

コンピュータグラフィックス

第8回

レンダリング技法 1

～基礎と概要, 隠面消去～

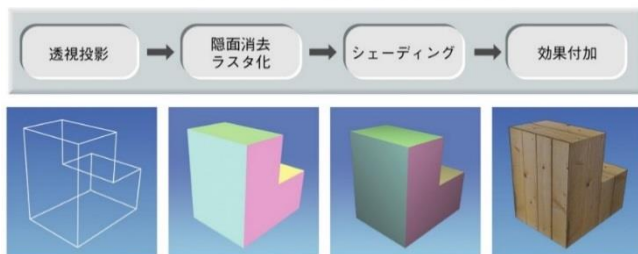
理工学部 兼任講師
藤堂 英樹

本日の講義内容

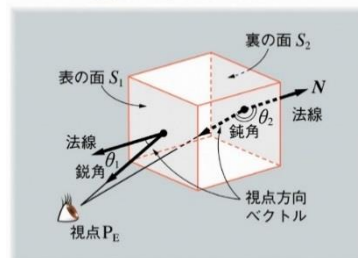
■ レンダリング技法 1

- レンダリング処理概要
- 隠面消去

■ 図4.2—レンダリングを構成する処理過程



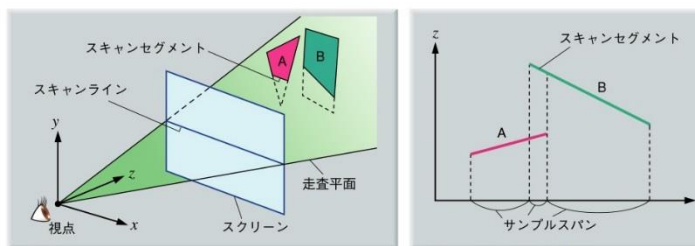
■ 図4.3—バックフェースカリング



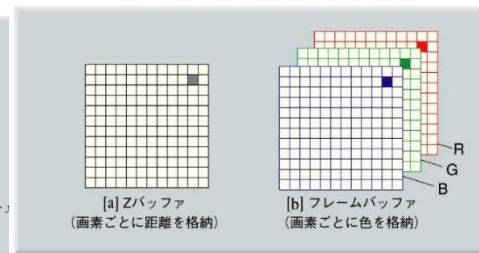
■ 図4.6—奥行きソート法の描画過程



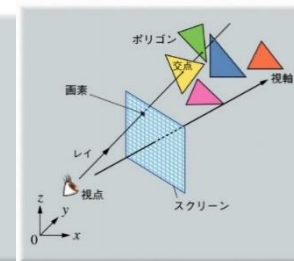
■ 図4.10—スキャンラインアルゴリズム



■ 図4.14—Zバッファとフレームバッファ



■ 図4.16—レイトレーシング法



レンダリングの基礎

■レンダリング

- 3次元シーンを入力とした**描画処理**
- 入力：3次元物体, カメラ, 光源情報
- 出力：画像



カメラから見た映像



レンダリングを構成する処理

■ 投影変換 (CG のための数学的基礎 2 参照)

- 透視投影, 平行投影等

■ 隠面消去 (今回の講義で説明)

- 見えない面を消す処理

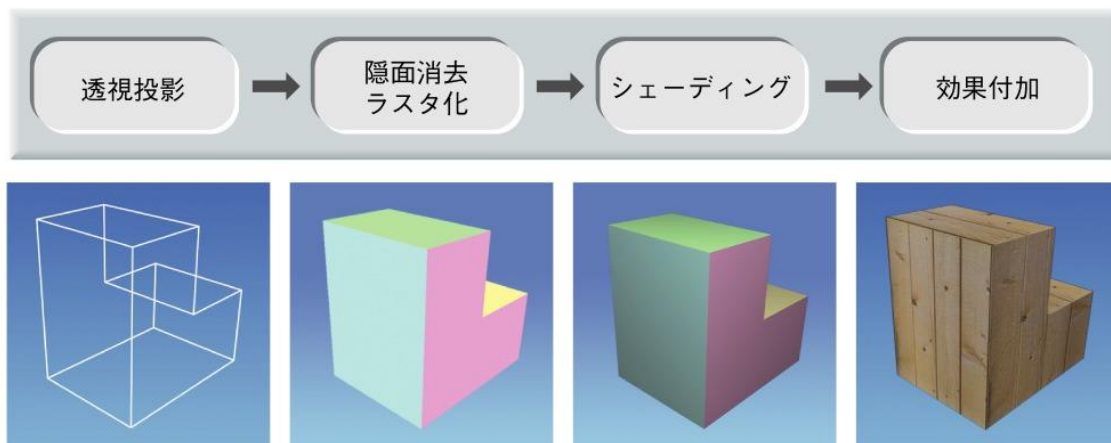
■ シェーディング

- 質感の表現

■ 効果付加

- テクスチャ

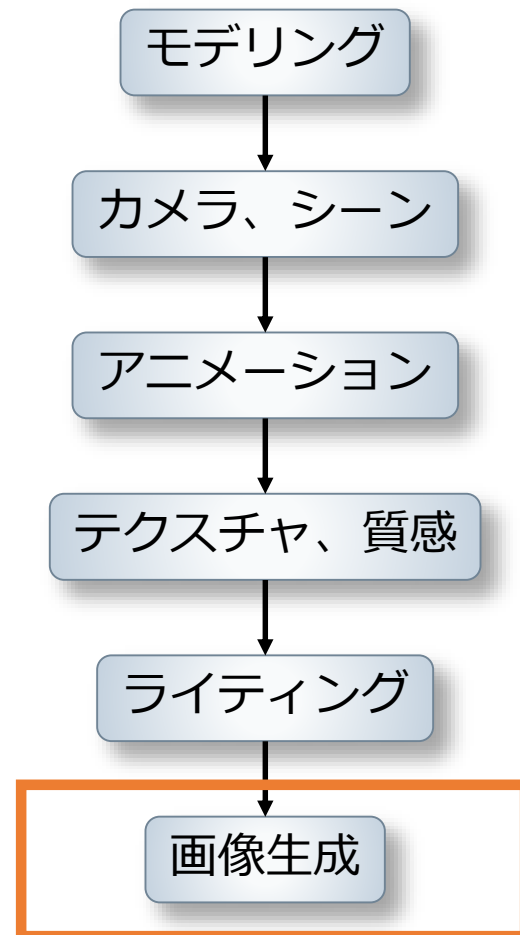
■ 図4.2——レンダリングを構成する処理過程



「コンピュータグラフィックス」2004年 / 財団法人画像情報教育振興協会 (CG-ARTS協会)

CG制作の主なワークフロー

■3DCGソフトウェアの場合



レンダリング関連

■実写においてはカメラがその役割を担当

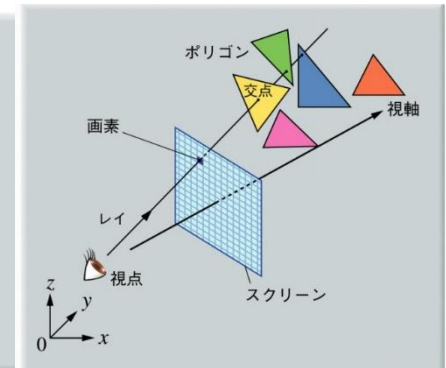
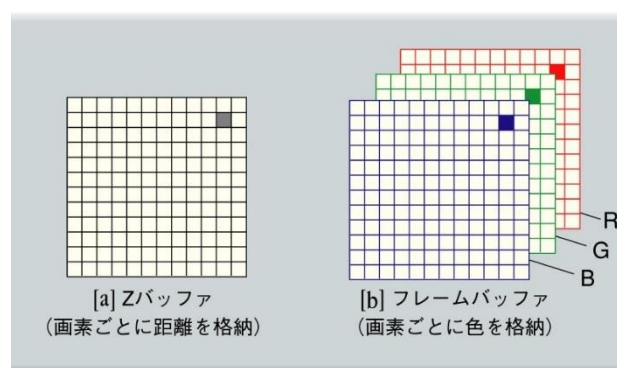
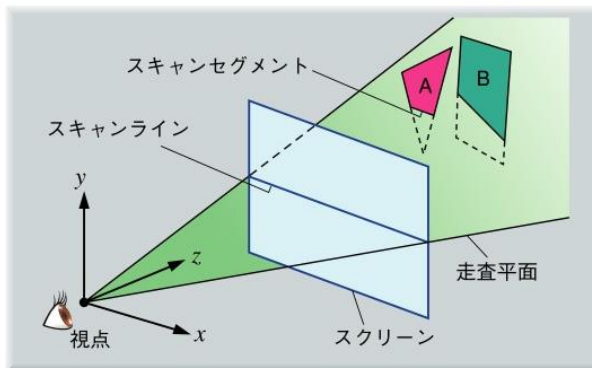
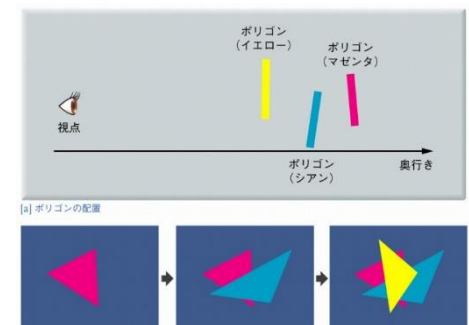
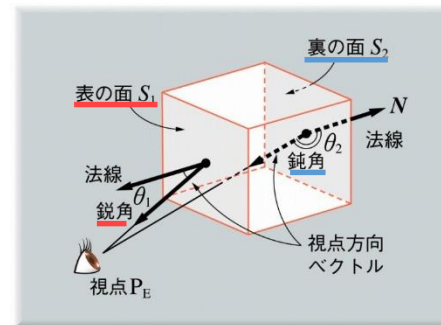
- 1. カメラでの撮影
- 2. **奥行関係が自然に撮影**される
- 3. 写真, ビデオの形で**画像, 映像を出力**

