コンピューターグラフィックスS

第14回　アニメーション

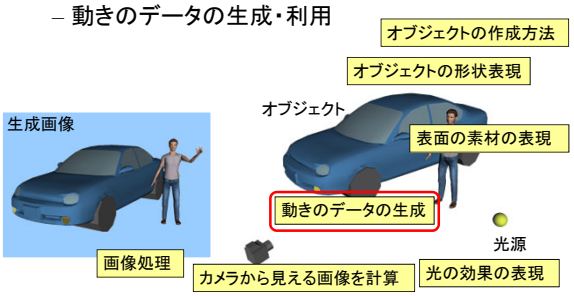
システム創成情報工学科 尾下真樹

2019年度 Q2

0

今回の内容

- アニメーション
  - 動きのデータの生成・利用



1

今回の内容

- 前回の復習
- コンピュータ・アニメーションの原理
- キャラクタ・アニメーション
- 人体以外の物体のアニメーション
- リアルタイム・アニメーション
- 授業のまとめ

2

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）
  - 5章

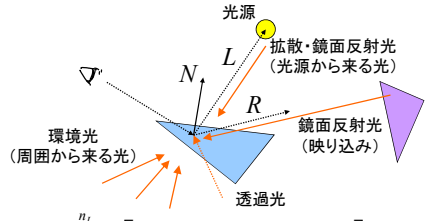
- 「ビジュアル情報処理 －CG・画像処理入門－」
  CG-ARTS協会 編集・出版（2,500円）
  - 5章
- 「3DCGアニメーション」
  栗原恒弥 安生健一 著、技術評論社
  出版（2,980円）

3

前回の復習

4

光のモデル



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

環境光　拡散反射光　鏡面反射光（局所照明）　鏡面反射光 透過光（大域照明）

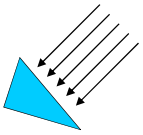
それぞれの光源からの光（局所照明）　大域照明

5

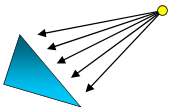
光源の種類と設定方法

- 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

無限遠に光源があると見なせる



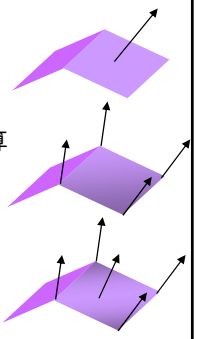
- 点光源
  - (x,y,z)の位置に光源がある
  - 光源位置のw座標を1.0に設定



6

シェーディング

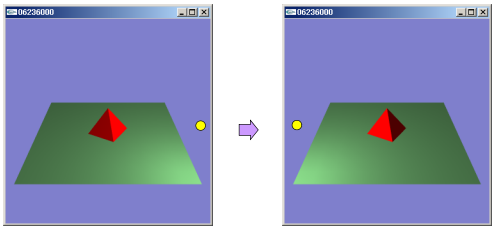
- フラットシェーディング
  - 面の法線から面の色を計算
- グローシェーディング
  - 頂点の法線から頂点の色を計算
  - 頂点の色から、各点の色を決定
- フォンシェーディング
  - 頂点の法線から、面内の各点(ピクセル)の法線を計算
  - 各点の法線から、色を計算



7

プログラミング演習

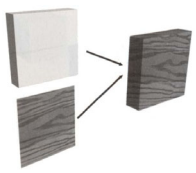
- 光源の種類(点光源・平行光源)や位置の設定を変更し、描画結果を確認



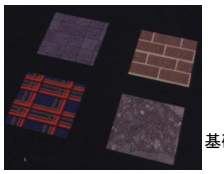
8

テクスチャマッピング

- 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

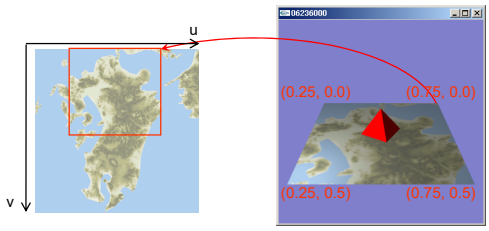


基礎と応用 図5.2

9

プログラミング演習

- 地面にテクスチャマッピングを適用して描画
  - 各頂点に、テクスチャ座標(u,v)を指定



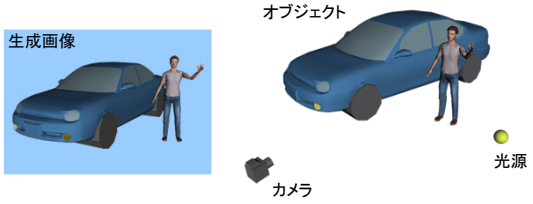
10

コンピュータ・アニメーションの原理

11

3次元グラフィックス（復習）

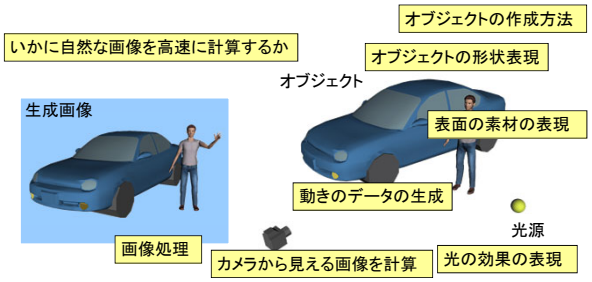
- CG画像を生成するためのしくみ
  - 仮想空間にオブジェクトを配置
  - 仮想的なカメラから見える映像を計算で生成
  - オブジェクトやカメラを動かすことでアニメーション



