遺伝的プログラミングを用いた
2足歩行運動生成に関する研究

工学研究科
筑波大学

2001年7月

玉 秀列

目次

図一覧

表一覧

要旨

生体工学, ロボティクス, 並びに神経生理学などの分野では, 人間の 2 足歩行のメカニズムを明らかにすることは主要な興味の 1 つであり, その研究成果は, リハビリテーションツールや人間型ロボットのような様々な応用に対する基礎技術として, 提供されることが期待されている. しかしながら, 身体における神経系と力学系による複雑な相互作用である歩行動作において, その実現メカニズムに関わる神経系の詳細については, その複雑さのために, まだ多くの部分が不明確なまま残されている.

近年の研究では, 人間が行う協調的歩行運動は, 身体筋骨格などの力学系が持つ振動リズムと, 神経系が持つリズムとの相互引き込み作用によって発生すると考えられている. しかし, そのような複雑な相互作用を伴うシステムに関して, 計算機を用いてシミュレートするためには一般に非常に困難なモデル構築が伴い, 従来は試行錯誤による解決に頼らざるを得なかった. そこで, 本研究では, 進化的計算手法を用いて, 引き込み現象による 2 足歩行運動の創発シミュレーションを計算機上で自動的に実現する手法を開発した. 具体的には, 遺伝的プログラミング（GP）を用いて, 神経振動子と呼ばれるリズム生成器を 2 足歩行に必要な関節毎に探索的に生成し, 得られた神経振動子群のネットワークと身体力学系のダイナミクスの相互作用として, 人間の歩行に近い 2 足歩行動作のシミュレーションを実現することに成功した.

第 1 章

序論

1.1 研究の背景

2 足歩行は人間の身体運動の中で最も重要なものの一つであり, その制御メカニズムは生物学, 工学の両面でこれまでも幅広く研究されてきた. しかし, ロボット工学やコンピュータアニメーションなどの分野に広く応用するには, 二足歩行運動の最適な制御モデルには問題が未可決のまだ数多く残されている.

生体システム機能を表現するためにはモデル化が不可欠である. モデル化は, 目的とする課題によって若干の相違はあるが, 既知の生理学的知見を基にして, できるだけ的確なモデルを設定することを目的として行なう. その際, 生体システムにおいては最適性や合目的性が実現されているとの前提をおくことが多い.

一般に, モデルにはかなり抽象的な概念追求型のものから, 具体的な生理あるいは心理機能を数量的に表現するものまで, いろいろなレベルのものがありうる. いずれにせよ, 事実を矛盾なく説明できるまで, 評価とフィードバックを繰り返し行なう必要がある. 実際には, このようにして得られたモデルが真実であるとするにはかなりの困難がある. しかし, 新たな矛盾が指摘されない限り, 一応正当なモデルとして, これをさらに応用面へ向けて展開を図っていくことが重要であると考える.

この考え方は, システム工学において, 合成による解析と呼ばれているものに相当す

る．この方法にも，もちろん数多くの限界がある．しかし，生体システムのように直接解析が困難なものに対しては，唯一の解析可能な手段と考えられ，さらに，その成果は工学基礎理論そのものにもフィードバックされ，新たなる工学基礎理論を展開する糸口にもなることが期待される．

近年，神経生理学分野における研究の発展から，脊髄にリズミカルなパターンを生成する神経回路網，即ち CPG （*Central Pattern Generator*) が存在することが明らかになってきた [37][53]．CPG は，中枢神経系に存在する神経細胞のグループであり，相互作用を行ないながら調和的なパターンの信号を生成して，筋肉を順次刺激することによりリズミカルな運動を生成する．CPG は各肢毎に存在し，それぞれの筋肉の収縮を制御すると考えられている．

過去の生体工学の研究では [54][42][43]，CPG は歩行運動や遊泳運動などのリズム運動に必要な振動パターンを生成するための神経振動子群としてモデル化されてきた．ここで，神経振動子は相互抑制で結合した 2 個の神経細胞の結合として構成されている．この神経振動子の重要な特徴は，相互引き込みと呼ばれる現象が生じることである．つまり，相互に結合された神経振動子の自己興奮振動は，ある範囲内の振動入力に対して一定の周期と振幅を保ちながら振動することが知られている．

神経振動子を用いた運動生成に関する過去の研究では，湯浅ら [4] は位相差のダイナミクスを導き，運動パターンを表すポテンシャル関数を与えて，エネルギー消費との関係で四足歩行の多様な歩行パターンを実現している．また，多賀ら [102] は，神経振動子系と，生体の筋骨格系とのリズムの引き込み現象から，柔軟な 2 足歩行運動を自己組織化的に生成する方法を提案している．これらのシステムにおいては，望ましい運動を実現するために専門家による地道な試行錯誤によって神経振動子ネットワークの詳細が設計されてきた．

一方，近年の計算機の発展により，相当量の計算機資源を必要とする手法を現実の適応最適化問題へ適用することが可能になってきている．それらの手法の 1 つに，生物の進化のメカニズムをモデル化した進化的計算手法 (Evlutionary Computation) が

ある. 進化的計算手法はその高いロバスト性などの理由から, 多様な分野に応用され盛んに研究されてきた. その結果, 進化的計算手法は回路の設計 [80] からスケジューリング [89] まで様々な最適化問題のための計算手法として, 大きな成功を収めている. 特に, ロボットの制御器をニューラルネットワークなどの低次レベルの記述法を用い, 進化的計算手法を通して構築する進化ロボティクス分野が注目を集めつつある. [29][34]. この分野では, ロボットの身体性を特に意識しなくとも制御器の構築が行なえるなどの利点があり, 新しい設計論を提供することが期待されている.

本論文では進化的計算手法の一つである遺伝的プログラミングを用い, 自律的な 2 足歩行運動を生成するための神経系の自動生成の方法について述べる. また, 遺伝的プログラミングの効率を向上させるための適応的ターミナル選択方法についても述べる.

1.1.1 2 足歩行運動生成

歩く, 走る, 泳ぐ, 跳ぶ, 飛ぶといった運動は, 一般的に移動運動と呼ばれ, 動物にとって最も重要で, かつ基本的な機能である. 移動運動は自分自身を維持し, 推進する肢を周期的に運動させることによって行われ, その基本は各肢のリズム運動と肢間の協調機構にある. これまでに様々な観点からの研究が行なわれてきているが, それらは大きく二通りのアプローチに分けられる.

一つは, 各肢が体幹をどのように推進し, そのとき関節, 筋などの器官がどのように働くかを力学的観点から明らかにしようとするアプローチである. このアプローチにおける制御の標準的な方法は, まず各関節の運動の軌道計画をすることである. そして, それらを安定に実現するようにフィードフォワード系またはフィードバック系を設計する.

さらに, 床と脚との接触点まわりの自由度は直接制御することができないために, 床反力に関するセンサ情報を利用して, 床反力作用点中心の軌道をモニタし, 系全体の安定性を確保するために運動軌道の一部をリアルタイムで変更する. これらのメカニズムがあれば, 原理的には安定な歩行が可能であり, 早稲田大学をはじめとする多くの日

本のグループが, 先駆的な歩行ロボットの開発を行なってきた.

また, 制御方法は公表されていないが, 最近発表された本田技研の人間型 2 足歩行ロボットの卓越したパフォーマンスの実現は画期的な成果である. しかし, ここでの根本的な問題点は, 歩き方を変えたときや環境が変ったときに, そのたびに運動軌道から決め直さなければならないことである. もちろん, 想定されるあらゆる場合について, 運動を記憶させておけばよいかもしれない. しかし, そのような方法だけでは制御が破綻するのは容易に想像できる.

もう一方は, 移動運動を支える神経系, および制御回路がどのような構成になっているか, また, どのようなメカニズムで働いているかを明らかにしようとするアプローチがある. 近年, 神経系の自律的な振舞いに着目し, 運動を生成する試みが行なわれている. これは神経系のリズム発生機構を非線形振動理論を基礎としてモデル化し, 歩行運動の生成を行なうものである. 非線形振動子による運動パターン形成の理論的な解析は, 動物の歩行運動で多くなされている.

Collins と Richmond [42] は, 四足歩行のパターン遷移を, 固定結合した発振器モデルにおけるパラメータ変化で実現させ, そのときの発振器間の結合の性質が発振器の種類によらないことを示した. Bay ら [28] は, 非線形振動の基本式である Van der Pol の方程式を用いて 2 足歩行運動パターンを生成している. また湯浅と伊藤 [4] は, 発振器の位相差空間でのポテンシャル関数により発振器間の位相差を任意の値に制御できることを示し, これを遊泳パターンと四足歩行パターンの形成に応用した.

また, 木村ら [76] は, 独自の発振器で昆虫の非線形発振器モデルを構成し, エネルギー効率により歩行パターンを生成した. しかし, これらの研究は神経振動レベルのパターン生成にとどまり, 筋骨格系という力学システムとの相互作用が考慮されていない. 力学システムを取り入れたものとして, 伊藤ら [74] は歩行パターンを, 消費されるエネルギー評価に基づく発振器を用いてモデル化し, 四足歩行を実現したが, 筋骨格系を考慮したリズム運動ダイナミクスおよび適応ダイナミクスを取り入れてはいない.

多賀 [102, 103, 104, 105], は人間の 2 足歩行における筋骨格系力学モデルと神経振動

子モデルを統合した解析を行なった. そこで, 身体力学系が持つ振り子的な振動リズムと, 神経系が持つリズムとの相互引き込みにより, 自律的な歩行運動を発生するモデルを提案している. ここで, 歩行運動を司る神経系ネットワーク構造は, 神経生理学的に必ずしも明らかにされてはおらず, そのモデル化には人為的な試行錯誤が必要であった. また長谷ら [68] は、神経系のパラメータの微小な修正を遺伝的アルゴリズムを用いて行なってきたが, 適当な（妥当な）神経系の構造を予めに作っておく必要があり, そのパラメータの調整だけでは生成できる運動の種類（バリエーション）も限られてしまう.

以上のような歩行のシミュレーション手法は工学的な手法から, より生体の運動発生機構に基づいたモデルへと発展してきており, これらの点を踏まえたモデルの構築を行なう必要があると考えられる.

1.1.2 生体システムの特徴

多くの生体制御系を考えてみると, 工学システムにおけるサーボ機構のように, 入力の変化に追従する追値制御系と, 恒常性と呼ばれる一定値を保持するプロセス制御系のような定値制御系とがあり, これを支配する負フィードバックの概念は細胞レベルのものから, 生体の全器官に及ぶものまで, そのあらゆる部分にわたってみられる.

このような生体制御系は, 生物の長い年月にわたる進化の過程を経て到達したものであって, 生物の生存に好都合なものが残ってきたと考えられる.

これに対して工学制御系は, 人間が特定の目的をもって構成したものであって, 技術的ならびに経済的にその設計と運転を容易にするような要素を組合せて構成されている. したがって, 同じように負フィードバックの概念によって支配されているので, 両者が非常によく似た面を多分に持っているのといえるが, その構成の理念が根本的に違うので, 生体制御系は, 工学制御系と多くの点で本質的に異なった特色をもっている.

これらは生体システムの数学モデルを構成する上で, 重要な指針を与えてくれる. 以下でその主なものを示す.

- 最適性と適応性

  生体では最適と適応はどのような場合にもごく自然に行なわれている．これを生体におけるエネルギー消費と情報伝達の面から考えてみると，たとえば人間の自然状態での定速歩行は必ずしも消費エネルギーが最小になる条件でなされているという確証はないが，定速歩行時の毎分歩数は消費エネルギーが最小，あるいはそれに近い値に選ばれているといわれている．また，呼吸する時に，ガス交換効率が最大となり，血液中炭素ガス濃度が保たれるように換気量と心拍出量の調節が協調的に行なわれていると考えられる．

- 冗長性

  工学的な制御系をはじめ，一般に人工のシステムではその経済性が問題になるので，要求される性能を満たす必要最小限の設計が目標とされる．したがって人工システムの各要素は，それぞれ全体のシステムに対して不可欠の役割を果たすことになり，要素の故障は全体システムの運転停止につながることになる．そこでシステムの信頼性をあげるためには多重並列構造にすることが必要になってくる．しかし，生体は元来経済性を目的として構成されていたシステムではなく，環境の中で生存するのに最も好都合なようにつくられ変容してきている．そのため，その生存にとって重要なものほど多重構造であり，信頼度が高くなっている．すなわち，冗長性が大きい構造になっているわけである．

- 非線形性とむだ時間

  生体システムには，入力と出力の間に非線形性とむだ時間が現れることが非常に多い．この特性は工学システムにおいても現れるものだが，生体システムに独特のものが多い．

1.1.3 進化型計算手法

工学問題の多くは最適解の探索に帰着される．従来の最適探索法は，現状より目的関数を改善する方向で次の探索点を求めることを繰り返す，局所的探索に基礎をおいている．しかし，この方法では，多峰性関数の最適化や組合せ最適化問題で，最適点に達することが困難なこと，また既存の探索点の情報を有効利用していないことなどの欠点が指摘されている．近年，情報科学の分野では，生物の進化過程ヒントにした探索手法である進化的計算手法が注目を集めている．