■コンピュータでは

- 1. カメラでの撮影⇒投影変換
- 2. **奥行関係が自然に見えるよう工夫**する必要がある
- 3. コンピュータ上で画像, 映像を出力

隠面消去

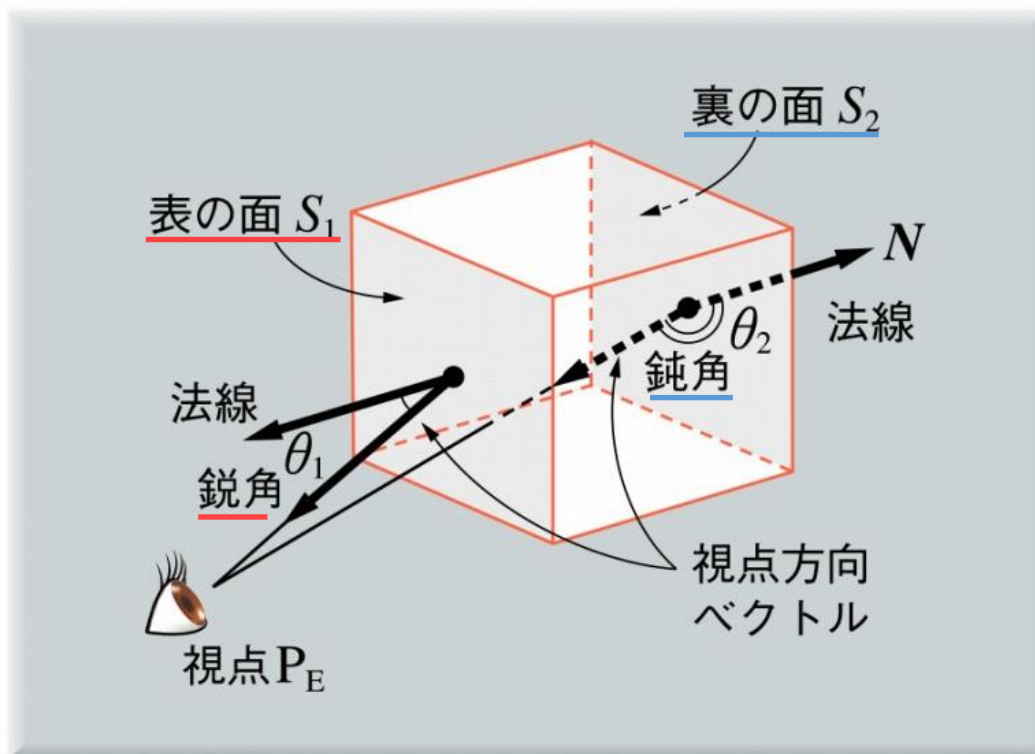
- バックフェースカリング
- 奥行きソート法
- スキャンライン法
- Zバッファ法
- レイトレーシング法



バックフェースカリング

■ 3次元形状の表を向いている面だけを描画

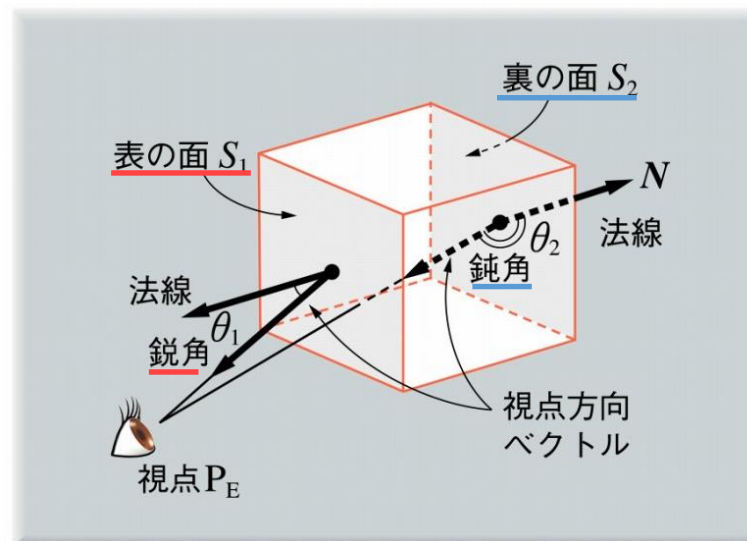
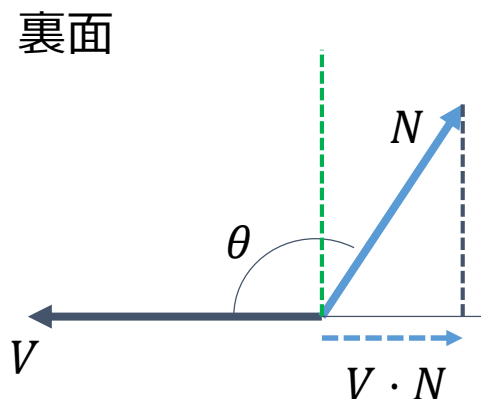
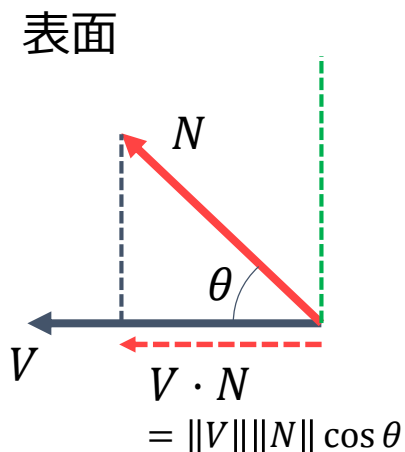
- 視点から見た時に反対方向の面を除去
- 視線(視点方向ベクトル)と法線のなす角で判定
 - 鋭角⇒表面
 - 鈍角⇒裏面



バックフェースカリング

■視線と法線のなす角で判定

- 視線 V , 法線 N
- 判別式: $D_f = V \cdot N$
 - $D_f > 0 \Rightarrow$ 表
 - $D_f < 0 \Rightarrow$ 裏



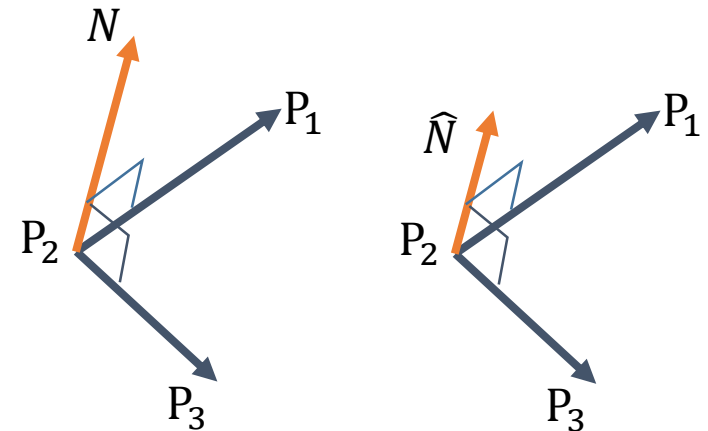
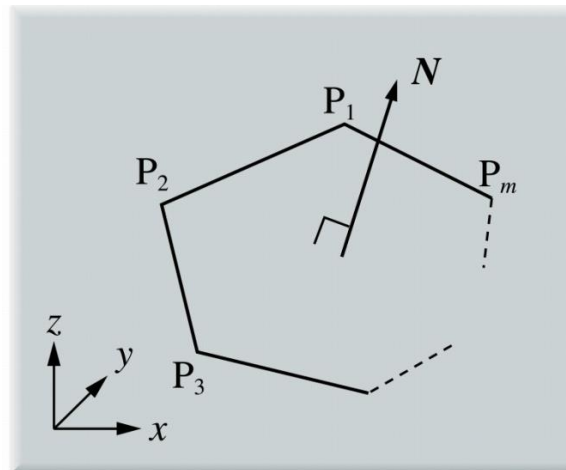
法線ベクトルの算出

