キャラクタの自律動作制御技術

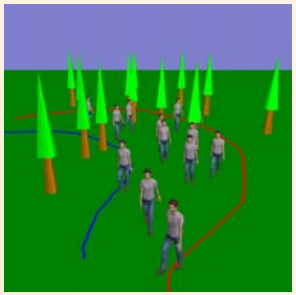
尾下 真樹

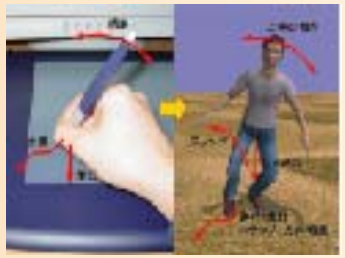
九州工業大学 情報工学部 准教授

尾下研究室の研究テーマ

- コンピュータアニメーション技術を主に研究
  - 特に、人体動作を扱う技術や、リアルタイムにアニメーションを生成するための技術を研究
  - コンピュータゲーム等への応用を目標

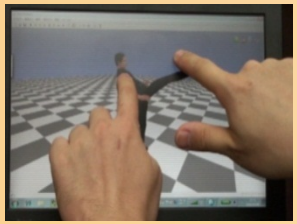


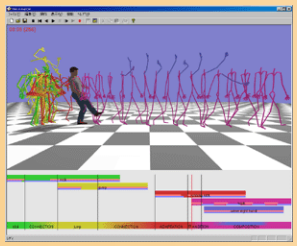

尾下研究室の研究テーマ

- 仮想人間の動作制御



- 動作操作インターフェース

- 衣服･髪･皮膚の
  実時間シミュレーション



- 簡単に使えるアニメーション
  制作システム

本発表の内容

- キャラクタの自律動作制御技術

- 現在の課題と解決案（研究成果）
- 研究動向
- 実装方法についても簡単に紹介
- 研究をゲーム開発に活用するには？

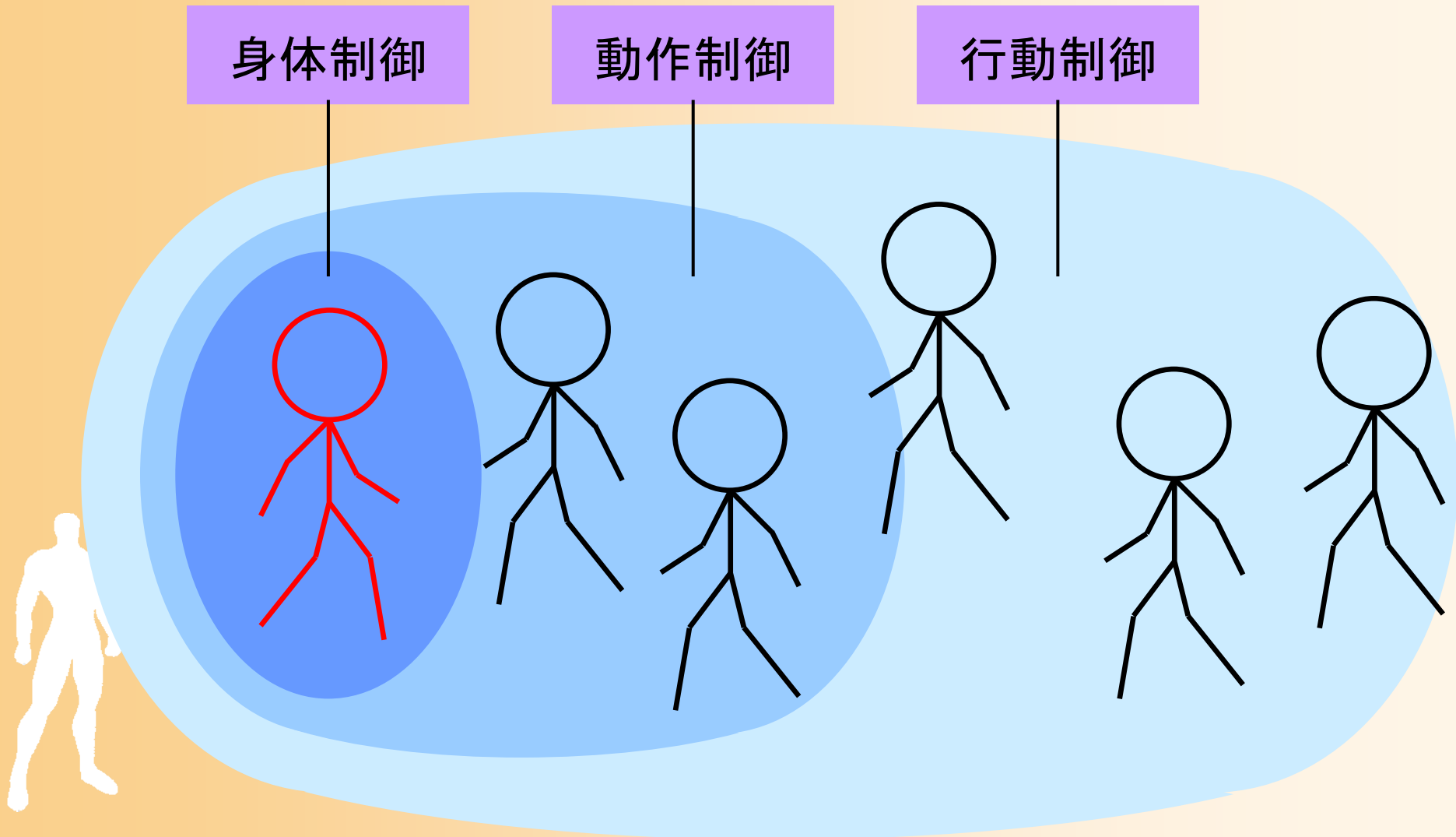
自律動作制御とは？

- ゲームにおける自律動作制御
  - Autonomous
    自律＝自動

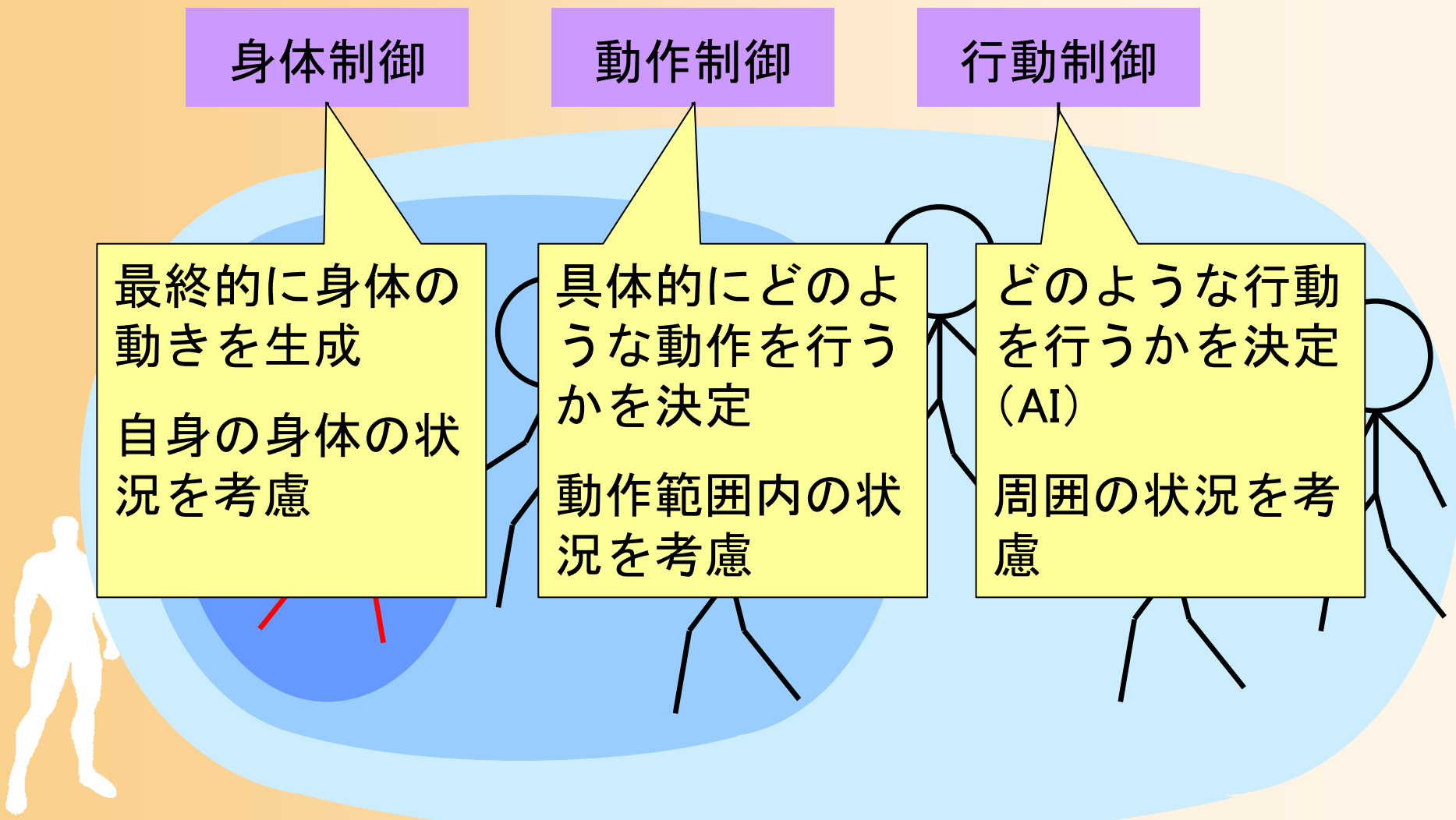
Non-Player Character
完全に自律的に動作を行う必要がある

Player Character
半自律的に動作を行う必要がある
（ex. プレイヤーの抽象的な操作に応じて具体的な動作を実現）

自律動作制御のレベル

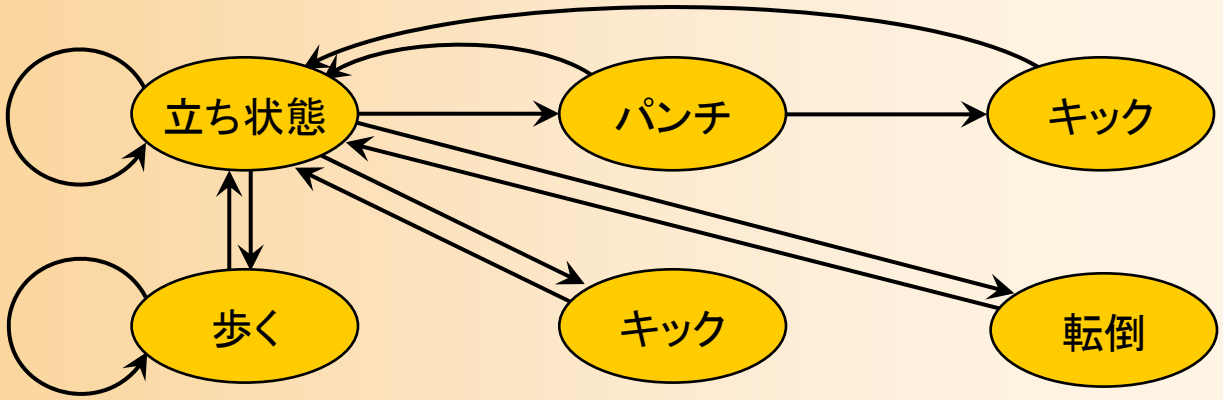


自律動作制御のレベル


身体制御
動作制御
行動制御
最終的に身体の動きを生成
自身の身体の状況を考慮
具体的にどのような動作を行うかを決定
動作範囲内の状況を考慮
どのような行動を行うかを決定（AI）
周囲の状況を考慮

現在のゲームでの制御手法

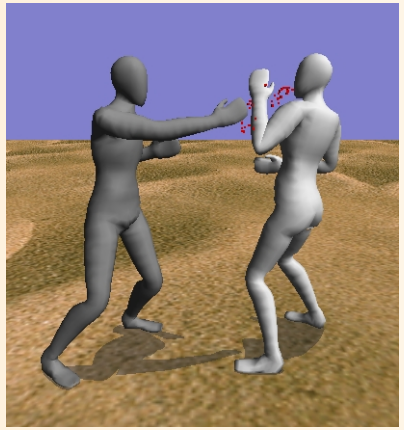
- 動作制御、身体制御
  - 動作ツリー（短い単位動作＋動作間の接続）
  - IK等



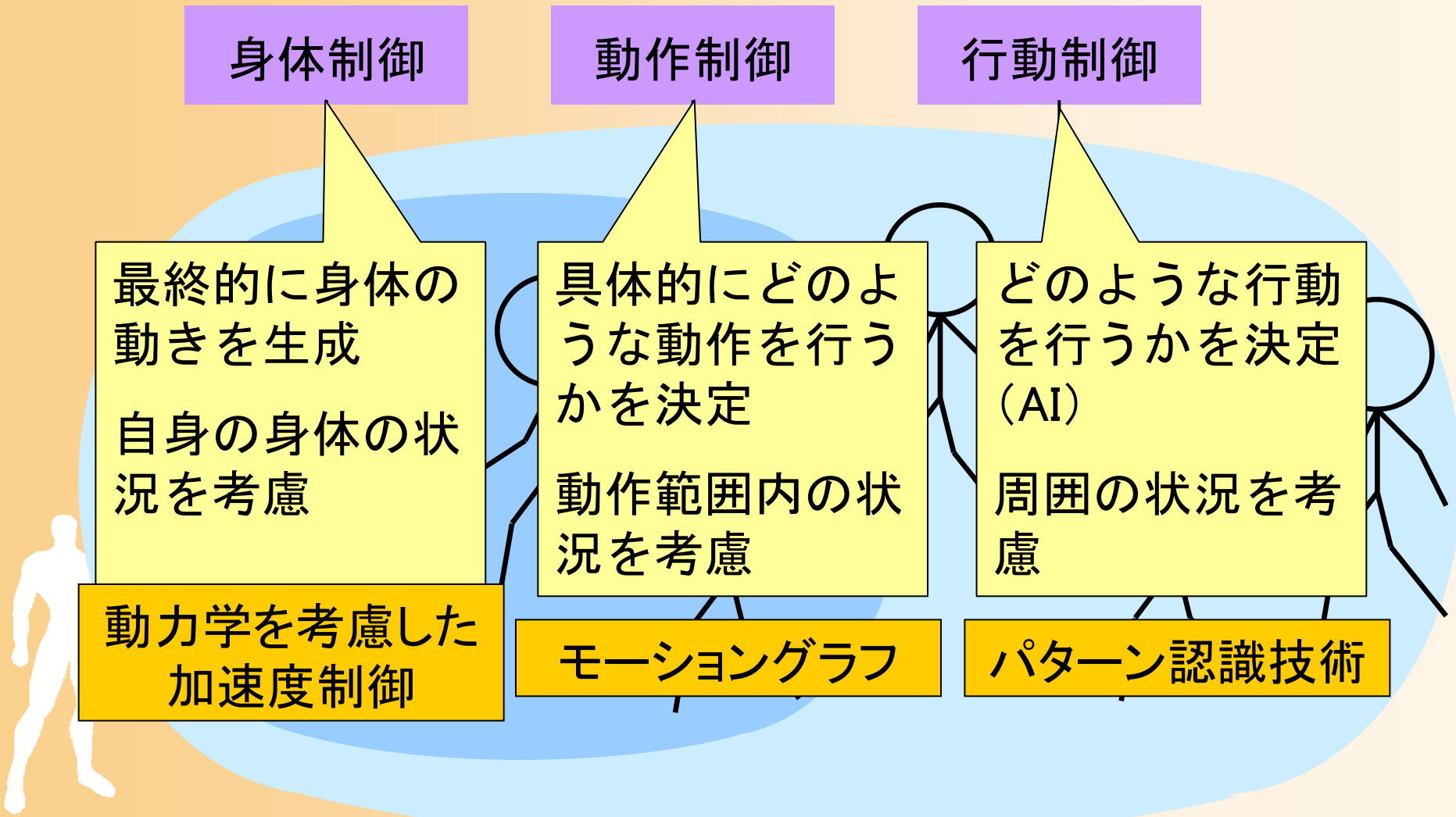
- 行動制御
  - ゲームに応じた行動ルール（AI）をプログラマが設計

現在の技術の問題点

- 一定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
- 自然な行動を行う行動ルールの記述は困難
  - 多くの条件を考慮する必要がある
- 動作ツリーの作成・行動決定ルールの記述には、多くの手間がかかる