進化的計算手法は，生物の進化のメカニズムをまねてデータを変形，合成，選択する手法である．この方法により，最適化問題の解法や，有益な構造の生成を目指す．進化的計算手法の大きな利点の一つは，「答えの解き方が分からない」あるいは「構造が明確でない」という問題に対して威力を発揮することである．進化的計算の枠組みには，大きく分けて，遺伝的アルゴリズム (Genetic Algorithm, GA)，遺伝的プログラミング (Genetic Programming, GP)，進化戦略 (Evolutionary Strategy, ES)，進化的プログラミング (Evolutionary Programming, EP) などがある．以下，これらについて簡単に説明する．

遺伝的アルゴリズムは進化的計算モデルの代表的な手法で，J. Holland が中心となって研究を進めてきた．遺伝的アルゴリズムは，生物の適応・進化過程を抽象化した概念的な枠組みであり，選択，交叉および突然変異と呼ばれる遺伝的操作の適用に基づく世代交代の繰り返しにより，ある問題に関するその解の優秀性を環境への適応度と読み換えることによって，組み合わせ最適化問題を解く手法である．遺伝的アルゴリズムのデータ構造は，GTYPE(遺伝子型) と PTYPE(表現型) の 2 層構造からなる．これらを適切に設計することで，遺伝的アルゴリズムは様々な分野に応用可能である．

遺伝的プログラミングは，遺伝的アルゴリズムの遺伝子に木構造的表現が扱えるように拡張したもので，John Koza によって提唱された．遺伝的プログラミングでは遺伝的アルゴリズムで行なわれる交叉や突然変異のような遺伝子操作を，同様に全て木構造に対して行う．遺伝的プログラミングの処理手順は遺伝的アルゴリズムと同様であ

る．しかし遺伝的アルゴリズムとの違いは，遺伝的プログラミングでは，進化の機構を用いてプログラムを自動生成することを目的としていることである．つまり，この方法を用いると，目的のプログラムの内部状態や処理手順について，明示的に設計することなくプログラムを生成することが可能である．この遺伝的プログラミングについて，次章でもっと詳しく述べる．

進化的プログラミングは，有限状態オートマトンの適応的学習を中心とした進化論的なプログラミングを実現しようとする研究である．最近では，実数値表現を用いた関数最適化などに拡張され，次の進化戦略とよく似た手法に発展した．個体集合と選択を用いることについては，遺伝的アルゴリズムと共通しているが，新しい探索点の生成には専ら突然変異を用いる．

進化戦略法は，ドイツの I. Rechenberg, H. P. Schwefel らが中心となった研究を進めてきた手法である．個体表現として，直接実数値を扱い，実際の適用対象に対する実験によるパラメータ調節などの手法として開発された．初めは，単一の親個体と，それに突然変異を加えた子個体の 2 つ個体を扱っていたが，後に多数の個体を扱う手法へと拡張された．進化的プログラミングと同様，個体集合の選択と突然変異を主とする探索手法である．

これらの手法の共通的な特徴は，ある構造を持った個体が多数で解析や探索を行うことにある．さらに，解析や探索の過程では，これら複数の個体は，交叉，突然変異といった操作によって構造を変化させる．各個体は，それぞれ適合度と呼ばれる値を持ち，この値によって確率的に選択が行なわれる．これを繰り返し行なうことによって，最適解が探索させる．

進化的計算手法では，多点により繰り返し計算を行うために計算コストが莫大となるといった問題点はあるが，最近はこの問題を解決するために，並列計算機上で進化的な計算を実現する試みが盛んに行なわれている．

最近では，歩行運動生成研究において，神経系と身体ならびに環境との相互作用ダイナミクスが果たす役割の重要性が認識されつつある．それらを考慮して，制御器を効

率的に構築するためのアプローチの一つとして，遺伝的アルゴリズムなどの進化的計算手法を用いて制御器を構築する手法が注目を集めている．特に，2 足歩行運動を実現するために必要となる神経系，即ち神経振動子を多く結合させたネットワークの結合係数を最適化するために，De Garis [52]，長谷ら [68] は遺伝的アルゴリズムを適用して，2 足歩行を評価することで神経振動子の結合係数を最適化する手法を提案した．

1.2 本研究の目的

歩行運動の研究は，その障害の原因や基本的なメカニズムに対する解析的な研究から，機能回復の予測などの合成的研究に移りつつある．このための運動発生手法として従来から最適化手法による制御工学的な軌道生成が行なわれてきたが，最近では身体力学系のリズムと神経系のリズム間の非線形引き込み現象を利用した自律的な運動発生手法が提案されている [102, 103, 104, 68]．

しかしながら，歩行運動を司る神経系ネットワーク構造は，神経生理学的に必ずしも明らかにはされてはおらず，そのモデル化には人為的な試行錯誤が必要であった．また，神経系のパラメータの微小な修正は，遺伝的アルゴリズムを用いて調整を行なってきたが，適当な (妥当な) 神経系の構造を予め作っておく必要があった．パラメータの調整だけでは生成できる運動の種類（バリエーション）も限られてしまう．例えば，歩行以外の運動を生成したい場合，試行錯誤で神経系の構造を作る直す必要がある．これには非常に大きな労力（経験と勘）が必要である．つまり，遺伝的アルゴリズムでは (1) あらかじめ神経系の構造を試行錯誤で決めておく必要がある．これには非常に労力がかかるし，必ずしも妥当な神経系構造が定められるとは限らない．(2) 調整できるのはパラメータだけなので，生成できる運動の種類，パターンは限られてしまうという問題がある．

遺伝的プログラミングの研究では，その初期段階から神経回路網の進化に関して，Koza や Yao ら [80, 113] により興味深い結果が得られてきた．また，進化的ロボティクスの

研究では, 遺伝的アルゴリズムを用いた多脚ロボット (4 脚～8 脚) の行動制御に, リカレントニューラルネットワークを用いた手法が提案され [60], 専門家による設計よりも優れた性能の制御器を得ることに成功したものもある. しかし, 転倒する心配が少なく, 歩き出し（立ちどまり）が簡単にできる多脚の歩行運動に比べ, 2 足歩行は不安定なリミットサイクルを持つため, 非常に転倒しやすい. そのため遺伝的プログラミングを用いた人間が介入しない自律的な 2 足歩行制御の生成に関する研究はまだ報告されていない.

本研究では, 人の介在なしに, 2 足歩行運動を生成するための神経回路網を自動的に構成することを研究目的とし, 計算機によるモデル化の手法や実験手法を研究について述べる. 現状では, 神経回路網に関する生物学的な研究はまだ十分でなく, また工学的な応用を考えると未知パラメータが多く, 望ましい運動を生成するための神経系の構造を定める体系的な方法は必ずしも明らかになってはいない. そこで, 本研究では遺伝的プログラミング (GP) を適用して, 進化的なアプローチにより神経回路網の構造を最適化する手法を提案する.

1.3 本研究の構成

前述のように本研究では, 人の介在なしに, 2 足歩行運動を生成するための神経回路網を自動的に構成することを研究目的とし, 計算機によるモデル化の手法や実験結果について述べる.

まず, 第 2 章では歩行運動を発生する神経系と力学系のモデル化について述べる. 本研究のシミュレーションでは, 身体形態や筋骨格系などを表す力学モデルと, 神経制御系を表す神経振動子モデルを用いたリズム発生回路をそれぞれ構築する. これらのモデルを用いれば非線形引き込み現象により, 自律的な運動発生を行なうことができる.

第 3 章では, 解剖学的に必ずしも明らかではない神経系のネットワーク構造とそのパラメータを求めるための遺伝的プログラミングを用いた探索手法について述べる. こ

の神経系モデルでは, 歩容の獲得のためには, 神経系のネットワーク構造や多くのパラメータ値を調節する必要がある.

第 4 章では, 神経系構造を探索するための遺伝的プログラミングの設計や, 探索効率向上のための, いくつかの工夫について述べる. その後, システムの実現手法について具体的に説明し, これまで行なった実験結果に関して評価を行なう.

第 5 章では, 遺伝的プログラミングの探索効率に多く左右させるノード設計の問題について述べ, その解決方法について述べる.

第 6 章では, 結論として、実験結果により明らかとなった課題について考察し, 今後の研究の方向性について述べる.

第 2 章

神経振動子モデルに基づく 2 足歩行生成

2.1　2 足歩行運動の特徴

歩行運動の特徴は, 脚の動きのパターンが驚くほど形式化されていることである. 色々な動物が歩いたり, 走ったりする様子は非常に複雑に見える. しかし, 脚の接地・離地のみに着目して, 連続写真などで歩行パターンを観察すると, いくつかの固有の歩行パターンを持っていることが分かる. このことから, 神経生理学における歩行運動の制御に関する従来の研究には, 大別すると 2 つの流れに分けられる.

一つは, 反射の連鎖によって運動が生成されると考える反射生理学研究であり, もう一方は脊髄内に想定された歩行リズムの生成中枢の構成を対象とした研究である.

前者は, 歩行運動は四肢の受容器ー反射中枢ー四肢の筋で構成された各々の反射系が一定の順序にしたがって持続的に活性化されることによって発生し, 中枢神経系には各々の反射系を一定の順序で活性化する機構が備わっていると考えるものである. これに対して後者は脊髄に歩行リズムを発生する中枢があり, これが脳から脊髄へ下降する神経系の活動によって活性化されることが歩行運動の基本的な機構であると考え, この機構を担う脊髄の歩行リズム生成中枢の構成について考察するものである.

しかしながら, 現在では両者の概念を融合し, 脊髄の歩行リズム生成するための中枢からの出力が四肢の受容器からの情報を取り入れることで, より協調的な歩行パター

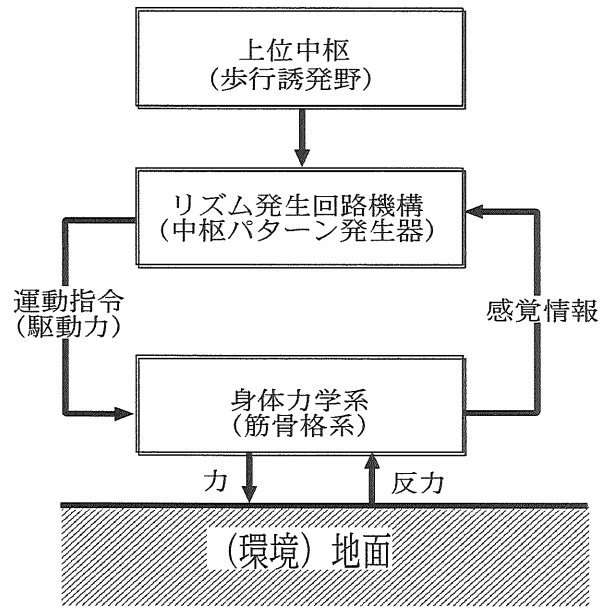

図 2.1: 相互引き込みによる歩行運動の発生モデル

ンを生成していると考えられるている.

たとえば, 多賀 [102, 103, 104] はこれらの生理学的知見に基づき, パターン発生器を用いた 2 足歩行の発生モデルを提案している.

これは図 2.1示すように身体力学系と神経系の振動が相互に引き込むことにより, 自律的に運動を発生するとしたものである. すなわち, 歩行運動は身体力学系と神経系からなる非線形力学系におけるリミットサイクルとして表すことができ, このリミットサイクルでは, ある程度, 初期値の変動や外乱によらず安定な周期解を得ることができると考えられている.

2.2 2 足歩行運動モデル

運動に基づいて身体形態を変化させるシミュレーション手法を構築するには何らかの運動発生メカニズムが必要となる. 従来のシミュレーションでは最適制御理論など主に機械工学的な制御理論により運動の発生が試みられてきたが, 実際の生体の運動

メカニズムとは異なることが問題となってきた。歩行運動の神経支配の構造については神経生理学の分野で盛んに研究が行なわれている.

最近の動物の歩行運動に関する神経生理学的研究の発展から, リズミカルな歩行運動は, CPG(Central pattern generator) と呼ばれる脊髄レベルのリズミックなパターン生成の神経回路網 (神経振動子) の働きにより発生するものであると考えられている.

たとえば, 大脳を切断された猫は, 自らの意志では歩けないのにもかかわらず, 中脳の特定の部位を刺激することにより歩くことが知られているが, これも下位の中枢神経系におけるリズムパターンの自律的な生成によるものであると考えられている.

こうした動物の周期的運動は, リズムパターンを自律的に生成する神経節や, 脊髄の中の神経回路網の周期的な興奮によって引き起こされるものであり, 神経回路網の構造自体にそのパターンが記憶され, 必要な時に生成されていると考えられている.

歩行運動の制御には随意的側面と自動的なとが考えられる. 随意的な制御とは歩行運動の開始や停止, または目的とする方向への姿勢の変換, 障害物回避などのように意識的な運動であり, 自動的な制御とは各部位の周期的な活動や様々な反射などである. ここで自動的というのは, 一定速度の歩行運動際して歩行器官の感覚情報などのフィードバック信号が中枢神経系内に常に受容されているにもか変わらず, その状態は必ずしも歩行者の意識にはのぼらず, 随意的な制御を必要としない, という意味で用いられる. 本研究ではこの歩行運動の自動性に重点を置いて考える.

2 足歩行運動のシミュレーションは, 歩行ロボットの分野で多く行なわれているが, 神経系の生理学的な特性など, ヒトの運動発生機構をより忠実に模擬したモデルもいくつか提案されている.