■面を構成する頂点から計算

- 頂点列 $P_i (i = 1, 2, \dots, m)$
- 任意の3頂点 P_i, P_j, P_k を選ぶ
- 外積による法線ベクトルの計算
 - $N = (P_i - P_j) \times (P_j - P_k)$
 - 単位法線 $\hat{N} = \frac{N}{\|N\|}$

$$\begin{aligned}n_x &= (y_i - y_j)(z_j - z_k) - (z_i - z_j)(y_j - y_k) \\n_y &= (z_i - z_j)(x_j - x_k) - (x_i - x_j)(z_j - z_k) \\n_z &= (x_i - x_j)(y_j - y_k) - (y_i - y_j)(x_j - x_k)\end{aligned}$$

$$\frac{N}{\|N\|} = \frac{N}{\sqrt{n_x^2 + n_y^2 + n_z^2}}$$

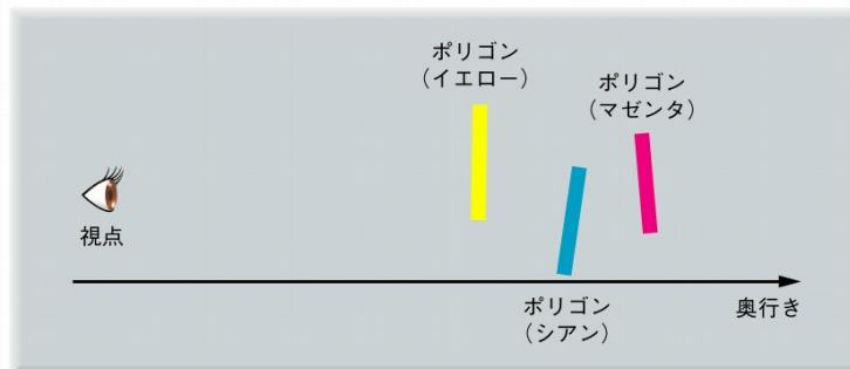


奥行きソート法

■奥行き順番に面を描いていく方法

- 奥行き順番に面を**ソート**する
- ポリゴンの**重心**を奥行き情報にする

■図4.6——奥行きソート法の描画過程

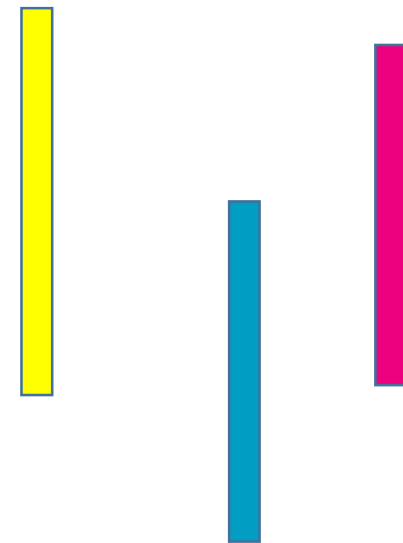


[a] ポリゴンの配置



[b] 描画過程(左から右の図へ描画が進行する)

「コンピュータグラフィックス」2004年 / 財団法人画像情報教育振興協会 (CG-ARTS協会)

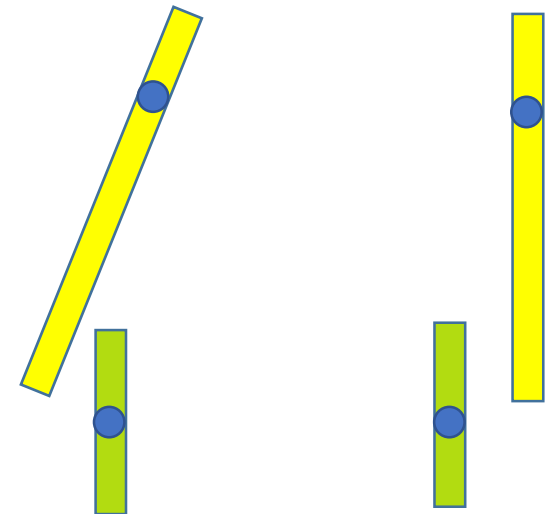
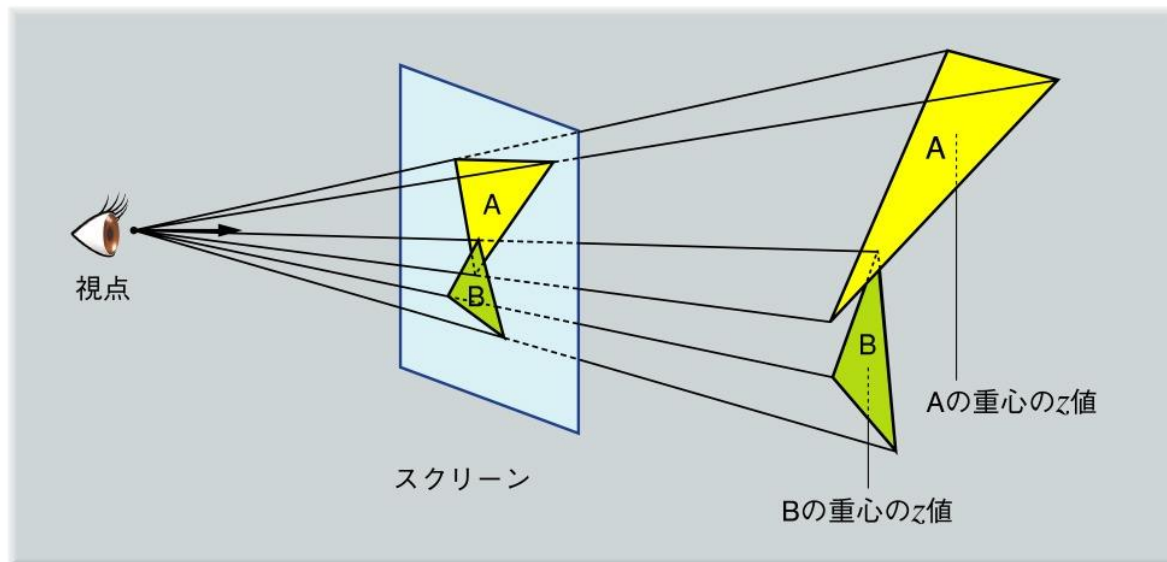


