コンピューターグラフィックスS

第5回　レンダリング

システム創成情報工学科　尾下 真樹

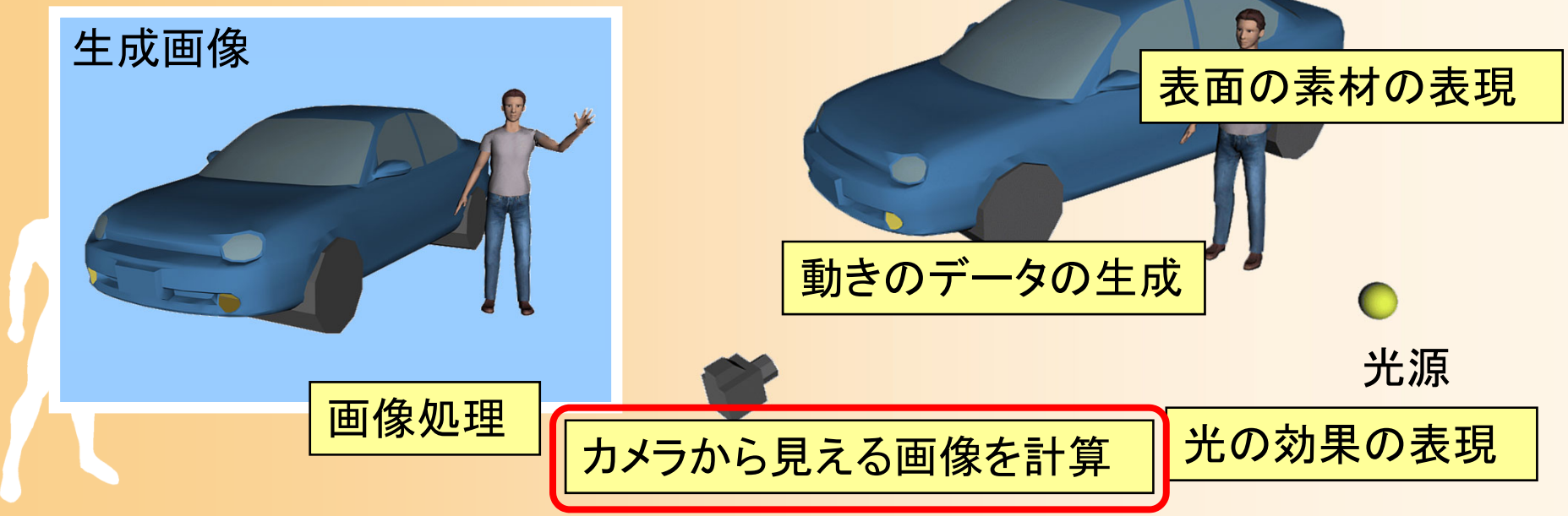
2019年度 Q2

今日の内容

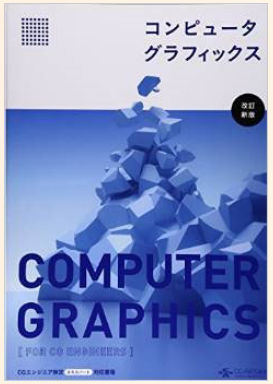
- レンダリングの種類
- レンダリングの予備知識
  - ポリゴンへの分割、隠面消去、光のモデル、反射・透過・屈折の表現
- レンダリング手法
  - Zソート法、Zバッファ法、スキャンライン法、レイトレーシング法
- レンダリングの高速化の工夫
- サンプリング

今回の内容

- レンダリング
  - カメラから見える画像を計算するための方法



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(3,200円)
  - 4章 4-1～4-2
  - 2-1、2-3、6-2-2 などの内容も説明
- 「ビジュアル情報処理 ―CG・画像処理入門―」
  CG-ARTS協会 編集・出版(2,500円)
  - 4章 4-1～4-2
  - 1-2-1、1-3-4 、1-5 などの内容も説明

参考書

- 「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）（次回の内容も含む）
- 「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3章（29～34ページ）、第6章（50～56ページ）、第8章（65～78ページ）
- 「コンピュータグラフィックスの基礎知識」
  塩川厚 著、オーム社 出版
  - 73～92ページ、102～113ページ

前回の復習

モデリングの技術

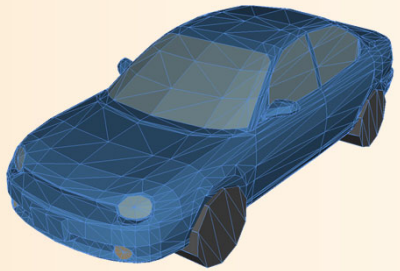
- ３次元オブジェクトの表現方法
  - オブジェクトを計算機上でどのように表現するか
- ３次元オブジェクトの作成方法
  - 実際にオブジェクトのデータをどうやって作成するか

  - 教科書によっては両者を混在して説明されていることもあるが、きちんと区別して考える

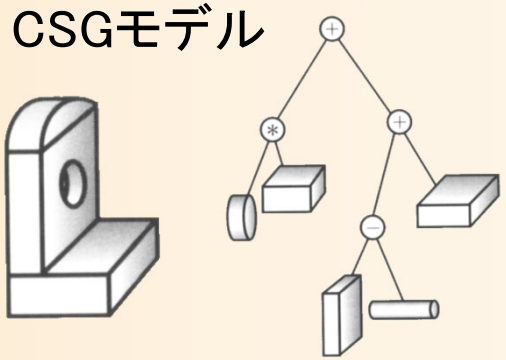
3次元モデルのデータ表現のまとめ

- サーフェスモデル
  - ポリゴンモデル
  - 曲面パッチ
  - サブディビジョンサーフェス
- ソリッドモデル
  - 境界表現
  - CSGモデル
- その他のモデル
  - 特殊なデータに向いた表現

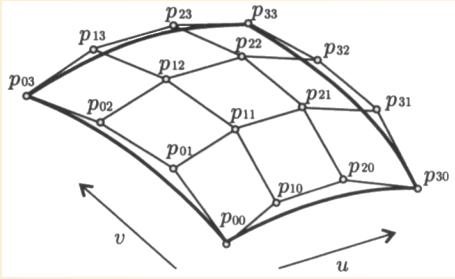
ポリゴンモデル



CSGモデル

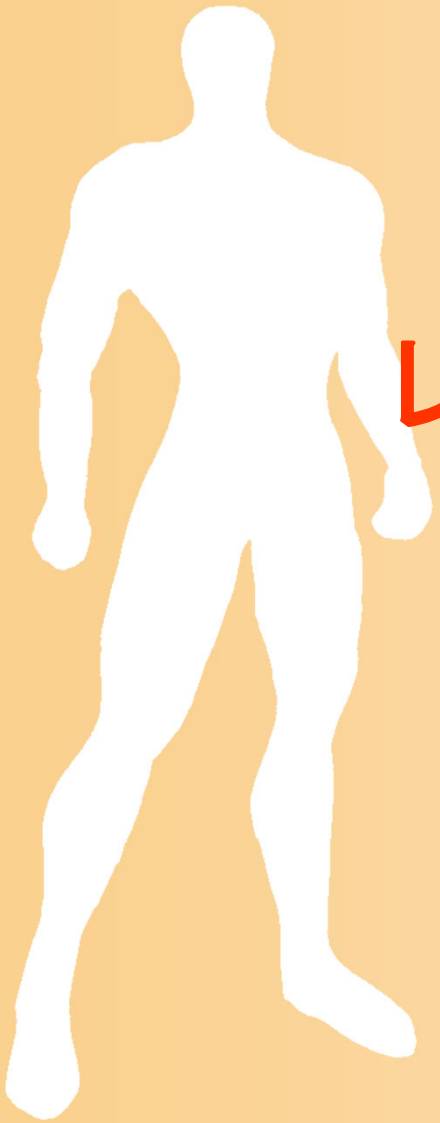


曲面パッチ



各用途ごとの主なモデリング方法

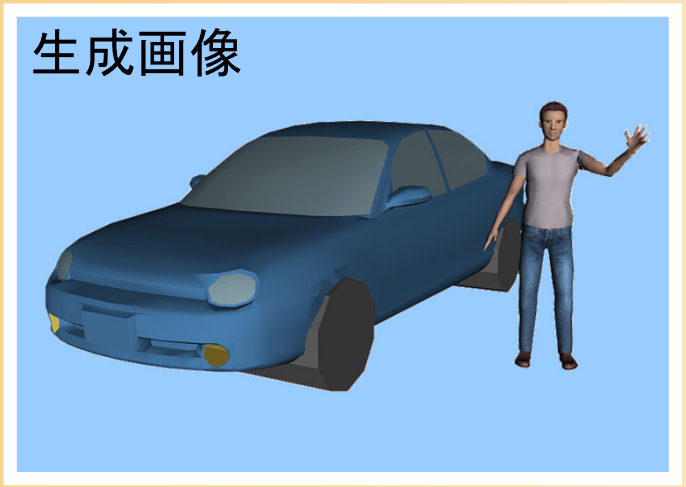
- 映画・工業製品設計などに使われる精緻なモデル
  - サブディビジョンサーフェス、ポリゴンモデル、曲面パッチ（NURBUS）
- コンピュータゲーム向けなどの比較的粗いモデル
  - ポリゴンモデル、サブディビジョンサーフェス
- 特殊なオブジェクトの表現
  - パーティクルやボクセル（炎・煙・水）
  - 高さによる表現（地面）
  - ボクセル（物体のスキャン結果or解析データ）
- 現在はあまり使われていないモデル
  - ワイヤーフレーム、CSG、メタボール、曲面パッチ

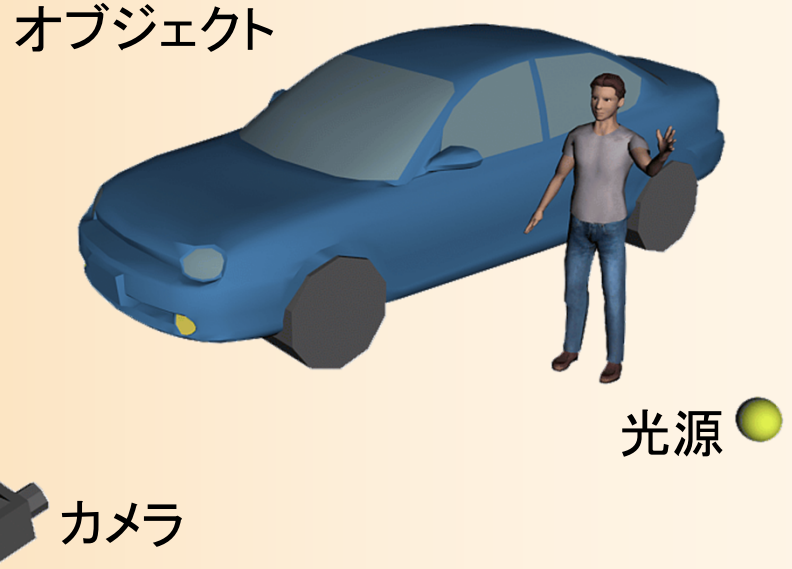


レンダリングの種類

レンダリング

- レンダリング（Rendering）
  - カメラから見える画像を計算するための方法
  - 使用するレンダリングの方法によって、生成画像の品質、画像生成にかかる時間が決まる

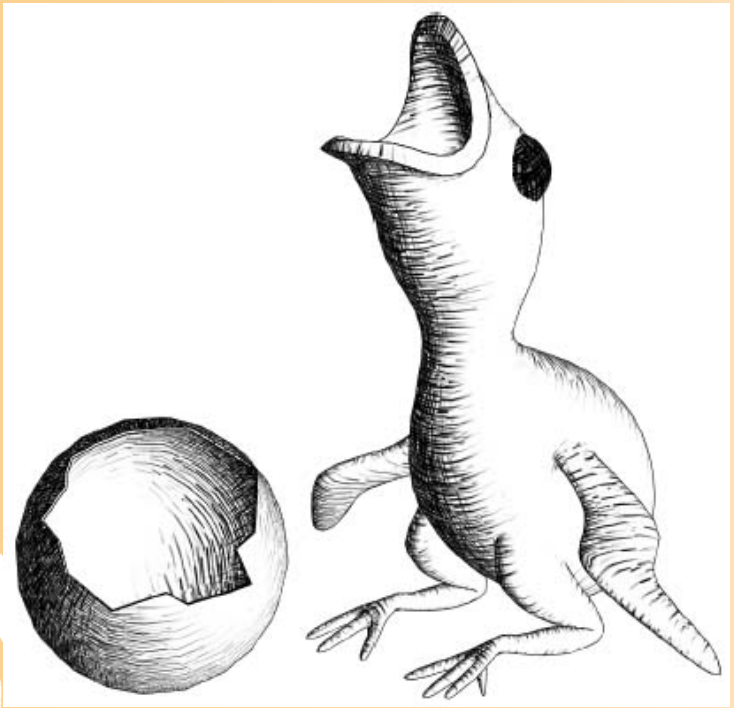




レンダリングの種類

- フォトリアリスティック・レンダリング（Photorealistic rendering）
  - 実際のカメラで撮った画像と区別がつかないような、写実的な画像を生成するための技術
  - これまでは、いかに写実的な画質を実現するかを追求しながらCG技術は開発されてきた
- ノンフォトリアリスティック・レンダリング（Non-photorealistic rendering）
  - 非写実的な、CGならではの画像を生成する技術