多賀ら [102] は, その考えを人間の 2 足歩行シミュレーションに適用し, 神経振動子系と生体の筋骨格系とのリズムの相互引き込み現象から, 柔軟な 2 足歩行運動を生成する方法を示した. 神経系は, 神経刺激により自律的にリズムパターンを生成し, 身体力学系はそのリズムパターンに同期しながらリズミカルな歩行運動を生成する. 足の接続状態や各関節の角度状態などのような生体センサーに関する情報は, 神経系に

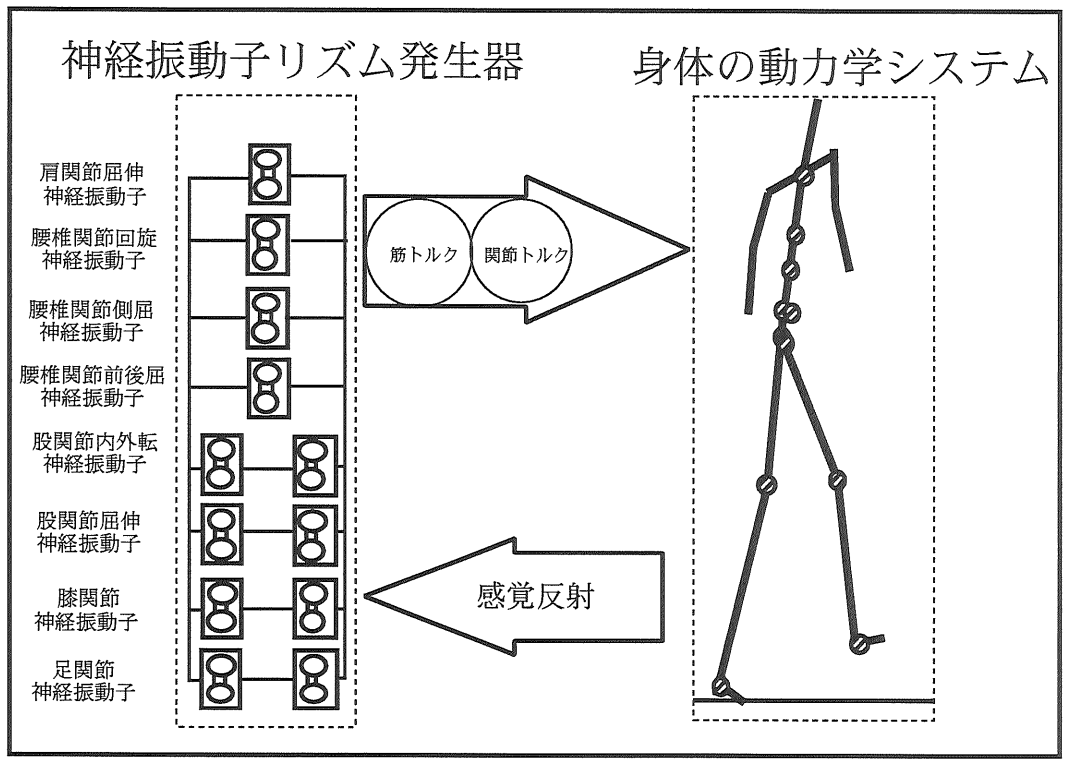

図 2.2: 自律的歩行運動発生モデル

フィードバックされ, 神経刺激のリズムパターンはこれらの情報に基づいて生成される. また, コンピュータシミュレーションによって, この理論に基づいて生成された2足歩行が斜面や歩行中での外乱などの環境の変化に対しても, 柔軟に適応し, 安定した歩行運動が実現できることが示された.

本研究のシミュレーションモデルにおいても, 神経系が発生するリズムパターンと身体力学系が持つ振り子的リズムパターンとが協調することで, 2足歩行運動が発生しているという仮説に基づき, 3次元2歩行運動生成システムを開発した.

図2.2は神経系リズムと, 身体力学系のリズムとの相互作用に基づく自律的な歩行運動発生モデルの概念図である. このような神経系と身体力学系の両者を考慮したモデルを神経筋骨格モデルとよぶ. この神経筋骨格モデルは, 筋骨格系を含んだ身体力学系と神経系からなり, これらの系と環境（地面, 重力）とが相互に協調することで歩行運動が実現できるとしている.

次に各モデルについて述べる. これらのモデルの具体的な運動方程式に関しては付録で述べる.

2.2.1 身体の動力学モデル

まず, 身体の剛体特性や関節自由度を表す剛体リンクモデルを構築する. 歩行運動の基本的なリンクモデルとしては, 下肢のもをモデル化し, 上体をリンクとした2次元モデルが多い.

しかし, 2足歩行体系の進化過程を見ると初期人類と現生人類とでは上肢と下肢との比率が大きく異るなり, 上肢と下肢とのバランスは体系的な特徴の一つであると考えられる. またモデルの精密化という点でも, 上肢のモデルが必要である. さらに歩行における上肢運動を考慮した場合, 上体と腰部とのねじれ運動の役割が重要となってくる.

たとえば, Gracovetsky [62] は歩行運動の基本的推進力は脊椎運動によって生じているとする spinal engine 理論を提唱している.

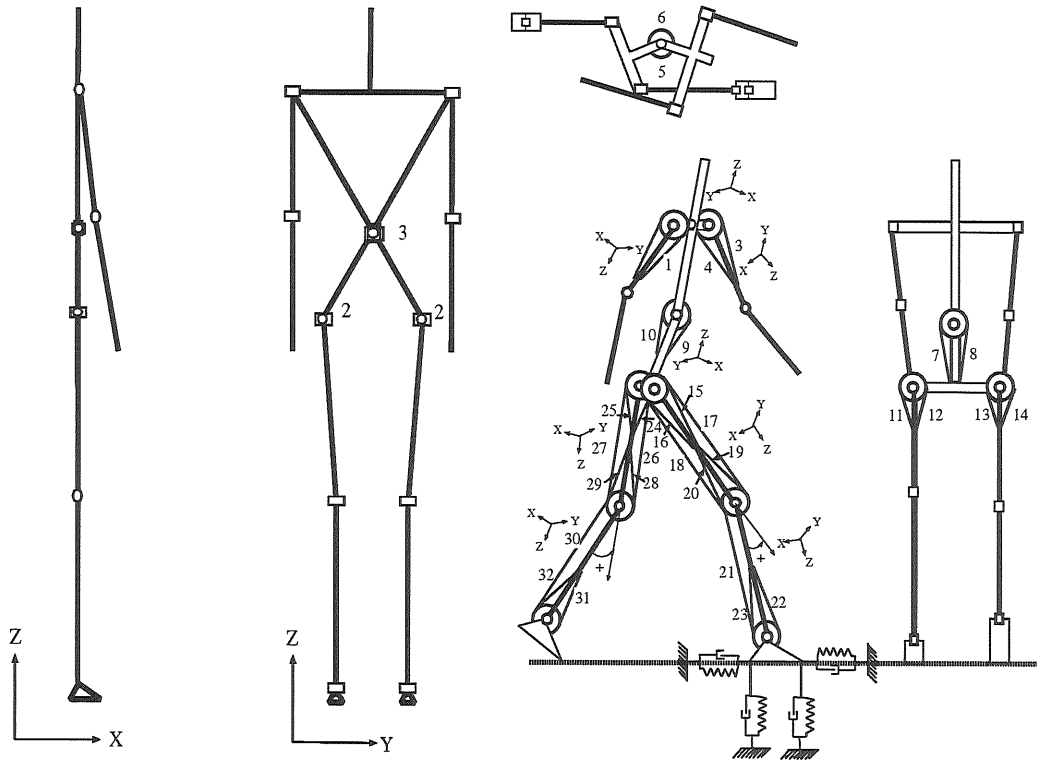

図 2.3: 3 次元の剛体リンクモデルと筋骨格系モデル

そこで本研究では, 上肢節の振り子運動や体幹節の回旋も考慮できるモデルとした. これらの運動を考慮するためには, 3 次元モデルを構築する必要があり, 本研究では従来の歩行モデルと比べて, 実際のヒトの歩行運動に充分に対応可能な 3 次元 2 歩行モデルを構築した. すなわち, このモデルは上肢を含めた 3 次元の身体構造, 筋骨格構造を持ち, 神経生理学的知見に基づいた運動発生機序を有するものを想定した.

本モデルの概要を図 2.3に示す.

- 矢上面モデル :

  図 2.3で示すように, 2 足歩行に関する基本的な身体形態を, 2 足歩行の身体力学系の剛体リンクモデルとして足, 下腿, 大腿, 腰部, 頭を含めた胸部, 上腕, 前腕, 左右合計 12 関節からなる 3 次元剛体リンクでモデル化した. このモデルは, 人間の 2 足歩行運動の特徴を表すための最も単純で基本的なモデルである.

  水平面内における体幹節の回旋を表すため, 股関節および肩関節は, 回転の自由

度の他に水平方向の並進の自由度を持つ．すなわち，屈伸と内外転の 2 自由度を持つ．また，この股関節，肩関節の移動量は水平面モデルの股関節，肩関節の移動量により表される．腰椎関節は前後屈，側屈，回旋の 3 自由度を有し，その他の関節は屈伸の 1 自由度を持つとした．

- 水平面モデル：
  体幹節の回旋運動を表すため，腰部と胸部の 2 節よりなる水平面 3 次元剛体リンクモデルを構築した．図 2.3では四肢もしめしてあるが，リンクとしては存在しない．この水平面モデルにおける肩関節と股関節には矢状面モデルの肩関節，股関節の水平方向反力が加わる．また水平面内の各節の重心位置や慣性モーメントの値は，矢状面モデルの体幹節の仰角に応じて変更される．

関節受動要素

矢状面内の各関節には，軟部組織による非線形粘弾性モーメントが作用するものとした．また水平面モデルには，線形に粘弾性モーメントが作用するとした．さらに床面とのねじれ抵抗を表すため，水平面モデルの腰部と絶対座標系との間にも粘弾性モーメントを作用させた．

これは本モデルでは足部と床面間に生じる床面鉛直軸回りのねじれモーメントを考慮していないため，シミュレーションにおいて，この粘弾性要素が存在しないと水平面モデルの向きが定まらず，体幹が横を向くようになってしまうためである．

床面モデル

歩行には左右の脚が床面に同時に接地している両脚支持期が存在する．この両脚支持の状態では，脚と床面とで閉ループ構造となり，内部不静定問題となる．閉ループ構造では運動の自由度に対して，拘束条件が存在し，通常のアルゴリズムにより運動方程式を解くことは不可能となる．そのためラグランジュ乗数を用いる方法 [2] や，ダランベールの原理と用いた方法 [1] などが提案されている．本研究では，多賀ら [102] や

Raibert [97] のモデルでも利用されているように粘弾性体により床面を表すことにした. この粘弾性体による床反力は踵2点, 爪先2点の計4点に作用するものとした。

座標系の定義

以上で述べた身体の剛体リンクモデルについての運動方程式を構築するため, リンク座標系をそれぞれ 2.3のように定めた. また, 空間の絶対座標系を図 2.3のように定め, x軸方向に向かって歩行するものとした. また, 水平面方向にy軸を, 鉛直上向きにz軸をとった.

各関節のリンク座標系の原点は近位の関節点上に定め, リンクの長軸方向にz軸を定めた. そのため, 立位姿勢では体幹節のz軸は上向きになるが, 四肢のリンクのz軸は下向きになる. また, 各リンクのリンク番号や, 関節の関節角度についても図に示すように定めた. リンク番号は腰部節を基準のリンクとし, これより末端節に向かってリンク番号が大きくなるように定めた. これは付録の運動方程式の順動力学問題の解法の計算順序を定めたものである. また角度は右回りを正にとり, リンクの絶対角度は鉛直軸とリンク座標系のz軸とのなる角で定義した. 関節角度は図のように解剖学的な屈曲伸展の定義によらず, 遠位節のリンク角度から近位節のリンク角度を差し引くことによって得られる角度により定義した.

2.2.2 筋骨格モデル

身体の解剖学的な筋走行の状態を表すモデルとして図 2.3に示すように全身３２の筋肉からなる筋骨格モデルを構築した. 図中の数字は各筋の番号を示す.

下肢については大腿直筋, ハムストリングスなどの二つ関節筋も考慮している. 肘関節を駆動する筋は省略してあり, 肘関節は受動的な動きのみを行なう. 股関節など2自由度以上の自由度を持つ関節について, 筋はそれぞれの関節自由度ごとに存在するものとした. すなわち, 中殿筋, 内転筋は股関節の内外転運動のみに関与する.

また, 腰椎関節については, 前後屈運動は腹直筋と脊柱起立筋肉, 側屈は外腹斜筋, 内

回旋運動は内腹斜筋により, それぞれ独立に行なわれるものとした. また, モデルの簡単化のため, 筋のモーメントアームは関節回転軸に対して垂直で, 関節角度によらず一定とした. すなわち, 筋の走行状態は滑車に巻き付く系のようにモデル化した. 各筋は神経系からの刺激によって力を発揮し, 剛体リンク系で表された身体モデルを駆動する. 筋肉の力学特性としては長さと力関係式および速度と力関係式を考慮した.

2.2.3 神経系モデル

神経系となるパターン発生器は, 比較的単純な運動を発生する運動のサブシステムで, その構成も比較的単純な神経回路から成り立っている. また, 動物にみられる体構造の多様性, 移動運動パターンの多様性とは反対に, 周期的な運動出力を発生する神経回路には基本的には数種類しかないと考えられる. しかし, リズムを発生するパターン発生器および相互作用の実体は必ずしも明らかになっているわけではい.