12

3次元グラフィックスの技術（復習）

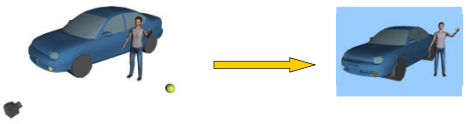
- コンピュータグラフィックスの主な技術



13

コンピュータグラフィックスの分類（復習）

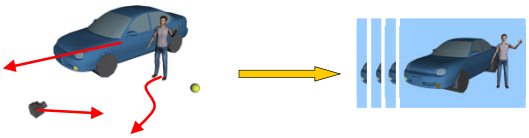
- 2次元グラフィックス
  - 画像データ（2次元）を扱う
    - 画像処理、符号化 などの技術
- 3次元グラフィックス
  - シーンデータ（3次元）→ 画像データ（2次元）
    - 出力データは、あくまで2次元になることに注意



14

アニメーション（復習）

- 2次元アニメーション
  - 少しずつ変化する画像を連続的に生成することで、アニメーションになる
- 3次元アニメーション
  - 動きのデータを与えて、連続画像を生成



15

アニメーションの原理

- 少しずつ変化する画像を連続して表示することでアニメーションとして見える
  - 1秒間に10枚～30枚毎程度（fps: frame per sec）
    - テレビ 30fps（60fps）、映画 24fps、アニメ 12fps、TVゲーム 30 or 60 fps
  - 3次元アニメーションは、少しずつ物体の位置・向きを変えながら、連続して描画することで実現

16

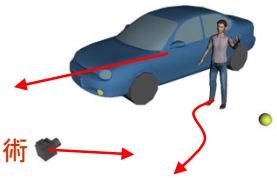
2D・3Dアニメーションの比較

- 2次元のアニメーション（セルアニメ）
  - モデリングに比べると絵を描くのは簡単
    - カメラから見える絵だけを描けばいい
  - 1枚1枚描く必要があるのでアニメーションは大変
  - 最近はこちらもほとんどコンピュータ上で処理
    - コンピュータ上で、原画の自動補間や彩色作業などを行う
- 3次元のアニメーション（CGアニメ）
  - モデリングに非常に多くの時間がかかる
    - きちんとした3次元のデータを作りこむ必要がある
  - 一通りモデリングが完成すれば、動きを与えるだけで、比較的簡単にアニメーションが作れる

17

アニメーションの生成

- アニメーション ＝ 動きの生成 ＋ 画像生成
  - 3次元空間中のオブジェクトに動きのデータを与える
  - 動きのデータをどう生成するか？
- 動きのデータを扱う技術
  - 単純な物体の動き
  - 人間（多関節体）の動き
  - 髪や衣服などのやわらかい物体、炎・流体など



18

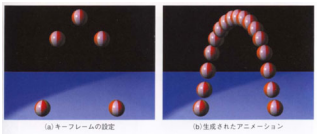
物体の動きのデータ

- 画像を描画するためには、各フレームにおける物体の位置・向きの情報が必要
  - 全てのフレームの情報をいちいち指定すると非常に時間がかかる
- キーフレームアニメーションが主に使われる
  - いくつかの代表的なフレームにおける、オブジェクトの位置・向きを指定
  - 指定されていないフレームは、前後のフレームの情報から位置・向きを決定

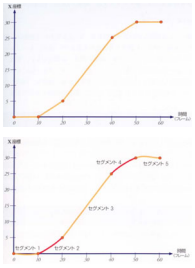
19

キーフレームアニメーション

- 動きのキーとなる（時刻,状態）の組のデータ列から、動きを生成
  - キーフレームの補間の方法には多くの種類がある
    - 線形補間
    - ベジェ曲線、スプライン曲線



参考書 図3.3



参考書 図3.9

20

物体の動力学シミュレーション

- 物理法則に従ったアニメーション
  - 実際の物体は物理法則に従う
    - 重力、衝突、摩擦力、力を加えると運動する、落下、等
  - キーフレームアニメーションでは、これらの物理法則は考慮されないので不自然に見える可能性がある
- 動力学シミュレーション
  - 初期状態を与えると、各フレームごとに運動方程式を数値的に解いていくことで、アニメーションが生成される
  - 望むような結果を得るような初期条件の設定が難しいという問題がある
  - キーフレームアニメーションとの使い分けが必要

21

キャラクタ・アニメーション

22

キャラクタ・アニメーション

- CGにより表現された人体モデル（キャラクタ）のアニメーションを実現するための技術
- キャラクタ・アニメーションの用途
  - オフライン・アニメーション（映画など）
  - オンライン・アニメーション（ゲームなど）
    - どちらの用途でも使われる基本的な技術は同じ（データ量や詳細度が異なる）
    - 後者の用途では、インタラクティブな動作を実現するための工夫が必要になる
- 人体モデル・動作データの処理技術