問題解決のための技術


身体制御
動作制御
行動制御
最終的に身体の動きを生成
自身の身体の状況を考慮
具体的にどのような動作を行うかを決定
動作範囲内の状況を考慮
どのような行動を行うかを決定（AI）
周囲の状況を考慮
動力学を考慮した加速度制御
モーショングラフ
パターン認識技術

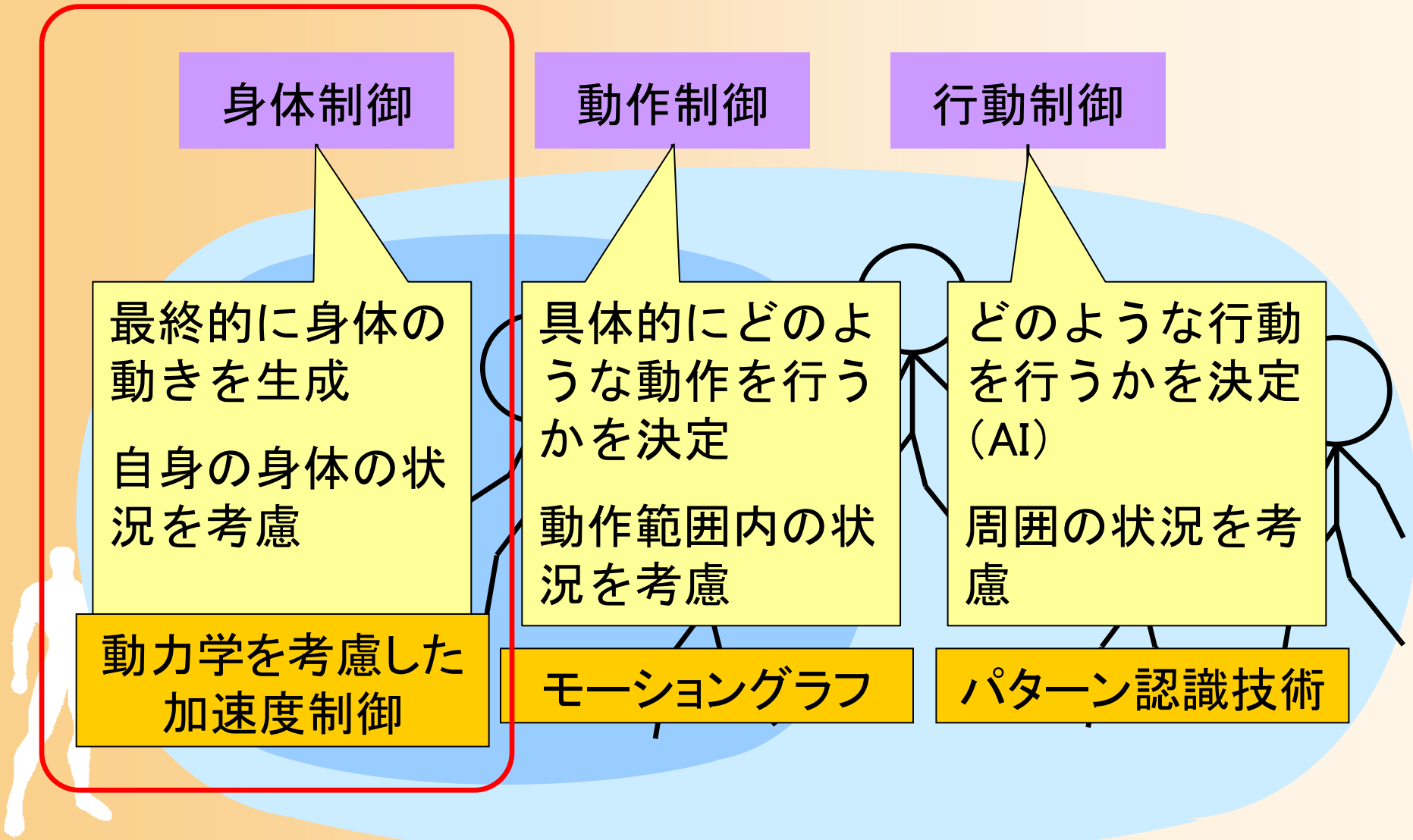
本分野の研究の最終目標

- 現実の人間と同様に動く 仮想人間 を実現
  - 人間と同様に、自律的に動作できる
  - 人間と同様に、動きを指示できる

現実の人間

仮想人間
（キャラクタ）

身体制御


身体制御
動作制御
行動制御
最終的に身体の動きを生成
自身の身体の状況を考慮
具体的にどのような動作を行うかを決定
動作範囲内の状況を考慮
どのような行動を行うかを決定（AI）
周囲の状況を考慮
動力学を考慮した加速度制御
モーショングラフ
パターン認識技術

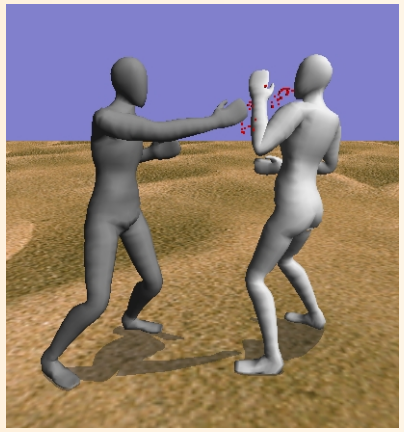
身体制御

動力学を考慮した制御

身体制御

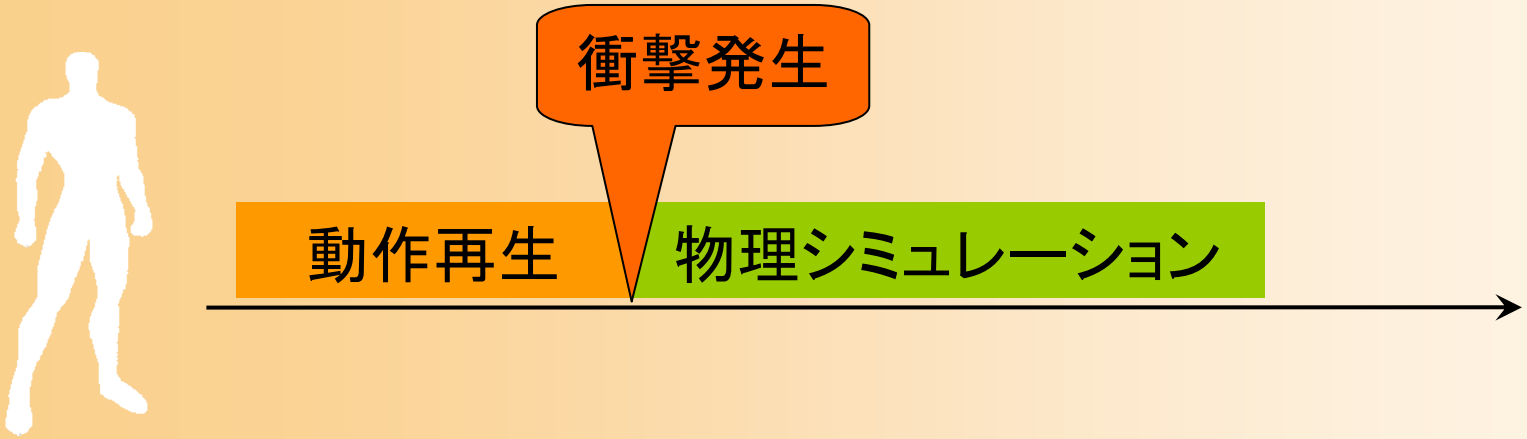
- 動力学を考慮した身体制御

- 加速度制御による動作生成
  - 目標動作を追従
  - バランス・関節負荷を考慮した制御



一般的な手法の問題点

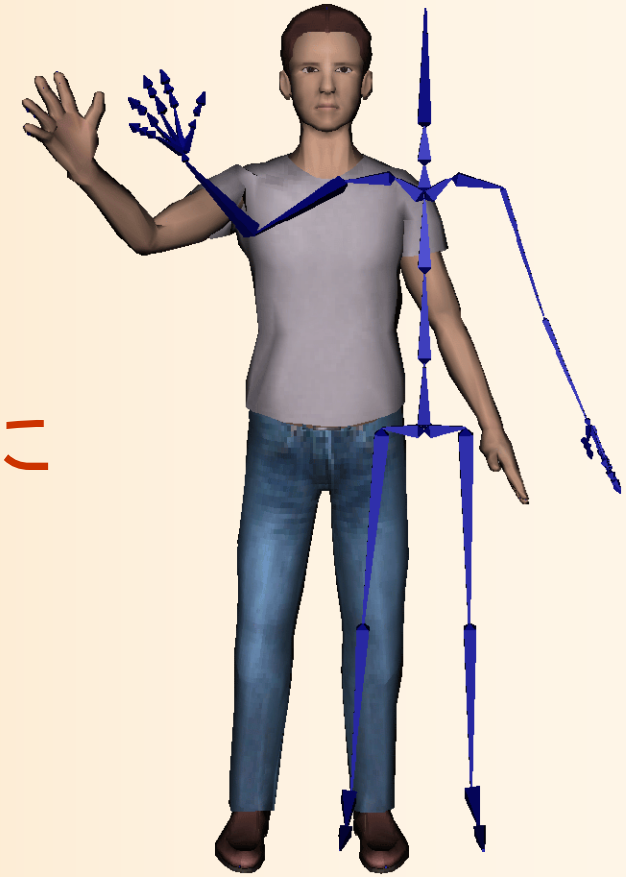
- 物理シミュレーションによる動作生成（ラグドール・シミュレーション）
  - 衝突時に、物理シミュレーションに切り替え
    - 受動的な動作のみで、能動的な動作は実現できない
    - 倒れずに回復するような動作は実現できない



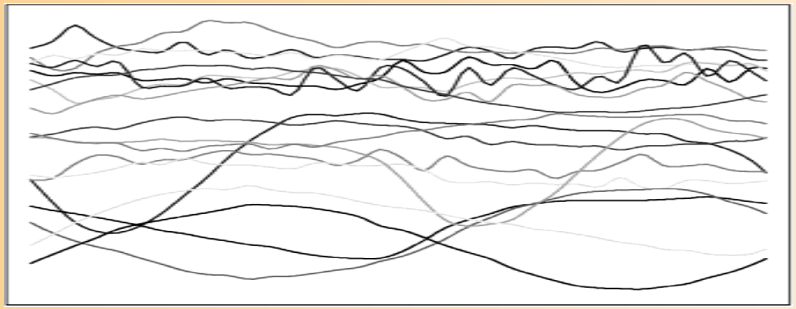
仮想人間の姿勢・動作の表現

- 多関節体の姿勢の表現
  - 腰の位置・向き（6自由度）
  - 各関節の回転角度（n自由度）

$$p_{root}\ q_{root}\quad \theta_0 \sim \theta_{n-1}$$



- 関節角度（姿勢）の時間変化により多関節体の動きを表現



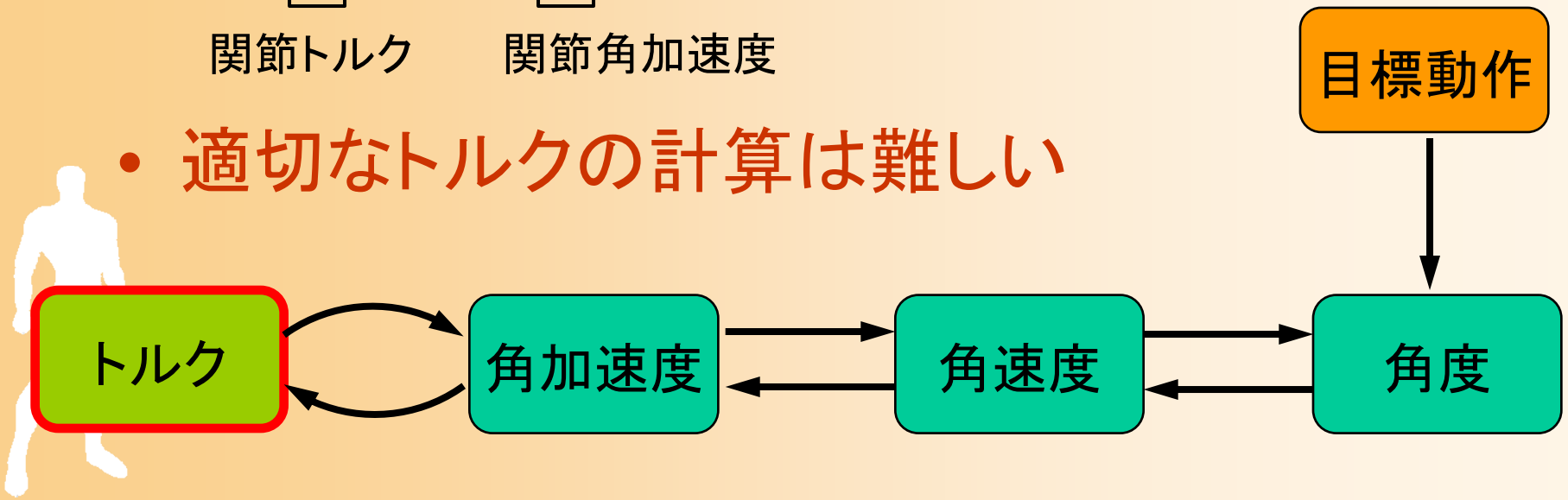
Popovic et al. © 1995

人体モデルの動力学

- 関節に回転力（トルク）が発生することで、運動が実現される
  - 多関節体の運動方程式

$$\underbrace{\tau}_{\text{関節トルク}} = H(\theta)\underbrace{\ddot{\theta}}_{\text{関節角加速度}} + C(\theta,\dot{\theta}) + G(\theta) + K(\theta)F$$