ノンフォトの例(1)

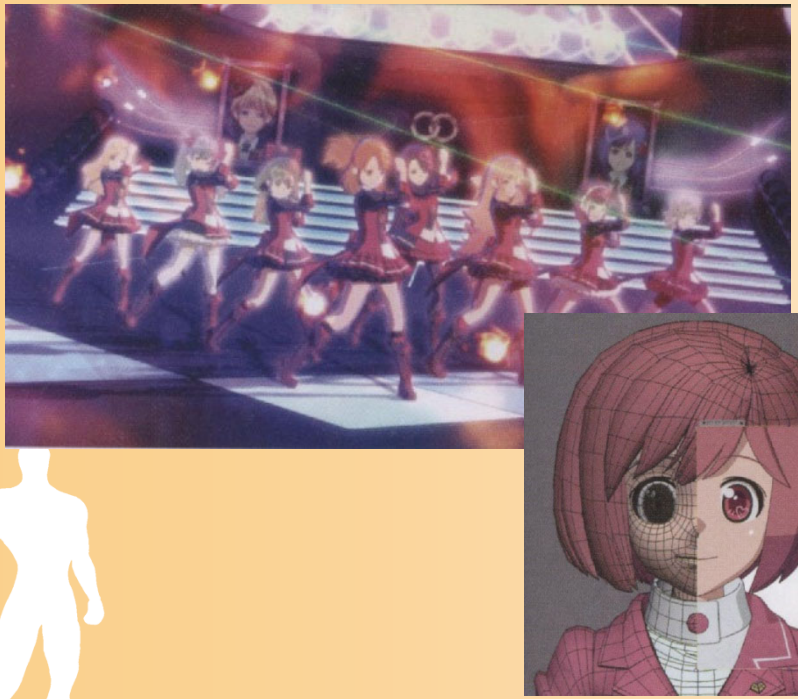


Hatching [Praun 2001]

JSRF ©スマイルビット 2002

ノンフォトの例(2)

- 最近は、セルアニメーション制作でも利用されつつある



AKB0048 © サテライト 2013

ジョジョの奇妙な冒険 © 神風動画 2013

CG WORLD 2013年3月号

ノンフォトの技術

- ノンフォトリアリスティック・レンダリングも、技術的にはフォトリアリスティック・レンダリングの技術をベースにしている
  - 通常のレンダリング手法の一部の処理を変更することで独自の効果を出す
- 今回の講義では、フォトリアリスティック・レンダリングの技術を中心に説明

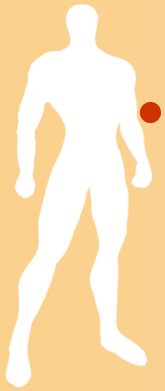
レンダリングの予備知識

レンダリングの予備知識

- レンダリング手法にはいくつか種類がある
  - 速度・画質のバランスを考えて、使用するレンダリング手法を選択する必要がある
- レンダリングを行う時にポイントとなる技術
  - 隠面消去
  - カメラとビューポート、座標変換
  - 面単位での描画
  - 光のモデル、反射・透過・屈折の表現

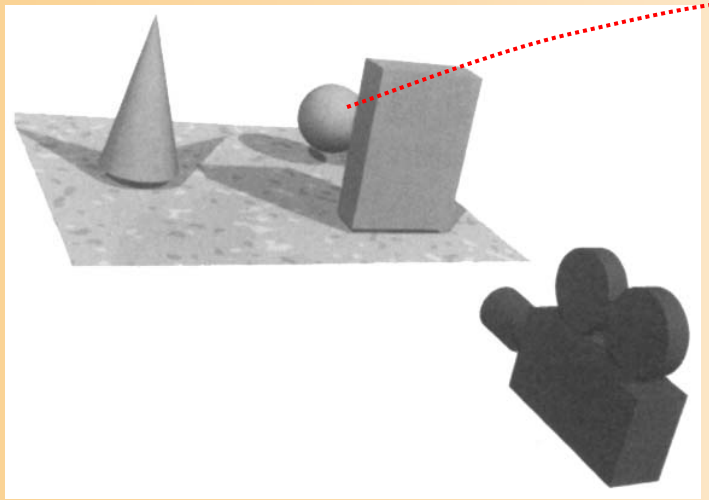
レンダリングの予備知識

- 隠面消去
- カメラとビューポート
- 座標変換
- 面単位での描画
- 光のモデル
- 反射･透過･屈折の表現

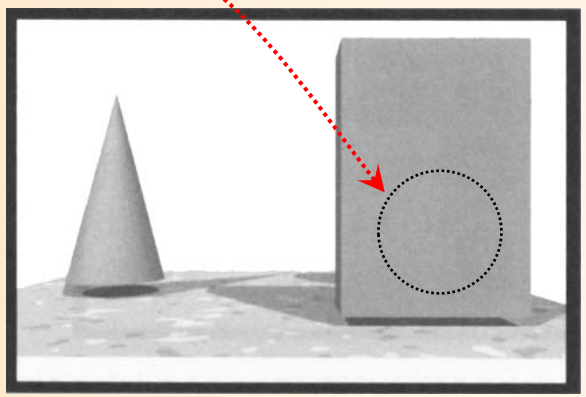


隠面消去(隠面除去)

- 隠面消去をどのようにして実現するか？
  - 見えるはずのない範囲を描画しない処理
  - 普通に存在する面を全て描いたら、見えるはずのない面まで表示されてしまう



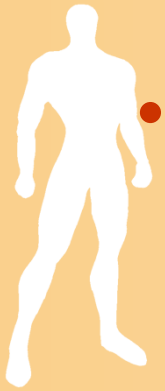
この球は手前の直方体で隠れるため描画しない



参考書「コンピュータグラフィックスの基礎知識」図2-21

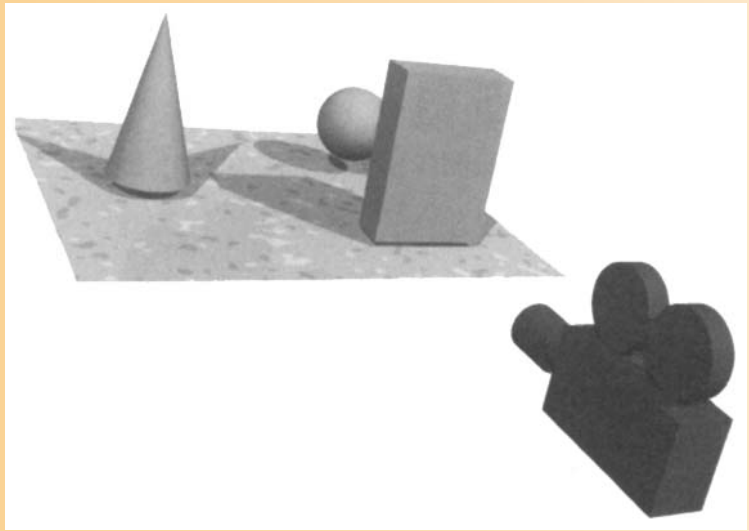
レンダリングの予備知識

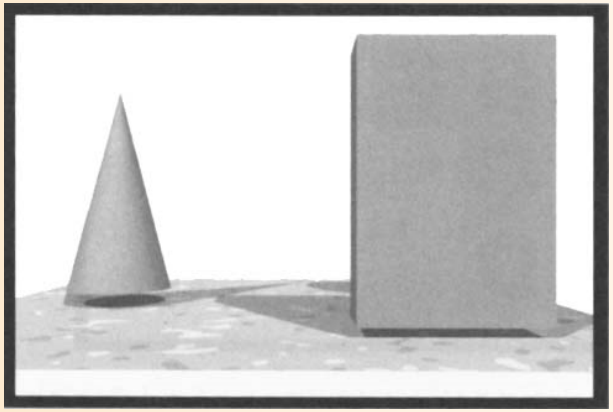
- 隠面消去
- カメラとビューポート
- 座標変換
- 面単位での描画
- 光のモデル
- 反射･透過･屈折の表現



カメラとビューポート

- ビューポート
  - カメラから見える範囲
  - カメラの視野によって見える範囲、見え方が決まる

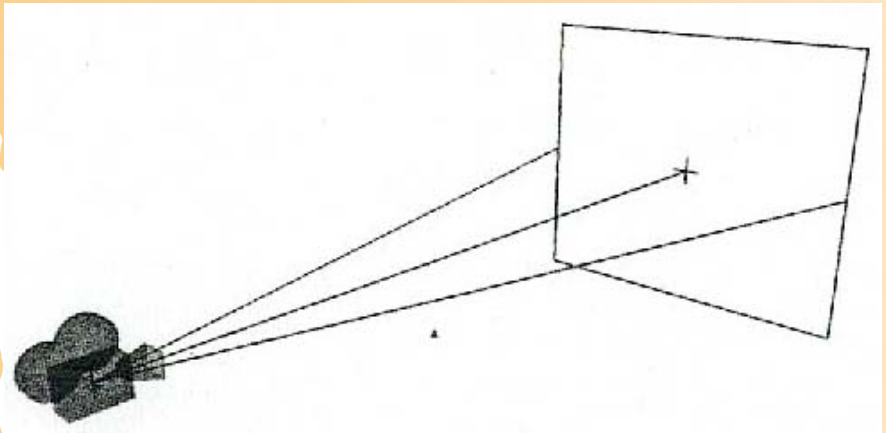




教科書 基礎知識 図2-21

カメラの視野

- カメラの視野(視野角)の設定
  - 対象物との距離や視野角の設定によって見え方が変わる

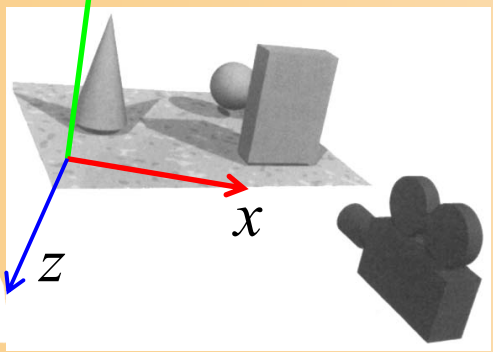


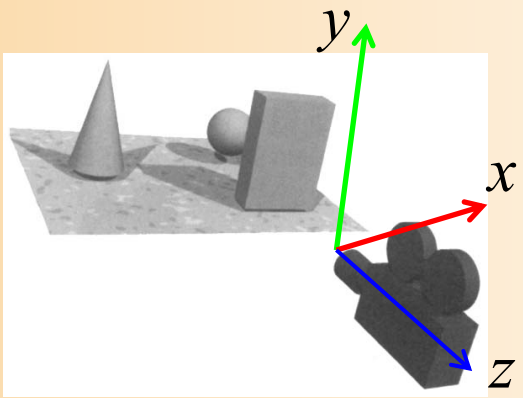
CG制作独習事典 p.11

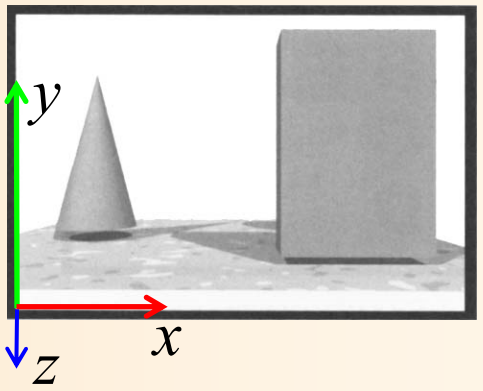
教科書 基礎知識 図2-27

座標変換

- 座標変換
  - ワールド座標からビューポート座標系への変換
  - ビューポート座標系からスクリーンへの透視変換
    - 行列演算によりこれらの処理を実現