奥行きソート法

■失敗する例

- ポリゴン単位より**細かい判定ができない**

■図4.7——優先順位をポリゴンの重心で定めた場合に隠面消去に失敗する例

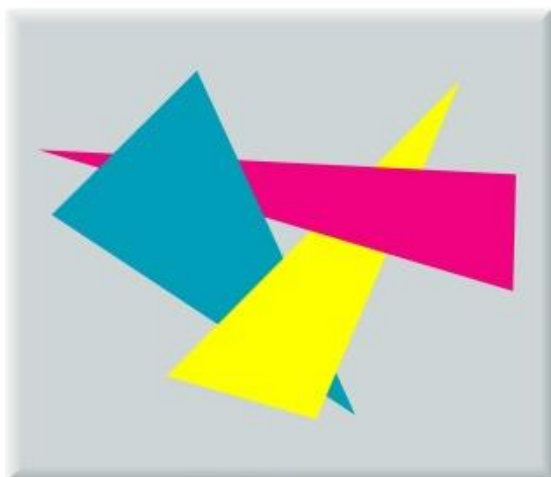


奥行きソート法

■失敗する例

- ポリゴン単位より**細かい判定ができない**

■図4.9——可視性の優先順位が決定できない場合



[a] 三すくみの場合



[b] 貫通している場合



[c] 凹ポリゴンの場合

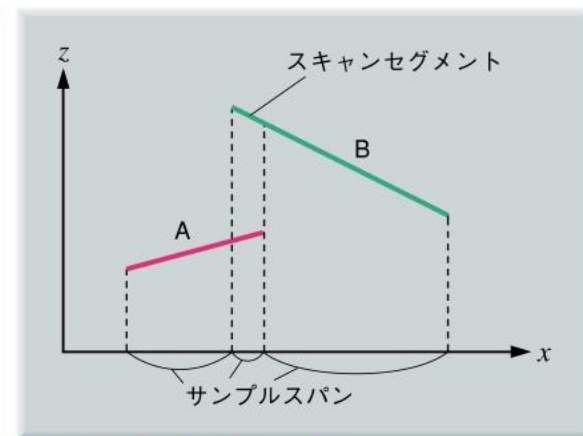
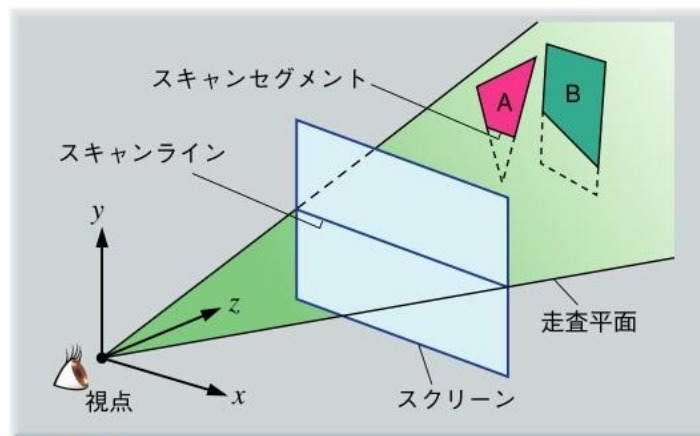
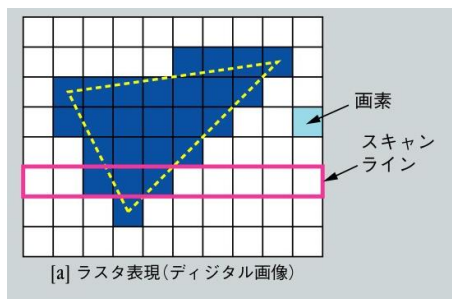
「コンピュータグラフィックス」2004年 / 財団法人画像情報教育振興協会 (CG-ARTS協会)

スキャンライン法

■ スキャンラインに沿って隠面消去

- 走査平面: 視点とスキャンラインで構成される平面
- スキャンセグメント:
 - ポリゴンと走査平面の交差線分
- スキャンセグメントの前後関係で可視判定

■ 図4.10——スキャンラインアルゴリズム

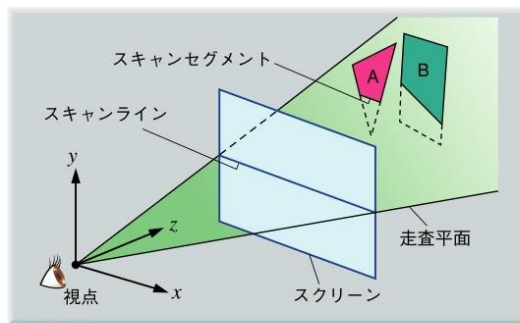


スキャンライン法

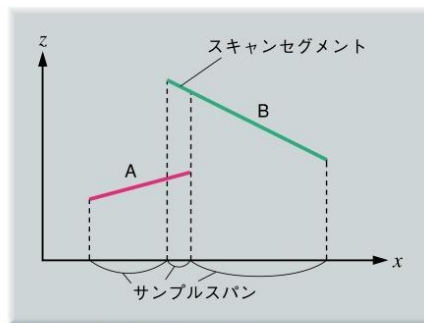
■ スキャンセグメントの前後関係で可視判定

- x 軸ソート: 端点を x 座標の小さい順にソート
- サンプルスパン: ソートした端点のサンプリング区間
- **サンプルスパン毎**に見えるセグメントを判定

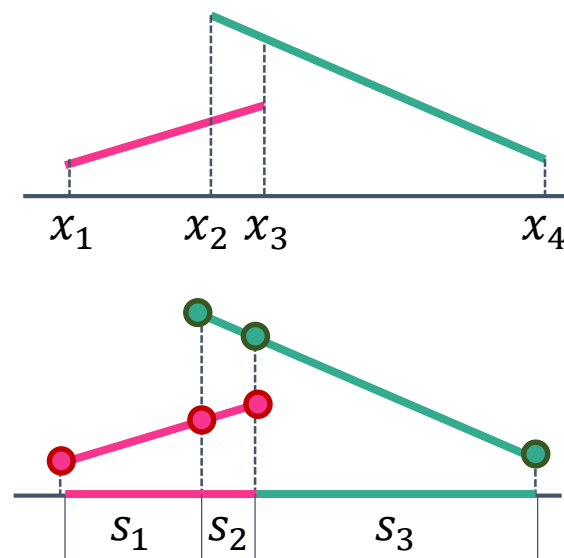
■ 図4.10—スキャンラインアルゴリズム



[a] 走査平面



[b] 奥行き判定 (走査平面断面図)



スキャンライン法の描画過程

- スキャンラインを上から下へ順番に移動
 - スキャンライン毎にスキャンセグメントの可視判定

■ 図4.11——スキャンライン法の描画過程
(ポリゴンの配置は図4.6[a]と同じ。左から右の図へ描画が進行する)



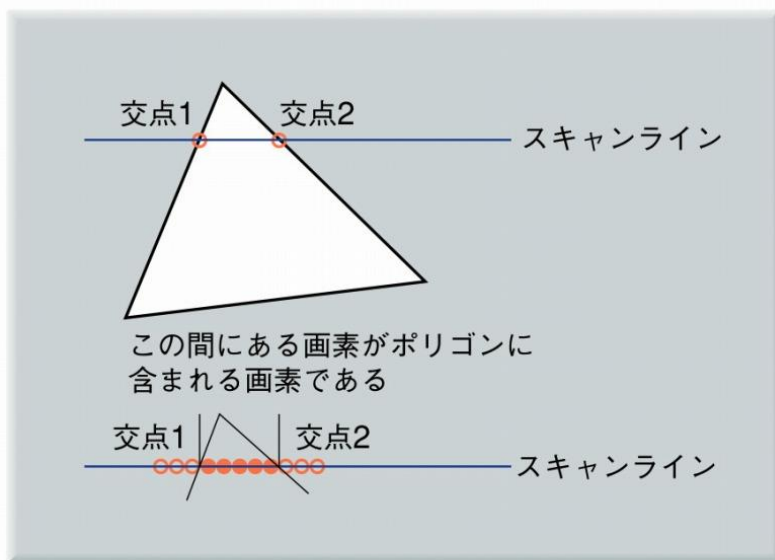
「コンピュータグラフィックス」2004年 / 財団法人画像情報教育振興協会 (CG-ARTS協会)

ポリゴンの走査変換

■ スキャンラインとポリゴンの交点の計算

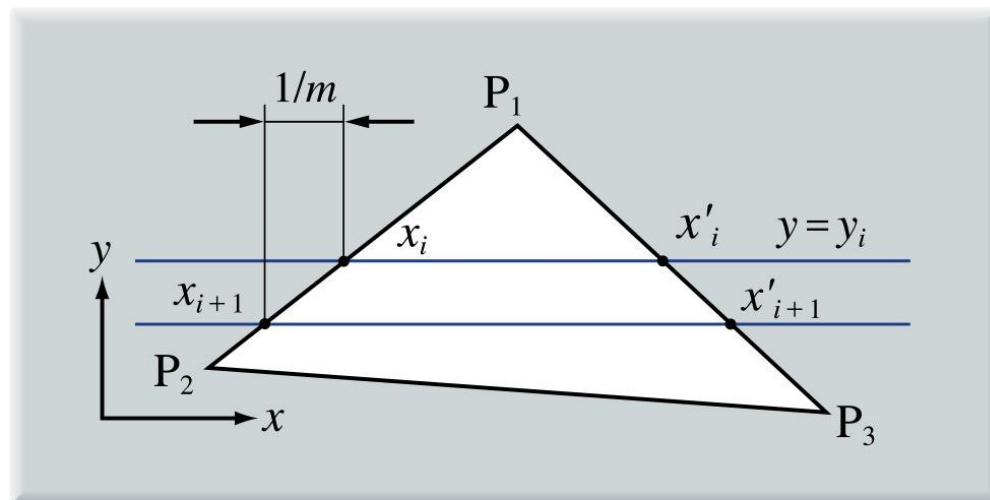
- **増分法**を用いた効率の良い計算
 - y方向の増分の計算: $x_{i+1} = x_i + \Delta x$
 - 増分の幅: $\Delta x = -\frac{1}{m}$ (m は辺の傾きを表す)

■ 図4.12——三角形の走査変換



「コンピュータグラフィックス」2004年 / 財団法人画像情報教育振興協会 (CG-ARTS協会)