23

キャラクタ・アニメーション

- 人体モデルの表現
- 動作データの作成
- 動作データの変形
- 動力学を考慮した動作の生成・変形

24

人体モデルの表現

25

人間の各要素の表現

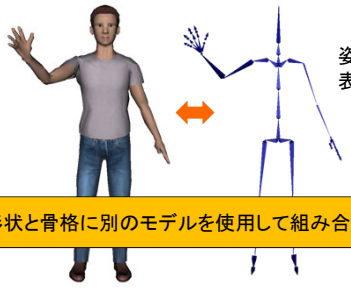
- 人体の表現
  - 身体の表現
    - 全身の骨格・形状の表現
  - 顔の表現
    - 細かい表情変化を表現するためには体とは別のモデルが必要
- 付属物の表現
  - 髪の毛や衣服など
    - シミュレーションによる動きの計算

26

人体モデルの表現

形状モデル（ポリゴンモデル）　骨格モデル（多関節体）

描画用



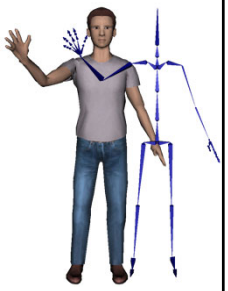
姿勢・動作の表現・処理用

形状と骨格に別のモデルを使用して組み合わせ

27

骨格モデルの表現

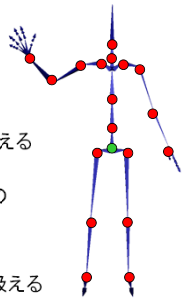
- 人間を多関節体として扱う
  - 人間の骨格をモデル化するためには、40～200 程度の自由度が必要になる
    - 基本的な関節だけで 40程度
    - 手の指や足の指なども入れると 200自由度くらい必要
  - 全関節の角度により人間の姿勢を表せる
  - 人体の形状モデルは、骨格の動きに応じて変形



28

骨格モデルの表現

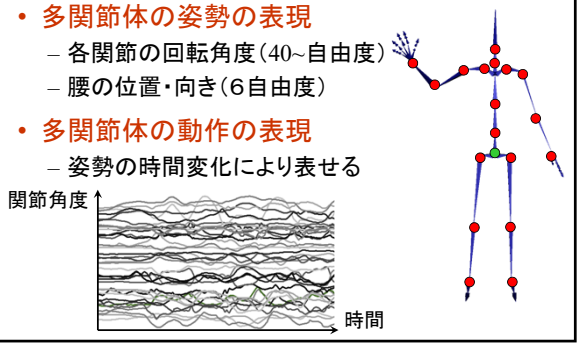
- 多関節体モデルによる表現
  - 複数の体節（部位）が関節で接続されたモデル
  - 体節
    - 多関節体の各部位、剛体として扱える
    - 複数の関節が接続されており、体節の長さや体節内での各関節の接続位置は固定
  - 関節
    - 2つの体節の間を接続、点として扱える
    - 関節の回転により姿勢が変化する



29

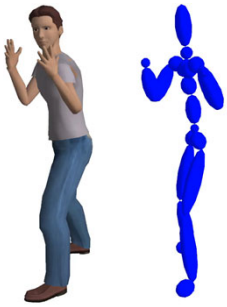
姿勢・動作データの表現

- 多関節体の姿勢の表現
  - 各関節の回転角度（40~自由度）
  - 腰の位置・向き（6自由度）
- 多関節体の動作の表現
  - 姿勢の時間変化により表せる



30

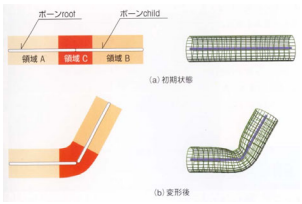
動作データの例

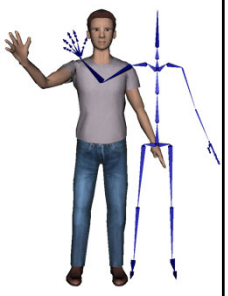


31

形状変形モデル（ワンスキンモデル）

- 人間の形状を全身で1つのポリゴンモデルとして作成
- 骨格モデルの変形に応じてポリゴンモデルの各頂点を移動



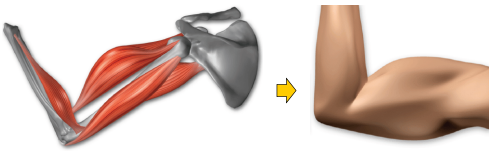


「3DCGアニメーション」図4.16

32

より高度な形状変形モデル（1）

- 筋肉モデルにもとづく変形
  - 人間の筋肉をモデリング
  - 骨格の動きに応じて筋肉を伸縮
  - 筋肉の動きに応じて皮膚を変形
    - ※ ワンスキンモデルよりもリアルな変形を実現できる

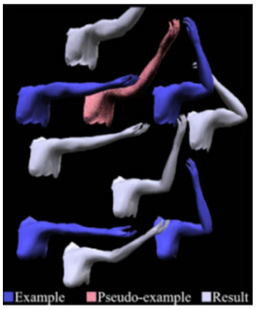


［Softimage XSI］

33

より高度な形状変形モデル（2）

- 形状補間
  - あらかじめ入力された複数のサンプル形状を適切な重みでブレンドすることで、変形を実現
  - 姿勢に応じて、適切なブレンドの重みを計算する必要がある