- 適切なトルクの計算は難しい



一般的なコントローラ

- 目標動作を追従するように関節トルクを計算

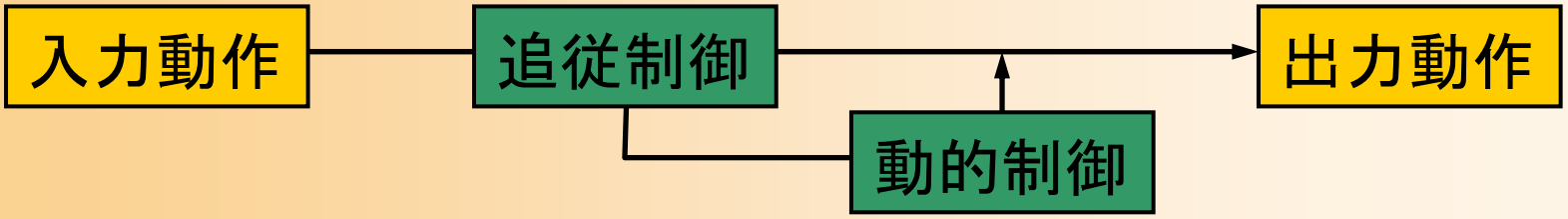
- PD制御（Proportional Derivative Control）
  - 各関節のトルクを独立に制御
    - 単純に目標角度との差に比例したトルクを適用

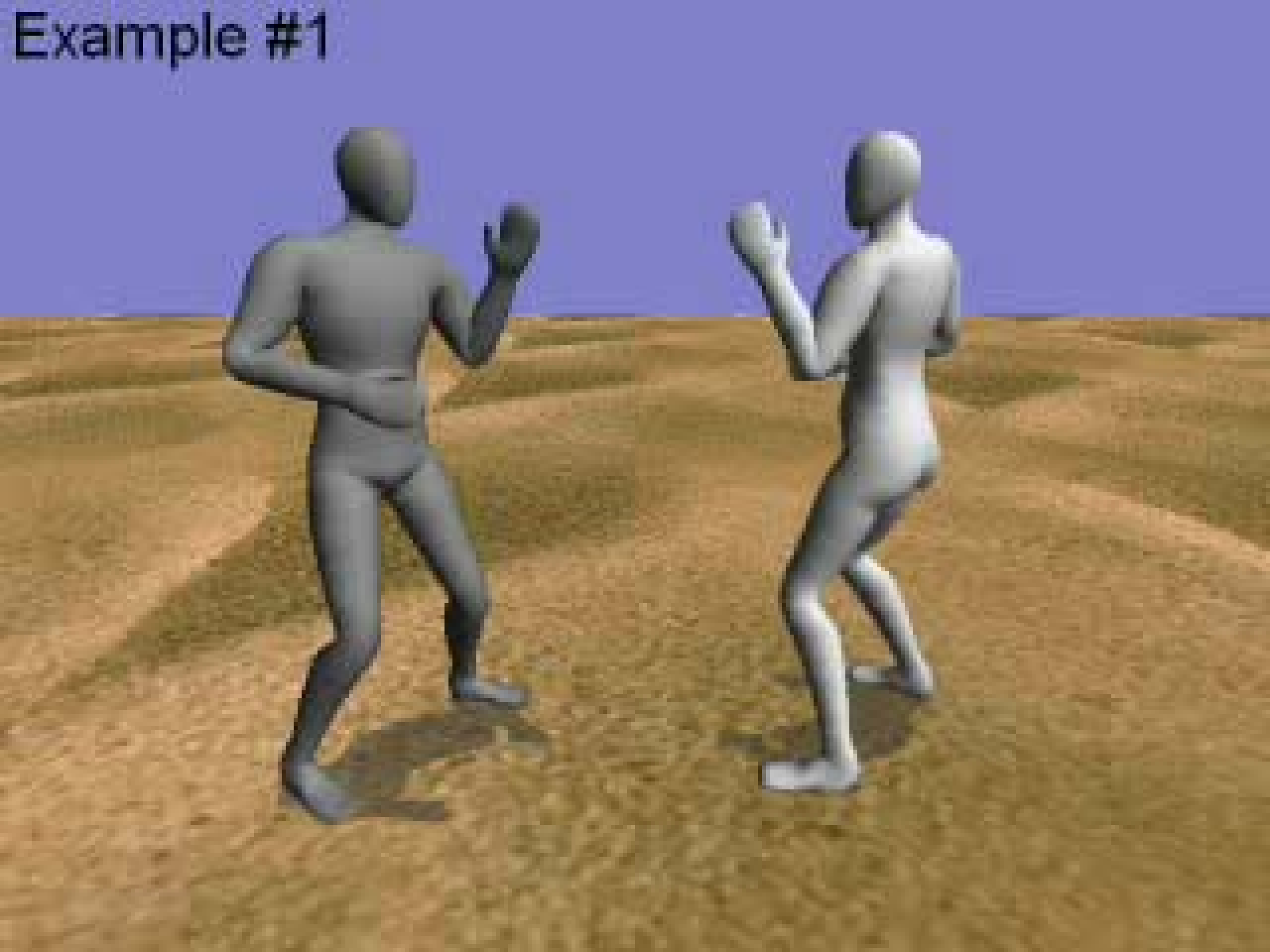
$$\tau = k_g\left(\theta_{target} - \theta\right) - k_d\dot{\theta}$$

- 問題点
  - パラメタの調整が困難（動作ごとに値が異なる）
  - 必ずしも人間らしい動作は生成できない

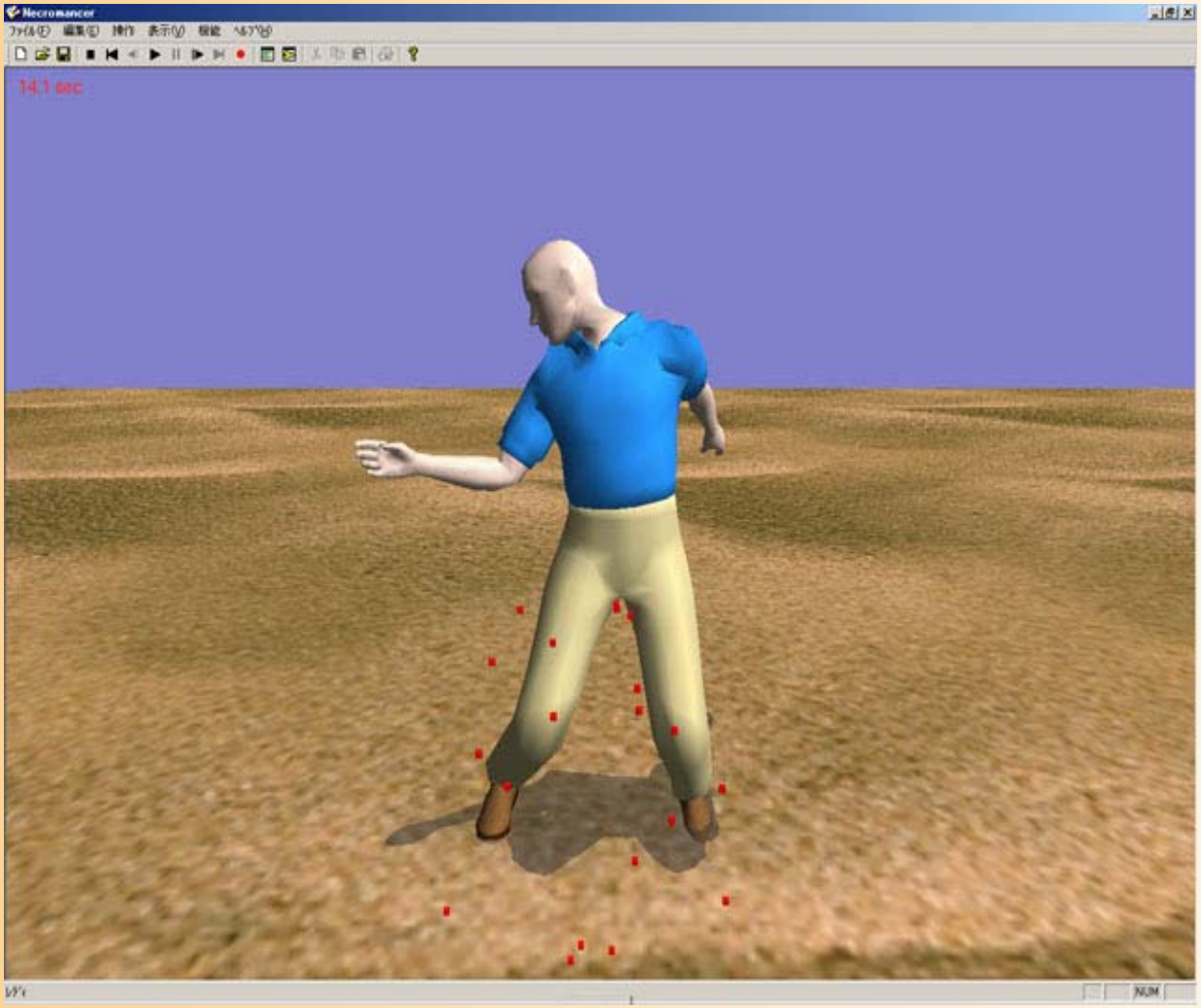
開発手法

- 動作データをもとにしたダイナミック制御
  - 基本的には入力動作を実行するよう追従制御
  - 力学的な影響に応じて動的制御を行う
    - 全身のバランスを保つような制御
    - 関節の負荷を軽減するような制御
  - 角加速度空間で関節の動きを制御することで、このような制御アルゴリズムを実現



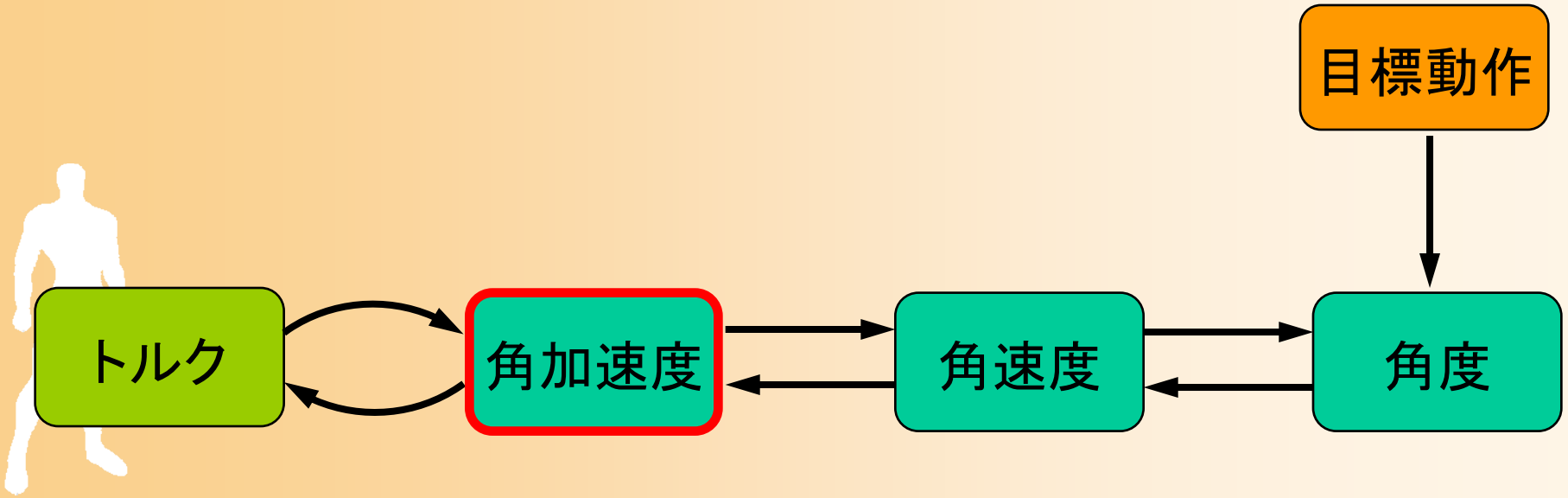

Example #1

プログラムのデモ



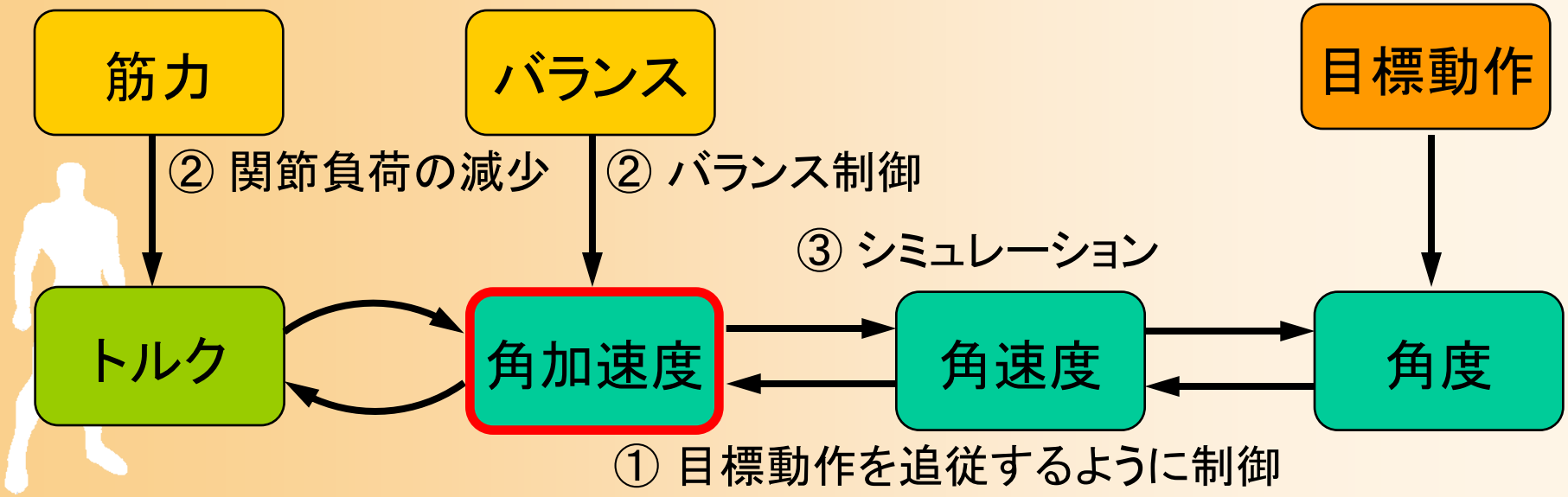
開発手法

- 角加速度空間で各関節を制御
  - トルク制御と比べて安定した制御が可能
  - 物理シミュレーションは不要



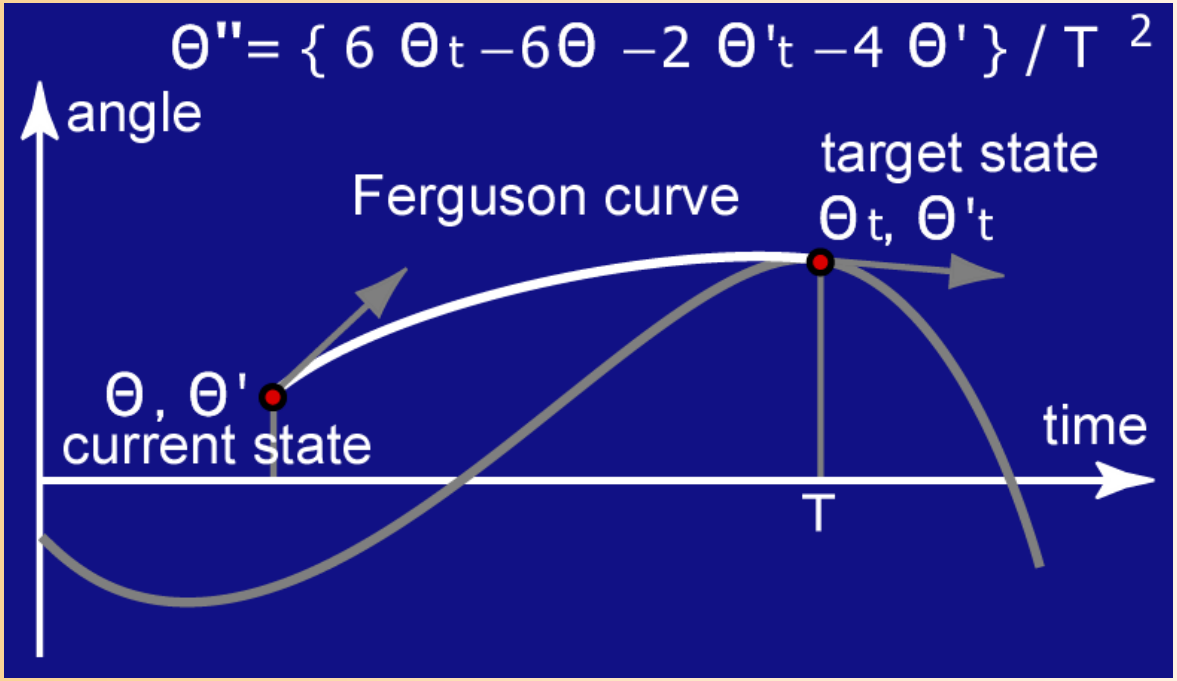
開発手法(続き)