本研究では, 神経系のモデルとして多賀らによる歩行運動の制御機構モデルを参考にして, 神経振動子ネットワークのモデルを構築することにした. ネットワークを構成する神経振動子は, 神経系の基本的リズム発生機構（CPG）をモデル化したものであり, 周期的なリズムを生成するパターン発生器は身体の各部位ごとに存在することに考えた. ここで, 一つの神経振動子は相互引き込み神経ユニット結合として構成されている. この神経振動子の重要な特徴は振動子を結合すると互いの振動数をそろえる相互引き込みと呼ばれる現象が生じることである. つまり, 神経振動子の自己興奮振動は振動入力に対してある振動範囲で一定の周期と振幅を保ちながら振動する.

神経振動子の個々の神経細胞のダイナミクスは次式のように表される.

$$\tau_i \dot{u}_i = -u_i - \beta v_i + u_0 - w_{ij} y_j + F_i, \tag{2.1}$$

$$\tau_i' \dot{v}_i = -v_i + y_i, \tag{2.2}$$

$$y_i = \max(u_i, 0) \tag{2.3}$$

ここで, u_i は第 i 神経細胞の内部状態, v_i は自己抑制を表す疲労度, y_i は神経からの

出力, u_0 は定常入力信号, τ_i と τ_i' は時定数, w_{ij} は神経振動子内のニューロン同士の接続関係, β は疲労定数であり, また, F_i は第 i 神経細胞の感覚受容器および他の神経振動子からの出力信号の関数である. 本研究では, これをフィードバックネットワークモジュールとよぶ. フィードバックネットワークモジュールは, 各関節の神経振動子出力 ($y_1 \cdots y_{24}$), 節絶対角度, 節絶対角速度, 脚の接地感覚情報から構成される関数である. ここで, 各神経振動子は, 上式で表される二つの神経細胞が互い抑制し合う組より構成され, 1 つの関節において, それぞれの神経細胞が内部状態に比例する出力により屈筋と伸筋を駆動する. このとき, 神経振動子と筋骨格系の間に相互に引き込まれる現像が生じた時には, 一定の周期と位相で振動を始める.

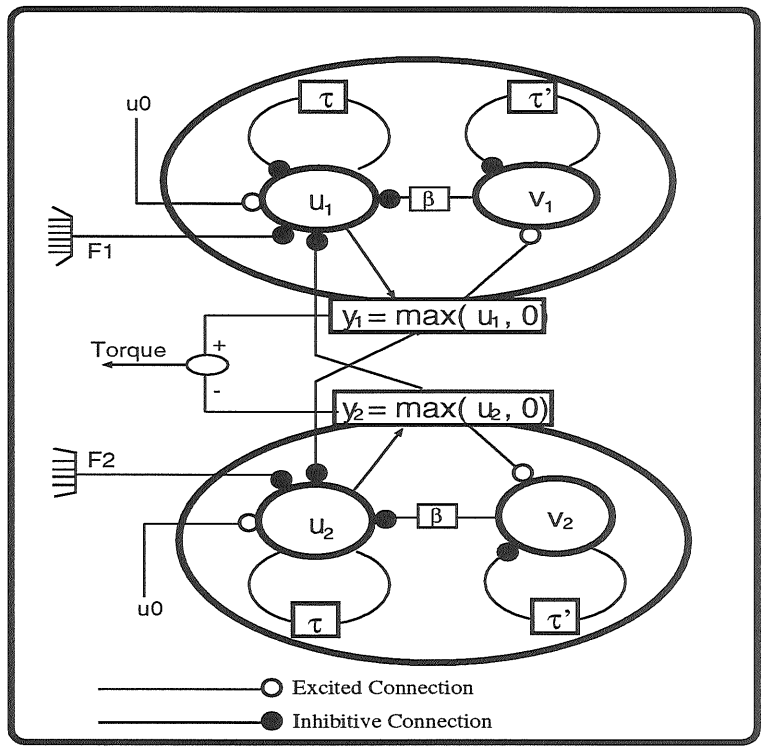

図 2.4: 神経振動子

図 2.4は相互抑制回路を持った神経振動子のモデルを示している. 本研究では, 神経回路網は以下の仮定に基づき構築されるものとする.

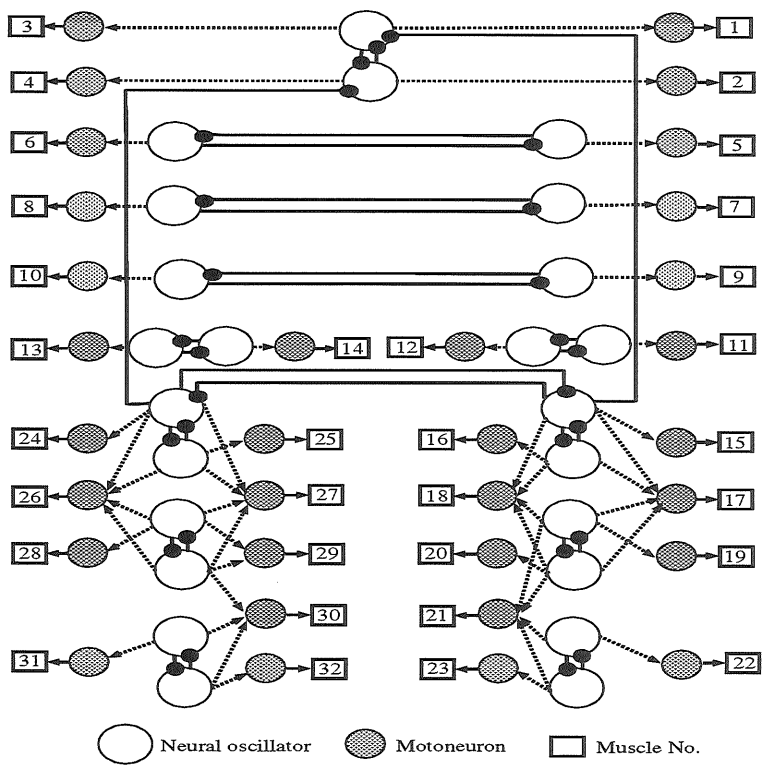

図 2.5: 神経振動子による神経系の例

1. 各関節自由度ごとに屈曲と伸展の一組の神経細胞からなる神経振動子の存在を仮定する．神経振動子とは神経系のリズム発生機構をモデル化したものであり，図 2. 3 のような構造を持ち，上式 (2.1〜 2.3) の微分方程式により表わされる．

2. 感覚受容器によりセンシングされる力学情報として，節絶対角度，節絶対角速度，および接地情報などの情報を想定する．

3. 感覚受容器からの出力がフィードバックされる場合，屈曲側と伸展側それぞれの神経細胞に同一の信号情報が入力される．ただし，その結合方法は片方が抑制（もしくは興奮）の結合の場合，もう片方はその逆の興奮（もしくは抑制）結合となる．また，左右脚の交互運動を実現するため，神経系の構造は左右対称とする．

図 2.5は，上の仮定に基づく神経系の例である．本研究では，図 2.5で示すように，神経系モデルは身体モデルの 12 関節に相当する 12 個の神経振動子 (24 神経細胞) で構成されている．

上のモデルに基づいて神経系を構築する際に最も難しい問題は，感覚受容器や他の神経振動子からの出力信号と神経系との結合関係を表すフィードバックネットワークを定式化することである (式の 2.1)．1 対の神経振動子の構造は図 2.5のように神経細胞の相互抑制結合で結合されているが，他の神経振動子や感覚受容器との結合関係はまだ神経生理学的に必ずしも明らかになっていない．

したがって，従来の 2 足歩行に関する研究では，人間の歩行パターンに近いリズム運動を生成する人工的な神経振動子を開発するために，フィードバックネットワークモジュールは研究者と工学者によって運動力学解析の結果を基に試行錯誤によって決定された．次章で 2 足歩行運動を自律的に発生するための神経系の構造やパラメータの自動決定手法について述べる．

第 3 章

遺伝的プログラミングによる神経系構造の探索手法

動物における歩行のような基本的な運動は, 一定運動が周期的に, 繰り返す比較的単純な運動であるが, その生成制御メカニズムは不明な点が多い.

その理由として,

(1) CPG が自律的な周期活動するのに本質的に関わっている神経細胞の同定が困難であるために, そのパターン生成機構が明確でないこと,

(2) CPG の振舞いは, 脊髄小脳路, 赤核脊髄路等を通じて脳幹, 小脳, 大脳の制御下にあり, また末梢感覚器による調節も受けていること,

(3) 運動ニューロンの活動により駆動される筋骨格系は非線形であり, かつ床反力を通じて外界と相互作用していること,

などがあげられる. (3) に関して, 同様な非線形性を持つ多関節マニピュレータの制御はロボット工学において重要なチャレンジング問題であるにも関わらず, 生体は容易にその適応的制御を成し遂げている. しかしながら歩行運動を司る神経系ネットワーク構造は神経生理学的に明らかにされてはおらず, そのモデル化には人為的な試行錯誤が必要であった. そこで, 本研究ではこの問題を解決するための方法として, 遺伝的

プログラミングを用いて神経振動子ネットワーク構造を自動生成することを試みた.

3.1 遺伝的プログラミング

遺伝的プログラミング (GP) は, 与えられた問題を解決するための望ましい計算機プログラムを探索する進化型探索アルゴリズムの1つである. 遺伝的プログラミングにおいて, 進化の対象となる個体は可変の長さと形を持つ階層的な計算機プログラムである.

したがって, 遺伝的プログラミングは、プログラムを遺伝子とする集団を保持し, それに対して再生, 交差, 突然変異などの遺伝子操作と選択淘汰を繰り返しながら望ましいプログラムを探索していく手法である.

また, 遺伝的プログラミングは関数ノードとターミナルノードによって組合される構造的な染色体を対象として扱う. すなわち, 関数ノードとターミナルノードの組合せにより, 染色体自体がプログラムとして扱われる.

3.1.1 基本アルゴリズム

各個体は一つの計算機プログラムに対応し, そのデータ構造は木構造である. そのプログラムは, 関数ノードの演算子とターミナルノードの定数または変数から構成されている. 演算子と変数の種類は, 対象とする問題に応じて予め容易しておく. プログラムの出力値は, 関数木の葉から根に向かって計算し, 最終的に根にあたる演算子での計算結果になる. 関数木のサイズを表す指標として, 深さとノード数が用いられる. 深さは, 根から最も遠い葉までの距離であり, ノード数は木を構成する演算子, 変数, 定数の総数である. 例えば, 関数ノードである演算子 $+, -, *, \exp, \sin$ と ターミナルノードである変数 x_1, x_2, x_3, x_4 および定数 0.1 から生成された関数

$f = (x_1, x_2, x_3, x_4) = (((x_1 + x_2) - \exp(x_3)) + ((x_1 + 0.1) * \sin(x4)))$ の関数木を図

3.1に示す.

この関数木の深さ, ノード数はそれぞれ 4, 13 となる. このように遺伝的プログラミングで生成する関数木の構成要素には様々な演算子を用いることが可能になる

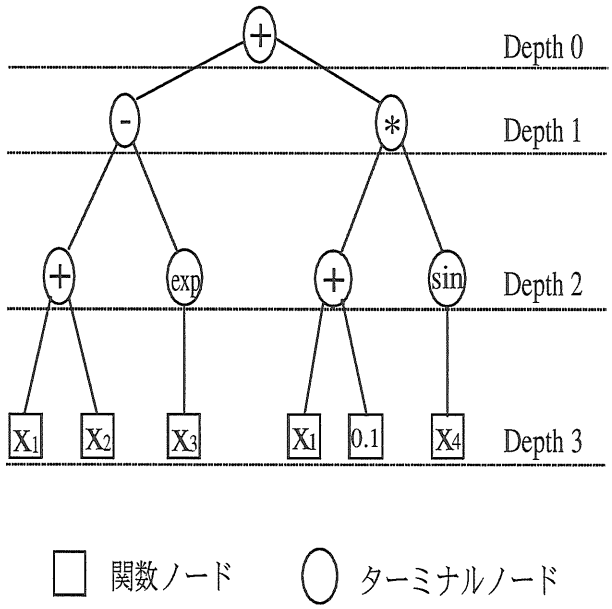

図 3.1: 遺伝的プログラミングの遺伝子

初期世代では遺伝子（木構造）をランダムに生成するのであるが, この生成の仕方には主に次のようなものがある.