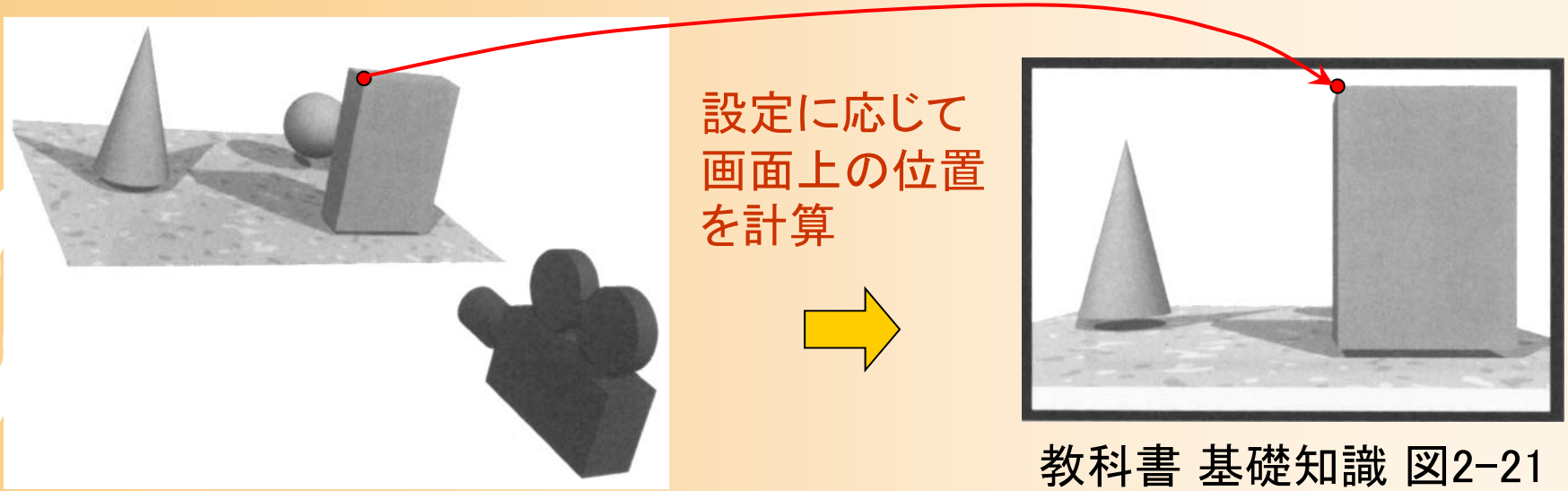




- 座標変換の詳細は今後の講義で説明

まとめ

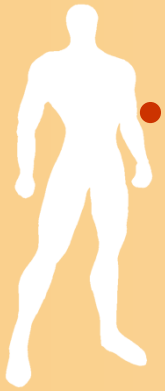
- カメラとビューポート、座標変換
  - レンダリングを行なうときには、あらかじめ、カメラのビューポートや変換行列を設定する
  - 座標変換を適用し、画面上での物体（ポリゴン）の位置を計算する（詳しくは後日の講義で説明）



教科書 基礎知識 図2-21

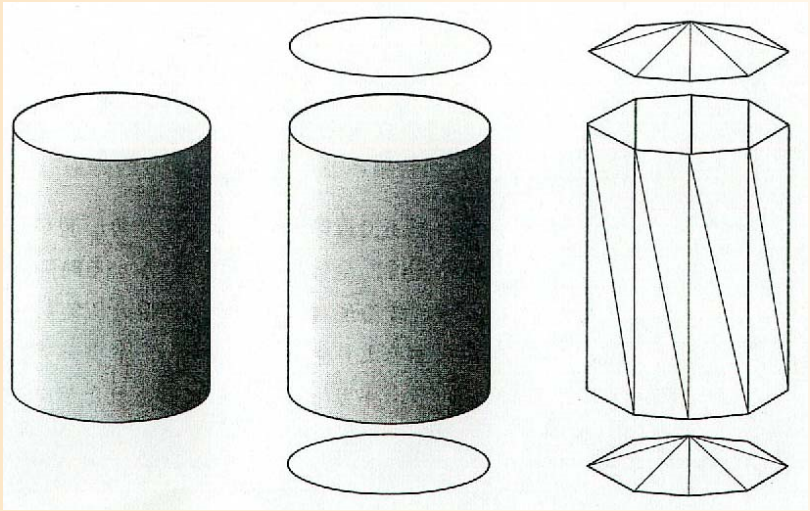
レンダリングの予備知識

- 隠面消去
- カメラとビューポート
- 座標変換
- 面単位での描画
- 光のモデル
- 反射･透過･屈折の表現



ポリゴン単位での描画

- 多くのレンダリング手法では、レンダリングの前に、曲面をポリゴン（多角形）の集合に変換して描画する
  - 実際には、全て三角面の集合に変換する
    - 多角形も三角面の集合に分割する
  - 変換の細かさは状況に応じて変更可能
    - Level of Detail（LOD）と呼ばれる技術



教科書 基礎と応用 図3-2

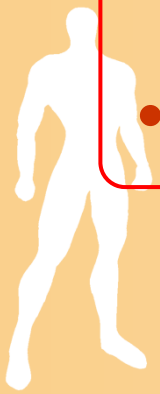
ポリゴン単位での描画

- 三角面(ポリゴン)に変換する理由
  - 三角面は単純なので高速処理に適している
  - 多くのレンダリング手法は、三角面に特化されたものである
  - 三角面に変換することによってデータ量(描画しなければならない要素の数)は大きくなるが、結果的には高速に描画できる