［Sloan 01］

34

顔のモデリングと表情生成

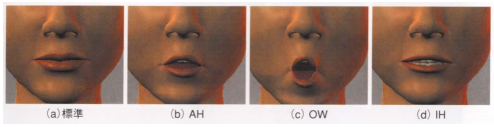
- 顔のモデリングと表情生成
  - 顔の表情は人間の感情を伝える上で非常に重要
  - 顔のモデリングや変形の手法が研究されている
    - テクスチャによる表情の表現
    - 筋肉ベースの顔の形状の変形
    - サンプル形状のブレンドによる変形

参考書 図4.18 [Pighin 98]

35

リップシンク

- セリフに合わせて唇の動きを生成する技術
  - あらかじめ、各音素ごとの口の形のデータを用意しておく
  - セリフを音素に分解し、各音素に応じて口の形を変形してゆく



参考書 図4.20

36

動作データの作成

37

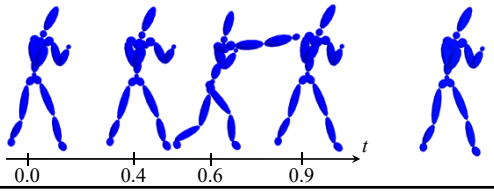
動作データの作成

- 動作データの主な作成方法
  - キーフレームアニメーション
  - モーションキャプチャ
  - （動力学シミュレーションによる方法）
    - 3番目の方法は、やや特殊な方法なので、後述

38

キーフレームアニメーション

- 動作のキーとなる姿勢を手作業で作成
  - キー姿勢の時刻・姿勢（キーフレーム）を作成
    - 各関節の回転を操作、各部位の位置を操作
- キー姿勢の間を自動的に補間して動作生成



39

キーフレームの姿勢の作成

- キーフレームにおけるキー姿勢の作成
  - 基本的にはアニメータが手作業で作成
  - かなりの時間・労力がかかる
- 基本的な姿勢の作成方法
  - 1つの関節の回転を操作
  - 1つの部位（関節・体節）の位置を操作
    - 部位の位置に合わせて関節の回転を自動計算（後述する逆運動学を使用）

40

モーションキャプチャ

- 人間の身体にセンサをつけて、人間の動作を計測・取得する方法
- モーションキャプチャ機器の方式
  - 光学式、慣性式、磁気式、RGB-Dカメラ式、等
  - 計測に使用するセンサーの種類の違いにより、さまざまな方式がある
    - センサーから得た計測データを、動作データに変換する処理が必要となる
    - 計測の時間間隔は、センサーの種類によって異なる
    - センサーの種類によっては、骨格モデルも推定可能

41

モーションキャプチャ機器の種類（1）

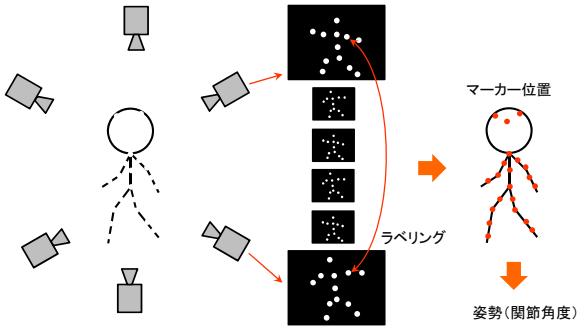
- 光学式
  - 現在、主に使われている方式
  - 体の各部位にマーカを付けてカメラで撮影
  - 複数のカメラから姿勢を計算
  - 姿勢の推定に時間がかかる
- 自発光学式
  - 光学式の拡張版
  - 各マーカが順番に発光
  - ラベリング処理が容易になる


Optitrack © NaturalPoint

Visualeyez © Phoenix Tech.

42

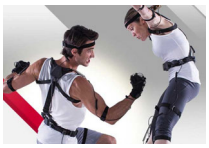
光学式モーションキャプチャ



43

モーションキャプチャ機器の種類（2）

- 慣性式
  - 加速度・ジャイロ・地磁気センサーの組み合わせにより、各部位の向きを計算
  - 小型・安価に実現可能

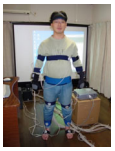

Perception Neuron © Noitom

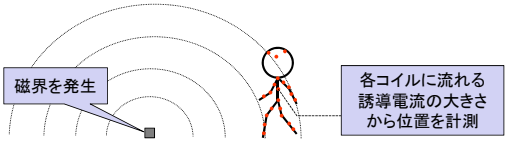

Xsens MVN © Xsens

44

モーションキャプチャ機器の種類（3）

- 磁気式
  - 磁界を発生させて、体の各部位につけたセンサの位置・向きを計算
  - 高速・高精度
  - ケーブルの拘束のため動きづらい

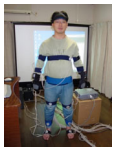

trackSTAR © Ascension



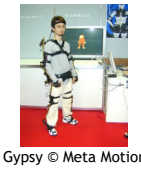
45

モーションキャプチャ機器の種類（3）

- 磁気式
  - 磁界を発生させて、体の各部位につけたセンサの位置・向きを計算
  - 高速・高精度
  - ケーブルの拘束のため動きづらい


trackSTAR © Ascension

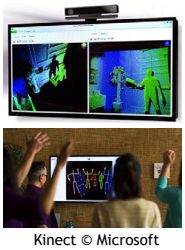
- 機械式
  - 体の各関節に回転角度を計測するための計測器を装着
  - 計測器の重さのため動きづらい


Gypsy © Meta Motion

46

モーションキャプチャ機器の種類（4）

- RGB-Dカメラ方式
  - 各ピクセルの奥行を計測
  - 人体部位・姿勢の推定
    - 完全な姿勢は求められない
  - 他の方式よりも精度は低い
  - Kinect が代表的
    - 事前に学習したモデルを使って、デプス画像中の各部位の位置を推定


Kinect © Microsoft

Shotten et al., “Real-Time Human Pose Recognition in Parts from a Single Depth Image”, IEEE CVPR 2011.