- FULL: 木を最大深さまで生成させる (図 3.2)
- GROW: ランダムにあるノードが終端になるかどうかを決定する (図 3.3)
- RAMPED HALF AND HALF: 集団内の個体毎に FULL と GROW をランダムに切替える

遺伝的プログラミングでの基本的な処理流れは図 3.4のとおりである. 最初に初期集団生成を行なう, これはあらかじめ決められた個体数の遺伝子をランダムに生成する操作である. 次に各個体に対して適応度を求める. そして終了条件のチェックでは処理

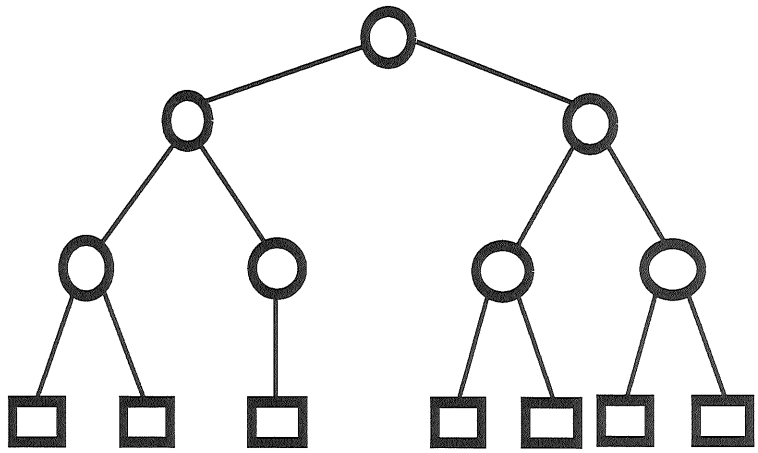

図 3.2: FULL Method

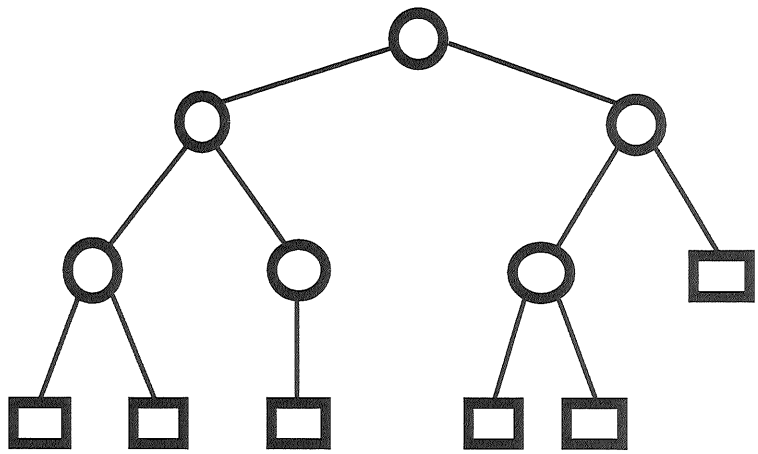

図 3.3: GROW Method

ループの終了判定を行なう．この終了条件には主に最大世代数や解が求まったかどうかが採用される．

選択戦略は次世代に生き残る個体を選び出す操作を指す．交叉, 突然変移, 複製などの遺伝子操作を用いて遺伝子の構造を変化させることにより, より良い解の探索を行なえる．適応度の評価は, 解かせたい問題ごとに設定した評価関数によって行なう．この評価した値によって選択戦略の各個体の次世代に生き残る確率が決まる．

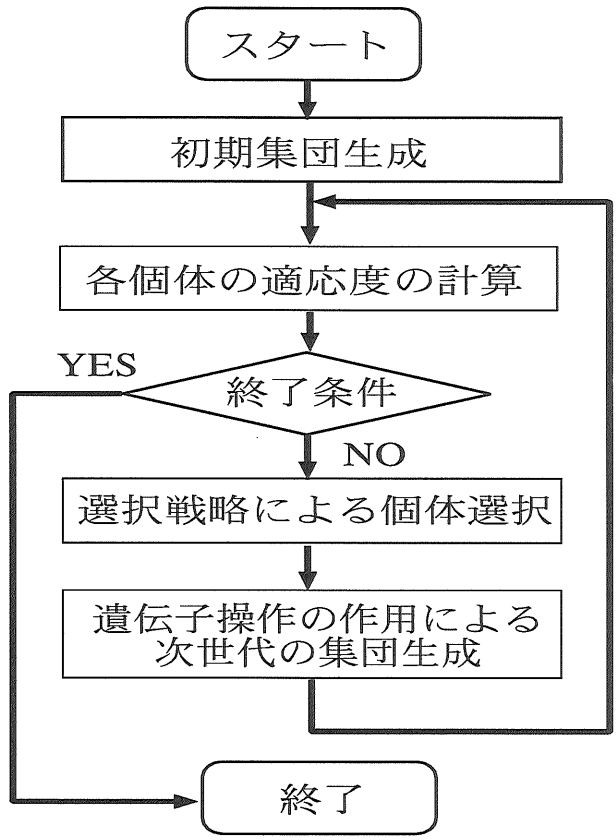

図 3.4: 遺伝的プログラミングの処理の流れ

3.1.2 選択戦略

選択は基本的に適応度の高い個体を次世代に残す戦略である．ここでは, その中で重要な手法について述べる．

ルーレット選択

ルーレッと選択は適応度に比例した重みによって確率的に親を選択する方式で, 適応度比例選択とも言われる. 個体 i(適応度 $f(i)$) が選べる確率 P_i は

$$P_i = \frac{f(i)}{\sum_{j=1}^{n} f(j)}$$

である.

トーナメント選択

トーナメント選択は集団の中から任意の数 T_s(トーナメントサイズ) の個体をランダムに選び, その中で最も適応度のよい個体を選ぶ方式である. たとえば, T_s を 5 とし, 選んだ個体の適応度が表 3.1のようになっているとすると, 選ばれる個体は i_5 である. (ただし, 適応度は大きいほど良いとしている). このトーナメント選択は, 初期収束を避けるために考案した選択戦略である.

個体	i_1	i_2	i_3	i_4	i_5
適応度	0.21	0.41	0.85	0.24	0.16

表 3.1: トーナメント選択の例

エリート戦略

エリート戦略とは, 親世代の集団のうち適応度のよい個体をある割合だけ次世代に積極的に残す方式であり, overselection とも言われる. この戦略は大きな個体集団（Koza は 500 以上の個体を持った集団）に対する探索速度を高めるために用いられる, しかし, エリートの個体が集団内で急速に広まることによって, 局所解に陥る可能性がある.

3.1.3 遺伝子操作

遺伝的プログラミングの遺伝子操作は遺伝的アルゴリズムと同様に交叉・複製・突然変異などがある. 各操作の意味は以下に示す.

複製

複製は, 最も簡単な操作であり, 選ばれた個体をある割合でそのまま次世代に複製する方法である.

交叉

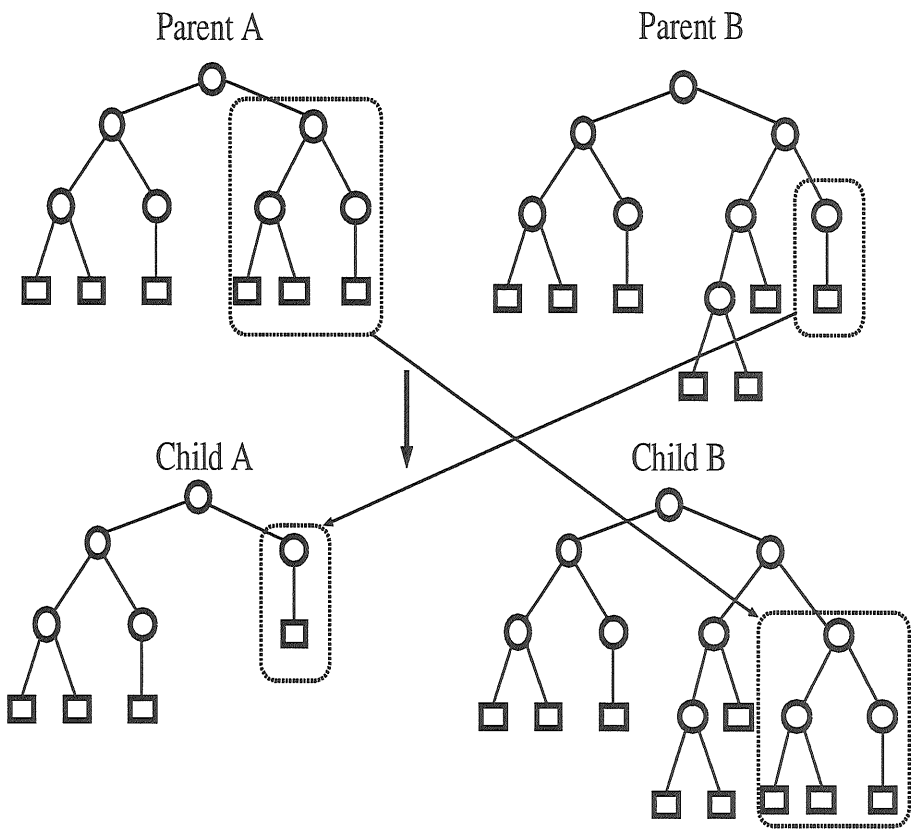

図 3.5: 交叉操作

交叉は, 遺伝的プログラミングにおいて重要な操作である. なぜなら, 遺伝的プログ

ラミングには突然変異, 複製などの操作があるが, 遺伝的プログラミングでの探索は主にこの交叉操作が担っているからである [81].

遺伝的プログラミングでは遺伝子構造が木構造であるため, 交叉は選択戦略で選ばれた個体 A の部分木と, 同様にして選ばれた個体 B の部分木を入れ換える操作となる. また, 交換する部分木は乱数によって選ばれる, たとえば, 図 3.5のように 2 つの個体のそれぞれ破線で囲った部分木に対して交叉を行なうことによって図 3.5のようなそれぞれ全く新しい遺伝子構造を持った個体となる.

突然変異

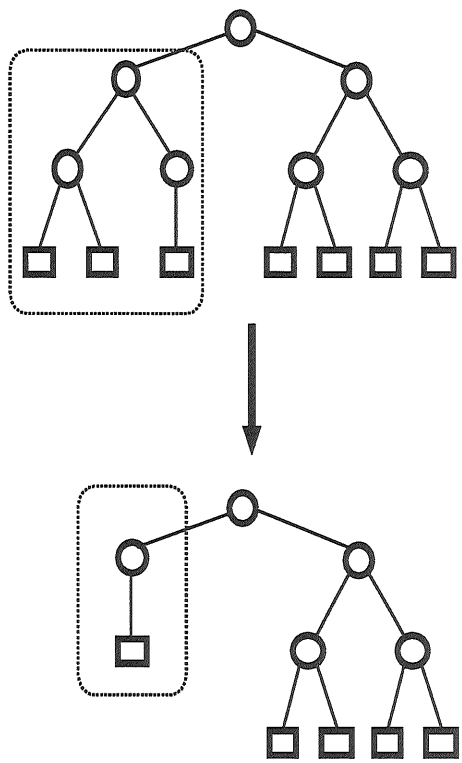

図 3.6: 突然変異

遺伝的プログラミングでの探索を行なう主な遺伝子操作は交叉であるが, 突然変異を用いることにより, 初期時点でランダムに生成された遺伝子による探索空間を広げることができる. 遺伝的プログラミングでの突然変異は, ランダムに生成された (木)

プログラムを遺伝子の部分木と入れ換える操作である. たとえば図 3.6のように破線で囲まれた部分木が, ランダムに生成された新しい部分木によって置き換えられる.

3.1.4 自動的関数定義 (Automatically Defined Function)

Koza らは, 遺伝的プログラミングのノードの設定問題に対し, 解の探索過程において解の探索・関数ノードの生成を同時に行なうシステムとして自動関数定義 (Automatically Defined Function) を提案している [83]. ADF は図 3.7のように関数ノード定義部を染色体内に持つ. この図では本体における ADF ノードの動作が ADF 部に記述されている. ADF 手法では, 遺伝的計算を本体と関数ノード定義部に同時に適用することで, 関数ノード生成と解の大域的探索を同時に行ない, 柔軟な解の探索を行なうことができ, 大規模な問題に対応可能であると考えられる.

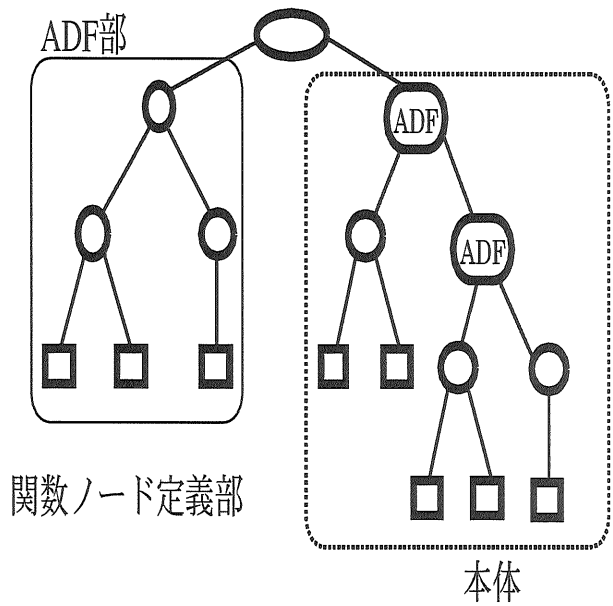

図 3.7: ADF の遺伝子構造

ADF を搭載していない遺伝的プログラミングでは, 木構造に同じ部分木が存在することによって, 重複した木の分だけ余計に木の深さが深まってしまう. ADF を用いることによって遺伝的プログラミングでは部分木であったところが関数名とその引数に

置き換わるため, 木の深さが同じならばADFを用いない場合よりも, ADFを用いた方が表現できる情報量が増える. すなわち, ある問題を解くのに必要なアトムの数がADFを用いた場合の方が少なくて済む.

3.2 神経振動子ネットワークの進化的生成

実世界において自律的な適応を見せる多くの動物の運動は, その体内に張り巡られた神経系によって制御されており, そのメカニズムを解明するために多くの研究が行なわれてきた. そして, 神経生理学などの実験の結果, 動物の歩行は主に脊髄にあるCPGと抹消からの感覚などによって発生する反射の組合せにより生成されていること, また, CPGや反射の機構は主に脊髄に存在し脳幹, 小脳, 大脳など上位中枢からの調節を受けていることは事実として広く受け入れられている.