- レンダリング手法によっては、曲面をそのまま描画することもできる
  - レイトレーシング法やスキャンライン法など

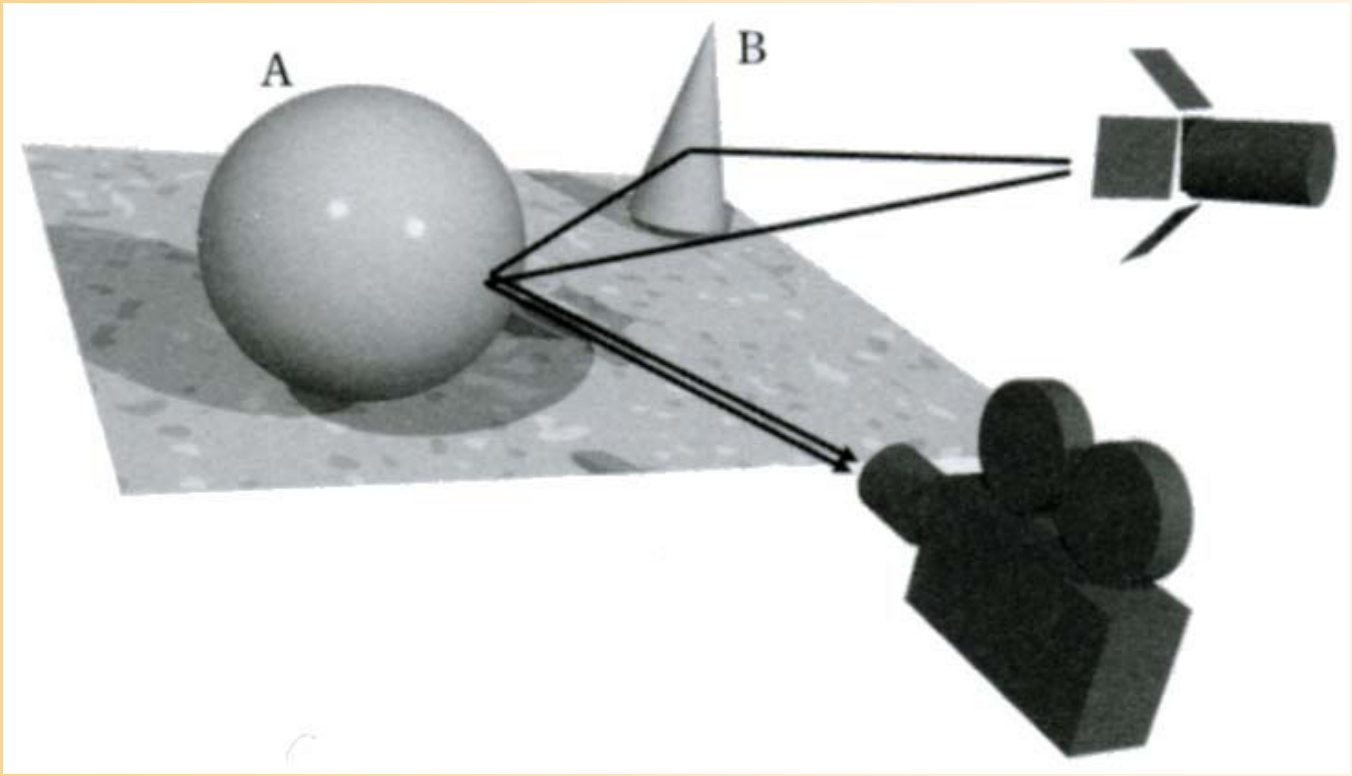
レンダリングの予備知識

- 隠面消去
- カメラとビューポート
- 座標変換
- 面単位での描画
- 光のモデル
- 反射･透過･屈折の表現



物体の色の計算

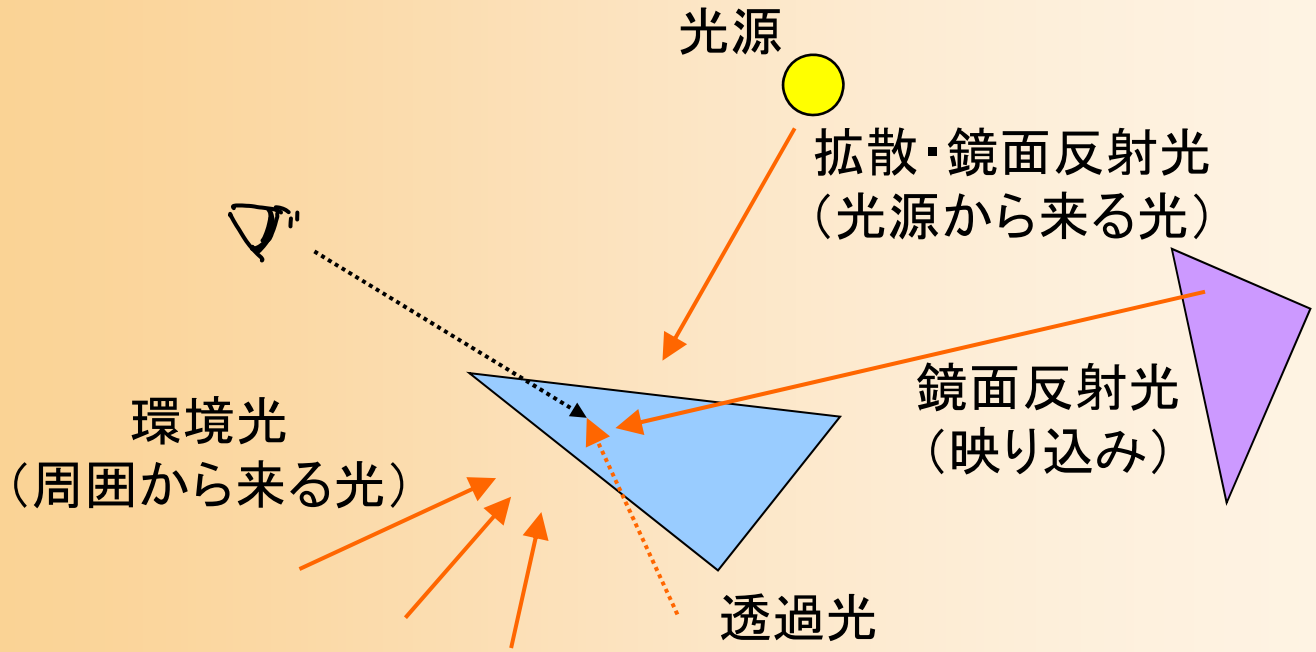
- 素材＋光の影響により物体の色は決まる



教科書 基礎知識 図3-2

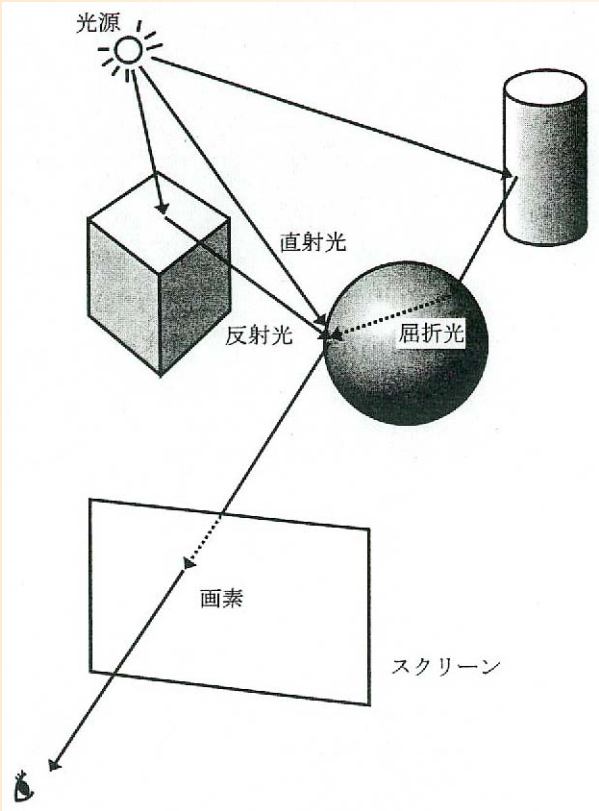
光のモデル

- 光を影響をいくつかの要素に分けて計算
  - 局所照明（光源からの拡散・鏡面反射光）
  - 大域照明（環境光、映り込み、透過光）



反射･透過･屈折の表現

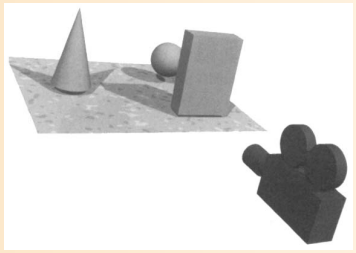
- 反射
  - ある物体に他の物体が映り込む
- 透過
  - 透明な物体の後ろが透けて見える
- 屈折
  - 透過が起こる時に、光線の角度が変化する現象

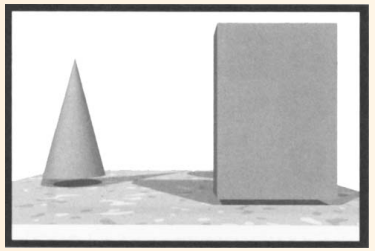


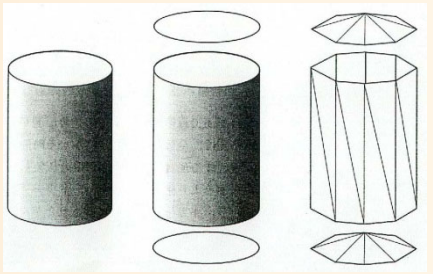
教科書 基礎と応用 図8-2

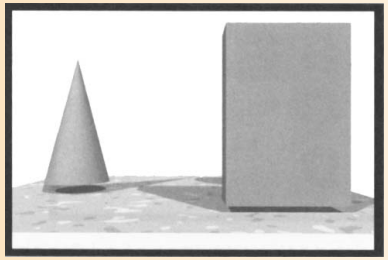
レンダリングの予備知識のまとめ

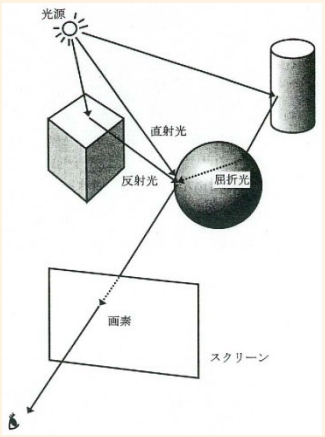
- 隠面消去
- カメラとビューポート
- 座標変換
- 面単位での描画
- 光のモデル
- 反射･透過･屈折の表現











レンダリング手法

レンダリング技術のポイント

- これまでに説明した内容（共通の手法）
  - 曲面はあらかじめ面の集合に分割し、面単位で描画する
  - 座標変換を使って、各面の画面上の位置を計算
- いくつかのレンダリング手法が存在する
  - 最終的に画面に描画する手順・方法が異なる
- 使用するレンダリング技術を決めるポイント
  - どのような方法で隠面消去を行うか？
  - 反射・透過・屈折などを表現できるかも重要

レンダリング手法

- いくつかのレンダリング手法がある
  - Zソート法
  - Zバッファ法
  - スキャンライン法
  - レイトレーシング法

低画質、高速度

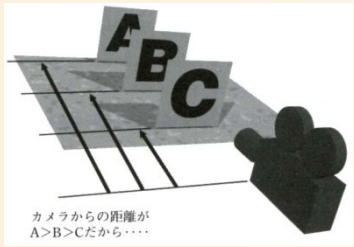
高画質、低速度

  - 速度と画質のトレードオフが存在する
  - 上記の4つは全て現在も実用的に使われている技術（それぞれ一長一短がある）

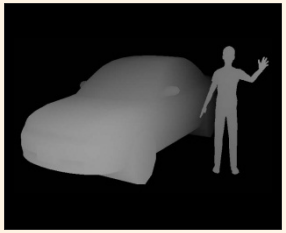
レンダリング手法

- Zソート法
- Zバッファ法
- スキャンライン法
- レイトレーシング法

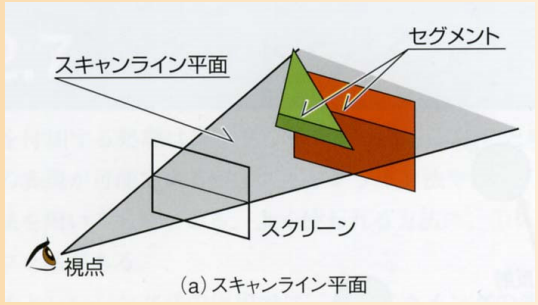
Zソート法



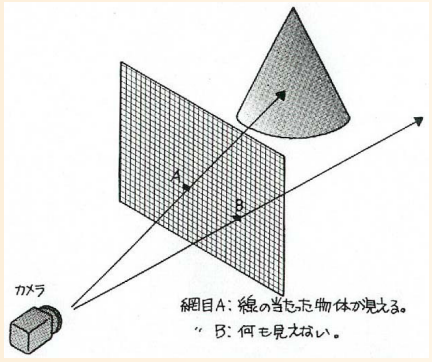
Zバッファ法



スキャンライン法

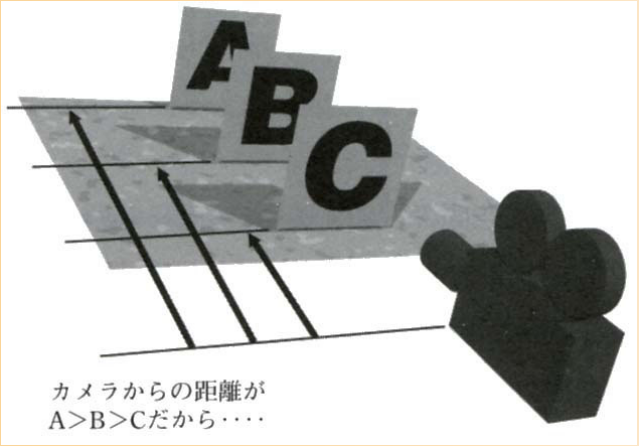


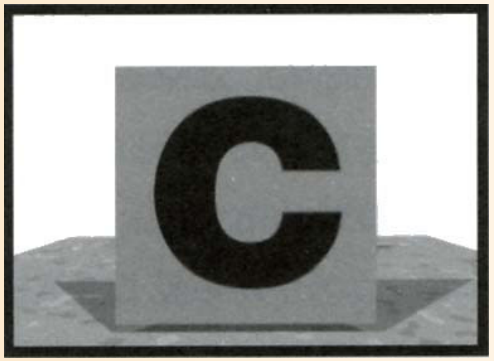
レイトレーシング法



Zソート法

- Zソート法（ペインタ・アルゴリズム）
  - 描画すべき面を奥にあるものから手前にあるものに順番にソート（整列）
  - 奥の面から手前の面へ順番に描画していく
  - 結果的に、奥の面は手前の面で隠れる

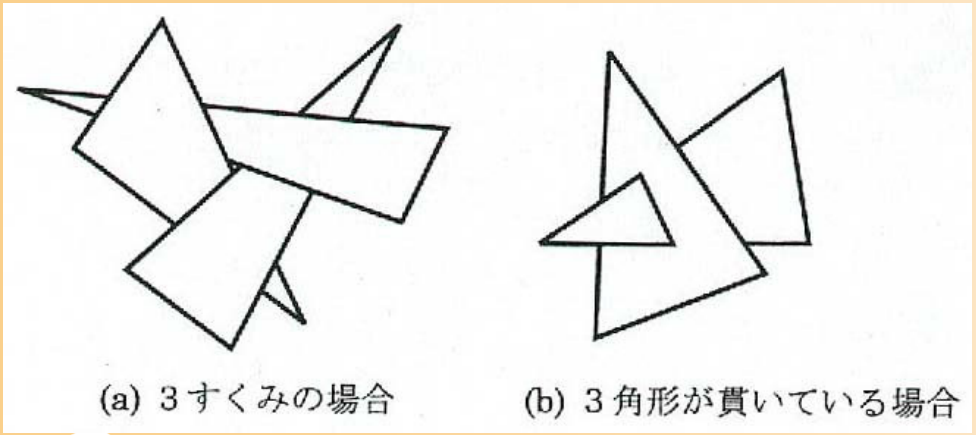


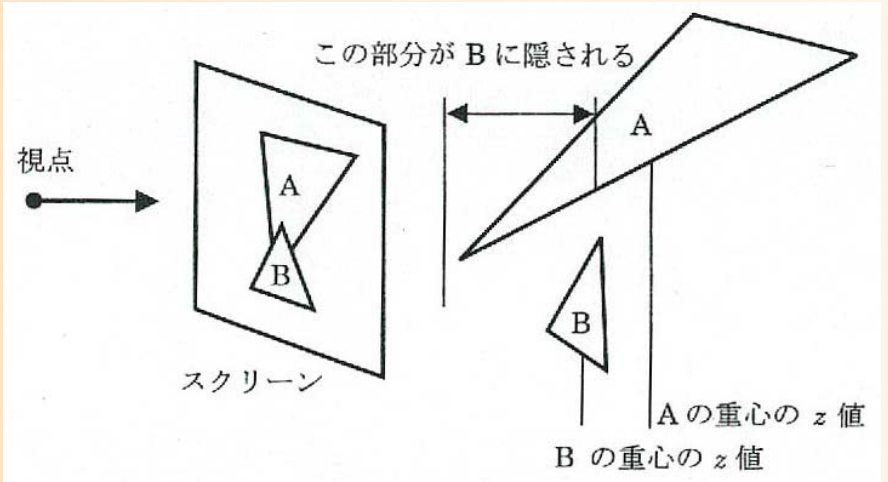


教科書 基礎知識 図2-30

Zソート法の問題点

- 面同士の前後関係を正しく判定できない
- 面同士が交差する時にうまく処理できない



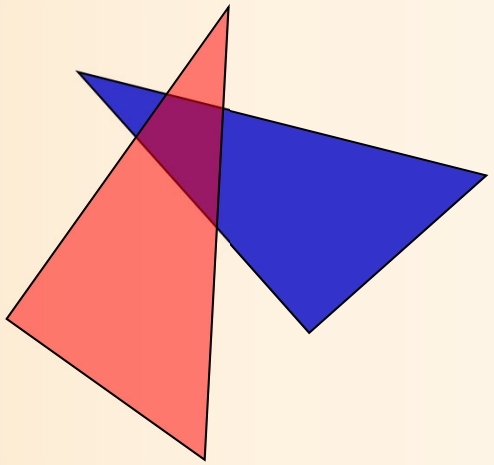


教科書 基礎と応用 図3.6

- 完璧な解決方法はない
  - 面単位で判定しようとすることが問題

Zソート法での透過の表現

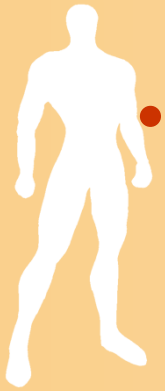
- 面を半透明で描画することで裏が透けて見えるので、透過のような結果が得られる
  - 面を描く時に、面の色と、後ろのピクセルの色を一定比率で混ぜ合わせる
  - ただし、混ぜ合わせには計算量がかかるので、ハードウェアがサポートしていなければ困難