- 基本的には目標動作を追従する
  - 目標動作を追従するような加速度を計算
- 関節負荷・バランスを考慮して加速度を修正
  - 関節相互の影響を考慮して加速度を制御



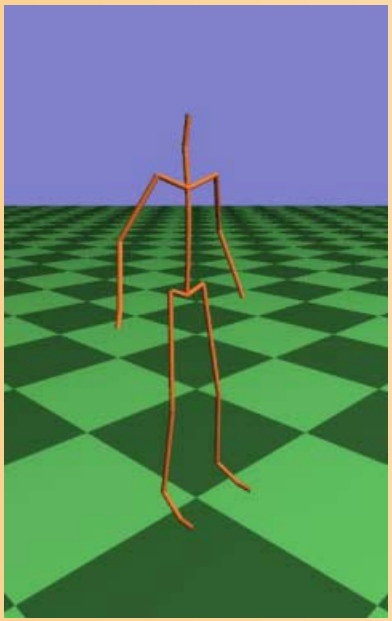
目標動作への追従

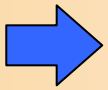
- 各関節の角加速度を計算
  - PD制御では、適切な加速度の決定は困難
  - 目標状態と Hermite 曲線から加速度を決定

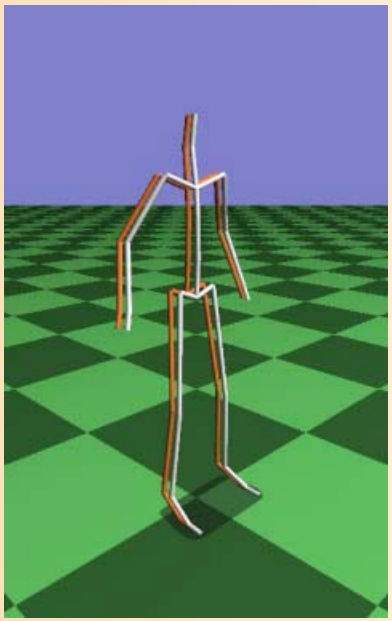


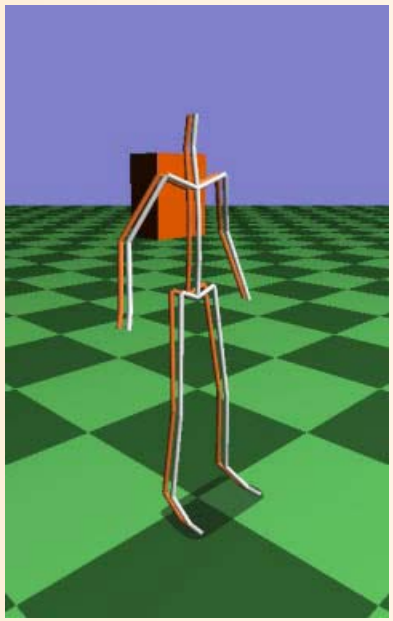
関節負荷に応じた制御(1)

- ある値以上のトルクが関節に発生したら、その負荷を減らすように、他の関節を運動
  - 関節相互の影響を考慮して、角加速度を修正









関節負荷に応じた制御(1)

- 関節負荷の計算

$$\underbrace{\tau}_{\text{関節トルク}} = H(\theta)\underbrace{\ddot{\theta}}_{\text{関節角加速度}} + C(\theta,\dot{\theta}) + G(\theta) + K(\theta)F$$

  - 関節負荷と加速度の関係は、慣性モーメント行列のみにより決まる
    → どの関節を制御すれば良いかが分かる

負荷のかかっている関節

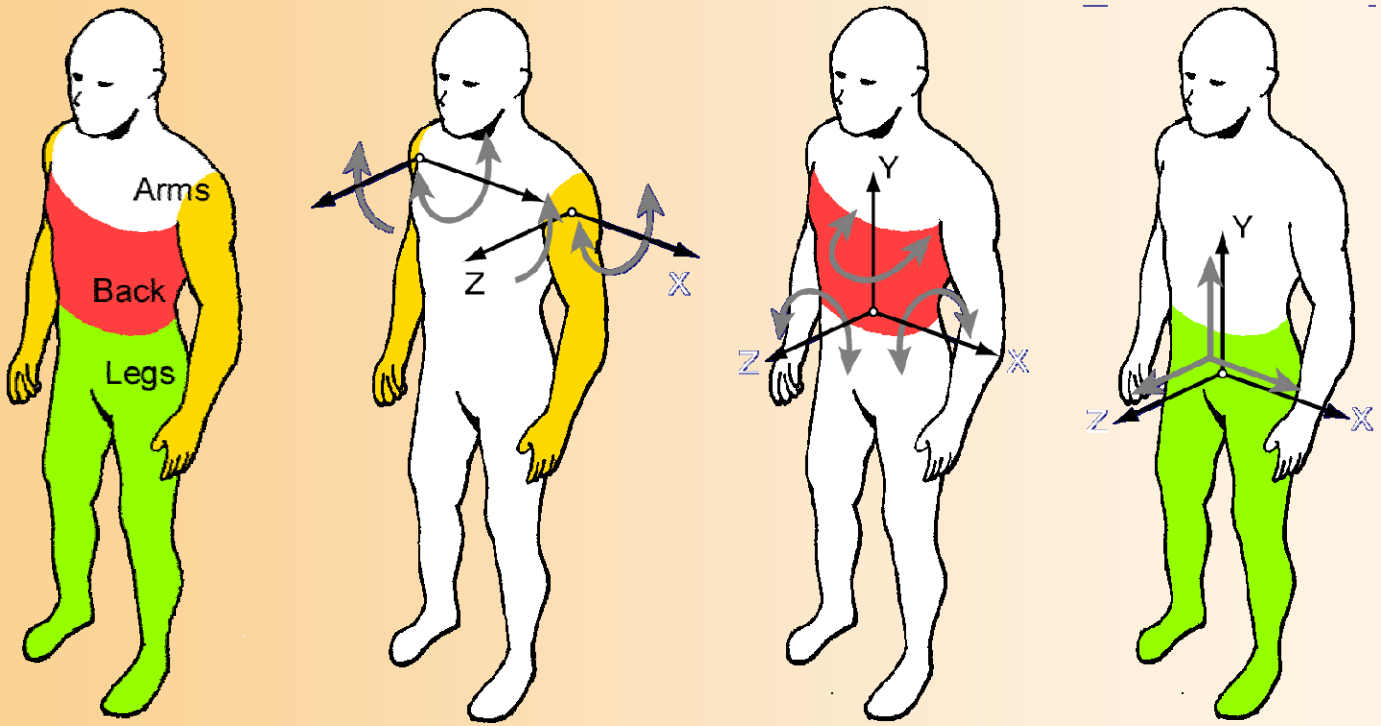
$$\begin{pmatrix} \square \\ \Delta\tau \end{pmatrix} = \begin{pmatrix} \square\!\square & \square\!\square \\ H(\theta) & \end{pmatrix} \begin{pmatrix} \Delta\ddot{\theta} \\ \square \end{pmatrix}$$

制御する関節

関節負荷に応じた制御(2)

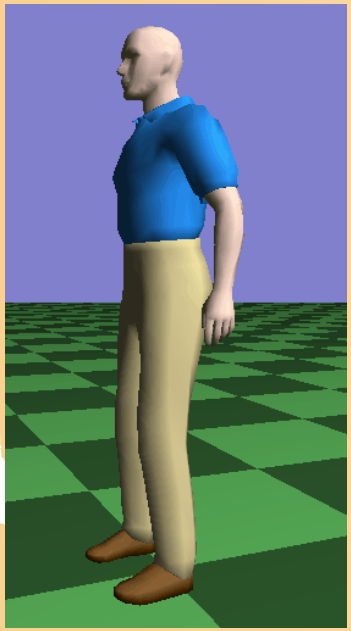
- 主要な関節のみに注目して制御

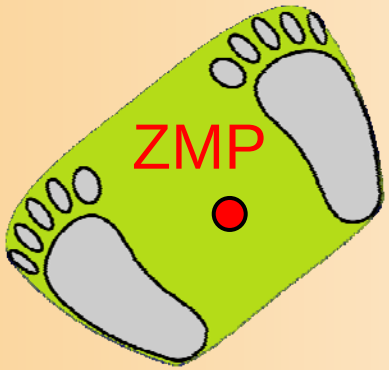
$$\Delta\tau_{\text{負荷を減らしたい関節}} = H(\theta)' \Delta\ddot{\theta}_{\text{制御する関節}}$$



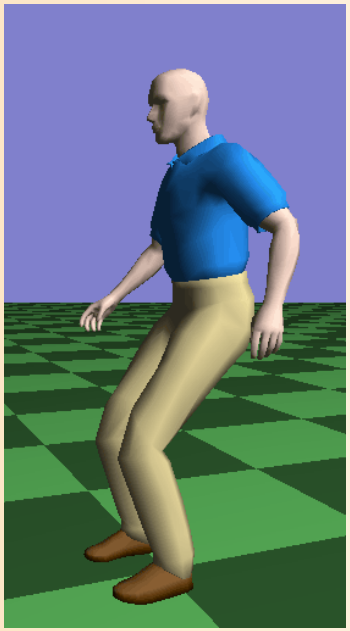
バランスに応じた制御(1)

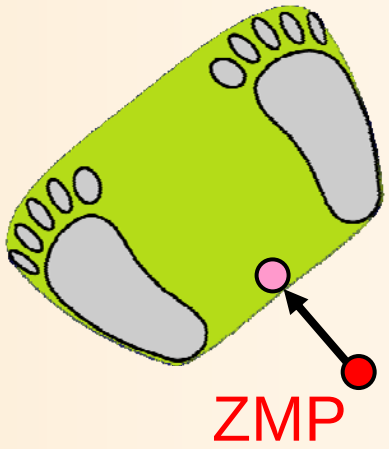
- バランスが崩れたら、バランスを保つように、複数の関節を制御
  - ZMP(Zero Moment Point)によりバランスを判定





バランスが取れた状態



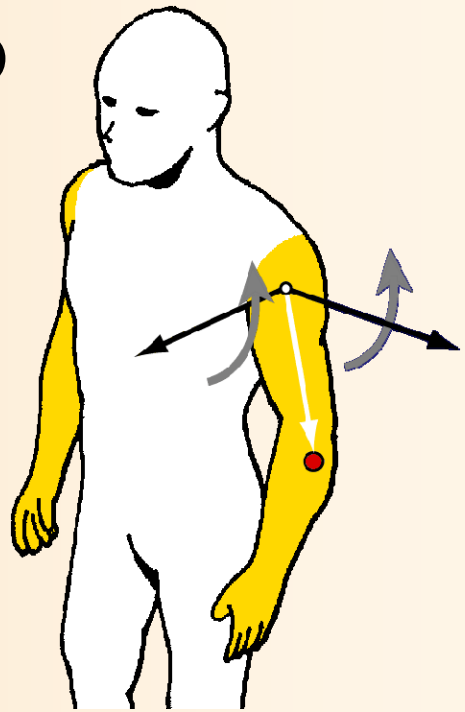


バランスが崩れた状態

バランスに応じた制御(2)

- ZMPは、全関節の角加速度から計算できる
- 各関節の角加速度が変化したときに、ZMPの位置がどれだけ変化するかも計算可能
  - ZMPを適切な位置に動かすための主要関節の角加速度変化を計算

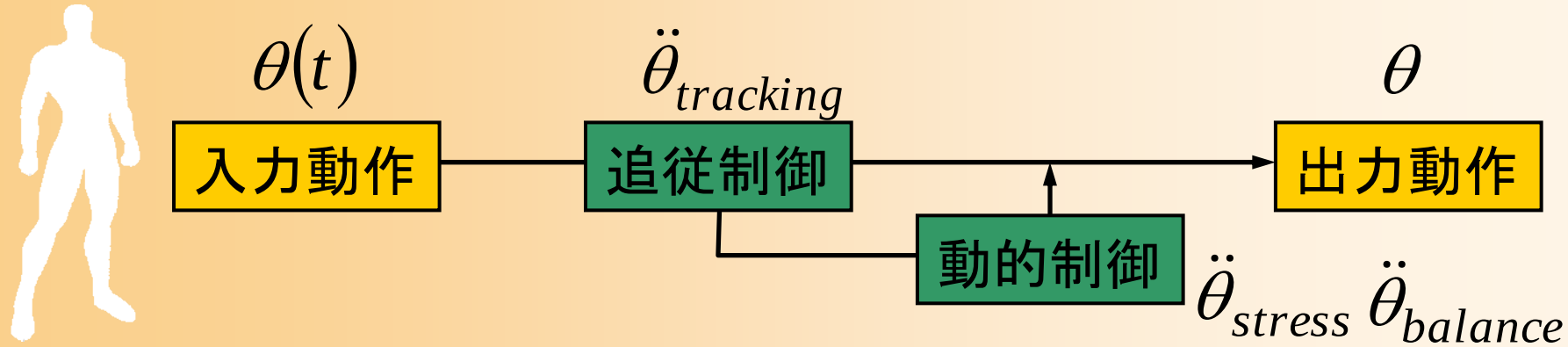
$$\Delta ZMP = \frac{\partial ZMP}{\partial \ddot{\theta}_{arms}} \Delta \ddot{\theta}_{arms}$$