■ 図4.13——増分法による交点の算出



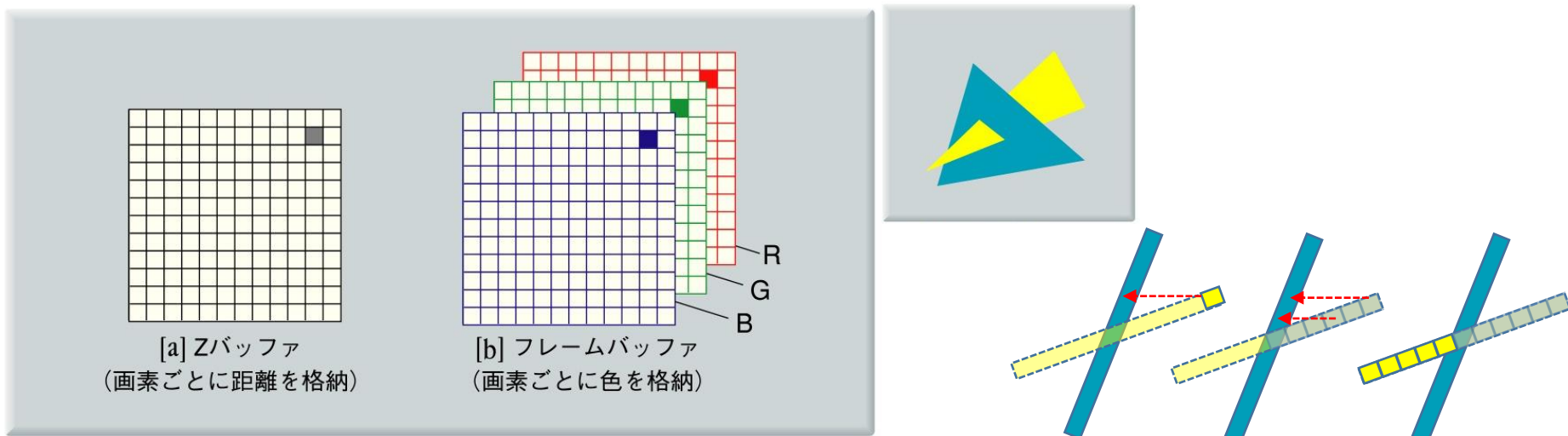
「コンピュータグラフィックス」2004年 / 財団法人画像情報教育振興協会 (CG-ARTS協会)

Zバッファ法

■画素毎に奥行き判定を行い隠面消去

- Zバッファ: 画像の奥行き情報を保存した一時領域
- フレームバッファ: 画素毎に色を格納

■図4.14—Zバッファとフレームバッファ



「コンピュータグラフィックス」2004年 / 財団法人画像情報教育振興協会 (CG-ARTS協会)

Zバッファ法の特徴

- アルゴリズムが簡単でハードウェア化しやすい
 - GPUによる高速なZバッファ処理
 - OpenGL, DirectX等の**3D APIで標準サポート**
⇒多くのソフトウェアで利用されている

Maya Shader FX © Autodesk

- 映像制作
 - デザイン時の**プレビュー**

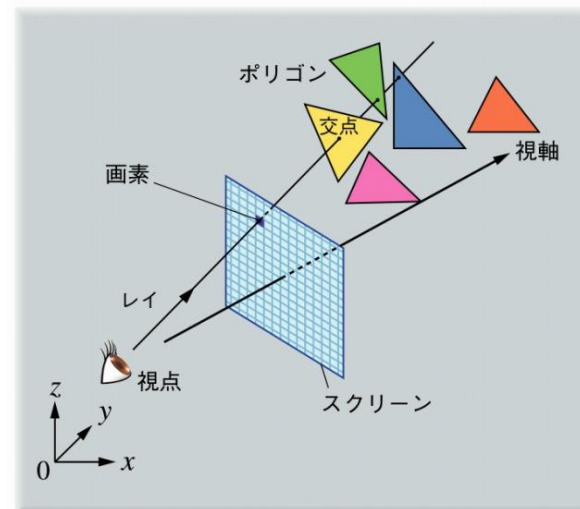
- ゲーム制作
 - ゲーム中の**高速な描画**

Angry Bots © Unity

レイトレーシング法

- レイ(視線)と物体の交差判定による隠面消去
 - **光線追跡法**とも呼ばれる
 - 反射・透過・屈折を扱える
 - **画素毎**にレイを計算
 - レイと**最初に交差するポリゴン**を求める
 - そのポリゴンの色で画素を塗る