- 屈折や反射はこの方法では表現できない

Zソート法の特徴

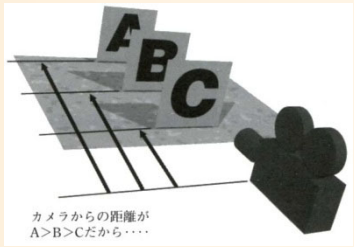
- メリット
  - 高速
    - ただし全ての面を描画するため無駄な点もある
  - それほど余分なメモリ領域を必要としない
    - 面をソートするためのメモリ領域のみ必要
- デメリット
  - 面同士の交差をうまく処理できない
- 主な用途
  - メモリの非常に限られる用途で、リアルタイムに描画する必要のあるような用途（PlayStation1）



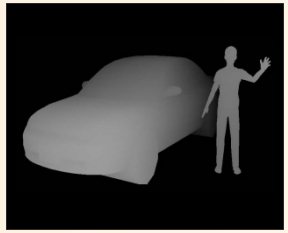
レンダリング手法

- Zソート法
- Zバッファ法
- スキャンライン法
- レイトレーシング法

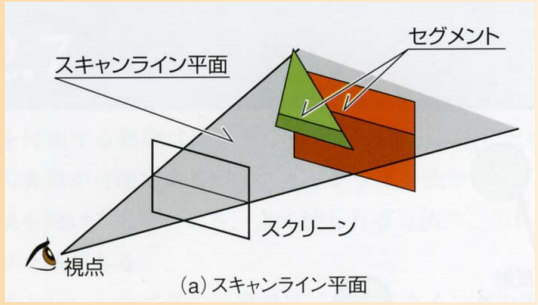
Zソート法



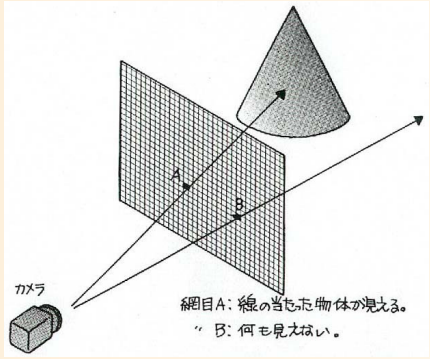
Zバッファ法



スキャンライン法

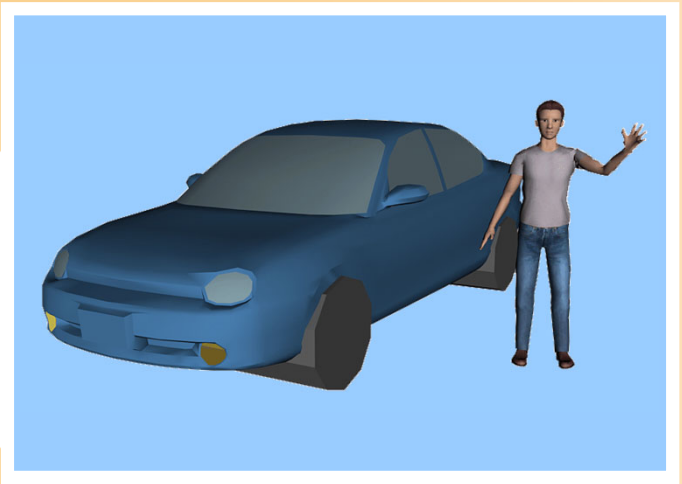


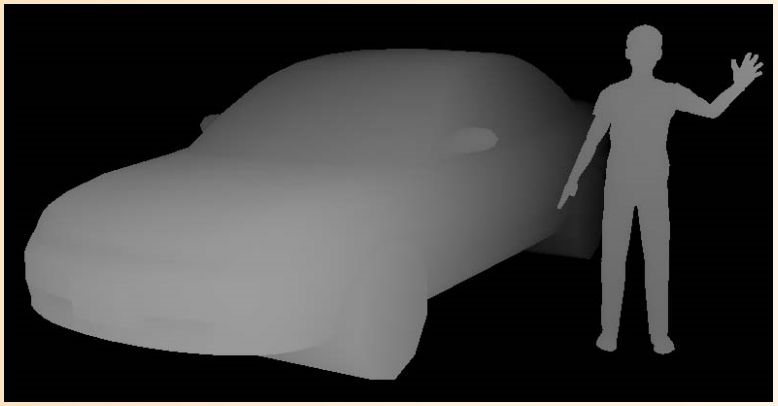
レイトレーシング法



Zバッファ法

- 画像とは別に、それぞれのピクセルの奥行き情報であるZバッファを持つ
  - Zバッファは画像とほぼ同じメモリサイズを使用
    - ピクセルあたり16ビット～32ビットを使用
  - ピクセル単位で処理するので交差も処理できる

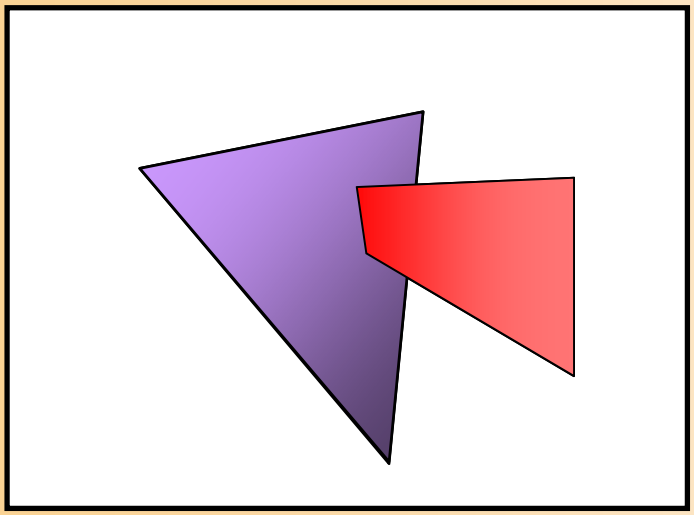


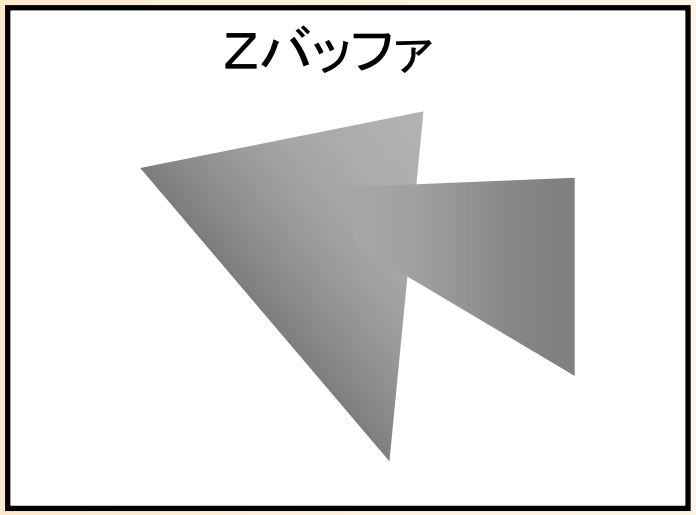


Zバッファの値（手前にあるほど明るく描画）

Zバッファ法による隠面消去

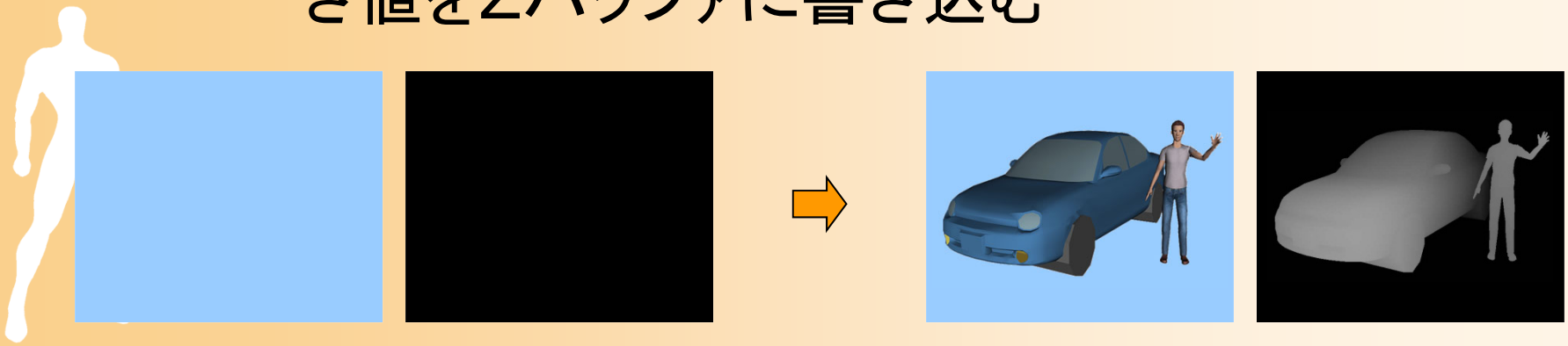
- Zバッファ法による面の描画
  - 面を描画するとき、各ピクセルの奥行き値（カメラからの距離）を計算して、Zバッファに描画
  - 同じ場所に別の面を描画するときは、すでに描画されている面より手前のピクセルのみを描画





Zバッファ法による描画の手順

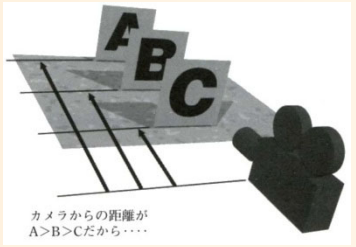
- アルゴリズム
  1. Zバッファの全ピクセルを無限遠で初期化
  2. 面を描く時に、各ピクセルの奥行き値を見て、これから描こうとしている面の方が奥にあればそのピクセルは描画しない
  3. 各ピクセルを書き込む時に、同時にその奥行き値をZバッファに書き込む



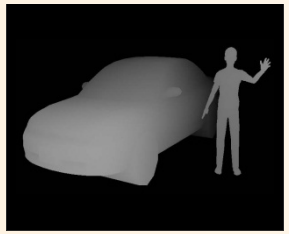
Zソート法とZバッファ法の比較

- どちらも面ごとに描画を行なう点で共通
- Zソート
  - 面単位で描画、面単位で隠面消去
  - 隠面消去がうまくいかないことがある
- Zバッファ
  - 面単位で描画、ピクセル単位で隠面消去
  - Zバッファのためのメモリが必要

Zソート法



Zバッファ法



Zバッファ法の特徴

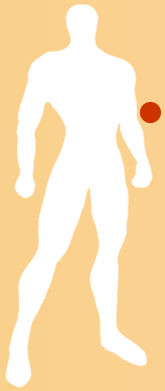
- メリット
  - ハードウェアでサポートされていれば、不必要なピクセルは描画しないで済むため、Zソート法より高速
  - 処理が単純なので、ハードウェアで実現しやすい
- デメリット
  - Zバッファのためのメモリを余分に必要とする
- 主な用途
  - メモリに余裕のある環境で、リアルタイムに描画する必要のあるような用途
  - 現在のパソコン用グラフィックスカード、家庭用ゲーム機はほとんどこの技術を採用している
    - ハードウェアによるZバッファをサポートしている

Zソート法とZバッファ法の速度比較

- Zソート法とZバッファ法はどちらが高速か？
  - ハードウェアがZバッファをサポートしていない場合は、Zソート法の方が高速
    - 各ピクセルの描画の度にZテスト（奥行き情報の比較）を行うのは非常に時間がかかるため
  - ハードウェアがZバッファをサポートしていれば、Zバッファ法の法が高速
    - Zソートが不要
    - 無駄なピクセルを描画する必要がなくなるため
    - リアルタイム処理では、なるべく少しでもメモリアクセス（特に書き込み）の回数を減らすことが重要

Zバッファ法での透過の表現

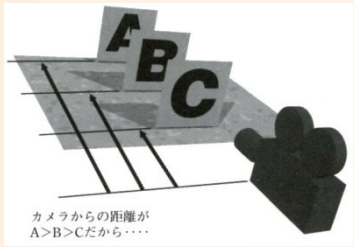
- Zソートと組み合わせることで透過を表現可能
  - 先に非透過の面を全て描画した後で、透過の面を後ろから順番に描画
  - 奥行きでソートするための処理時間がかかるが、非透過の面については前方の面から順番に描画することで、不要な面の描画を減らし、結果的に高速化できる可能性もある
- 反射や屈折はそのままでは表現できない