47

モーションキャプチャ機器の比較

方式 （製品例）	光学式 （Optitrack）	慣性式 （Perception Neuron）	RGB-Dカメラ方式 （Kinect）
精度	高	中	低
スーツ着用	要	要	不要
専用スペース	必要	不要	不要
動作（姿勢）取得	○	○	△
骨格モデル推定	○	×	△
指の姿勢取得	△※1	○	△※2
顔の表情取得	△※1	×	△※2
最大秒間フレーム数	120	60	30
価格	数百～数千万円	数十～一千万円	数万円

※1 全身の姿勢とは別にキャプチャを行う必要がある
※2 限定された状態の識別しかできない

48

動作データの作成方法の比較

- モーションキャプチャ
  - 人間の動きをそのまま取り込めるので、一見手軽そう
  - 実際には、ノイズや、俳優とキャラクタの骨格の違いなどのため、かなり編集が必要になる
  - そのままではキーフレームがないので編集がしづらい
- キーフレーム編集
  - ゼロから作成しなければならないため大変
  - キーフレームが設定されているので、修正はしやすい
  - 人間らしい細部の動きや自然さを実現するのは難しい
- 両者の使い分けや組み合わせが必要

49

動作データの変形

50

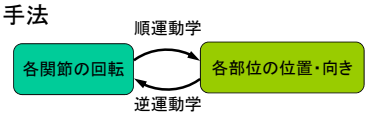
動作データの変形処理

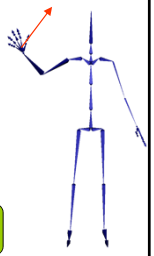
- 既存の動作データを編集したり、動作データをもとに新しい動作を生成するための技術
  - インバース・キネマティックス（逆運動学）
  - 動作補間、動作遷移・接続
  - タイム・ワーピング
  - モーション・ワーピング
  - モーション・リターゲッティング
- これらの技術を利用することで、毎回新しい動作データを作成する手間を省ける

51

運動学

- 運動学（キネマティックス）
  - 多関節体の姿勢表現の基礎となる考え方
  - 人間の姿勢は、全関節の関節回転により表現できる
  - 各関節の回転と、各部位の位置・向きの間の関係を計算するための手法





52

順運動学と逆運動学

- 順運動学（フォワード・キネマティックス）
  - 多関節体の関節回転から、各部位の位置・向きを計算
  - 回転・移動の変換行列の積により計算
- 逆運動学（インバース・キネマティックス）
  - 指定部位の位置・向きから、多関節体の関節回転を計算
    - 手先などの移動・回転量が与えられた時、それを実現するための関節回転の変化を計算
  - 動きを指定する時、関節回転よりも、手先の位置・向きなどを使った方がやりやすい
  - ロボットアームの軌道計画等にも用いられる

53

54

逆運動学計算における問題

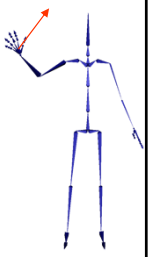
- 逆運動学での制約条件に関する問題
  - アンダー・コンストレインツ（少な過ぎる制約）
    - ある部位の位置・向きを指定しただけでは姿勢が一意に決まらない
      - 例：手の位置がきまっても、その手の位置を実現するような全身の関節角度の組は無数に存在する
    - 何らかの評価基準を入れて適当な解を決定する
      - 例：前ステップの姿勢との差がなるべく小さくなる解を選択
  - オーバー・コンストレインツ（多過ぎる制約）
    - 逆に制約が多すぎて、与えられた制約を満足する姿勢がない

55

IKの計算方法

- いくつかの計算方法がある
  - 数値的解法
    - 擬似逆行列を使った計算方法
      - 各関節の微少回転に応じた指定部位の移動・回転量の関係をもとに、条件を満たすように関節を少しずつ回転
    - 非線形計算による計算方法
      - 最適化問題と考えて、指定部位の位置・向きの条件を満たすような関節角度を探索
  - 解析的解法
    - 人間の骨格に注目した効率的な計算方法

※ 詳しい説明は省略（「ロボット工学」等の授業で習うかも）

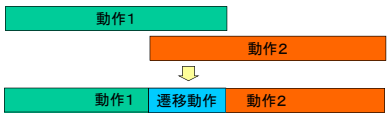


56

57

動作接続・遷移

- 動作接続・遷移・合成（トランジション、ブレンド）
  - 2つの既存の動作をつなげて新しい動作を生成
  - 前後の動作の間を滑らかにつなぐ手法
    - 時間に応じた比率で前後の動作の姿勢を混合
    - 前の動作の終了部分と、次の動作の開始部分が、同じような動作である場合に適用可能