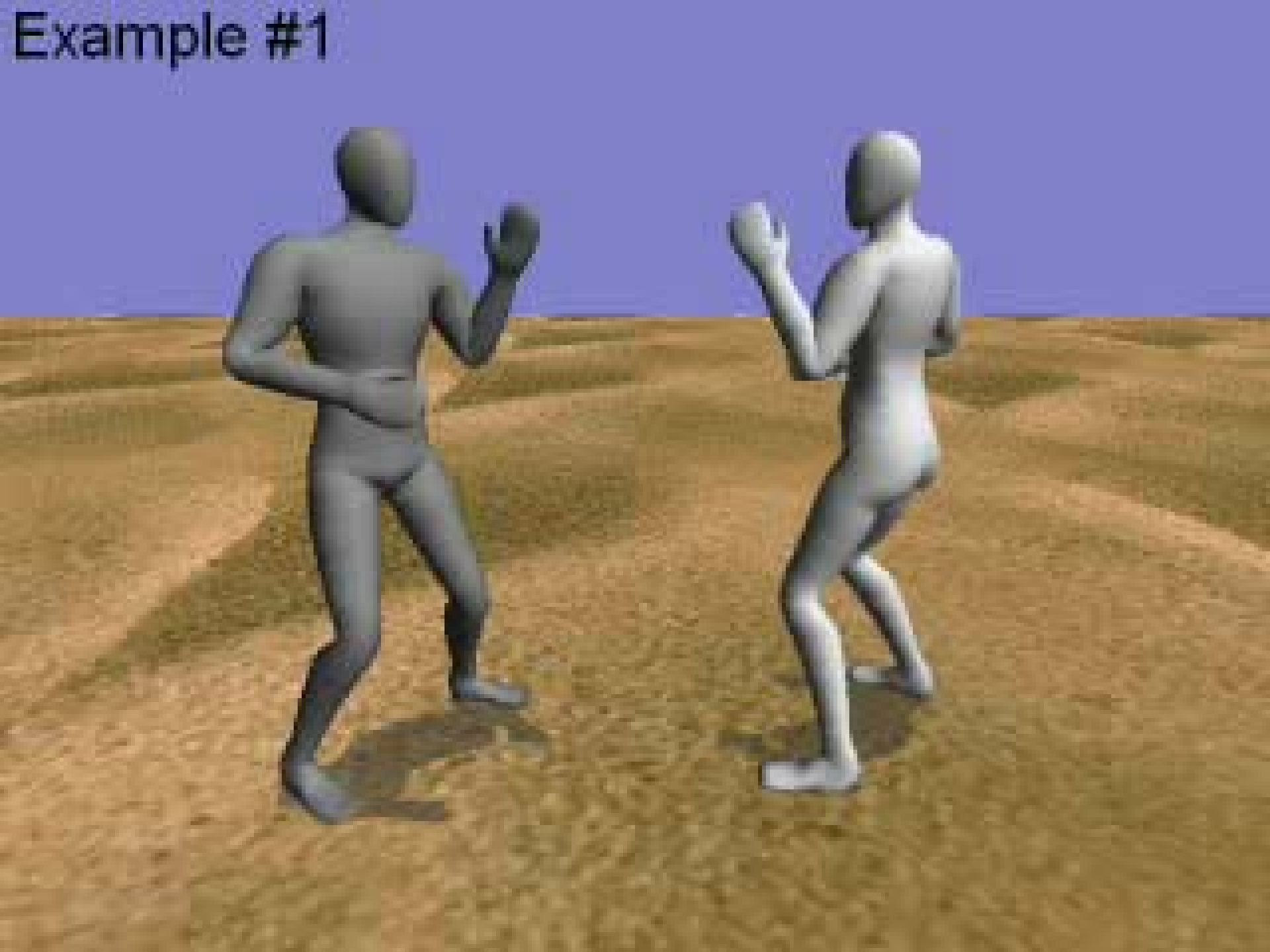


動的制御のまとめ

- 加速度の計算
  - 基本的には入力動作を追従
  - 力学的な影響に応じて動的制御を行う
    - 関節の負荷を軽減する制御、バランスを保つ制御
- 加速度にもとづき運動を計算

※ 実際は、下半身の姿勢は、腰と足の位置より計算



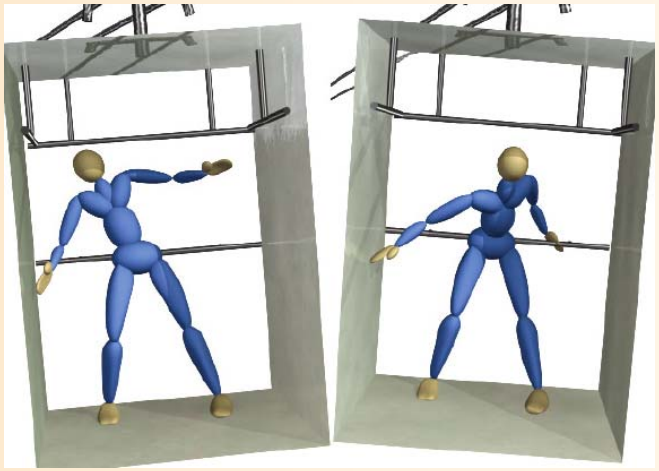

Example #1

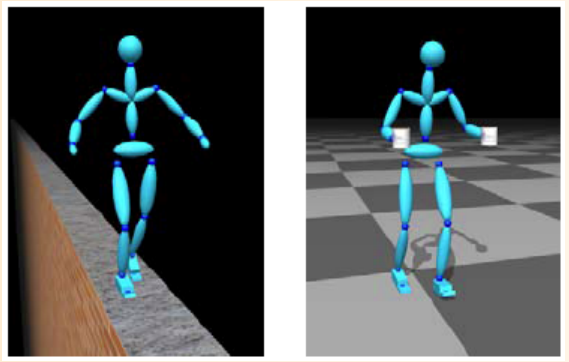
最近の研究の動向

- モデルベース手法 vs データベース手法
  - モデルベース手法
    - 何らかのモデル(プログラム)に従って動きを生成
    - 個人ごとの動きの変化の実現などは困難
  - データベース手法
    - 大量のデータを用意しておくことで、問題を解決
    - 準備が必要、適切なデータの検索・変形は困難
  - ハイブリッド手法
    - 両方をうまく組み合わせる手法
    - どのように組み合わせるかは、難しい

関連研究（モデルベース手法）

- 最近のSIGGRAPH論文
  - 関節角度制御 [Jain 2009]
    - 力学的要素を考慮して、最適な角度（＋速度、加速度）を決定
    - 最適化問題に帰着
  - 歩行制御モデル
    - 動力学シミュレーションによる歩行動作
    - 安定性を上げるための工夫

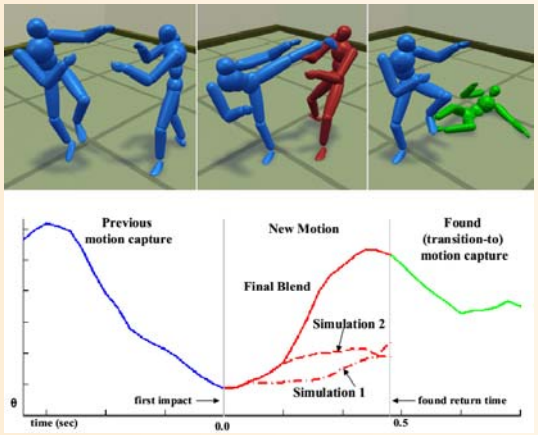




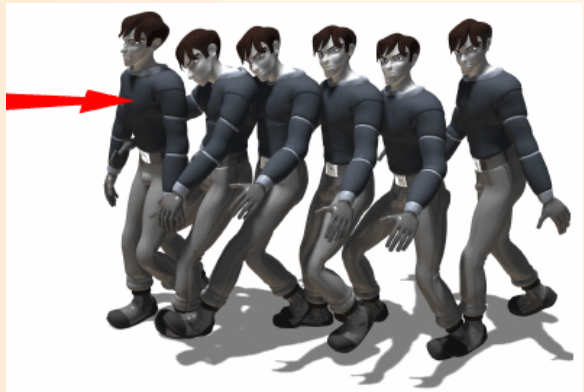
[Wang 2010]

関連研究（ハイブリッド手法）

- シミュレーションと動作データの利用 [Zordan 05]
  - 衝撃後の動作を物理シミュレーションで計算（ODE）
  - 後にうまくつながるような転倒動作をデータベースから検索



- 動作データの選択と変形 [Arikan 05]
  - 衝撃・姿勢に応じたリアクション動作データを検索
  - 各部位に一定の速度を加えてIK により姿勢を変形



関連技術（ハイブリッド手法？）

- Endorphin, Euphoria（Natural Motion 社）
  - 基本的には、本研究と同様、人間の動作制御モデルが実装されている
    - さまざまなビヘイビアの制御モデルが用意されており、ビヘイビアの種類やタイミングを指定することで、自動的に動作が生成される（endorphin）
    - 詳細な内部技術は公開されていない
    - モデルベースなので、キャラクタの個性を出すことは難しい？

Natural Motion Ltd

ロボット工学との関連

- ロボット制御とはやや目的や手段が異なる
  - ロボット制御
    - 力学的に正確な制御が必要
    - 現在ではごく単純な動作しか実現できていない
    - 人間らしい動きの実現はあまり考えられていない
    - 人間とはモータの機構も異なる
  - アニメーション
    - 力学的には厳密でなくても人間らしい動きを実現
  - 動力学などの基礎的な理論はかなり共通
- 将来的にはヒューマノイドロボットの技術がゲームにも応用可能になる？

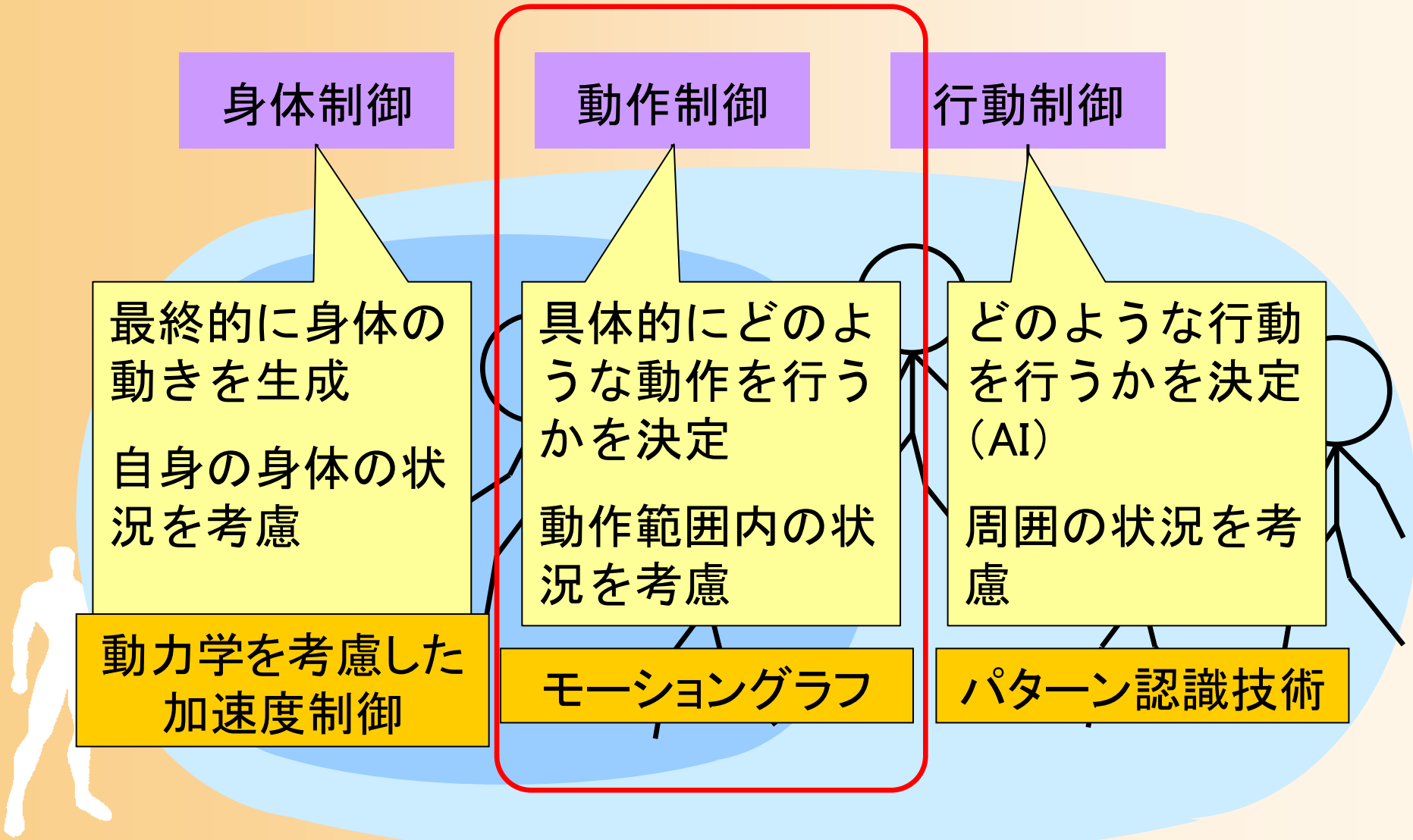
まとめ

- 動力学シミュレーションを用いた動作制御
- 動力学シミュレーション（ラグドール・シミュレーション）は容易に実現可能
- 人間らしい能動的な動きを実現する手法が課題

実装のヒント

- 動力学シミュレーション
  - 既存のエンジンが利用可能（ODE, Havok, PhysX）
  - 多関節体モデルの生成（変換）が必要
- PD制御は容易に実装可能
  - 安定動作には調整が必要
- 簡単な制御モデルを試すことは可能
  - 逆動力学計算、ZMP計算、慣性行列計算等の計算方法はロボット工学の教科書を参照

動作制御


身体制御
動作制御
行動制御
最終的に身体の動きを生成
自身の身体の状況を考慮
具体的にどのような動作を行うかを決定
動作範囲内の状況を考慮
どのような行動を行うかを決定（AI）
周囲の状況を考慮
動力学を考慮した加速度制御
モーショングラフ
パターン認識技術