■ 図4.16——レイトレーシング法

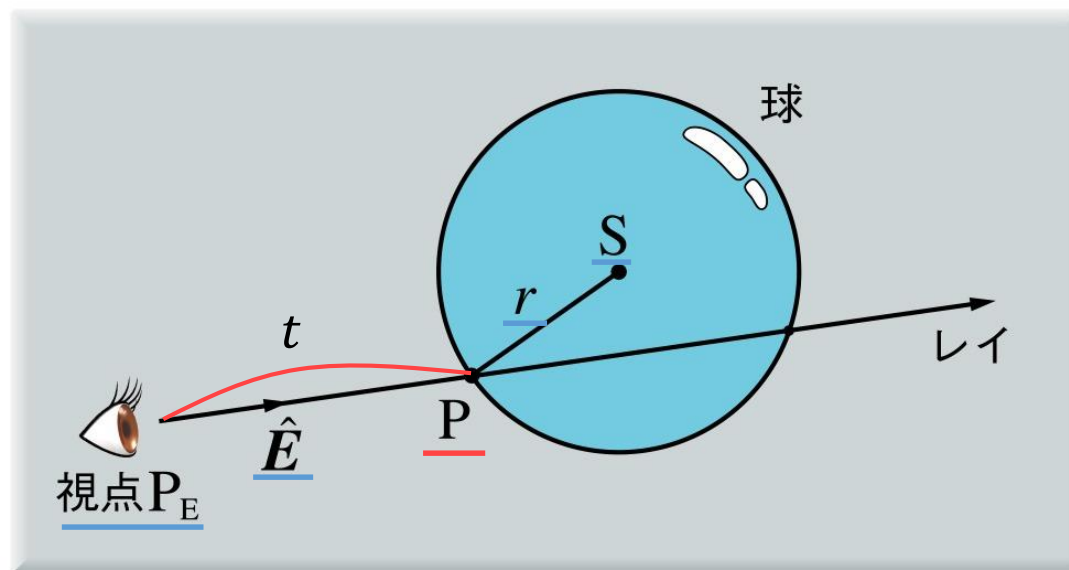


NVIDIA IRAY
© NVIDIA

レイと球の交差判定

- 入力: 視点 P_E , レイ \hat{E} , 球の中心 S , 半径 r
- 出力: 視点からの距離 t , 一番手前の交差点 P

■ 図4.17——レイと物体との交差判定



レイと球の交差判定

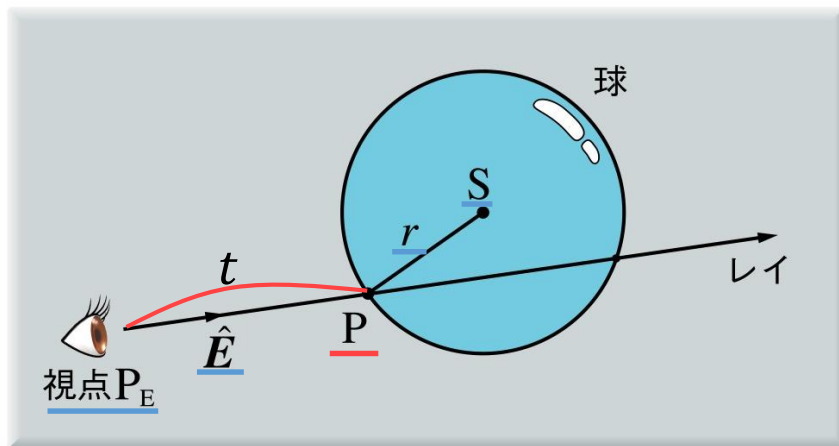
■ レイ上の点の方程式

- $P = \hat{E}t + P_E$

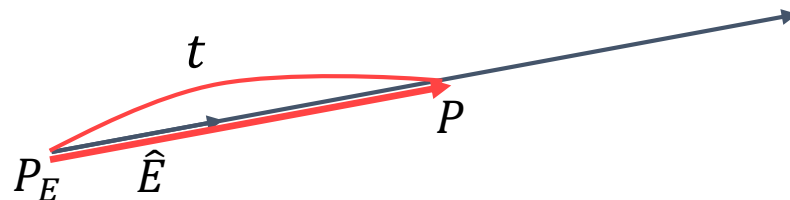
■ 球面上の点の方程式

- $\|P - S\| = r$

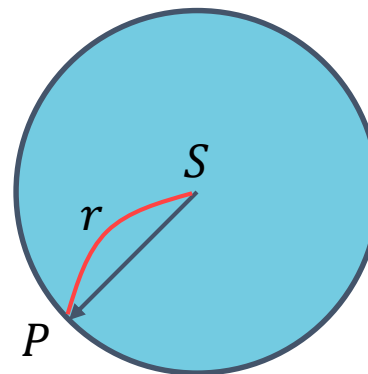
■ 図4.17——レイと物体との交差判定



レイ上の点の方程式



球面上の点の方程式



レイと球の交差判定

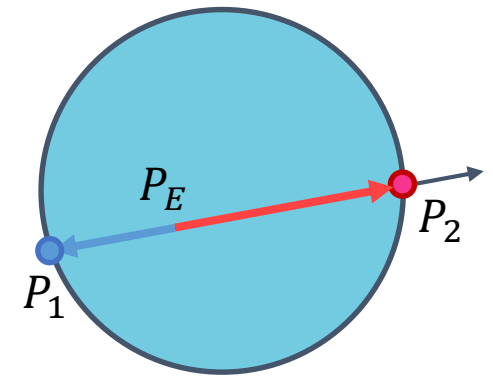
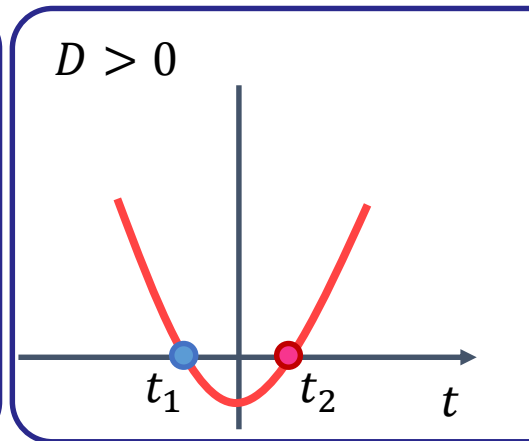
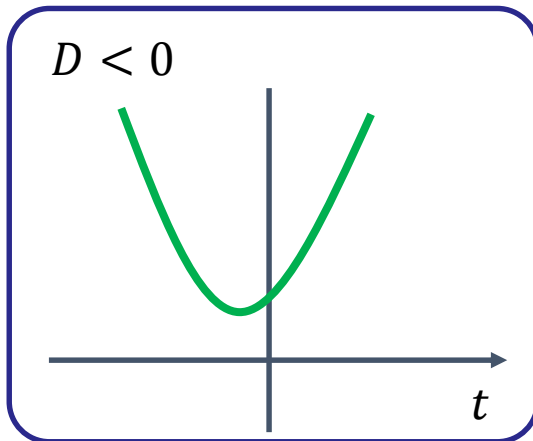
■方程式の解

- $P = \hat{E}t + P_E$ (1)
- $\|P - S\| = r$ (2)
- (2)の両辺を2乗
 - $(P - S) \cdot (P - S) - r^2 = 0$ (3)
- (3)に(1)を代入して整理
 - $at^2 + 2bt + c = 0$ (4)
 - $a = \|\hat{E}\|^2 = 1$
 - $b = \hat{E} \cdot (P_E - S)$
 - $c = \|P_E - S\|^2 - r^2$
- (4)を解いて t の値を求める

レイと球の交差判定

■ (4)を解いて t の値を求める

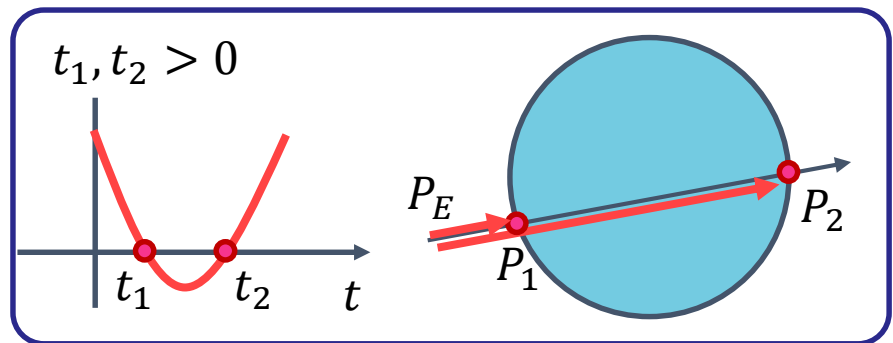
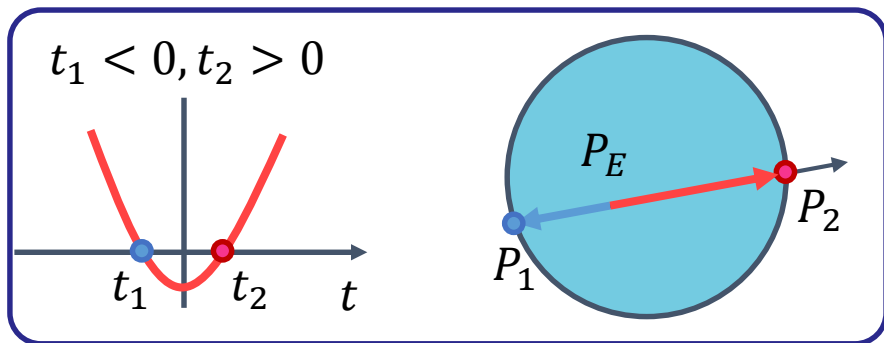
- $at^2 + 2bt + c = 0$ (4)
- 判別式: $D = b^2 - ac$
 - $D > 0$: レイは球と交差
- t の解の種類
 - $t < 0$: 交点が視点の背後にある
 - $t > 0$ の解の内もっとも小さいものを交点とする



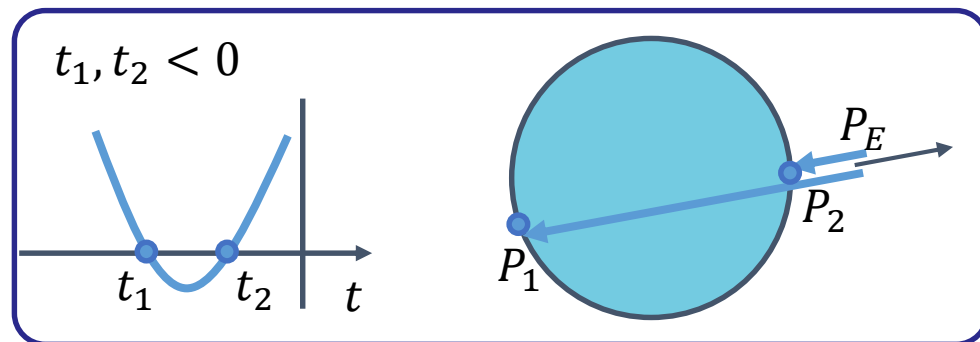
レイと球の交差判定

■ t の解の種類

- 可視点が存在する場合



- 可視点が存在しない場合

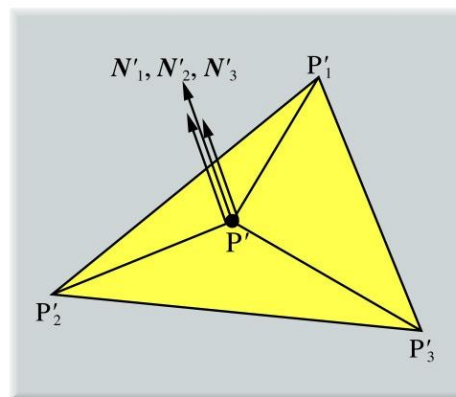
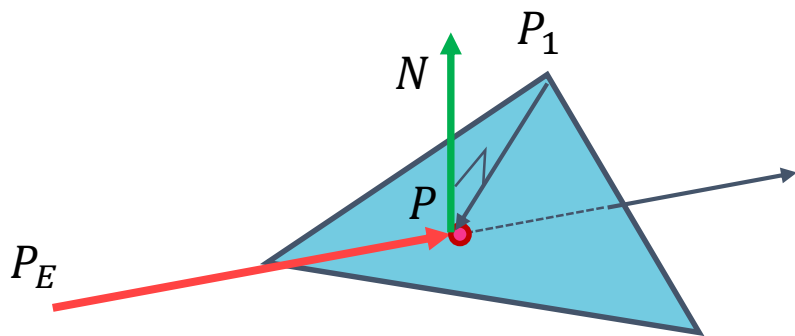