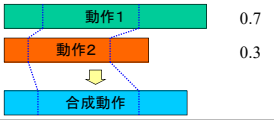


58

59

動作補間

- 動作補間（モーション・インターポレーション）
  - 複数の動作をブレンドして新しい動作を生成
    - 例：ゆっくり走る動作と早く走る動作から、中くらいの速度で走る動作を生成
    - 任意の比率で混ぜ合わせることができる
  - 事前に複数の動作を同期しておく必要がある
    - 同じタイミングにキー時刻を設定しておく

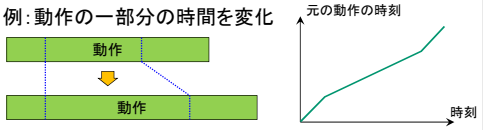


60

61

タイム・ワーピング

- 動作データの再生時間を変化させることで、動作の速度を変更する
  - タイムワープ関数により時間変化を指定
    - 例：動作の一部分の時間を変化

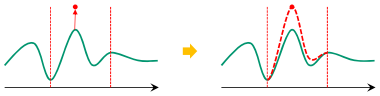


  - 再生時間を大幅に変化させたり、再生速度が急激に変化したりすると不自然な動作になるので、タイムワープ関数を滑らかにするなど工夫が必要

62

モーション・ワーピング

- 動作データ中のキーフレームの姿勢変化に合わせて前後の動作を滑らかに変化させる
  - キーフレームの時刻、変形後の姿勢、変形を適用する区間の開始・終了時刻を指定
  - 区間中の各時刻の姿勢に、姿勢ブレンドを適用
    - 時刻に応じてブレンド比率を変化（ブレンド比率関数）

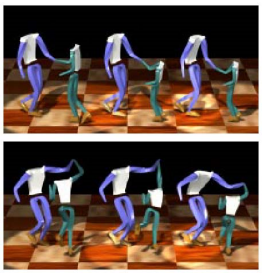


  - 姿勢変化が大きいと、不自然になる可能性がある
    - 変形区間やブレンド比率関数の設定が重要

63

モーション・リターゲッティング

- 骨格の異なるキャラクタ間で動作データを変換
  - 例：モーションキャプチャデータを、元の俳優の骨格から、実際のキャラクタの骨格用に修正
  - 単純に関節角度をコピーするだけでは、足の位置がずれたりするため、修正する



[Gleicher 98]

64

動力学を考慮した動作の生成・変形

65

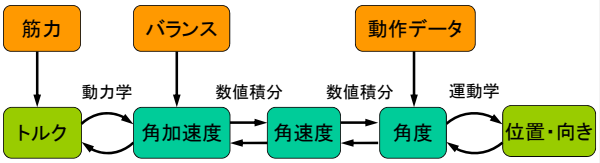
動力学を考慮したアニメーション

- 運動学（キネマティックス）
  - 多関節体の機構を扱うための手法（関節回転と各部位の位置・向きの関係を扱う）
  - 現在は、こちらが主に使われている
  - 力学的に正しい動きが生成されることは保証されない
- 動力学（ダイナミックス）
  - 運動学に加えて、力学的な要素を考慮する手法

66

動力学を考慮したアニメーション

- 主に2種類の方法がある
  - 動力学シミュレーションによるアプローチ
  - 動作最適化によるアプローチ
- 多関節体の動力学



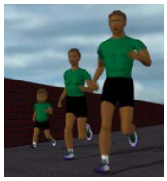
67

動力学シミュレーション

- 動力学シミュレーション
  - 本物のロボットと同様に、制御理論に基づいて関節のトルクを制御するコントローラを構築
  - 動力学シミュレーションを行って動きを生成
    - 人間らしい動きを生成することは困難
    - 結果をうまく制御することが難しい

$$\tau_i = K\left(\theta_{i,goal} - \theta_{i,curr}\right) - K\left(\dot{\theta}_{i,goal} - \dot{\theta}_{i,curr}\right)$$

PD制御
目標角度－現在角度に適当な係数をかけることで各関節のトルクを決定

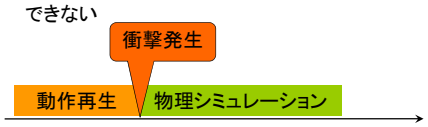


[Hodgins 97]

68

動力学シミュレーションの応用例

- ラグドール・シミュレーション
  - 一部のコンピュータゲームで使われている
  - 通常は、動作データを再生
  - 衝突時に、物理シミュレーションに切り替え
    - 受動的な動作のみで、能動的な動作は実現できない
    - 倒れずに途中で回復して元に戻るような動作は実現できない



69

動作最適化

- 動作最適化
  - 関節トルクやバランスの不自然さを評価する関数 f（動作）を設計
  - f（動作）がなるべく小さくなるように動作データを少しずつ修正する
    - 計算に非常に時間がかかる
    - 人間のような多関節体には、自由度が高すぎてそのまま適用するのは困難

[Popovic 98]

70

人体以外の物体のアニメーション

71

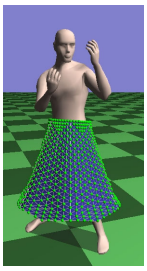
髪・衣服のアニメーション

- 髪や衣服などのキャラクタの付属物
  - 自然なキャラクタアニメーションには非常に重要
  - これらの柔らかい物体の動きを人手で設定するのは困難
  - 動力学シミュレーションを使って計算する技術が開発されてきた
  - コンピュータゲームなどでは、全身の形状と同様に扱い、骨格に応じて単純に変形することが多い

72

衣服のシミュレーション

- 衣服の動力学シミュレーション
  - 映画などで実用的に使われている
- 粒子モデルによる計算手法が主に使われている
  - 衣服を粒子モデルにより表現
  - 粒子に働く弾性力、屈折力、重力、摩擦力、空気抵抗などをモデル化
  - 物理法則に従って粒子の運動を計算 → 衣服の変形を計算



73

髪の表現・シミュレーション

- ポリゴンによる大まかな表現方法
- 一本一本を表現する方法
  - 代表的な髪を与えて、髪形を作成する方法
  - 流れ場を与えて、髪形を生成する方法

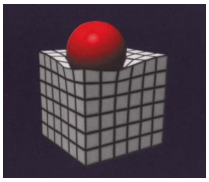
[Koh and Huang, CAS 2001]

[MIRA Lab.]