動作制御

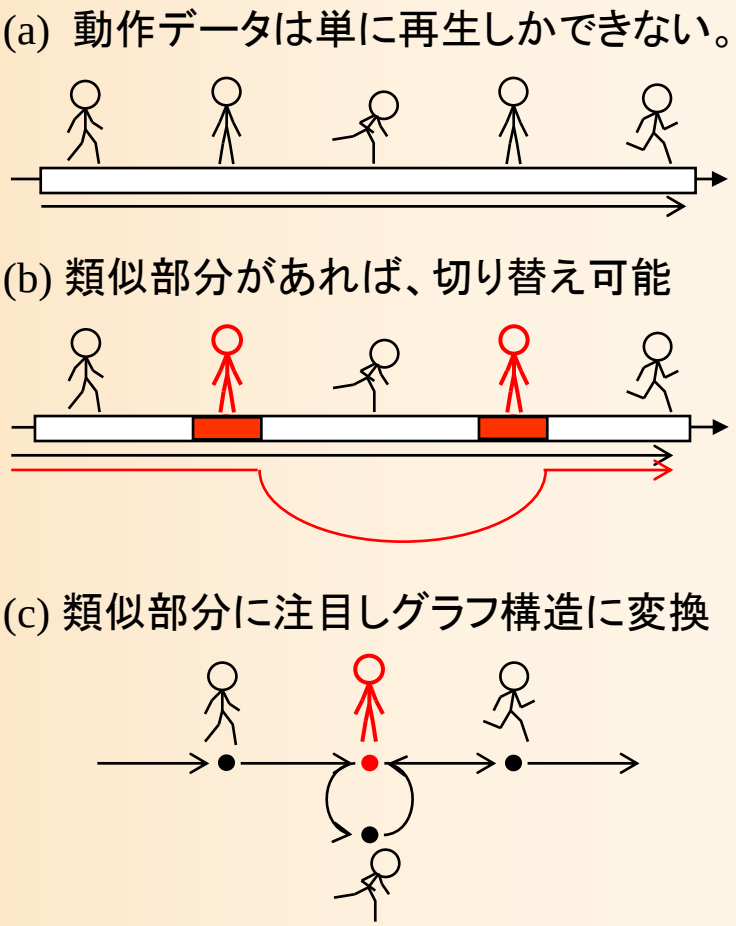
モーショングラフ

動作制御

- モーショングラフを利用した動作制御
  - 動作データを自動的に組み合わせてさまざまなバリエーションの動きを生成できる
- 格闘動作（回避動作）への応用例

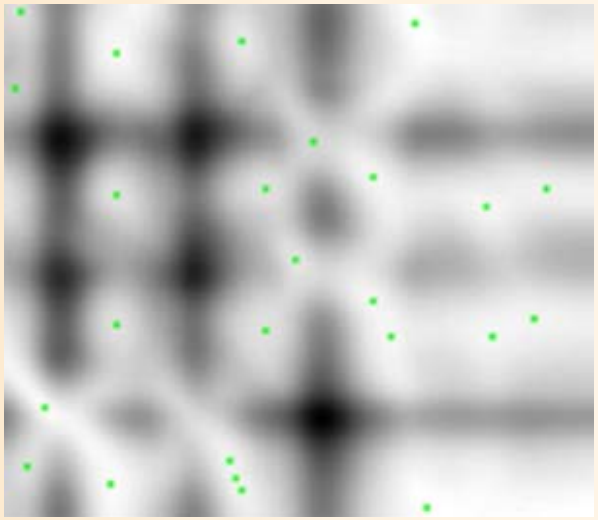
モーショングラフ

- モーショングラフ [Kovar 02]
  - モーションキャプチャデータは、有向グラフ構造に変換できる
  - ノードを順に遷移することで、連続的な動作を生成できる
    - 遷移のルールが重要



モーショングラフの作成方法

- モーション中の各フレーム同士の姿勢間の距離を計算（マッチウェブ）
  - 姿勢同士の距離の評価方法も重要
    - 例：全主要部位同士の距離の和など

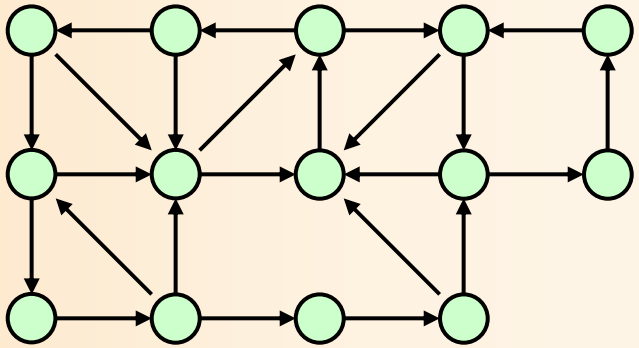


Kovar 02

- マッチウェブ中の極小点をモーショングラフのノードとし、ノード間の動作をエッジとする

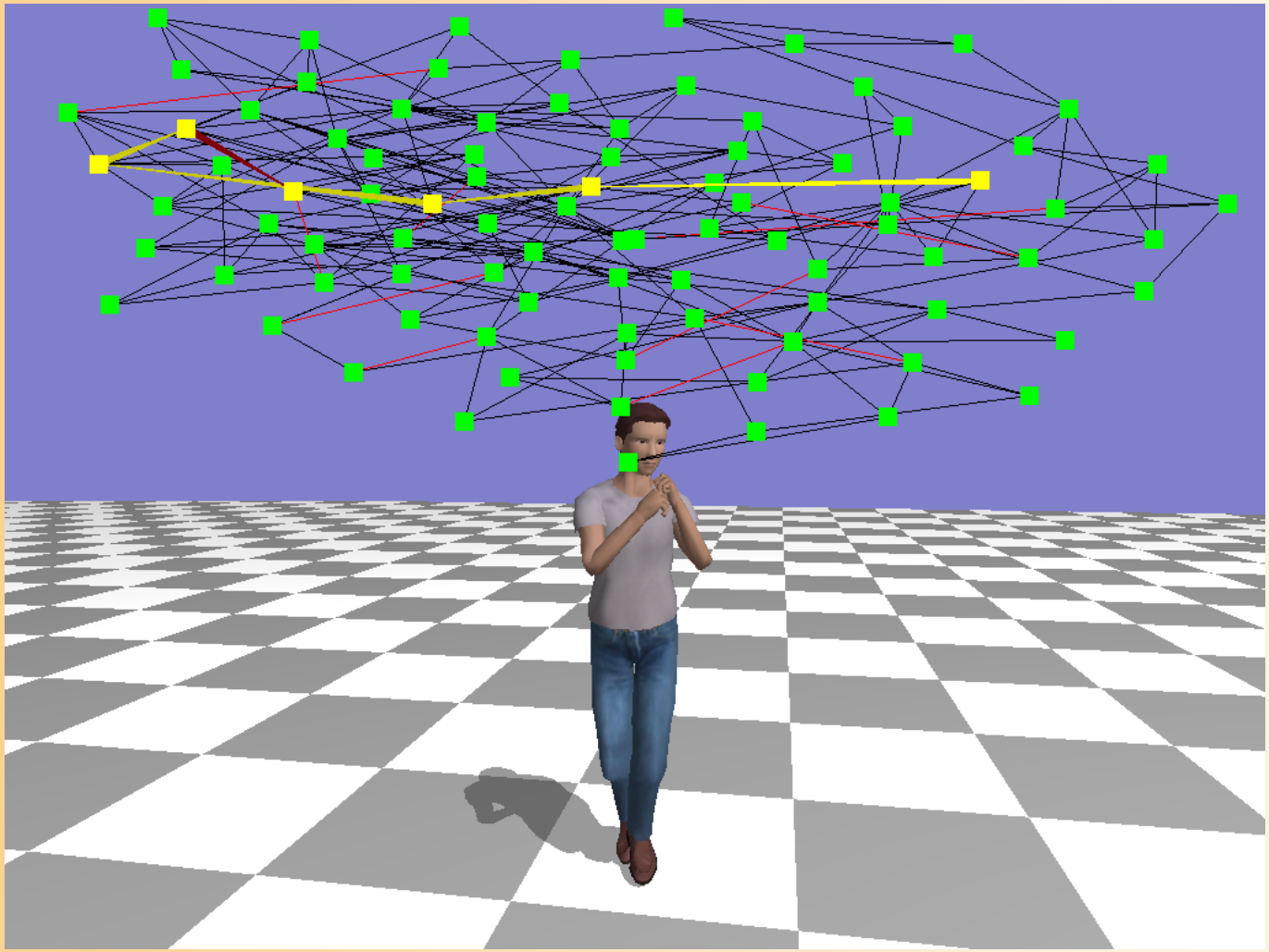
モーショングラフによる動作生成

- データ構造
  - ノード・・・姿勢
  - エッジ・・・動作
    - 一般的な動作ツリーとは逆



- 動作生成
  - ノードを辿りながらエッジの動作を再生していくことで、動作を生成
    - 実際には、ノードの前後で、動作ブレンディングや、足を地面に固定するための IK が必要
  - ノード遷移のためのルールが重要

デモ

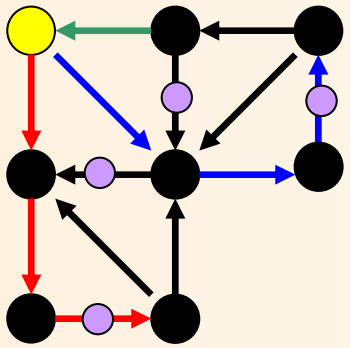


モーショングラフによる回避動作の実現

- 研究目的
  - さまざまな攻撃を紙一重で回避するような華麗な回避動作を実現したい
    - 現在のゲームでは、基本的に、移動による回避しかできない
    - アクション映画のような動作に近づけたい

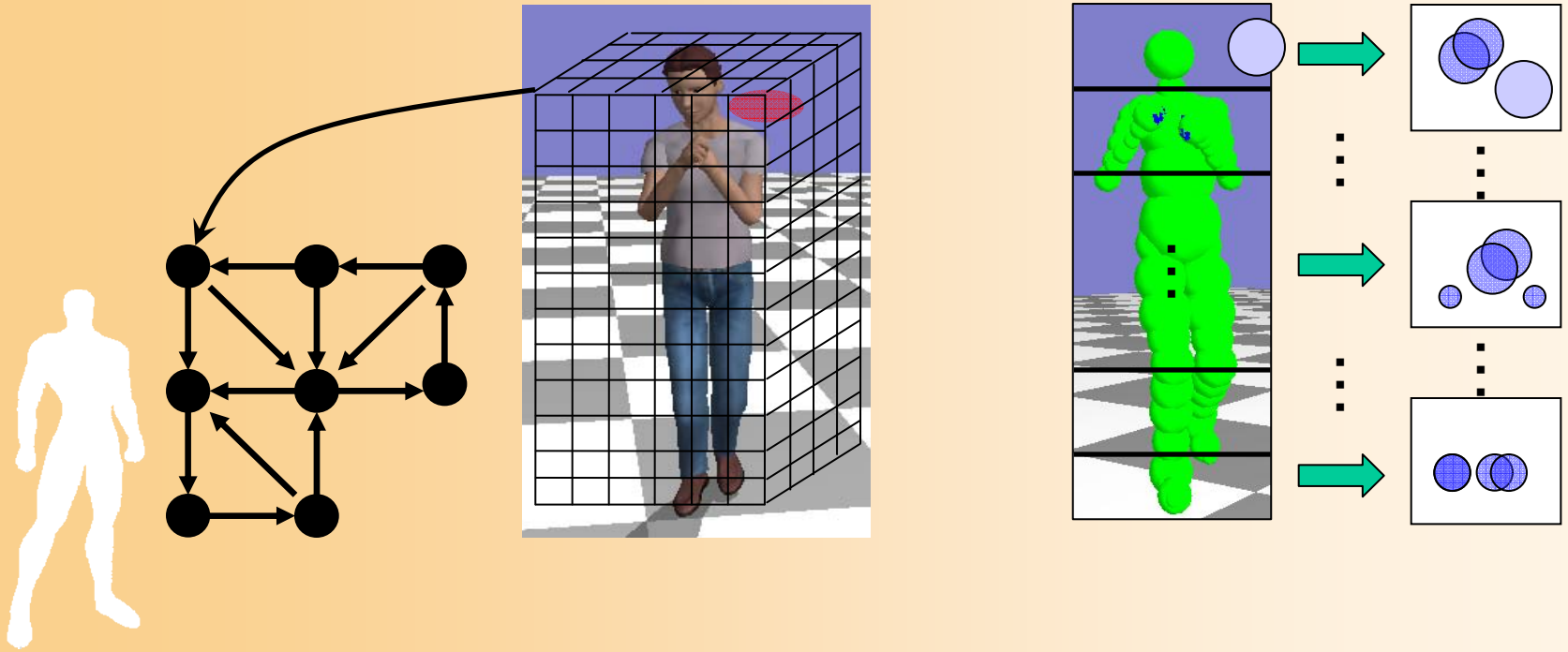
提案手法(1)

- 攻撃情報(領域・時間)に応じて、回避動作を実行するパスを決定
  - 各パス候補の再生速度を調節して、回避のタイミングを合わせる
  - 各パス候補を評価
    - 攻撃領域と回避領域が重なる動作を選択
    - 攻撃領域と身体領域が重ならず、なるべく近い動作を選択



提案手法(2)

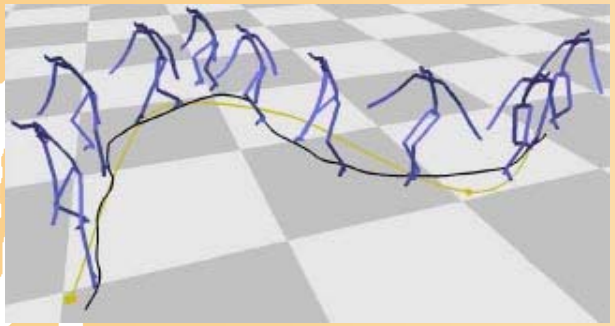
- 各候補の領域交差判定を高速に行うため、インデックスやGPUによる判定を利用



デモ

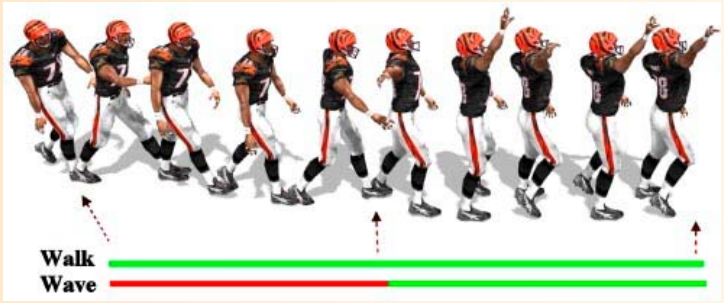
関連研究

- モーショングラフにおけるノード選択条件
  - 歩行パスによる条件 [Kovar 02]
  - 手先位置の条件 [Lee 05]
  - 動作の種類の条件 [Arikan 04]
  - 格闘動作 [Shum 08]　音楽に合わせた動作



[Kovar 02]

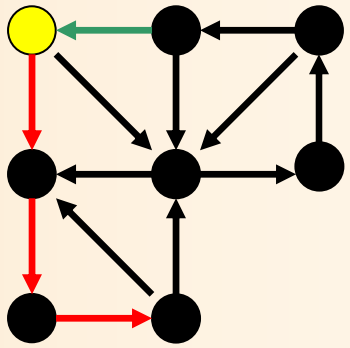
[Lee 04]



[Arikan 04]

関連研究（2）

- パス探索アルゴリズム
  - 適切な動作を実現するためには、次のエッジだけを決めるのではなく、ある程度先まで考慮してパス（複数のエッジ）を決定する必要がある
  - 優先探索法、A*法、強化学習、Min-Max法、などの探索アルゴリズムが利用可能