レイとポリゴンの交差判定

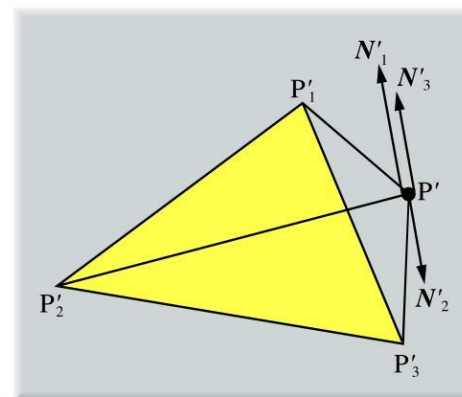
■ 三角形ポリゴンとの交差判定

- ポリゴンを含む平面とレイとの交点を求める
- 交点がポリゴン内部にあるかを判定

■ 図4.20——外積の方向を用いた点の内外判定



[a] P'が内部の場合



[b] P'が外部の場合

レイとポリゴンの交差判定

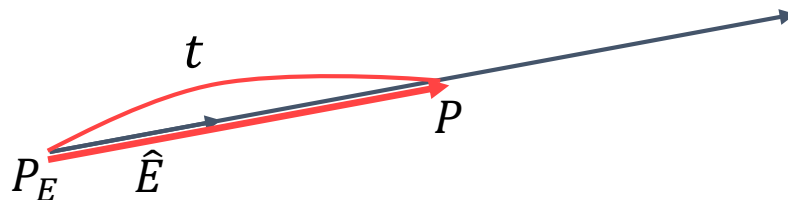
■ レイ上の点の方程式

- $P = \hat{E}t + P_E$

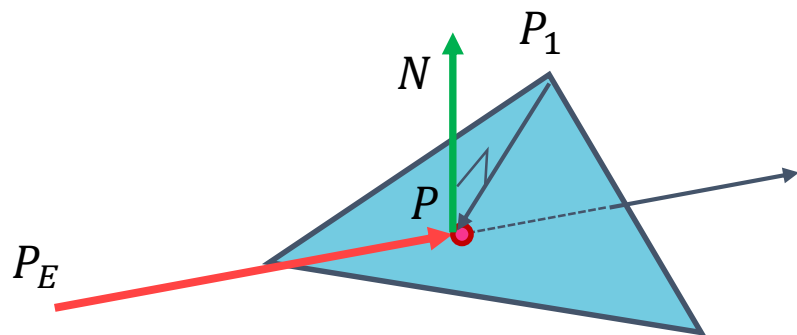
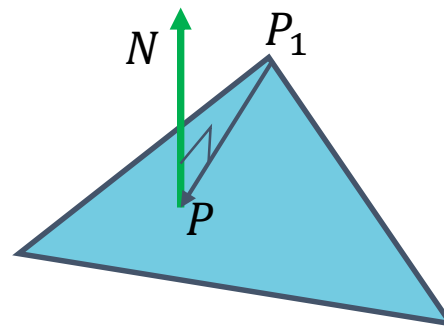
■ 平面上の点の方程式

- $(P - P_i) \cdot N = 0$

レイ上の点の方程式



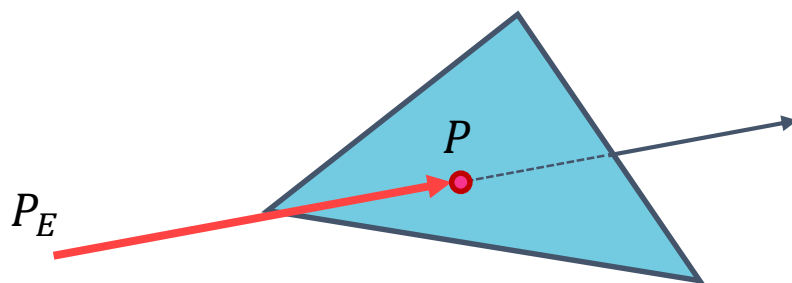
平面上の点の方程式



レイとポリゴンの交差判定

■ 方程式の解

- $P = \hat{E}t + P_E$ (1)
- $(P - P_i) \cdot N = 0$ (2)
- (2)に(1)を代入
 - $t = -\frac{(P_E - P_i) \cdot N}{\hat{E} \cdot N}$
- 球の時と同様に $t > 0$ の時に可視点となる

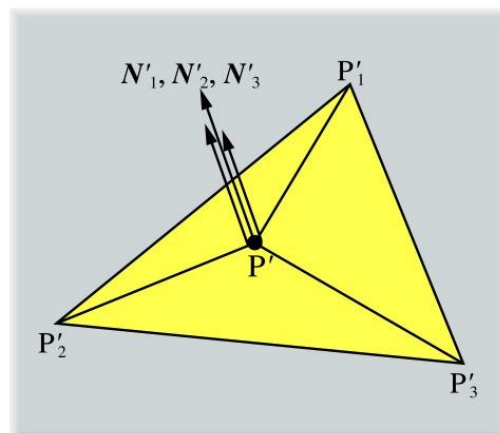
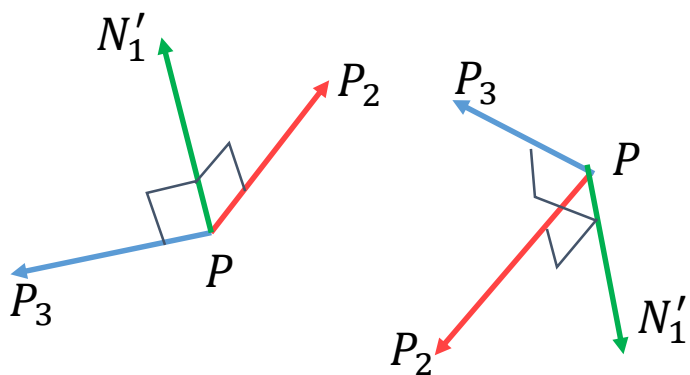


レイとポリゴンの交差判定

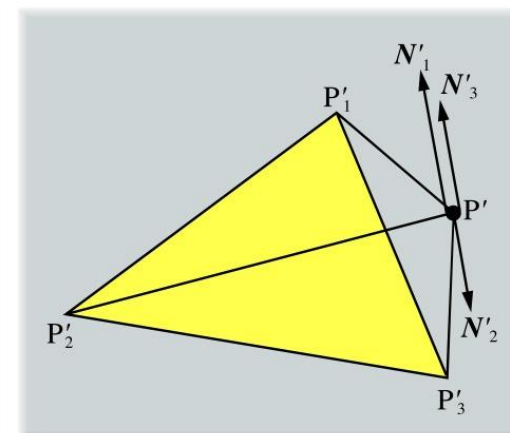
■ポリゴン内部にあるかを判定

- 点とポリゴンの各頂点を結ぶ**ベクトルの外積**を利用
 - $N'_i = (P'_{i+1} - P') \times (P'_{i+2} - P')$
 - 時計周りか反時計周りかで**向きが逆転**
 - **全て反時計周り: 内部にある**
 - 1つでも時計周り: 外部にある

■図4.20——外積の方向を用いた点の内外判定



[a] P' が内部の場合

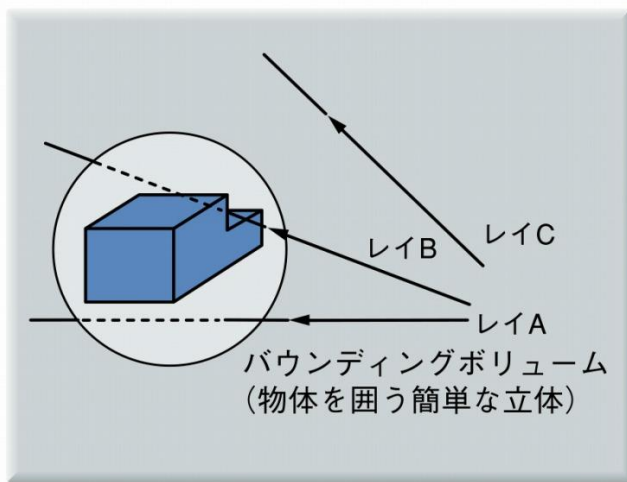


[b] P' が外部の場合

レイトレーシング法的高速化

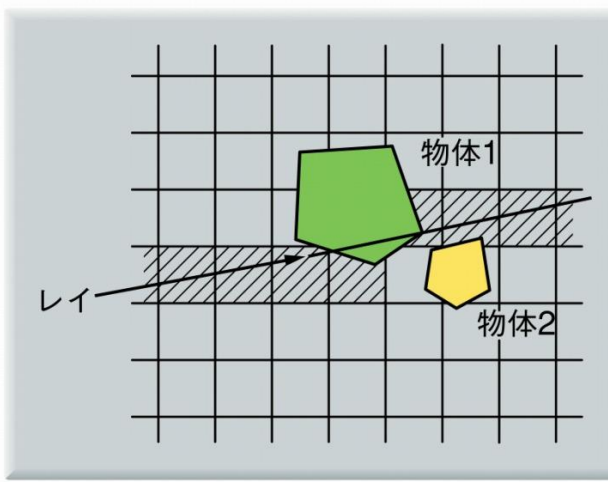
- 画素毎に交点計算を必要とするため時間がかかる
⇒ さまざまな高速化が提案されている
 - バウンディングボリューム
 - 空間分割法
 - 並列計算