このようなパターン発生器と反射を組み合わせた制御系により, ロボットを含めた機械系を制御する手法は, 最近注目を集めているNeuro-Mechanical Systemの典型である.

特に, 歩行は神経系, 身体力学系, 環境としての路面などとの動的な相互作用があるという意味で, 興味深い対象となる. ここで, 身体の剛体特性と解剖学的特性に関しては, 過去の研究により既に多くの知見が得られている.

しかしながら, 神経系のフィードバックネットワークの結合関係に関しては未だ充分な研究成果が蓄積されていない. すなわち, 実際に歩行を生成している感覚－運動神経系全体の挙動を総合的に計測することは現時点では困難であり, ある筋の運動指令の発生にどの感覚情報がどの時点で影響しているのかという神経系の具体的な働きは必ずしも明らかになっていない.

したがって, 2足歩行などの運動に関する計算機シミュレーションを実現する際に, 生物学的な見地から最適なフィードバックネットワークの構成をあらかじめ決定するのは不可能と考えられる. 近年, ロボットなどの制御系を進化的計算手法を用いて自律

的に構築する進化的ロボティクスの計算手法の研究が注目を集めつつある [29].

設計者がトップダウン的にコントローラを構築していた従来の手法と比べて, ボトムアップ的に構築する進化ロボティクスでは, ロボットと環境との相互作用ならびにロボットの身体性 (センサやアクチュエータの特徴など) を陽に意識しなくとも, コントローラの構築に反映される設計が可能となるため, 新しい設計論を提供することが期待されている. 本研究でも, 身体力学系の協調制御を行なうための神経系構築における一つの解法として進化的なアプローチを用いた. 即ち, 本研究では神経系構築を神経系のパラメータと構造を同時に求める進化的探索過程として, 遺伝的プログラミングを用いてモデル化を行なった.

3.2.1 フィードバックネットワークモジュール

本研究で用いた神経振動子を相互結合することにより, 相互結合された神経振動子の間には, 引き込み現象が発生し, 互いの振動が同期する. しかし, その同期の際の位相差は結合方法に依存するので, 神経振動子で構成された神経系が適切な機能するためにはそのネットワーク構造を決定する必要がある. このネットワークにおいての構造決定を適切かつ自動的に行なう方法に関する検討・研究は十分ではない.

さらに, パターン発生器を含む脊髄の神経回路は歩行パターンの形成に不可欠であるが, それが歩行運動のすべてではない. 歩行運動は外乱に対して適応しなければならない. 感覚受容器からの感覚フィードバックが, 歩行パターンの推移と歩行の周期の調節に重要な役割を果たしている.

基本的に, 力学系からのフィードバックは感覚受容器からの入力を受けて行なう反射機構を表したものである. 歩行形成にはパターン生成機構のみならず, 反射系の関与が明らかになっている. たとえば, 猫の脳幹の中脳以下を残して, それ以上を取り去った場合でも, 電気刺激の条件を変えずに, トレッドミルのスピードをあげると, 猫の歩容はウォークからトロット, ギャロップまで変化する. この間, 遊脚相の期間は一定で, 支持相がトレッドミルのスピードが増やすにつれて短くなる. 明らかに, これは感覚

フィードバックの寄与である. しかしながら, その詳細な反射経路は明らかではない.

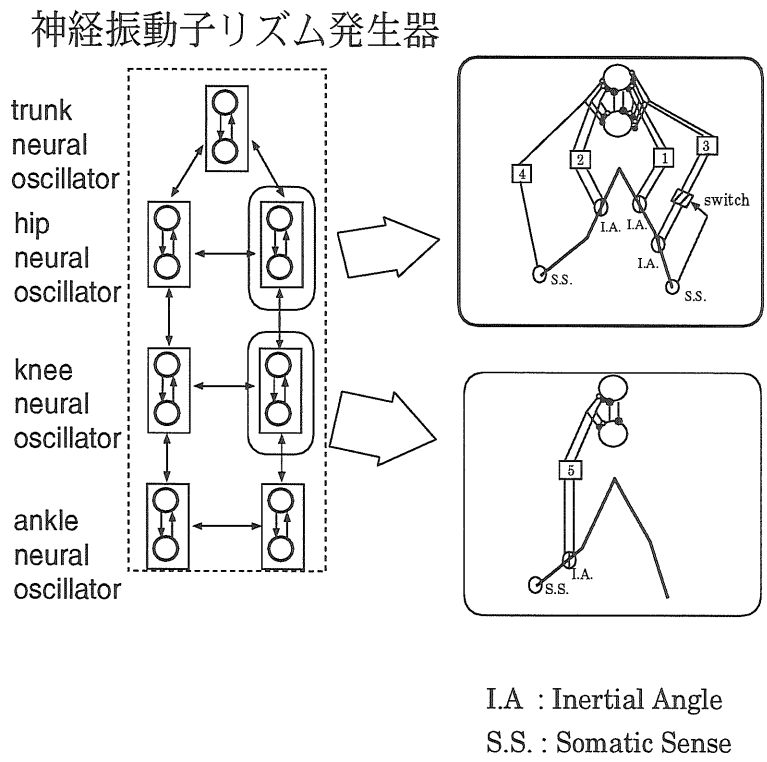

図 3.8: フィードバック経路

たとえば, 図 3.8は, 多賀 [102] が下半身だけの神経振動子と用いた自律的な 2 足歩行生成に用いたフィードバック回路の例である. この図を見みると, 関節ごとに大きく異なるフィードバック回路の構造を持っている. このモデルの神経系の構造は基本的には歩行観察結果に基づき人為的な試行錯誤により決定したと考えられる.

このように, フィードバック回路を含めた神経系に関しては, まだ生理的な知見が十分に得られていないので, 今までは設計者の人為的な試行錯誤により構造決定を行なってきた. しかし, 体形の変化や, 異なる形態のモデルの状況変化に対しては, 新たに人為的な試行錯誤により決定しなければならない. これは非常に時間がかかり, 労力を必要とする.

このような神経振動子ネットワークと感覚フィードバックからなる神経系の構造生

成が自動化されれば, ニューラルネットワークを初めて設計する初期設計問題だけでなく, 体型などの状況変化に応じた設計変更問題も自動的に行なえることになり人間の手を必要としない自律的なシステムの構築が可能となる. 本研究では遺伝的プログラミングを用いて神経系構築を自動化することにした.

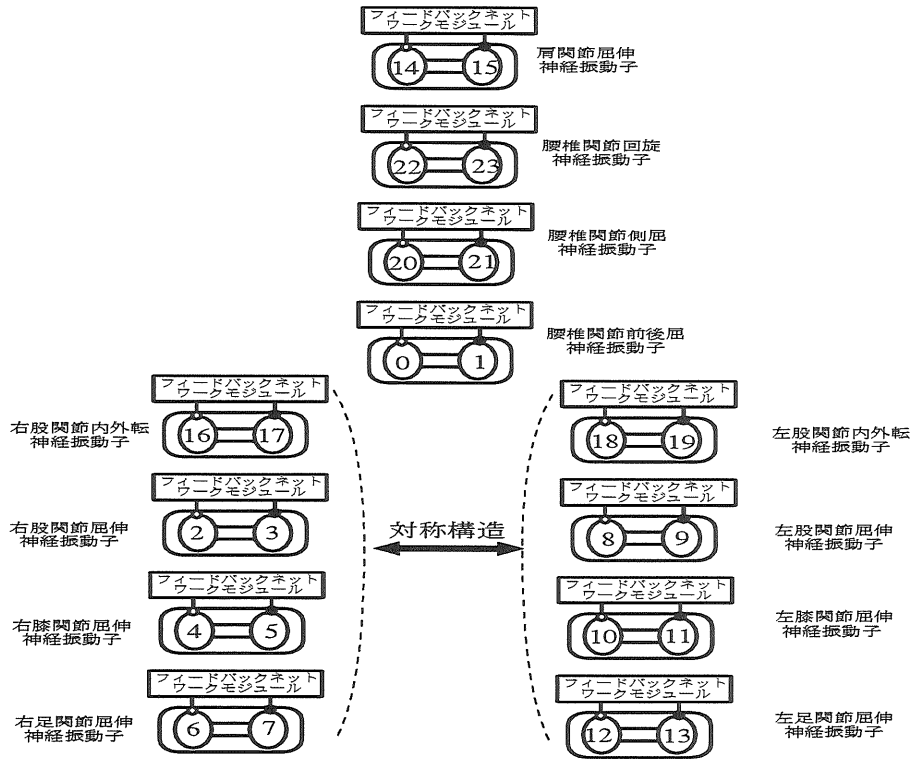

図 3.9: フィードバックネットワークモジュール

本研究では, 神経振動子は相互抑制結合で結合されており, その基本構造は固定とする. すなわち, 各関節を独立的に駆動する基本的なリズム発生機構である神経振動子をあらかじめ持ち合わせているものとし, 神経系と力学系との間の結合関係であるフィードバックネットワークの生成に重点を置いた. 自律的に構築された神経系の評価は神経系と力学系, そして環境とのダイナミクスの相互作用によって生成された歩行運動に対して, 予め定められた評価基準に基づいて行なわれる.

各関節の神経振動子に結合される他の神経神経子からの出力と感覚受容器からの出

力からなる信号情報をフィードバックネットワークモジュールとよぶ．このフィードバックネットワークモジュールは，各関節に屈曲と伸展それぞれが結合された2つのフィードバックネットワークモジュールを持つという仮定に基づいている．また，屈曲側と伸展側それぞれに同一の信号情報が入力される．その結合方法は片方が抑制（もしくは興奮）結合の場合は，もう片方はその逆の興奮（もしくは抑制）結合となる．図 3.9にその概念図を示す．

本研究での神経系は身体における全神経振動子へのフィードバックネットワークモジュールとして，24 個のフィードバックネットワークモジュールから構成されている．24 個の全フィードバックネットワークモジュールをグローバルフィードバックネットワークとよぶ．このグローバルフィードバックネットワークは，神経振動子にある各フィードバックネットワークモジュールを通じて，協調的に他の神経振動子と相互作用を行なうことにより，統一的な働きをする．

本研究では，遺伝的プログラミングを用いて各フィードバックネットワークを進化対象の遺伝子としてモデル化することにより，神経系のグローバルフィードバックネットワークを異種の遺伝子集団の集合として，同時に進化的に生成する手法を提案した．

3.2.2 遺伝的プログラミングによる神経振動子ネットワークの生成

遺伝的プログラミングは，与えられた問題を解決するための望ましい計算機プログラムを探索する進化型探索アルゴリズムの１つである．遺伝的プログラミングおいて，進化の対象となる個体は可変の長さと形を持つ階層的な計算機プログラムである．したがって，遺伝的プログラミングはプログラムを遺伝子とする集団を保持し，それに対して再生，交差，突然変異などの遺伝子操作と選択淘汰を繰り返しながら望ましいプログラムを探索していく手法である．

遺伝的プログラミングの実現は，初期集団の生成から始まる．そして各世代ごとに，神経系と力学系の微分方程式を数値的に解くことにより，歩行運動を発生させる．発生した2足歩行運動に対して，あらかじめ定められた評価基準に基づき歩行運動パター

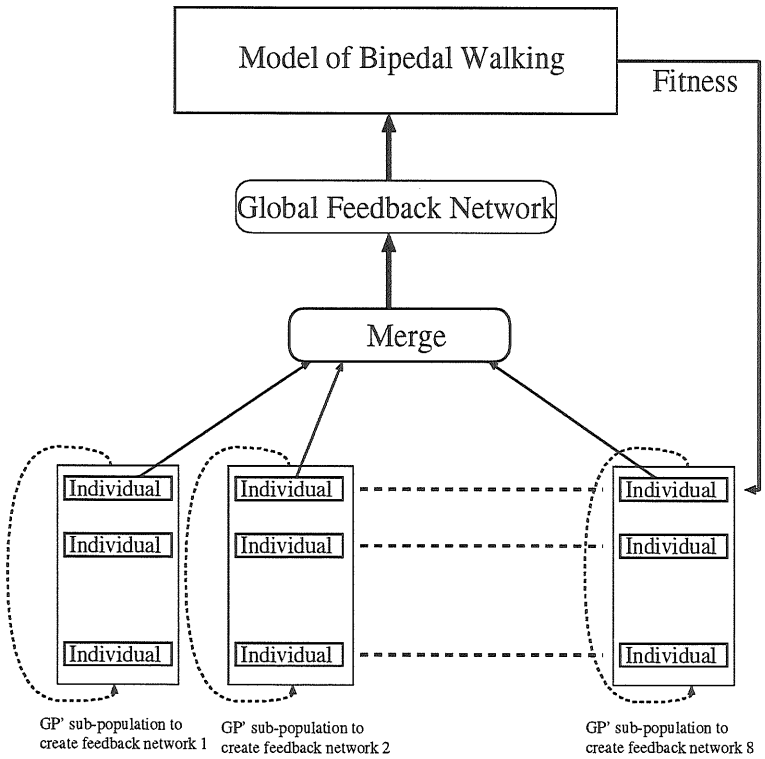

図 3.10: フィードバックネットワークの進化

ンを評価する. 各個体は, 評価値に基づいて確率的に遺伝子操作 (再生, 交叉, 突然変異など) が加えられ, 新しい個体が生成される. この手順を, あらかじめ設定した終了条件が満たされるまで繰り返して行い, 遺伝的プログラミングにおける進化過程が進められる.