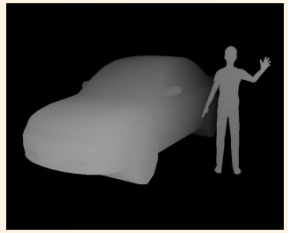
レンダリング手法

- Zソート法
- Zバッファ法
- スキャンライン法
- レイトレーシング法

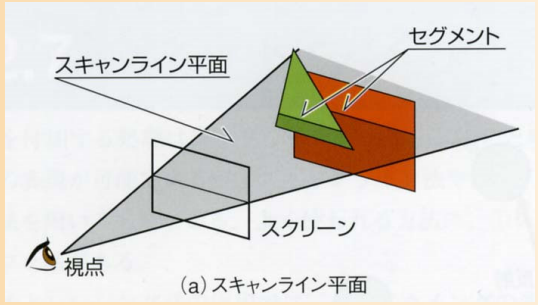
Zソート法



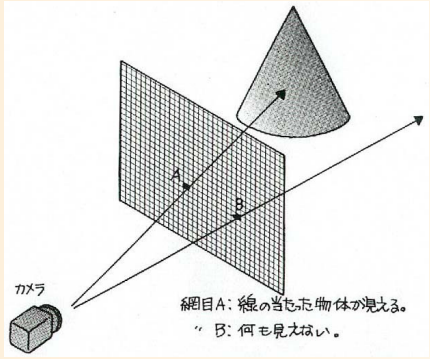
Zバッファ法



スキャンライン法

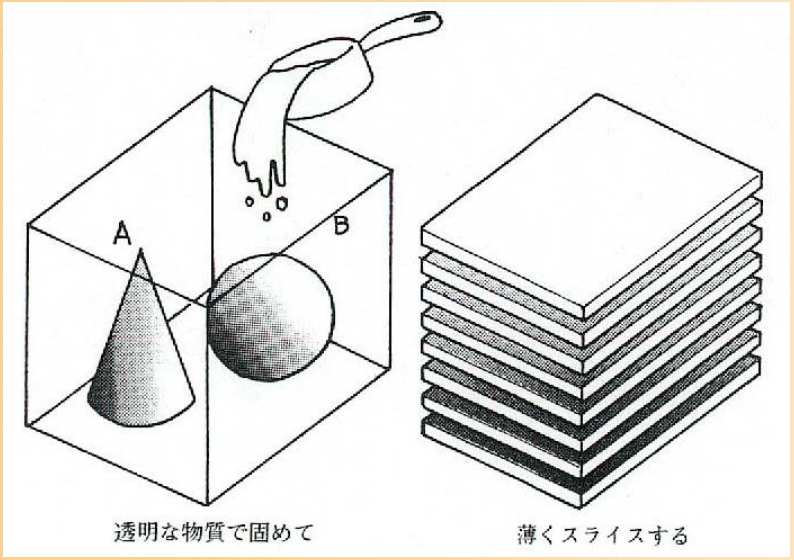


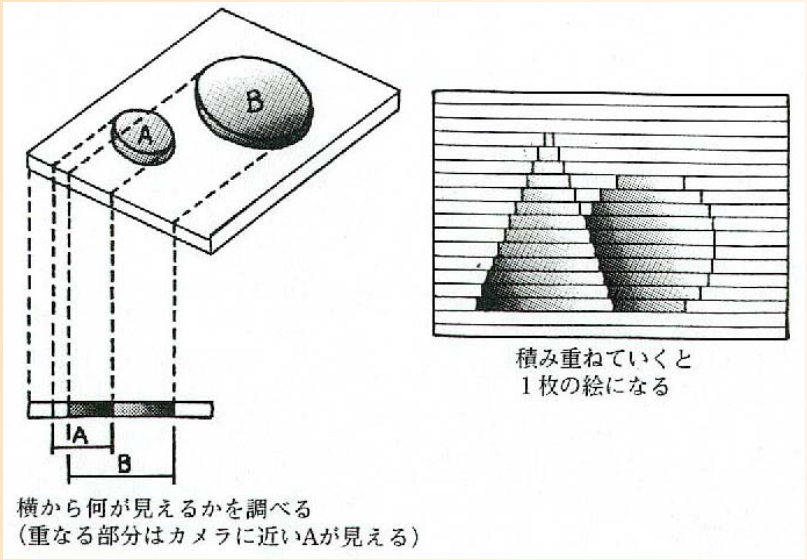
レイトレーシング法



スキャンライン法

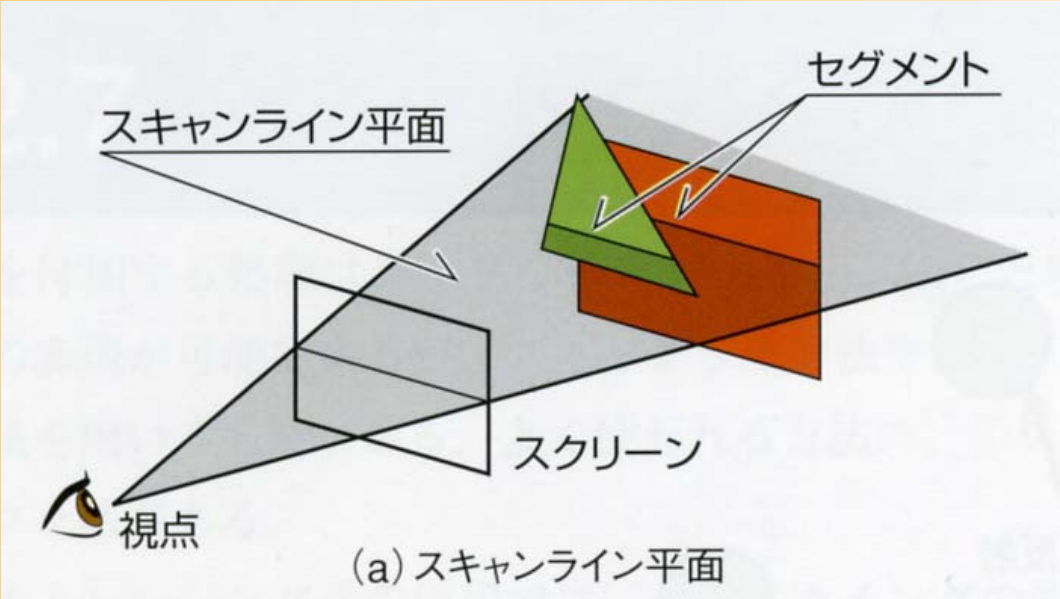
- 画像のそれぞれのラインごとに描画
- ラインの各スパンごとにどの線分が描画されるかを判定し、それぞれのスパンを描画

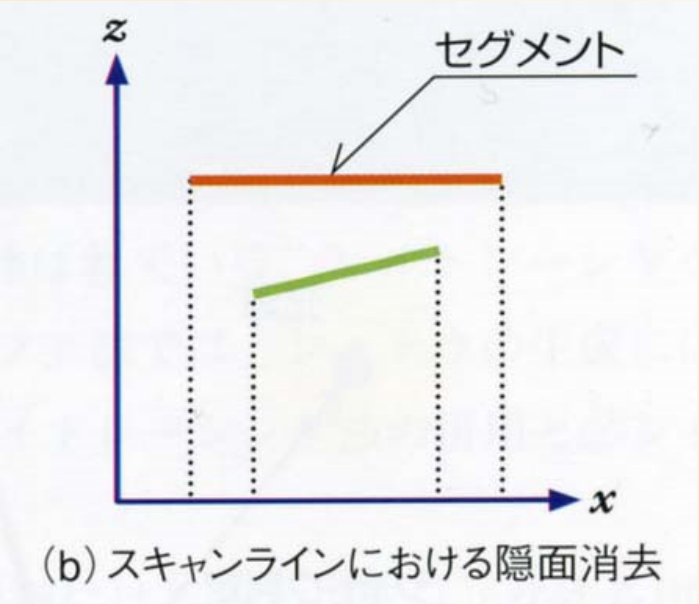




教科書 基礎知識 図2.31

スキャンライン法の説明





教科書 3DCGアニメーション 図2.31

スキャンライン法の特徴

- メリット
  - メモリが少なくて済む
  - より正確な透過が計算できる
  - 境界を正確に判定できるため、ジャギー（後述）を減らすことができる
- デメリット
  - 処理が複雑（ハードウェア実装は困難）
  - 時間がかかる（リアルタイムではまず無理）
- 主な用途
  - 画質が非常に綺麗で、時間的にも許容範囲であるため、現在映画などに主に使われている

これまでの手法の問題

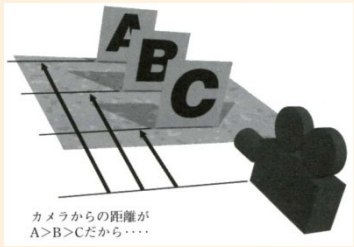
- これまでの手法は、面単位で描画する技術
- 透過は処理できても、反射・屈折などはうまく再現できない
  - ただし、擬似的にこれらを実現する方法もあるので、次回紹介する
    - 環境マッピング

反射を考慮したレンダリングの例
基礎と応用 図8.9

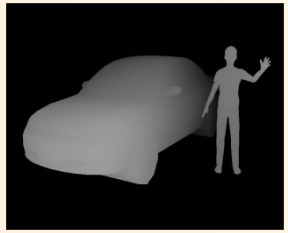
レンダリング手法

- Zソート法
- Zバッファ法
- スキャンライン法
- レイトレーシング法

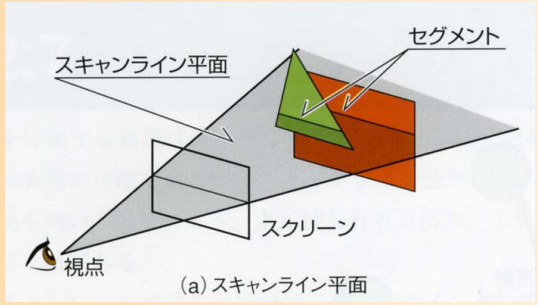
Zソート法



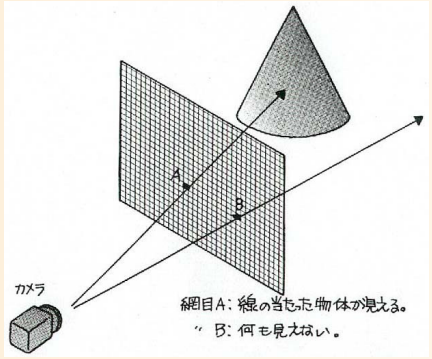
Zバッファ法



スキャンライン法

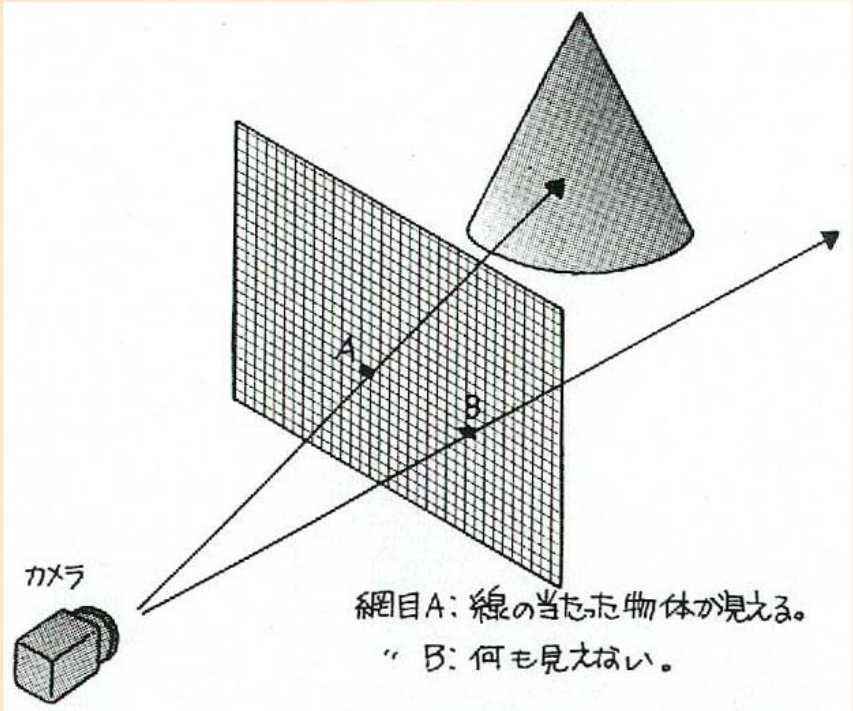


レイトレーシング法



レイトレーシング法

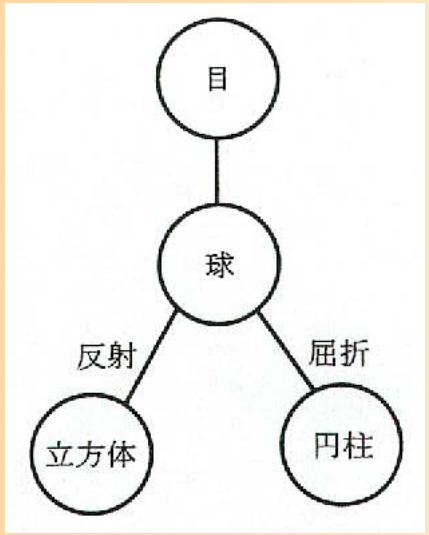
- 最も高品質な(最も実写に近い)画像が得られる方法
  - カメラからそれぞれのピクセルごとに視線方向に半直線(レイ)を飛ばし、物体との交差判定によりピクセルの色を計算