■ 図4.22—バウンディングボリュームを用いた高速化



「コンピュータグラフィックス」2004年 / 財団法人画像情報教育振興協会 (CG-ARTS協会)

■ 図4.23—空間分割法による高速化

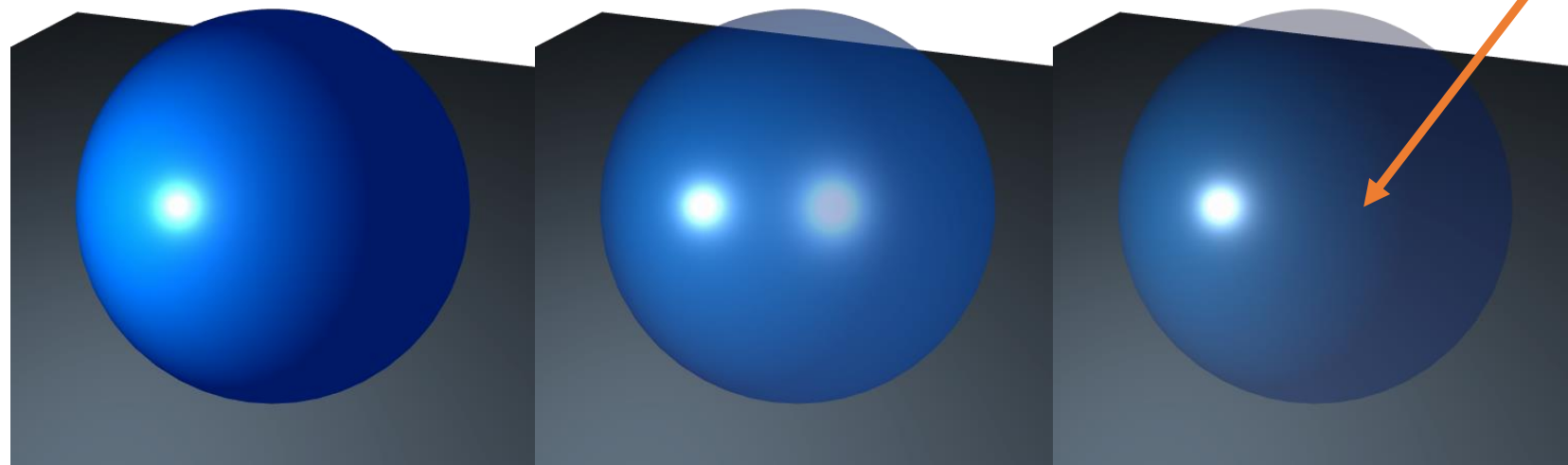


「コンピュータグラフィックス」2004年 / 財団法人画像情報教育振興協会 (CG-ARTS協会)

CG制作での隠面消去

■バックフェースカリング

- 効果的に描画するポリゴン数を削減できるが...
- 透明オブジェクトでは裏面も必要



裏面の反射が消える

透明度0の球

透明度0.7の球

透明度0.7の球
+バックフェース
カリング

CG制作での隠面消去

■奥行ソート

- アニメによく見られる眉毛や目が髪の前にある表現

SurfacePiercing
© Sakana-Ya

CG制作での隠面消去

■ スキャンライン

- Zバッファとレイトレーシングの中間
⇒CG制作では少し選択しづらい

■ Zバッファ

- CGソフトウェアの**プレビュー表示**
- ゲーム中の**高速な描画**

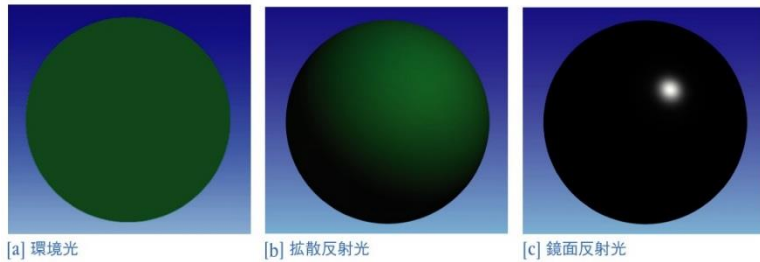
■ レイトレーシング

- 最終用の**高品質な映像**
- **最近は高速化**も進んでいる

次回

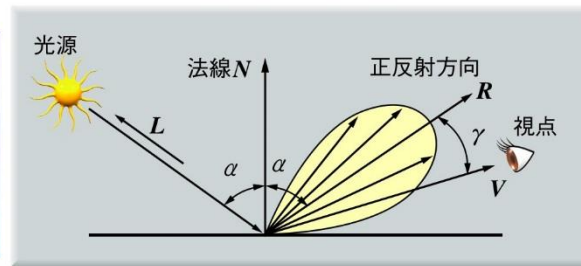
■ レンダリング技法2 ～シェーディング, マッピング～

■ 図4.29—環境光による反射, 拡散反射, 鏡面反射の各成分



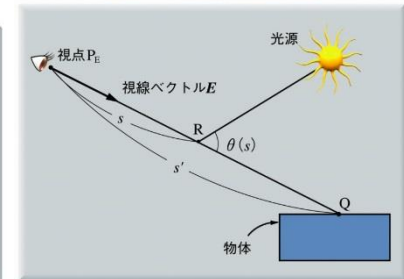
「コンピュータグラフィックス」2004年 / 財団法人画像情報教育振興協会 (CG-ARTS協会)

■ 図4.38—鏡面反射光



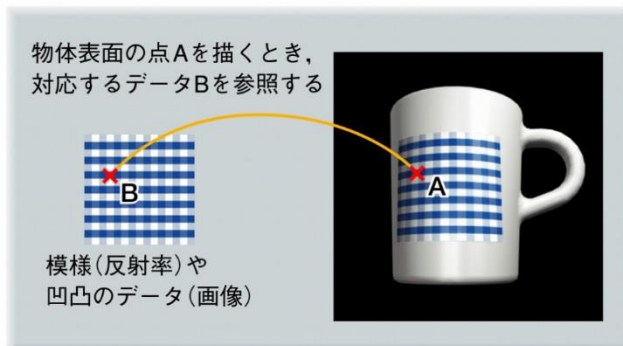
「コンピュータグラフィックス」2004年 / 財団法人画像情報教育振興協会 (CG-ARTS協会)

■ 図4.44—散乱粒子による光の散乱



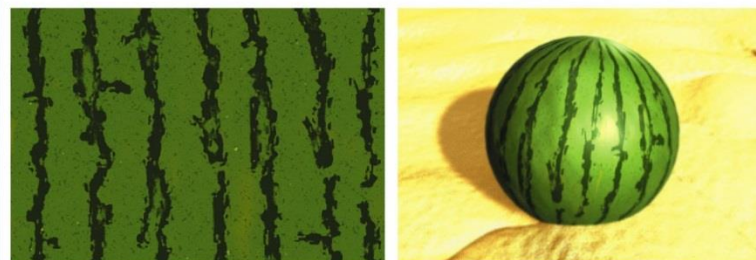
「コンピュータグラフィックス」2004年 / 財団法人画像情報教育振興協会 (CG-ARTS協会)

■ 図4.64—(2次元) マッピングの考え方



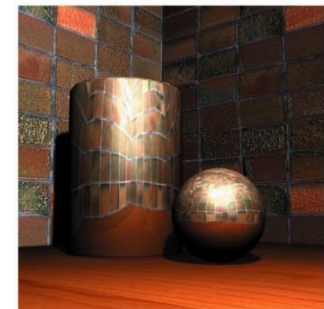
「コンピュータグラフィックス」2004年 / 財団法人画像情報教育振興協会 (CG-ARTS協会)

■ 図4.67—極座標変換を用いる方法によるテクスチャマッピングの例



「コンピュータグラフィックス」2004年 / 財団法人画像情報教育振興協会 (CG-ARTS協会)

■ 図4.79—環境マッピングによる画像



「コンピュータグラフィックス」2004年 / 財団法人画像情報教育振興協会 (CG-ARTS協会)