まとめ

- モーショングラフを用いた動作生成
- 回避動作への適用例

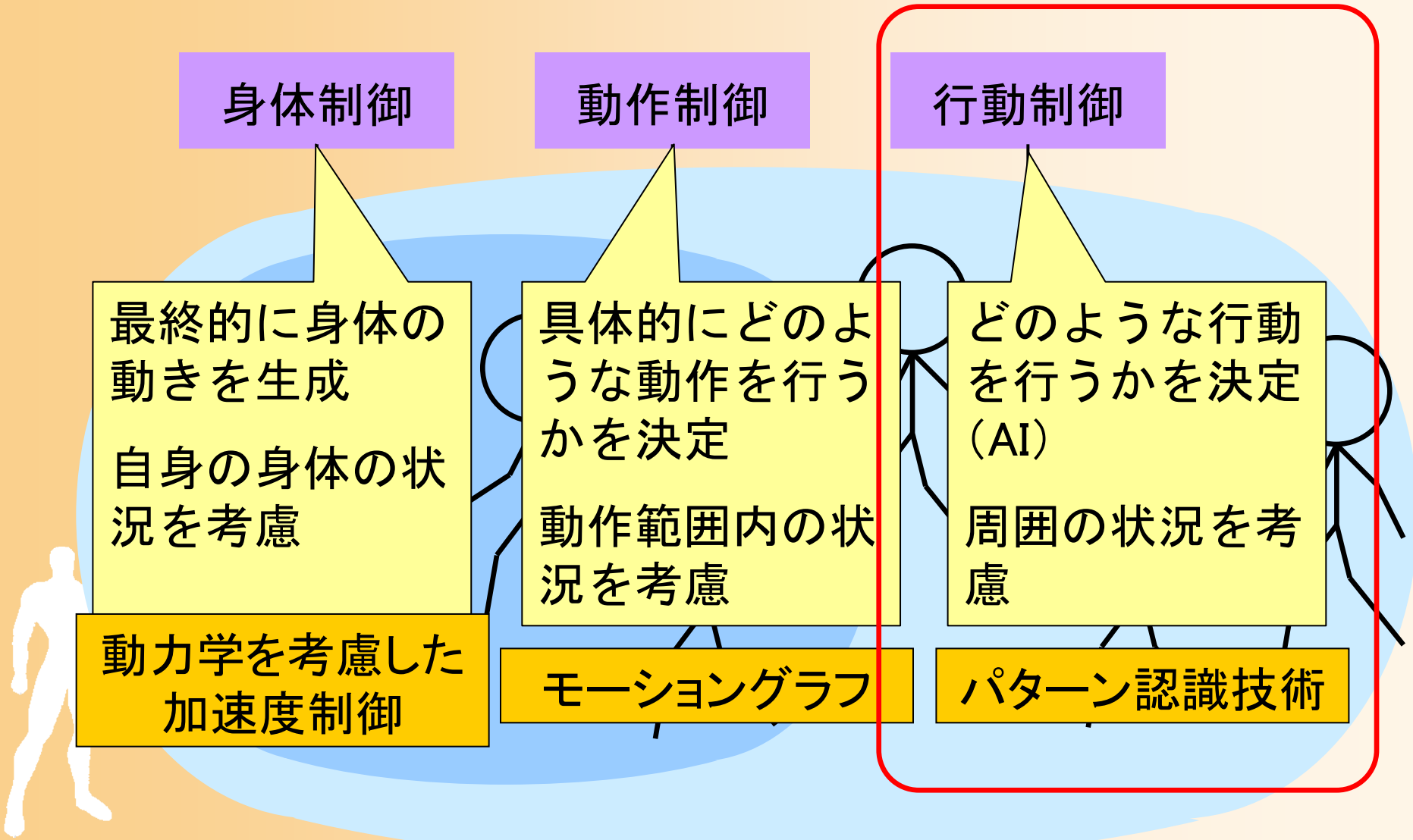
モーショングラフの利点

- モーションデータから自動的に構築できる
- 連続的な動作が無限に生成できる
  - 毎回異なる動作が生成される
  - 基本的には同じ動作の繰り返しだが、ノード数が十分にあれば、十分に自然に見える
- デメリット
  - 毎回異なる動作が生成される
    - ゲームによっては必ずしも良いとは限らない？
  - 適切な遷移ルールが必要（計算時間も考慮）

実装のヒント

- モーショングラフの生成・再生プログラム
  - それほど難しくはない
  - 公開されているサンプルプログラム等はないので、論文を参考に実装する必要がある
  - 再生処理も、ランダムに再生するだけなら簡単
- モーショングラフを利用した動作生成
  - 適切な遷移ルールの実装が必要
    - 本研究の回避動作を全て実装するのはやや困難
    - 探索等の処理には既存の基礎手法が利用可能

行動制御


身体制御
動作制御
行動制御
最終的に身体の動きを生成
自身の身体の状況を考慮
具体的にどのような動作を行うかを決定
動作範囲内の状況を考慮
どのような行動を行うかを決定（AI）
周囲の状況を考慮
動力学を考慮した加速度制御
モーショングラフ
パターン認識技術

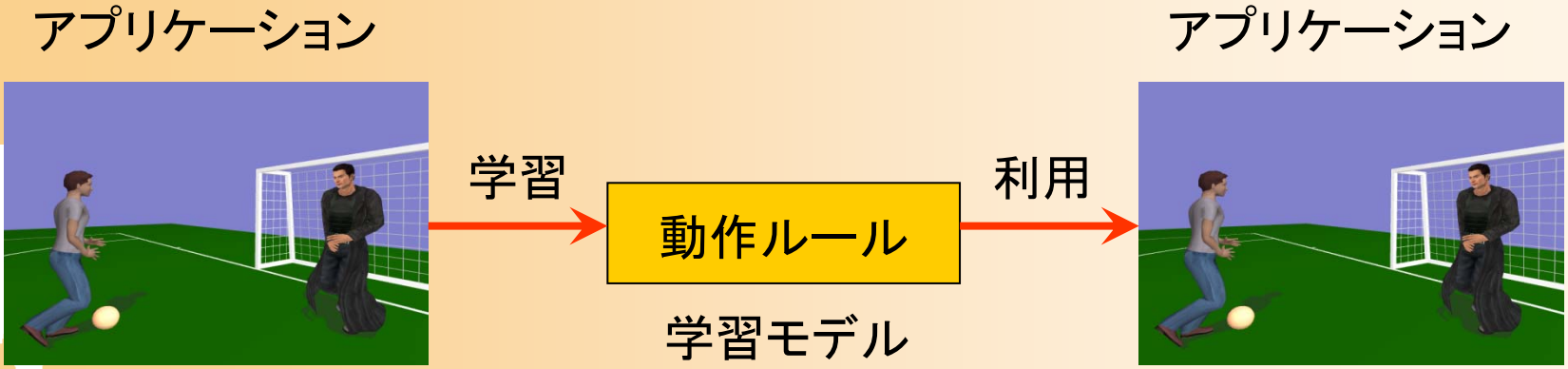
行動制御

パターン認識技術

行動制御

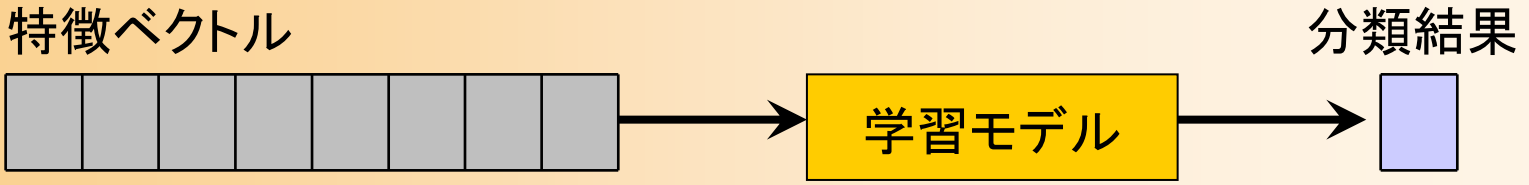
- パターン認識技術を応用した行動制御

- 人間（プレイヤー）から行動制御のルール（動作ルール）を自動的に学習

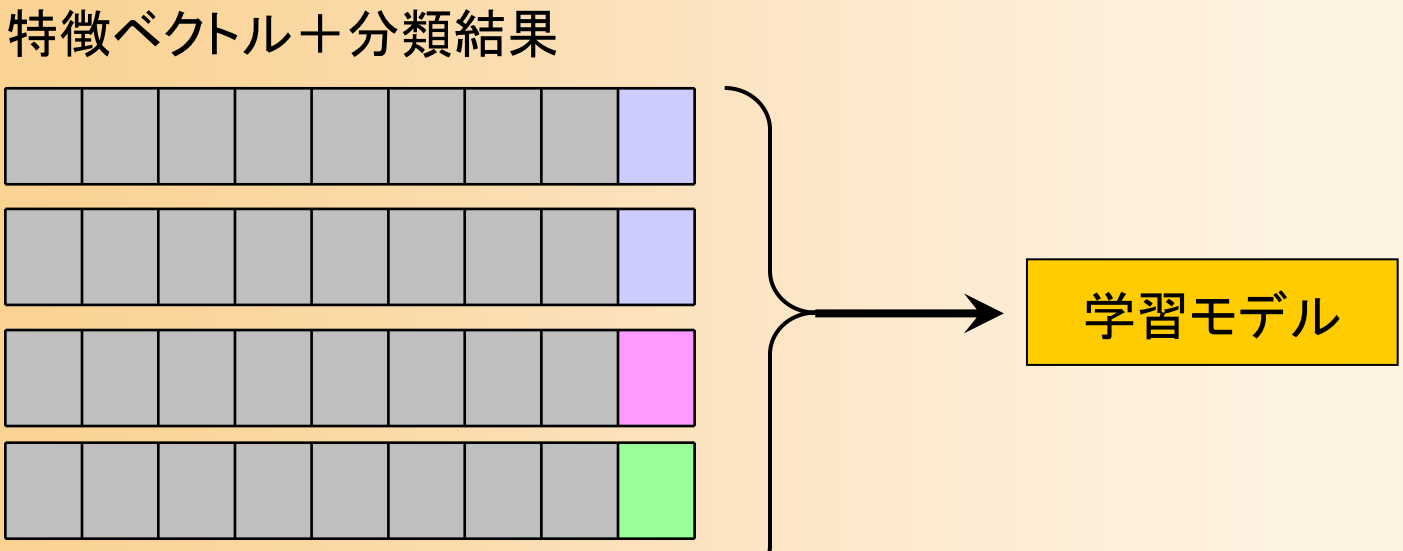


パターン認識技術とは？

- 入力された特徴ベクトルを判別（分類）

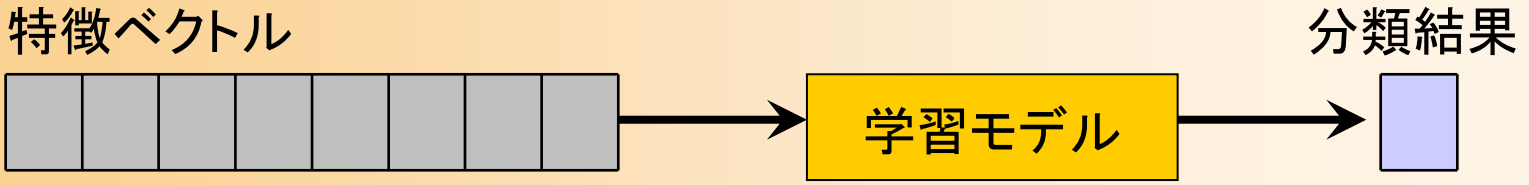


- 事前学習（教師あり学習）



パターン認識技術の応用

- パターン認識技術の行動決定への応用



行動を決定する上での条件となる変数をまとめたもの（自分の位置、敵との位置関係、味方との位置関係、等）

行う動作のラベル（前に移動、攻撃、等）

- 人間（プレイヤー）がどのような状況でどのような行動を行うか、データを集めて学習
- パターン認識技術としてSVMを利用（後述）

操作ログからの動作ルールの学習

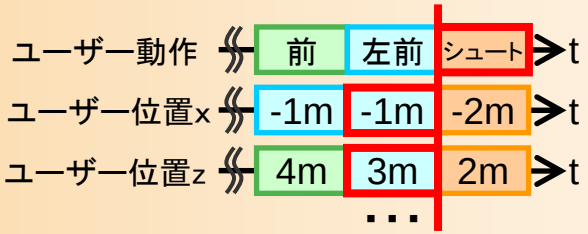
操作ログ記録アプリケーション

入力：ユーザーの操作

各時刻で記録

操作ログ

ユーザが行った操作+特徴量



動作開始時の特徴量と開始動作を抽出

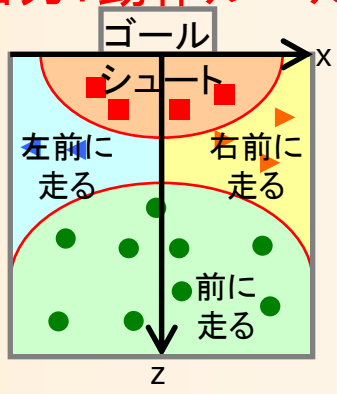
動作開始条件の教師データ

動作開始時の特徴量			動作
位置x	位置z	…	
-1m	4m	…	前に走る
-1m	3m	…	シュート
…	…	…	…

SVMを用いて動作ルール抽出

動作開始条件の範囲を推定
任意の特徴量での開始動作がわかる

出力：動作ルール



サッカーの動作ルールの実験例

入力 操作ログ(1~2分)
利用者がキャラクタを操作

出力アニメーション
動作ルールでキャラクタを制御

- キャラクタ:オフェンス、ディフェンス、キーパー
- 特徴量: 各キャラクタの位置(計6次元)

戦闘動作の動作ルールの実験例

学習に使った入力データ（人間が操作）

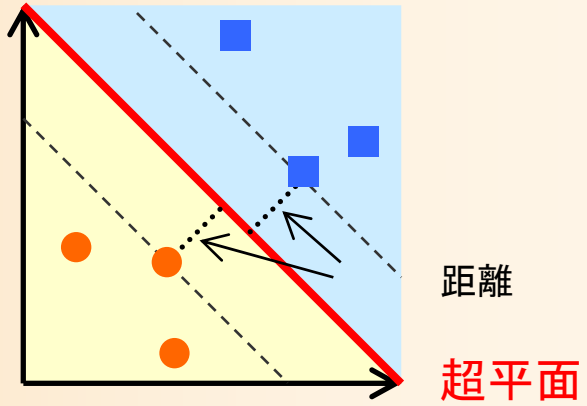
学習した動作ルールによる動作生成

一般的なパターン認識技術（学習アルゴリズム）

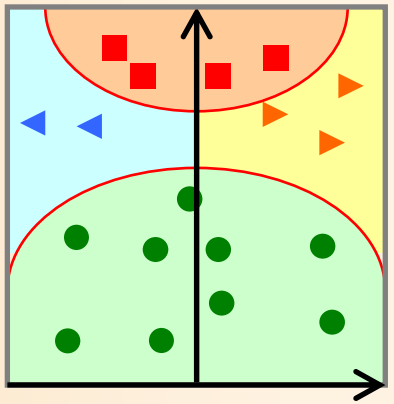
- Support Vector Machine（SVM）
  - 高次元のパターン認識、本研究で利用
- ニューラルネット
  - 高次元のパターン認識、未知のデータの認識には弱い？
- 自己組織化マップ（SOM）
  - データ分類手法（教師なし学習）
- 隠れマルコフモデル（HMM）
  - 1次元の時系列信号の認識に向いている
- 遺伝的アルゴリズム（GA）、強化学習
  - 最適化問題の解法（≠パターン認識）

Support Vector Machine

- Support Vector Machine (SVM) [Vepnik 95]
  - 代表的な認識手法
  - 特徴空間を、各サンプルと超平面の距離が最大になるように、超平面で分割