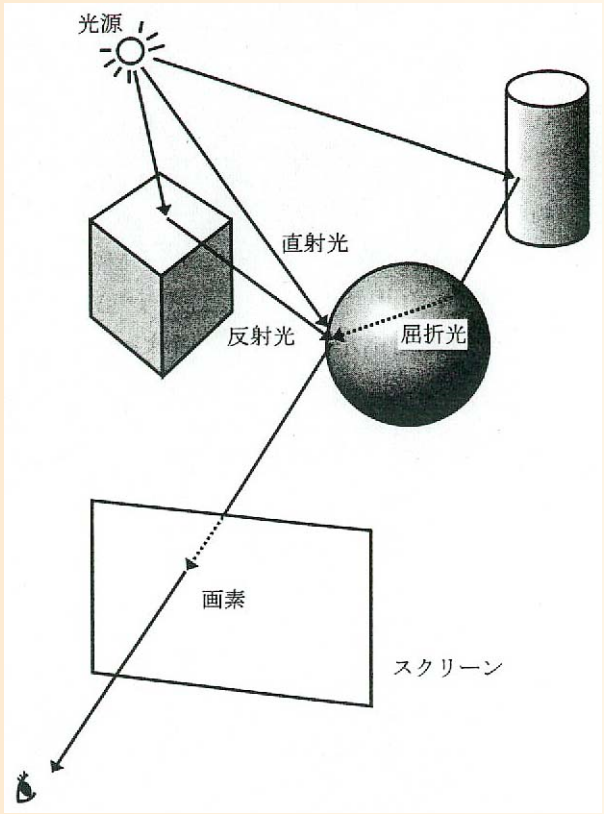


教科書 基礎知識 図2-33

レイトレーシング法の計算

- 反射・透過・屈折の計算
  - 物体にぶつかったら、反射・屈折成分を物体の色に加えることで、そのピクセルの色を決定

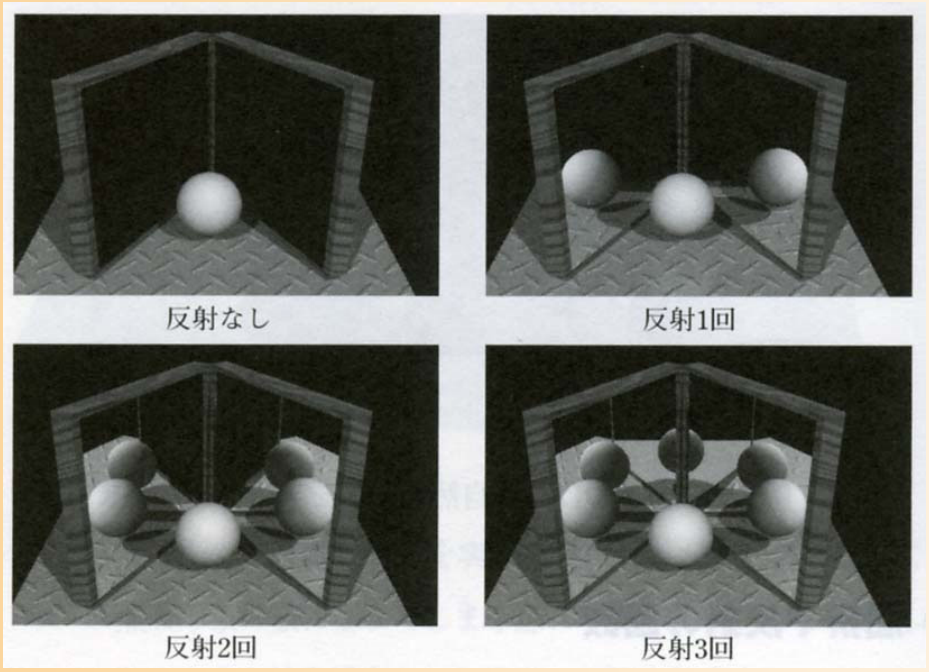




教科書 基礎と応用 図8.2 図8.3

レイトレーシング法の計算

- 反射・屈折のくり返し計算
  - 何回も反射を計算しているときりがないので、一定回数できりあげる（シーンに応じて設定）



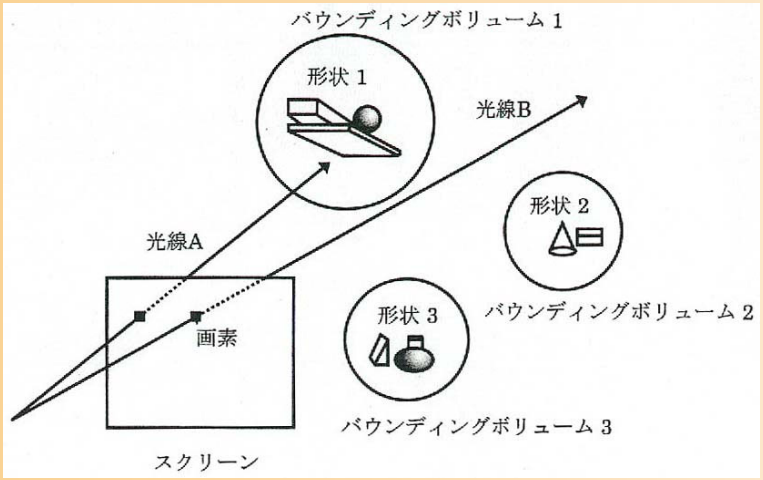
教科書 基礎知識 図3-32

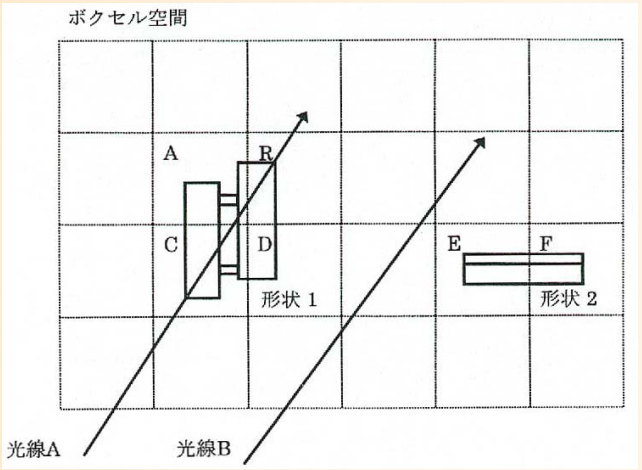
レイトレーシング法の特徴

- メリット
  - 最も高品質
  - 反射・透過・屈折もある程度正しく表現できる
- デメリット
  - 計算に非常に時間がかかる
- 主な用途
  - とにかく品質の要求される静止画像（ポスタ等）
  - 映画などに使うには時間がかかりすぎる
    - 実際には屈折の表現が必要な状況は少ないので、スキャンライン法でも十分な画質が得られる

レイトレーシングの高速化の工夫

- 空間インデックスの導入による高速化
  - あるレイに対して、すべてのオブジェクトとの交差を求めるのは非常に時間がかかる
  - 空間インデックスにより、交差する可能性のあるオブジェクトを絞り込むような処理をまず行う





教科書 基礎と応用 図8.11 図8.12

レンダリング手法のまとめ

- Zソート
  - 面単位で描画、面単位で隠面消去
- Zバッファ
  - 面単位で描画、ピクセル単位で隠面消去
- スキャンライン
  - ライン単位で描画、ピクセル単位※で隠面消去
- レイトレーシング
  - ピクセル単位で描画、ピクセル単位※で隠面消去

※ 実際には、さらに細かい単位で描画が可能
（詳細は、後のアンチエイリアシングの所で説明）

レンダリング手法の比較

- Zソート
  - リアルタイム処理、たまに不自然な画ができる
- Zバッファ
  - リアルタイム処理、それなりの画質が得られる
- スキャンライン
  - まあまあの速度で、かなりの画質が得られる
- レイトレーシング
  - 画質は非常に高いが、速度が遅すぎて実用には向かない

レンダリング手法の用途

- コンピュータゲーム等のリアルタイム・アニメーション
  - Zバッファ法
  - ハードウェアが対応しておらず、計算速度も遅ければ、Zソート法
- 映画等のオフライン・アニメーション
  - スキャンライン法、（レイトレーシング法）
- 静止画像
  - レイトレーシング法、（スキャンライン法）

レンダリングの高速化のための工夫

レンダリング関連の技術

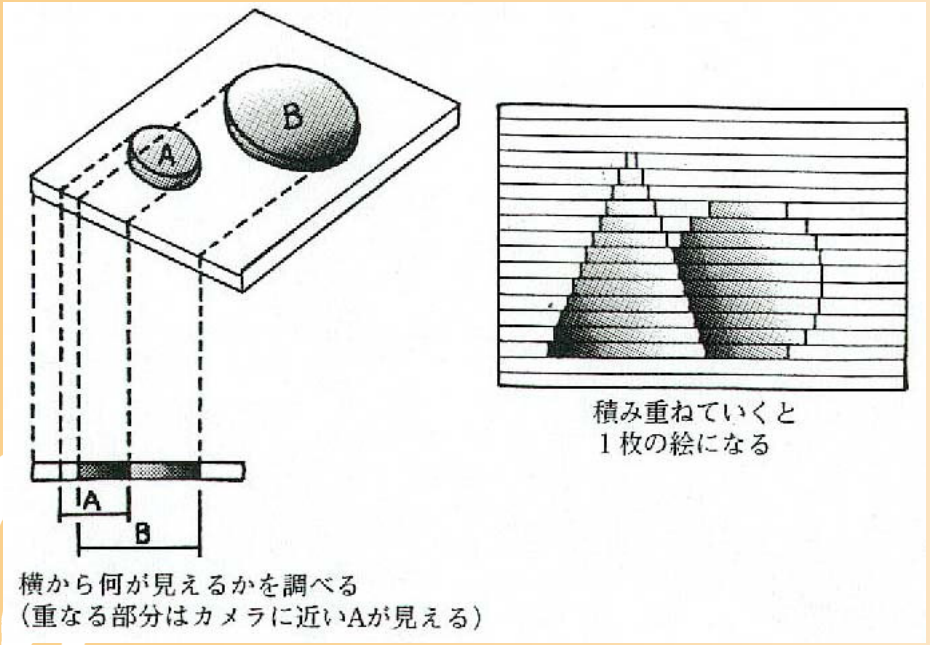
- レンダリング処理の高速化のための工夫
  - 並列処理による高速化
  - 背面消去
- その他のレンダリング関連の重要な技術
  - 光の表現方法
    - シェーディング、ラジオシティ
  - 素材の表現方法
    - マッピング
  - これらについては今後の講義で説明