本研究で提案する, 2 足歩行の神経系を構築するためには、神経振動子の 12 個のフィードバックネットワークモジュール（フィードバックネットワークモジュールの相互抑制性により 24 個のフィードバックネットワークの半分として）を生成する必要がある. さらに, 両脚の神経振動子系は, 左右対称的な構造を持っているという仮定により, 遺伝的プログラミングシステムは各フィードバックネットワークの探索空間に相当する 8 個の個別の部分集団を持っている. 図 3.10は, 本研究で遺伝的プログラミングシステムを用いて, 各関節のフィードバックネットワークモジュールから成るグローバルフィードバックネットワークを進化させるための概念図を示している.

各フィードバックネットワークは遺伝的プログラミングシステムの各部分集団で独立的に生成される. しかしながら, フィードバックネットワークの適応度評価は, 他のフィードバックネットワークと分離して行なうことができない. その適応度は神経系のグローバルフィードバックネットワークを通じて集団全体としての神経振動子の相互作用によって発生した 2 足歩行の評価に基づいて決定させる必要がある.

したがって, 個々のフィードバックネットワークの評価はグローバルフィードバックネットワーク全体の評価値から推定しなければならない. これはマルチエージェントシステムによる問題解決における困難な問題として良く知られている報酬割り当て問題に相当する. つまり, 他のエージェントとの協調動作により解を求めたときに, 各エージェントの問題解決に対する貢献度をどのように振り分けるのかを求める問題である.

この問題を解決するため, 遺伝的プログラミングに関しても様々な研究活動 [72] がなされている. しかし, 合理的に報酬割り当て問題を解決するための一般的な方法論は未だ存在しない. この問題を解決するために, 本研究では図 3.2.2のような遺伝的プロ

グラミングにおける簡単な異質集団モデルを考案した．そのアルゴリズムを以下で説明する．

1. 初期集団の生成:

   ランダムに遺伝的プログラミングの各部分集団の個体を生成する．

2. グループの生成:

   このグループは生成された各部分集団から適応度に従って選択された個体集合であり，グローバルフィードバックネットワークとして機能する．初期世代でのグループの生成は，ランダムに生成された各部分集団の個体の配列順にしたがって生成する．

3. グループの適用:

   グループ内の個体は同時に適用され，2 足運動シミュレーションによりグループとして評価される．グループ中の個体には同じ適応度が等しく振り分けられる．

4. 次世代の集団生成:

   次世代の集団生成は，部分集団ごとに通常の遺伝的プログラミングと同様な選択戦略と遺伝子操作によって行われる．

以上の手法は遺伝的プログラミングを用いて異質の集団を同時に生成するための最も簡単な方法である．

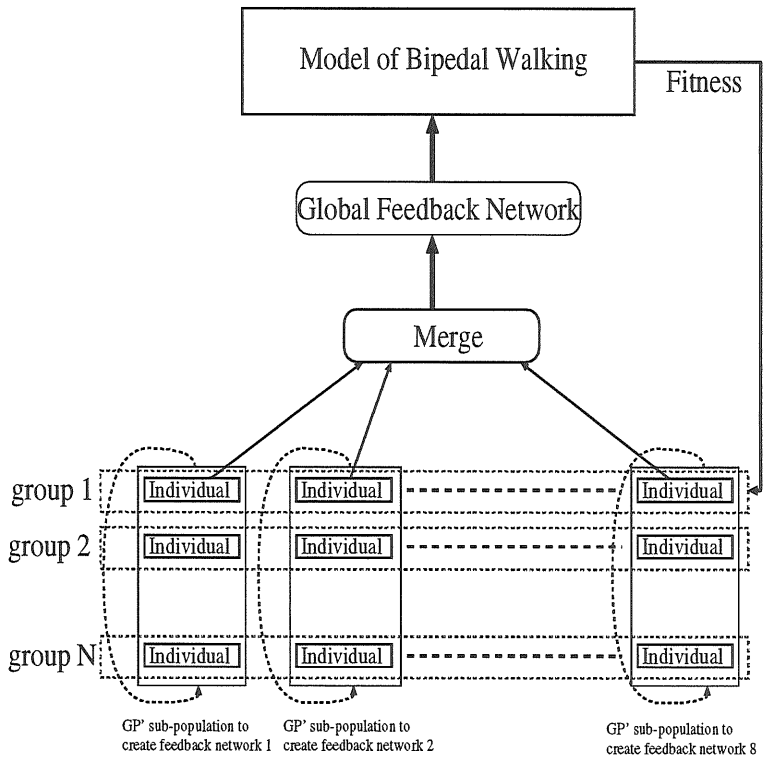

図 3.11: フィードバックネットワークモジュールの進化ための異種 GP モデル

第 4 章

2 足歩行運動シミュレーションの実際とモデルの評価

4.1 シミュレーションモデルの特徴

本章では 2 足歩行運動の自律的に発生するシミュレーション方法について述べる。前章で述べたように、力学系リズムと神経系リズムの非線形相互引き込み現象を利用した自律的運動発生手法が提案 [102, 3] されている. しかしながら, 歩行運動を司る神経系ネットワーク構造は神経生理学的に必ずしも明らかにされてはおらず, そのモデルは人為的な試行錯誤によって定められてきた.

本研究では, この問題を解決するために, 2 足歩行運動を発生する神経系の構造を自動的に決定する計算機シミュレーション実現した. 神経系の探索手法として, 遺伝的プログラミングを用いて複雑な神経系構造を柔軟に表現, 変更できるようにした. 図 4.1は前章までに説明した神経系と筋骨格系のモデルに基づいた 3 次元 2 足歩行運動シミュレータの概念図を示している.

神経系ネットワーク構造やパラメータ探索において, 本モデルによる歩行運動の生成シミュレーションの性質のまとめると次のような特徴があると考えられる.

歩行運動は複雑な非線形方程式により記述され, 導関数などの算出が困難である. ネットワークの構造やパラメータの変動に対して評価基準のポテンシャルは複雑な形状を

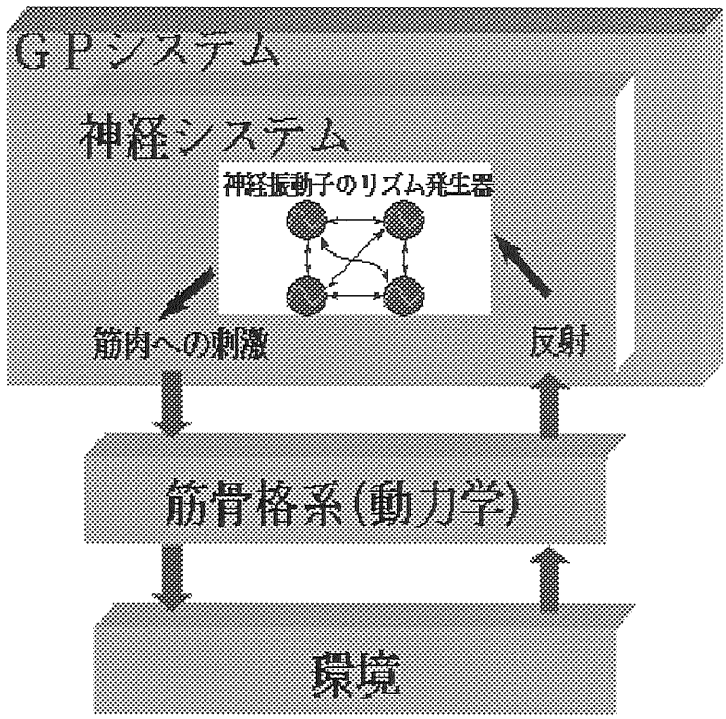

図 4.1: 実現したシステムの概念図

しており, 多峰性をもつことが予想される.

特に, 僅かなネットワークの構造やパラメータの変動で定常歩行から転倒してしまうことが考えられる. このような場合の評価関数の勾配は非常に複雑になっていると考えられる.

いわゆる正常歩行と呼ばれる歩容はある程度の許容範囲があり, 厳密な最適歩行を選択するのは困難であり, またその必要もない, むしろ広く複雑な解空間を如何に探索するかが問題となる. つまり厳密な最適解でなくとも大域的な準最適解が求まれば良いのである.

これらのことを考察し, 神経系の構造やパラメータを探索にするにおいて遺伝的プログラミングの特徴を考えると, 以下のことが考えられる.

最初に系の導関数を用いる必要がない, また多峰性の評価関数に対しても初期集団を十分大きくすることで, 局所解に収束することなく, 探索することができる.

確率的な突然変異を用いるので局所解な厳密解を得るのは不得手であるが, 大域的

な準最適化を求めることができる．現在提案されている学習モデルは教師信号が必要であったり，限られた神経系のモデルにしか適応できない場合が多い．

したがって，本研究では神経系の学習機能を特別にモデル化せずに，遺伝的プログラミングを用いることにした．遺伝的アルゴリズムを使う場合，適用する問題の特徴値(パラメータ)を個体の遺伝子として表現し，その個体が進化させることで探索や大域的最適解を求めるが．これにはあらかじめ決定された神経系の構造を用いる必要がある．

それに対して，遺伝的プログラミングは，個体の遺伝子にプログラムの関数を表現し，遺伝子である染色体によってで目的の機能を果たすプログラムを表現する．これは歩行運動の生成ための有効な機能を果たす神経系の構造を進化させることにおいて遺伝的プログラミングの方が効果的であることを示している．つまり，遺伝的アルゴリズムが解そのもの（パラメータ）だけを進化させるのに対して，遺伝的プログラミングは機能（神経系の構造とパラメータを同時に）を進化させることができる．

4.2 実現したシステムの詳細

本研究では, 提案した手法の妥当性を検証するために計算機シミュレーションを行なった. 図 4.2はそのシミュレーションの流れの概念図を示している.

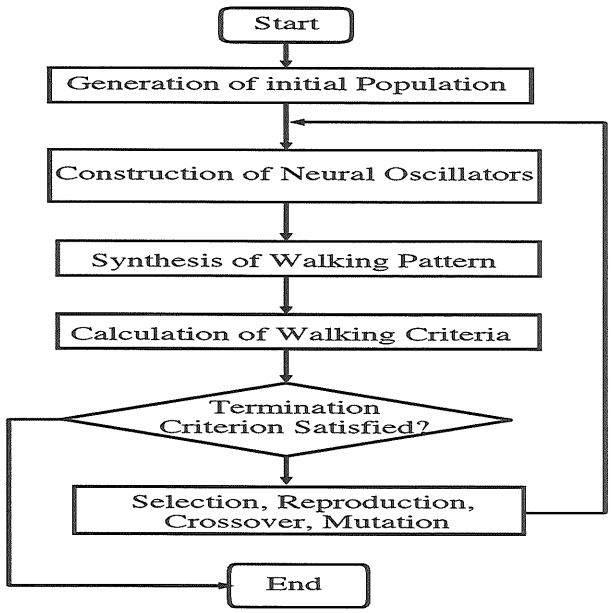

図 4.2: GP による神経系の構造やパラメータの探索の流れ

遺伝的プログラミングを用いた 2 足歩行生成のための神経系構造とパラメータの探索の流れは以下の通りである.

1. 初期集団群の生成

   ランダムに表 4.9のターミナルノードと関数ノードを用いて神経系におけるフィードバックモジュールの初期遺伝子集団を生成する. ここで, 各関節の神経振動子のフィードバックモジュールの探索空間として図 4.3ように遺伝的プログラミングシステムは 8 個の部分集団を持つ. 神経振動子数の 12 個の部分集団が必要になるが, 両脚の神経振動子系の左右対称的な構造を持っているという仮定により, 8 個の部分集団を持つことにした.

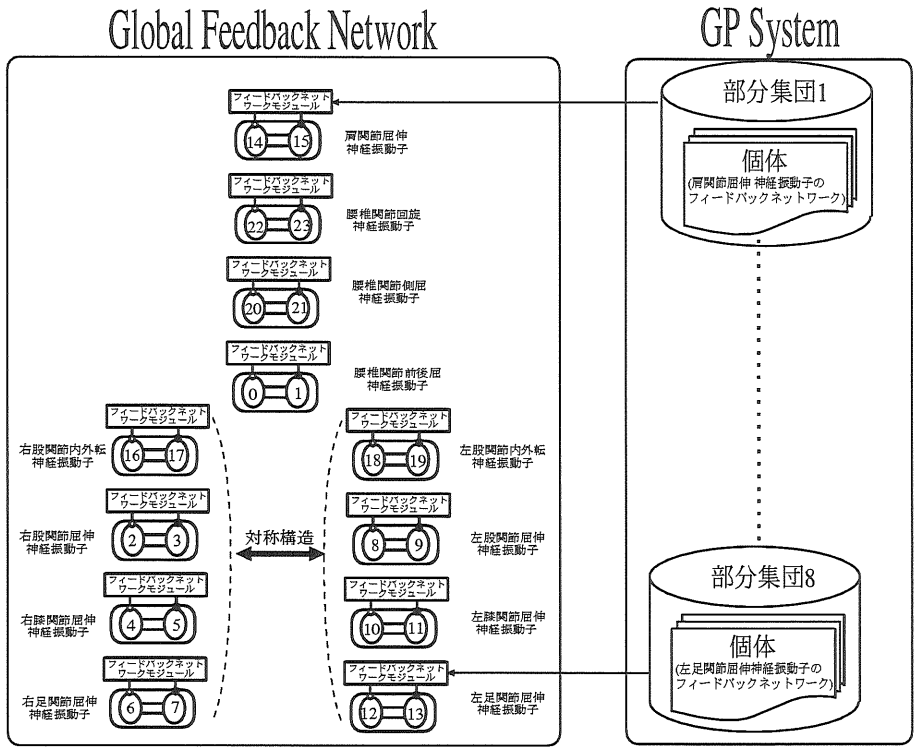

図 4.3: 神経系のフィードバックモジュールの遺伝子集団の生成

そして, 生成された各部分集団ごとの木構造のプログラムはグローバルフィードバックネットワークとして適用され, 記述された情報より神経系の構造を決定する.