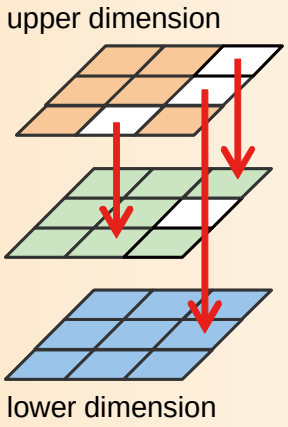


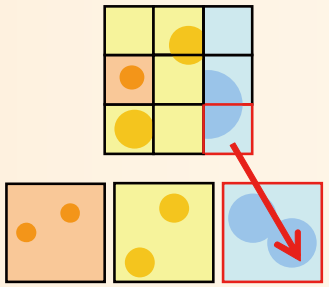
- LibSVM [Chang and Lin]
  - 多クラスの識別に対応
  - BRF基底関数を使用した非線形の空間分割



SVMの問題と改良

- SVMの問題
  - 特徴空間の次元が高くなると多くのデータが必要
  - 適切なパラメタ(領域の半径)の設定が必要
    - 特徴空間の領域ごとに適切なパラメタは異なる
- SVMの改良(SVMの階層化)
  - 複数の特徴量ベクトルで階層化
  - 複数のパラメタで階層化





SVMs with different parameters

比較

- 固定の特徴量やパラメタでは、不自然な動作が生成されることがある

提案手法
特徴量とパラメタの
階層化を使用

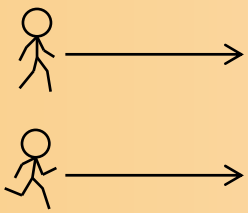
固定の特徴量を使用

固定のパラメタを使用

今後の課題

- 複数の演技者のモーションキャプチャデータを解析し、人間の動作ルールを抽出・再現

例：戦闘やスポーツ等



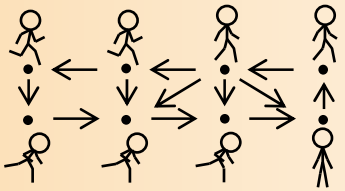
複数の演技者のモーションデータ

解析

例：後ろから攻撃されたら回避、
敵に囲まれたらパスする、等

動作ルール

＋



モーショングラフ

動作生成に適したデータ構造に変換

アニメーション
（多数のキャラクタの動作を生成）

関連技術

- Massive
  - GUI環境によるルールの記述
    - ファジィルールや状態遷移をGUI画面上で設計
  - 多数の映画で利用
  - 利用にはかなりの知識や労力が必要

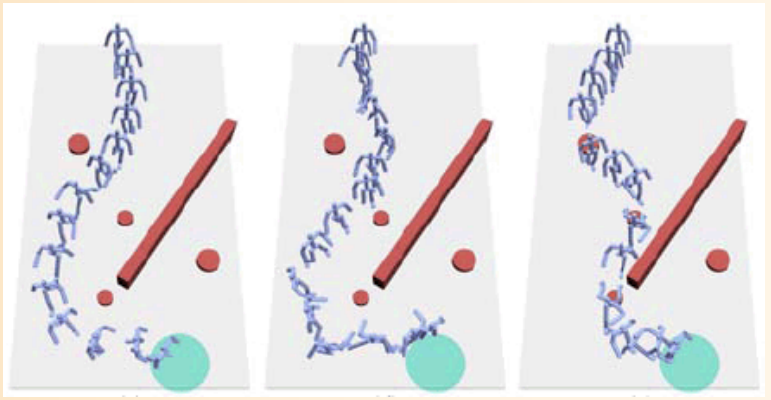
Massive

Load of the Rings, 2002

Chronicles of NARNIA, 2005

関連研究

- 行動ルールの自動学習
  - 逆強化学習 [Lee 2010]
    - 動作計画のための評価基準を自動学習



- 確率の学習
- 高度なルールベースの手法（自動学習はなし）
  - 視界にもとづく動作決定 [Ondrej 2010]

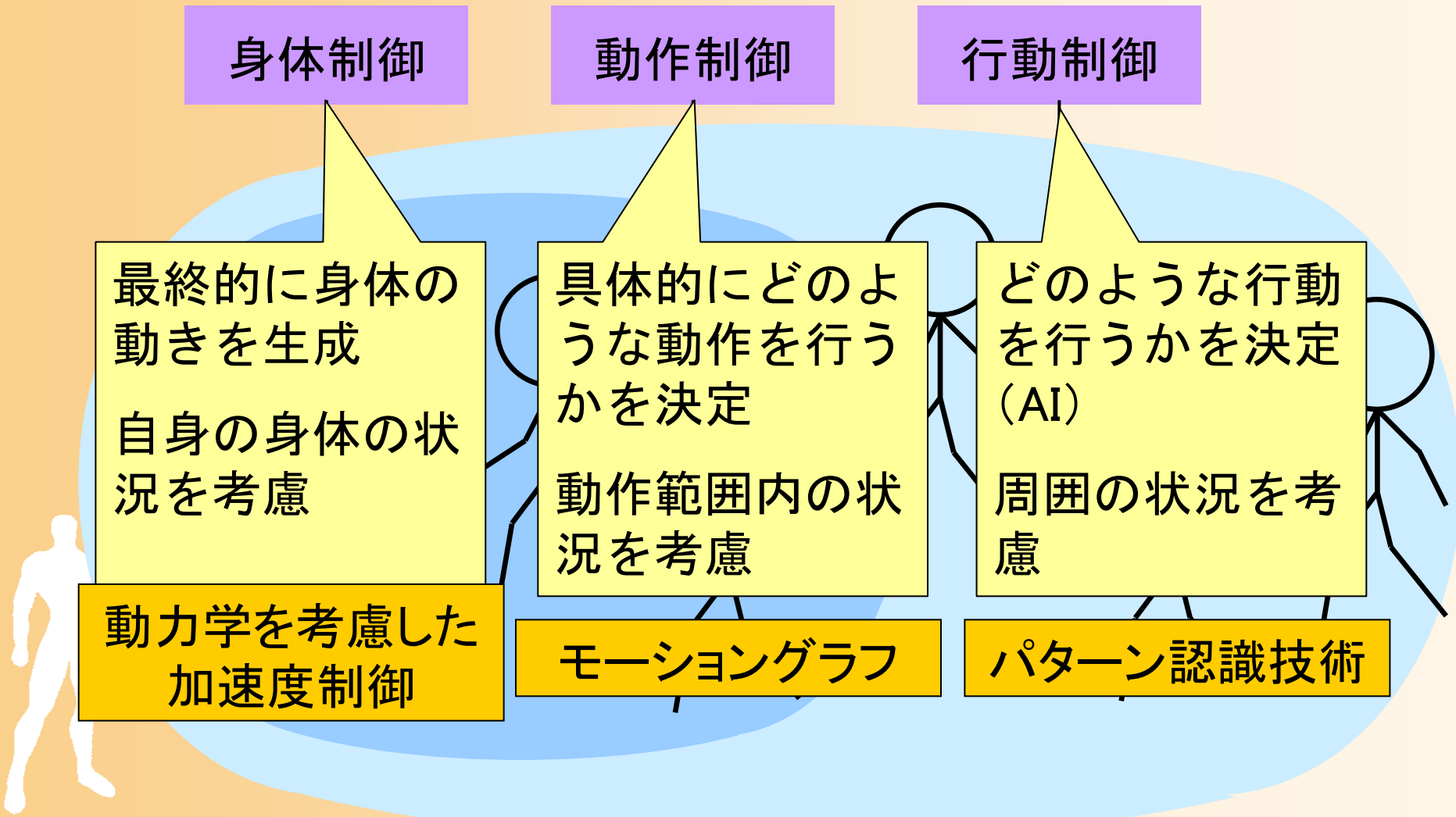
まとめ

- パターン認識技術を用いた行動決定
  - 自動的に行動制御モデルの作成（学習）を行うことが可能
  - ゲーム中にプレイヤーの動きを学習して成長するような機能も実現可能？
  - うまく動かない可能性もある
  - （ゲームレベルの）調整が難しい？

実装のヒント

- パターン認識技術
  - 一般的に利用可能なライブラリも多数公開されている
    - 商用利用は不可のものも多いので要注意
- 行動決定
  - パターン認識が利用可能であれば、それを利用した行動決定は容易に実現可能
  - 学習モデルの作成(とテスト)には、ある程度、試行錯誤が必要になるかもしれない

まとめ

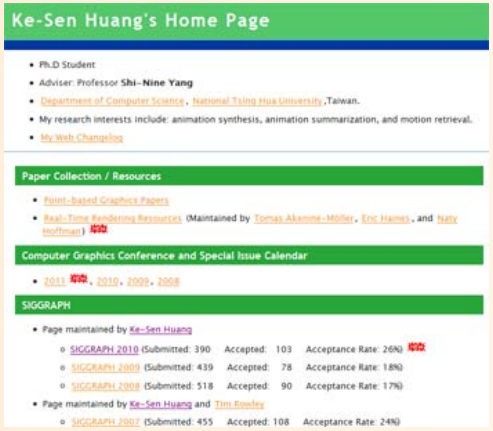

身体制御
動作制御
行動制御
最終的に身体の動きを生成
自身の身体の状況を考慮
具体的にどのような動作を行うかを決定
動作範囲内の状況を考慮
どのような行動を行うかを決定（AI）
周囲の状況を考慮
動力学を考慮した加速度制御
モーショングラフ
パターン認識技術

ゲーム開発に役に立つか？

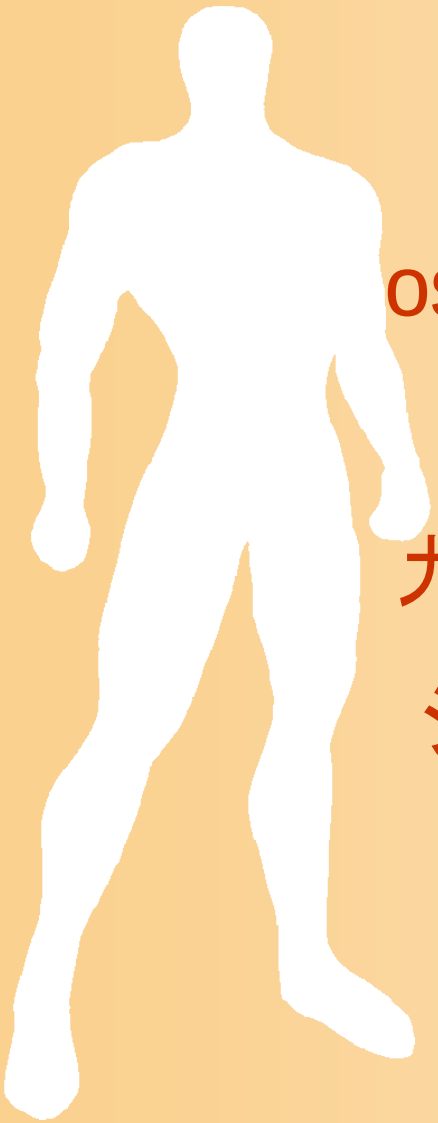
- プレイヤーの問題意識？
  - 一般のプレイヤーは、現在のゲームでキャラクタの動作が不自然とは感じていない？
- 研究とゲーム開発の違い
  - 研究では、一般的な手法の方が評価される
  - 研究手法を具体的なゲーム開発に利用するにはチューニングが必要

研究のゲームへの応用

- 文献調査
  - SIGGRAPHなどの国際会議の論文へのリンクをまとめたページ
    - http://kesen.huang.googlepages.com/
  - 学会のデジタルライブラリ（ACM、IEEE CS）



- 産学連携（共同研究）
  - 研究成果のゲームタイトルへの応用
  - 問題の持ち込み、海外の論文の技術の実装
  - 学生への共同研究テーマの割り当て（求人活動？）



尾下 真樹
oshita@ces.kyutech.ac.jp

九州工業大学 情報工学部
システム創成情報工学科
尾下研究室
www.oshita-lab.org

参考文献(1)

- 本発表に関連する筆者の研究
- 身体制御
  - Masaki Oshita, Akifumi Makinouchi, "A Dynamic Motion Control Technique for Human-like Articulated Figures", presented at *EUROGRAPHICS 2001*, *Computer Graphics Forum*, Vol. 20, No. 3, pp. 192-202, September 2001.
- 動作制御
  - 正岡 直樹, 尾下真樹, "モーショングラフによる回避動作の実現", Visual Computing / グラフィックスとCAD 合同シンポジウム 2009 予稿集, 6 pages, 旭川, 2009年6月.
- 行動制御
  - Masaki Oshita, Tomomasa Yoshiya, "Learning Motion Rules for Autonomous Characters from Control Logs Using Support Vector Machin", International Conference on Computer Animation and Social Agents 2010 (CASA 2010), 4 pages, Saint-Malo, France, May-June 2010.

参考文献(2)

- 身体制御関連
  - Sumit Jain, Yuting Ye, C. Karen Liu, “Optimization-Based Interactive Motion Synthesis.” ACM Transactions of Graphics (Proc. of SIGGRAPH 2009), 28(1), Article No. 10, 2009.
  - Zordan, V. B., Chiu, B., Majkowska, A., Fast, M. “Dynamic Response for Motion Capture Animation.” ACM Transactions of Graphics (Proc. of SIGGRAPH 2005), 24(3), 697-701, 2005.
  - Okan Arikan, David A. Forsyth, James F. O'Brien. “Pushing People Around.” Proceedings of the 2005 ACM SIGGRAPH/Eurographics Symposium on Computer Animation, pp. 59-66, 2005.

参考文献(3)

- 動作制御関連
  - Kucas Kovar, Michael Gleicher, Fedderic Pighin, “Motion Graphs”, ACM Transactions on Graphics (SIGGRAPH 2002), Volume 21, Issue 3, pp. 473-482, 2002.
  - Jehee Lee, Kang Hoon Lee, "Precomputing Avatar Behavior From Human Motion Data", Eurographcis/ACM SIGGRAPH Symposium on Computer Animation 2004, pp. 79-87, 2004.
  - Okan Arikan, David A. Forsyth, James F. O'Brien, "Motion Synthesis from Annotations", ACM Transactions on Graphics (SIGGRAPH 2003), Volume 22, Issue 3, pp. 402-408, 2003.
  - Hubert Shum, Taku Komura, Masashi Shiraishi, Shuntaro Yamazaki, "Interaction Patches for Multi-Character Animation", ACM Transactions on Graphics (SIGGRAPH Asia 2008), Volume 26, Issue 3, Article No. 114, 2008.

参考文献(4)

- 行動制御関連
  - Chih-Chung Chang and Chih-Jen Lin, LIBSVM - A Library for Support Vector Machines, http://www.csie.ntu.edu.tw/~cjlin/libsvm/
  - Seong Jae Lee, Zoran Popovic', “Learning Behaviour Style Using Inverse Reinforcement Learning”, ACM Transactions on Graphics (SIGGRAPH 2010), Article No. 122, Volume 29, Issue 4, 2010.
  - Jan Ondřej, Julien Pettré, Anne-Hélène Olivier, Stéphane Donikian, “A Synthetic-Vision Based Steering Approach for Crowd Simulation”, ACM Transactions on Graphics (SIGGRAPH 2010), Article No. 123, Volume 29, Issue 4, 2010.