74

その他の柔らかい物体の表現

- 有限要素法
  - 物体を細かいメッシュに区切る
  - それぞれのメッシュのエネルギーを数式で定義
  - エネルギーが安定するように変形を計算

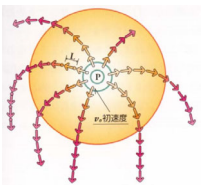


参考書 図6.9
[Terzopoulos 87]

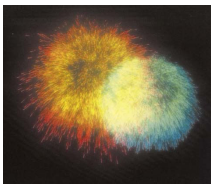
75

パーティクル

- 細かい粒子の運動によって、炎などの自然物を表現する技術
  - パーティクルの運動、生成・消滅条件を定義
  - 自動的にアニメーションが生成される



参考書 図7.2



76

リアルタイム・アニメーション

77

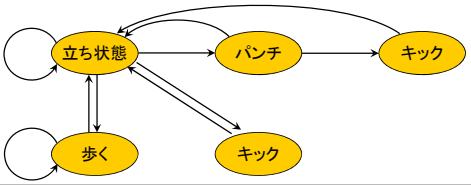
リアルタイム・アニメーション

- コンピュータゲームが代表例
  - 映画などの動画製作とは異なり、動きのデータを動的に生成する必要がある
- 動的な動きの生成方法
  - 単純な物体の制御（車・飛行機など）
    - キー入力や自動制御にもとづき制御
    - 物理法則も考慮
  - キャラクター（多関節体）の制御
    - 動作グラフによるアニメーション
      - 必要な動作データをあらかじめ作成して用意
      - 入力などに応じて適切な動作データを順番に再生
    - 現在は物理法則はほとんど考慮されていない（不自然）

78

動作状態機械

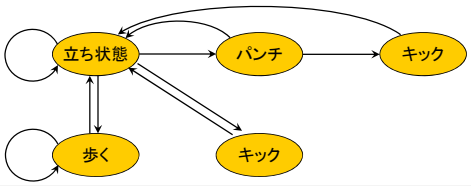
- あらかじめ各動作データ間がつながるように考慮して動作データとそのつながりを作成
  - 前動作の終了姿勢と次動作の開始姿勢が同じになるように動作を作成する
  - 各動作間の遷移のタイミングや方法を設定



79

動作状態機械

- 状態機械（State Machine）、動作ツリー、動作グラフなどの呼び方がある
  - データ構造としては、ノードが短い動作、エッジが前後の動作の接続を表す有向グラフとなる



80

リアルタイムアニメーションの問題点

- 現在の主なアプローチ
  - あらかじめ必要な動作データを全て作成しておき、ユーザの操作に応じて動作データを再生
- 問題点
  - 決められた動作の繰り返ししかできない
  - 特に衝突や外力などの力学的な影響に応じた自然な動作が困難
- 状況に応じて動的に動作を作成する技術が必要

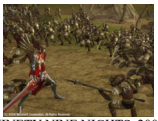
81

群集モデル

- 多数の人間の動きをシミュレートする技術
  - 映画やコンピュータゲームでの群衆シーンで利用
    - 個々のキャラクタの動きを作成するのは不可能
  - 一定のルールに従って各キャラクタを制御
    - 単純なルールでも、キャラクタ同士が相互作用し合って、ある程度リアルな動きが生成される


Lord of the Rings, 2002


Chronicles of NARNIA, 2005


NINETY-NINE NIGHTS, 2005

82

群集モデルの実現方法

- プログラムによる動作ルールの記述
  - if（条件）then（動作）の形式の動作ルールを多数、ゲームなどのプログラム内に直接記述
- GUI環境による高度な動作ルールの記述
  - 群衆アニメーションソフトMassive が有名
  - 多数の映画で使用されている
  - 状態マシン＋ファジィルール による条件を、GUI画面上で記述
  - かなりの知識や労力が必要


Massive

83

まとめ

- 前回の復習
- コンピュータ・アニメーションの原理
- キャラクタ・アニメーション
- 人体以外の物体のアニメーション
- リアルタイム・アニメーション
- 授業のまとめ

84

授業のまとめ

85

本科目の達成目標(シラバスより)

- コンピュータグラフィックスの歴史や、どのような応用に利用されているかといった背景知識を理解させる。
- コンピュータグラフィックスを扱う上で必要となる、レンダリング技術や変換行列などの、基礎的な知識を理解させる。
- 3次元グラフィックスを用いた簡単なプログラムを開発できる技術を習得させる。