並列処理による高速化

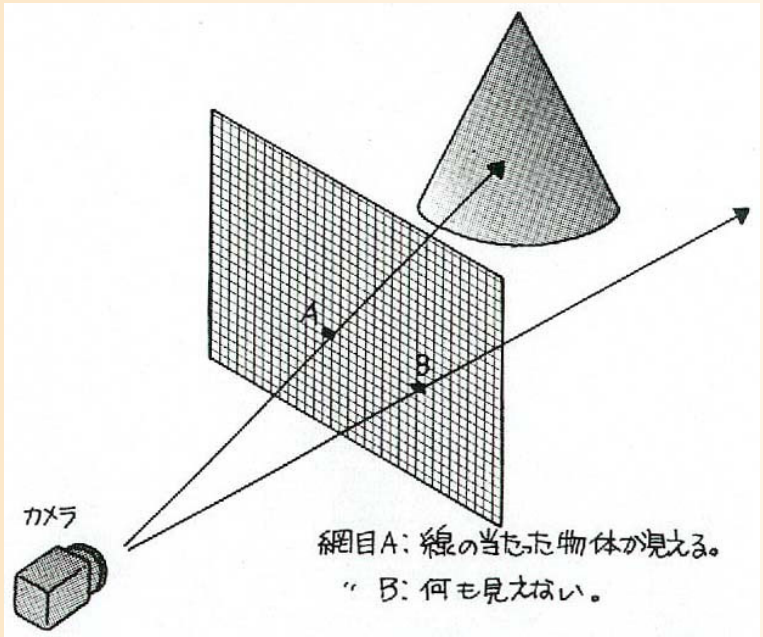
- スキャンライン法、レイトレーシング法は並列処理が可能
  - スキャンライン法はそれぞれのラインごと
  - レイトレーシング法はそれぞれのピクセルごと
    - に独立に処理することができる
- 現在の主要な商用CGツールはネットワークレンダリングの機能を持っている
  - 映画などの高品質の画像の生成では、多数のコンピュータを使ってレンダリングされている

並列処理による高速化

- スキャンライン法とレイトレーシング法の例



教科書 基礎知識 図2.31



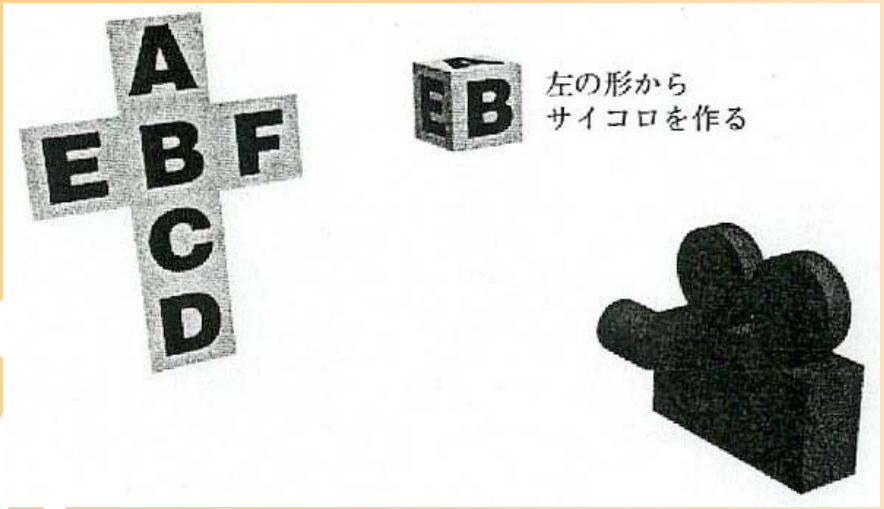
教科書 基礎知識 図2-33

背面消去

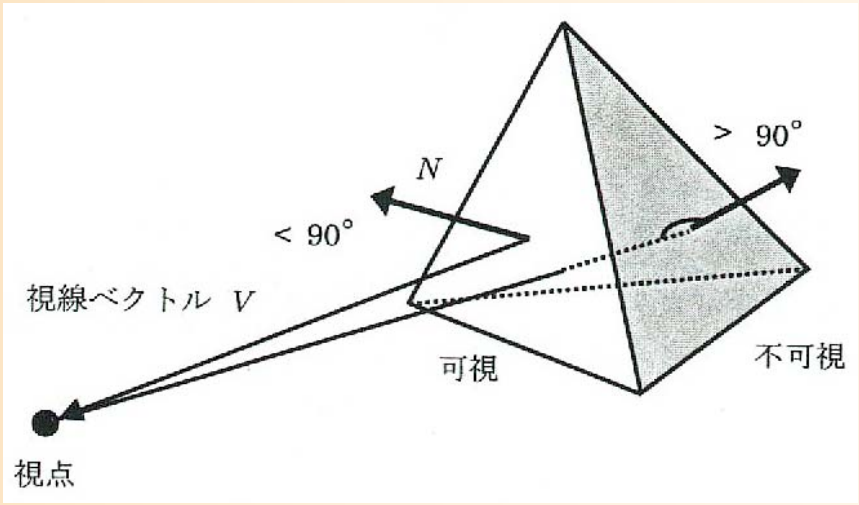
- 背面消去（後面消去、背面除去、後面除去）
  - バックフェースカリング、とも呼ぶ
- 後ろ向きの面の描画を省略する処理
- サーフェスモデルであれば、後ろ向きの面は描画は不要である点に注目する
  - 仮に描画したとしても、その後、手前側にある面で上書きされる
  - 裏向きの面の描画を省略することで処理を高速化できる（単純に考えると、約半分に減らせる）

背面消去

- 後ろ向きの面の判定方法
  - 視線ベクトル(カメラから面へのベクトル)と面の法線ベクトルの内積により判定



教科書 基礎知識 図2-22



教科書 基礎と応用 図3.5

背面消去

- 背面消去が使えない場合
  - 内側が見えている箱や壺など、裏面も描画する必要がある場合
  - 一時的に両面を描画する or 裏面も描画する必要がある場合は2枚の面を張り合わせておくことによって対処

表面のみ描画

裏面も描画

サンプリング

サンプリング

- サンプリング（Sampling）
  - コンピュータグラフィックスでは、ピクセル単位で処理を行う
    - 本来は連続的な映像を、ピクセルごとに離散化して扱うことになる
  - そのままではカメラや人間の眼とは異なる画像となってしまうことがある
    - エイリアシング、モーションブラー、被視界深度
    - このような問題を解決するために、サンプリングの方法を工夫する技術がある

サンプリングの関連技術

- アンチエイリアシング
- モーションブラー
- 被視界深度

アンチエイリアシング

- ジャギー
  - ピクセル単位で描画を行うことにより、物体の境界にギザギザが生じてしまう
  - アンチエイリアシングによりジャギーを軽減

(a)アンチエリアシングなし

(b)アンチエリアシングあり

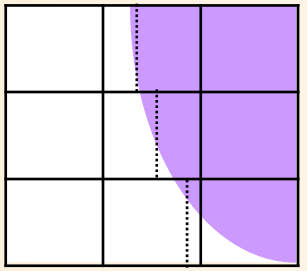
教科書 3DCG アニメーション 図2.47

アンチエイリエシングの手法(1)

- オーバーサンプリング
  - 視点を少しずつ変えて複数の画像をレンダリングし、その平均を取る(合成する)
    - 高品質だが時間がかかる
- リサイジング
  - 画面よりも大きなサイズでレンダリングを行って、縮小して表示する
- フィルタリング
  - レンダリングした画像に対して、周囲のピクセルの色を混合するようなフィルタを適用する

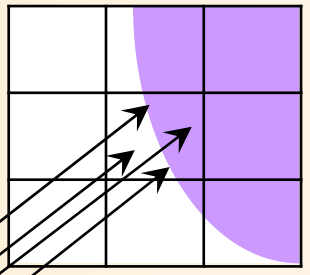
アンチエイリエシングの手法(2)

- スキャンライン法の場合
  - 境界が明示的に判定できるため、境界のみでの色の混合を行うことができる
    - 比較的効率的に実現できる



ピクセル内の境界の位置により
ピクセルの色(混合比率)を決定

- レイトレーシング法の場合
  - ピクセルよりも細かい単位でレイを飛ばして、色を混合を行なうこともできる
    - リサイジングよりも効率的



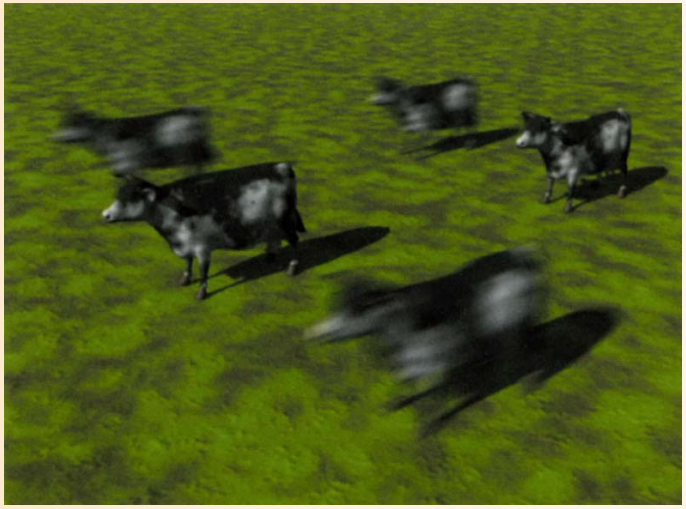
ピクセルに飛ばした複数のレイの
色を混合してピクセルの色を決定

サンプリングの関連技術

- アンチエイリアシング
- モーションブラー
- 被視界深度

モーションブラー

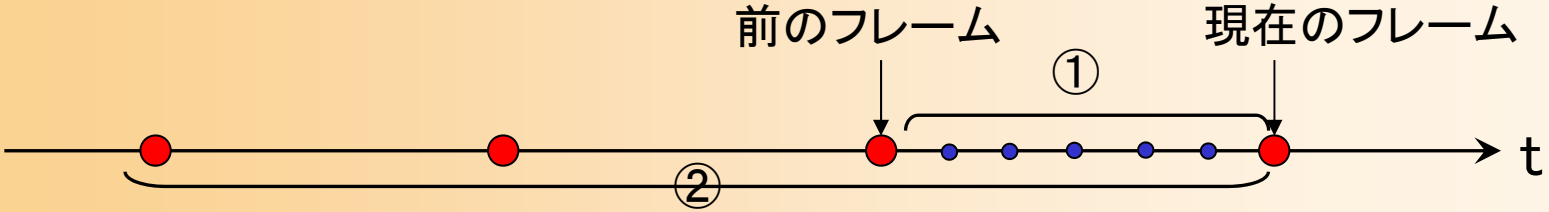
- モーションブラーブラー
  - カメラや人間の眼では、高速に運動する物体はぶれて見える
    - カメラの撮影や人間の認識には、一瞬ではなく、一定時間内の多少の間があるため
  - CGでは、普通に描画すると全ての物体がくっきりと見えてしまう
  - モーションブラーの効果を擬似的に実現



教科書 3DCGアニメーション 図2.48

モーションブラーの実現方法

- オーバーサンプリング
  - 時間を少しずつ変えて複数の画像をレンダリングし、その平均を取る
    - アンチエイリアシングとの組み合わせも可能
  - 高速で動く物体は、少しずつずれて描画される
- 残像
  - 前フレームの画像に重ねて描画
  - 厳密にはモーションブラーとは異なる

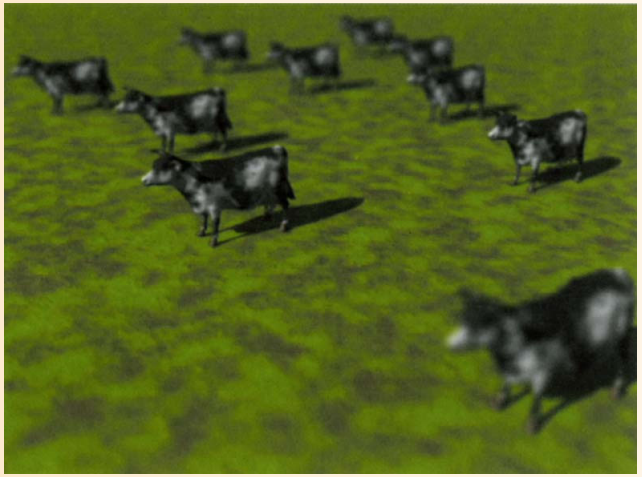


サンプリングの関連技術

- アンチエイリアシング
- モーションブラー
- 被視界深度

被視界深度

- 被視界深度
  - フォーカスの合う範囲
  - カメラや人間の眼では、フォーカスの合わない範囲はぼやけて見える



教科書 3DCGアニメーション 図2.49

- 実現方法
  - 焦点は固定したままでカメラの位置を変えながら複数の画像をレンダリングし、平均を取る
  - フィルタによる実現方法もある

まとめ

- レンダリングの種類
- レンダリング技術
  - ポリゴンへの分割、隠面消去、光のモデル、反射・透過・屈折の表現
- 各種のレンダリング手法
  - Zソート法、Zバッファ法、スキャンライン法、レイトレーシング法
  - レンダリングの高速化の工夫
- サンプリング

次回予告

- 次回（講義）
  - レンダリング・パイプライン
  - ポリゴン描画
- 次々回（演習）
  - プログラミング演習
  - ポリゴンモデル描画（モデリング、レンダリング）