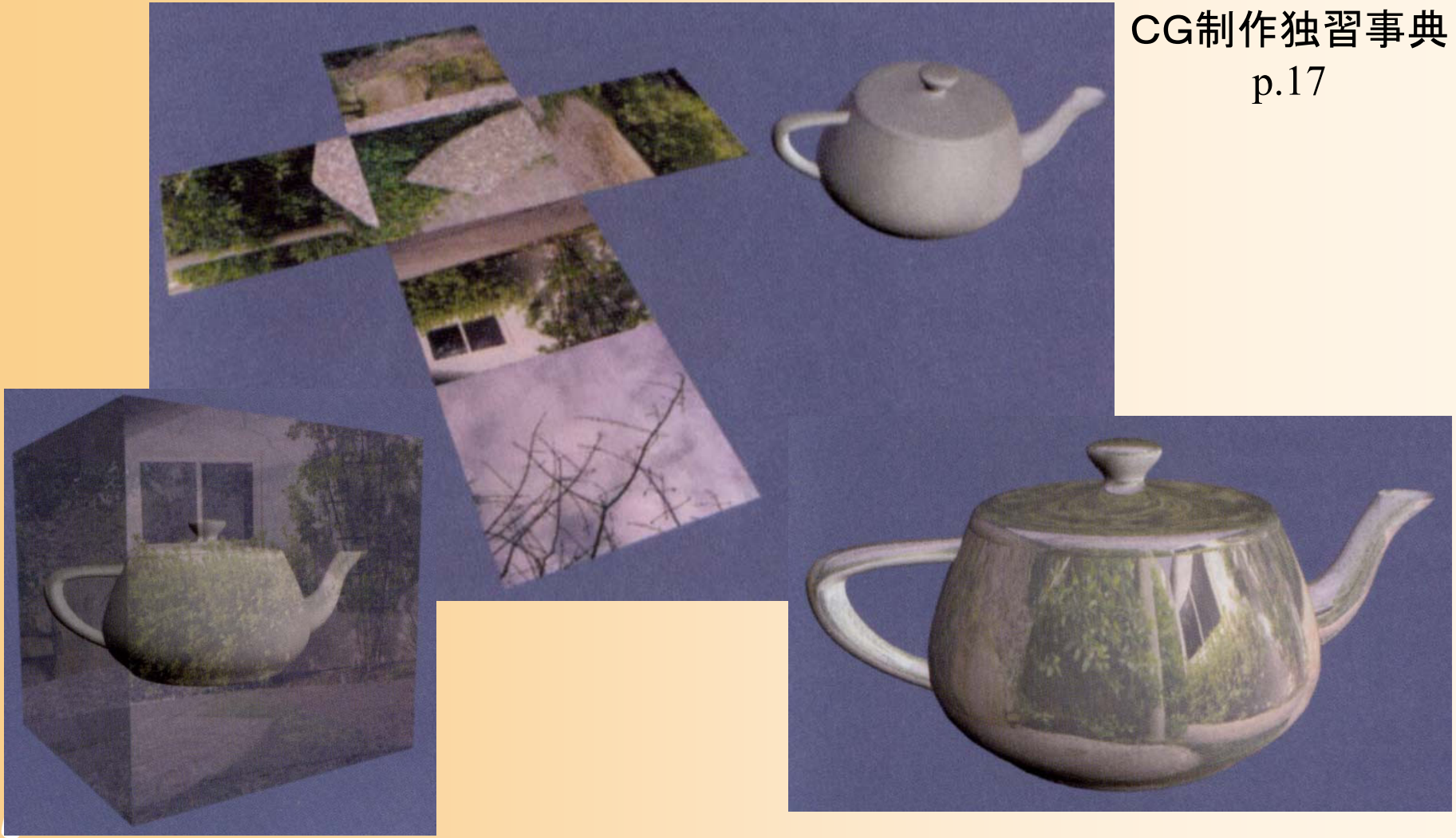
- 環境マッピング
  - 物体の周囲の風景の画像を、テクスチャマッピングを使って物体に貼り付けることで、周囲の風景の映り込みを表現する

CG制作独習事典 p.17

環境マッピングの手順

- 物体の周囲の画像をレンダリング
  - カメラを物体の中心に置いて、各方向を見たときの画像を生成
    - 例えばキューブ環境マップならば6枚分の画像を生成
  - 周囲の物体が動かなければ、周囲の画像の生成は最初の一回だけで良い（2回目からは高速に描画可能）
- レンダリングした画像を物体に貼り付ける
  - テクスチャ座標は、各点の法線から自動的に計算できる

環境マッピングの例



CG制作独習事典
p.17

マッピングの合成

- 複数の画像によるマッピング
  - 実際の応用では、複数のマッピングを重ね合わせる（合成する）場合が多い
    - 基本となる模様（1枚）＋
      環境や光などの影響（1～数枚）
  - 昔のハードウェアは、１枚しかマッピングができなかった
  - 最近のハードウェアは、4枚～16枚程度の同時マッピングに対応している

まとめ

- 前回の復習
  - シェーディング
- シェーディング（前回の続き）
  - OpenGLでの光源情報の設定
  - ラジオシティ
  - 影の表現
  - BRDF
- マッピング

次回予告

- OpenGL演習
  - シェーディング（光源情報の設定）
  - マッピング