86

本科目の位置づけ

- 情報の選択必修科目
- 学科の学習・教育目標(B) に対応
  - コンピュータ応用とシステム理論を学び、時代の要請に呼応した新たな情報システムを創造し、開発を行うための基礎能力を身に付ける

87

成績評価

- 期末試験(40点)
- 期末レポート(40点)
- 毎回の授業の演習問題・演習課題(20点)
  - 授業中の演習問題、プログラミング演習課題
- 出席
  - 成績には考慮しない、一定回数の欠席で不合格

※ 上記の説明以上の具体的な評価方法や配点、個別の成績評価状況に関する質問には、回答しない

88

期末試験・レポート

- 期末試験
  - 8月7日(水) 3限目 2101講義室
    - 正確な時間・会場は、掲示を確認すること
  - 試験範囲は、講義で扱った内容全て(演習も)
    - 記述式の問題にもきちんと回答できるように勉強すること(例:変換行列、プログラム、等)
- 期末レポート
  - 8月26日(月)17:00(厳守)
    - 締め切り(+30分後)以降の提出は一切認めない
  - 端末室の利用可能期間・時間に注意すること

89

レポート課題

- レポート課題
  1. ポリゴンモデルの描画
  2. 視点操作インターフェースの拡張
  3. アニメーション
  4. テクスチャマッピング
  - 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こと

90

再試験・再レポート

- 再試験・再レポート提出は一切行わない
- 最初から試験・レポートに全力を尽くすこと
- 特に、試験に自信がない人は、レポートを頑張ること
- 万一、再試験・再レポート等を実施する場合は、Moodleで連絡する

91

講義のまとめ(1)

- 講義
  - 第1回 ガイダンス、グラフィックスのしくみ
  - 第2回 グラフィックスの応用、要素技術
  - 第3回 演習(1): OpenGL&GLUT入門
  - 第4回 モデリング
  - 第5回 レンダリング
  - 第6回 レンダリング・パイプライン
  - 第7回 演習(2): ポリゴンモデルの描画
  - 第8回 座標変換(1)
  - 第9回 演習:座標変換によるアニメーション

92

講義のまとめ(2)

- 講義(続き)
  - 第10回 演習(3):座標変換
  - 第11回 シェーディング
  - 第12回 シェーディング、マッピング
  - 第13回 演習(4):シェーディング、マッピング
  - 第14回 アニメーション
  - 第15回 演習
  - 演習と講義を交互に行なう
  - レポート課題
  - 期末試験

93

卒業研究との関連

- 尾下研究室で卒業研究や大学院研究を行う人は、本講義の内容は最低限の基礎となる
  - コンピュータアニメーション技術の研究
    - キャラクタの動き生成、操作インターフェース、物理シミュレーション、アニメーション制作システム
  - 他の研究室でも、一部の内容は関連する
- 研究では、より高度な内容を扱うことになる

94

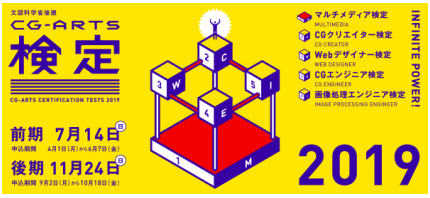
より詳しく勉強したい人へ

- OpenGLプログラミング
  - 多数の参考書がある
- アプリケーション開発
  - ゲームエンジンの利用(Unity、Unrealなど)
    - モデルデータ、モーションデータなども入手可能
  - 携帯端末用ゲーム・アプリ開発(iOS、Android)
- 資格
  - CGエンジニア検定(エキスパート・ベーシック)
    - 授業の参考書が公式テキスト

95

参考：CGエンジニア検定

- CGエンジニア検定（CG-ARTS協会）
  - ベーシック／エキスパート
  - 年2回実施、福岡でも受験可能
  - 本授業の内容を修得できていれば合格可能



96

ゲーム・アニメ制作の勉強方法

- アニメーション制作の独習
  - 市販のソフトウェアが利用可能
    - Maya, Softimage, 3ds max 等
    - 教育目的であれば無償利用可能
- ゲーム開発の独習
  - ゲーム開発環境が利用可能
    - Unity, Unreal 等
    - 自分の開発したゲームを販売することもできる
  - プログラミング言語の習得

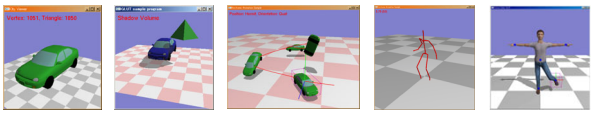
3ds max © Autodesk

Unity © Unity

97

大学院授業

- コンピュータグラフィックス特論Ⅱ
  - コンピュータグラフィックス技術を用いるソフトウェア開発で必要になる実用的な技術を学ぶ
    - 視点操作、幾何形状データの読み込み
    - 影の描画（高度な描画技術）
    - キーフレームアニメーション
    - 物理シミュレーション、衝突判定
    - キャラクタアニメーション



98

授業アンケート

- 授業アンケート（選択式・記述式）（学部共通）
  → 他の科目と一緒に、Webから入力

  スマートフォンからも回答可能

  https://virginia.jimu.kyutech.ac.jp

99

まとめ

- 前回の復習
- アニメーション
  - アニメーションの原理
  - キャラクタ・アニメーション
  - 人体以外の物体のアニメーション
  - リアルタイム・アニメーション
- 授業のまとめ

100