2. 歩行運動の発生

2 章で示したように身体力学系と遺伝的プログラミングにより自動的に決定された神経系モデルのの連立微分方程式を, 4.2.2 節の歩行運動の初期条件から数値的に解くことにより, 時系列的に歩行運動を生成する. シミュレーションは規定の歩数に達するか, 転倒するまで続ける. 転倒の判断は股関節位置が, 下肢長 (大腿長と下腿長) の 3/5 の高さより低くなった場合, そして両脚が地面から離れた場合を転倒したと判断した. 両脚が地面から離れた場合に転倒と判断した理由は 4.4.2 節で説明する.

3. 適応度計算

生成された歩行運動の結果から, 式 のような評価関数に基づいて歩行運動の評価値を求める. この評価値は遺伝的プログラミングの適応度とする.

4. 終了条件

本シミュレーションでの終了条件は定めた世代数まできた場合と、 歩数が 10 歩以上になったとき, シミュレーションを終了する.

5. 選択と遺伝子操作

適応度が改善されるように, 表 4.8のような遺伝子操作と選択戦略を用いて各関節のフィードバックモジュールを進化させることで, 神経系の構造とパラメータを同時に修正する.

以上の過程を終了条件を満たすまで繰り返す.

4.2.1 運動評価基準

遺伝的プログラミングを用いて適切な神経系の構造を決定するためには, 歩行運動パターンを評価する何らかの基準が必要である. しかし, 人間の 2 足歩行運動はその複雑な身体構造により多くの自由度が有しており, 多様な運動が可能となっている. そのため移動運動を行なう場合, 無数の運動パターンが選択できる可能性があることになる. しかしながら実際の歩行は多少の個人差などはあるものの非常に高い定型性, 再現性を持っている. このような定型動作の存在より, 身体の随意運動には何らかの合目的性が存在し, この合目的性を達成するように運動が獲得されていると考えられるが, このような定型動作がどのような基準で達成されているかは, まだ明らかになっていない.

そこで, 本研究ではまず初期段階として, 歩行不可能な状態からある程度の正常歩行運動が獲得できるまで, 歩数と歩行距離などを簡単な評価基準を用いる. 次段階はより生体力学的に妥当な歩容の自律的生成のため, 生体力学的な運動評価基準を用いる. この次段階の評価基準はまだ定めておらず, 本研究の今後の課題として考えている.

初期段階の歩行運動評価基準は, 転倒するまでの歩行距離, 歩数, 身体の横ずれ, そして体重心の高さの変化する重み付き線形和であり, 次式のようなものを用いた.

$$I = a_1\mathcal{D} + a_2\mathcal{S} + a_3\mathcal{Z} + a_4\mathcal{Y} \tag{4.1}$$

だだし, a は重み係数で本研究では試行錯誤的に $a_1 = 0.09$, $a_2 = 0.10$, $a_3 = -0.001$, $a_4 = -0.005$ とした. $\mathcal{D}$ は転倒するまで歩行距離, $\mathcal{S}$ はその時の歩数である. $\mathcal{Z}$ は転倒するまでの体重心の高さ変化の時間的平均である. そして $\mathcal{Y}$ は初期の体重心の Y 軸の位置と転倒する時の体重心の Y 軸の位置との変位である. 上式の評価関数の重み係数によっては, 通常の歩行からジャンプに至るまでの多様な運動パターンを発生させることが可能である. 本研究のシミュレーションでは力学系の運動方程式と神経系を表す連立方程式 (2.1 - 2.3) を, 積分の時間刻み幅を $0.5ms$ としてオイラー法を用いて数値的に解いている.

4.2.2 遺伝的プログラミングの探索効率化

遺伝的プログラミングはさまざまな分野に応用され, その有効性が確かめられている. 遺伝的プログラミングの適用範囲は, AI の問題解決から, ロボット, 分子生物学などの実際的な問題まで多岐にわたっている. しかし, GP では木構造に交叉と突然変異を適用することで木を変形し, プログラムを探索する. この方式の効率や有効性についてはまだ不明な点も多い.

特に, 遺伝的プログラミング研究の現状での最大の問題点は計算量であると考えられる. 本研究では, 計算時間を短縮するために遺伝的プログラミングの並列計算を行なった. これに関しては後で述べる. さらに本研究では, 遺伝的プログラミングの探索探索空間を削減するために以下のようないくつかの工夫を加えた.

1. 計算打ち切り

   本研究では正常歩行歩容の獲得を目指したので, ジャンプのように両足が床面を離れることを防ぐために, 両足が 10cm 以上床面から離れた場合, その評価値を 0 とした.

   これは遺伝的プログラミングによって自動的に生成された神経系ネットワークを用いて評価シミュレーションを行なう際, シミュレーションでは動力学的計算を行なうので, 評価計算時間が多くかかる, そこで, 見通しがない歩行運動の場合はその評価シミュレーションを中止し, 次の評価シミュレーションを行なうことで評価計算時間を削減するためのものである.

2. 位相センサ

   本シミュレーションでは右足だけの接地状態を表すセンサー, 左足だけの接地状態を表すセンサー, 両足の接地状態を表すセンサーの 3 種類の位相センサを用いた. 次式はそれを具体的に表したものである

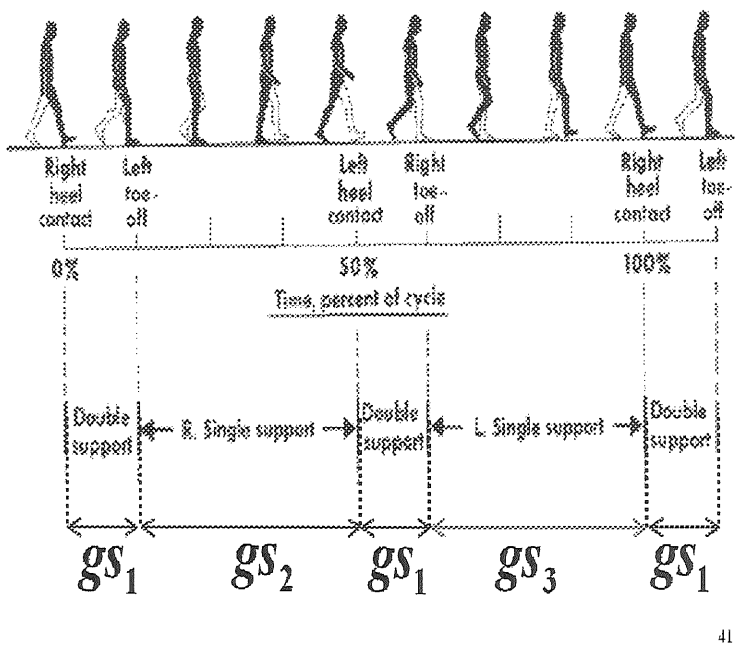

図 4.4: 歩行中の脚位相状態

$$
\begin{aligned}
gs_1 &= s_r * s_l \\
gs_2 &= s_r(1 - s_l) \\
gs_3 &= s_l(1 - s_r)
\end{aligned}
$$

だだし, s_r, s_l はそれぞれ, 右足と左足の接地センサーであり, 接地している時は 1 で, 遊泳の場合は 0 である. また gs_1 は両足の接地の位相センサ, gs_2 は右足の接地の位相センサ, gs_3 は左足の接地の位相センサである.

この位相センサは, 身体のグローバル的な歩行状態を表すためのものである. ただし, 身体のグローバルな位相状態を表す位相センサそのものが感覚受容器に存在は生理学的には考えにくい. しかしながら, 足の接地センサの情報を総合して考えれば, 結局このような位相情報と等価な情報として扱えると考えられるので, 本モデルでは位相センサのような比較的抽象度の高いフィードバック量を用いた.

図 4.4は歩行中の位相状態を表したものである

3. 禁止制約条件

関数の引数となり得るターミナル同士の関係を制約として定式化し, 初期集団生成時に制約違反となるターミナルの組合わせを禁止した.

遺伝的プログラミングは, 関数ノードとターミナルノードとを組合せた木構造の染色体を改造しながら行なう解探索手法である. 本研究ではその染色体を, 表現型で表すとフィードバックネットワークモジュールになる. そこで, ターミナルノードの組合せにおいて, 運動学的に意味を持たない組合せを制約として定式化,

その組合せを初期集団生成時に禁止する．これは探索空間から制約条件を充足しない領域を削除することにより，その求解性能を向上させることができる．

本研究での組合せ禁止制約を表 4.1にまとめる．

表 4.1: 禁止制約の例

NO	禁止制約
1	各関節の相対的角度同士の掛け算はない
2	各関節の相対的角速度同士の掛け算はない
3	各関節の絶対角度同士の掛け算はない
4	各関節の絶対角速度同士の掛け算はない
5	各重心位置同士の掛け算はない
6	各重心速度同士の掛け算はない
7	各重心速度と重心位置同士の掛け算はない
8	各関節の相対的角度と重心位置同士掛け算はない
9	各関節の相対的角度と重心速度同士掛け算はない
10	各関節の相対的角速度と重心位置同士掛け算はない
11	各関節の相対的角速度と重心速度同士掛け算はない
12	各関節の絶対角度と重心位置同士掛け算はない
13	各関節の絶対角度と重心速度同士掛け算はない
14	各関節の絶対角速度と重心位置同士掛け算はない
15	各関節の絶対角速度と重心速度同士掛け算はない

次は遺伝的プログラミングの初期世代の個体における禁止制約付きの遺伝的プログラミングのプログラム木の生成のアルゴリズムを示す．

```
generate_tree(depth){
  if(depth = 0){
     loop:
     T = random_int(terminal_set);
     if(!restricted_constraint_check(T)){
      return T;
    }
    else{
      goto loop;
    }
    F = random_int(function_set);
    for(i = 0; i<arity_of_F;++i){
        generate_tree(depth-1);
     }
  }
```

図 4.5はそのアルゴリズムの例を示している.

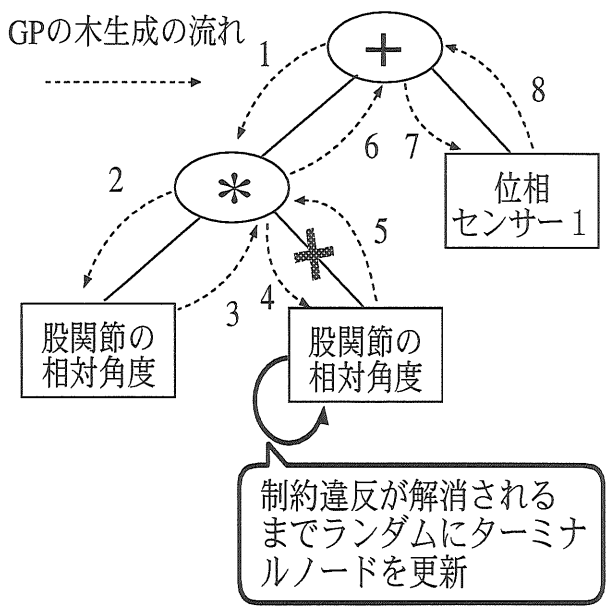

図 4.5: 禁止制約を用いた木の生成

たとえば, 図 4.5のように関数ノードである “*” ノードの第 1 番目の引数と第 2 番目の引数のターミナルノード組合せが制約違反になる場合, 第 2 番目の引数のターミナルノードを, 制約違反が解消されるまでランダムに繰り返して更新を行なう.

4.2.3 遺伝的プログラミングの並列処理

遺伝的プログラミングは多くの繰り返し計算を行うため, 莫大な計算コストを必要とするといった問題が存在する. 一方で, 遺伝的プログラミングのような進化的計算手法は多点探索を行う場合が多いため, 並列計算に適していると言われている.

一般的に遺伝的プログラミングを並列化する主な方法は次の 3 つである.

- 適応度評価の並列化：通常の遺伝的プログラミングの適応度評価は時間がかかるので, その評価のうち並列に実行できるものを並列化する.
- 個体レベルでの並列化：各個体の評価を別々のプロセッサに割り当てる.
- 実行の並列化：プロセッサごとに別々の初期化により遺伝的プログラミングを走らせて, 最も良い成績を採用する.

また, 並列計算機では, CPU とメモリの位置関係によって, 共有メモリ型と分散メモリ型 (図 4.6, 図 4.7) に大別される. 共有メモリ型では, 1 つのメモリを中心として複数のプロセッサがこれにつながっている. この型の利点は, プログラミング時にデータ分割を考慮に入れる必要がないため, 自動並列化を容易に行うことができる点である. さらに, メモリ間通信が必要ないため, プロセッサ数が少ない場合は性能を高めることができる.

しかし, プロセッサ数が多くなると, 他タスク・他ジョブとのメモリアクセス競合により通信が混み合い, 性能が低下してしまう欠点もある. 分散メモリ型では, 1 つのメモリと 1 つのプロセッサを 1 つの節（ノード）として, このノードを相互結合網で複

数接続している. この型の利点は, 他タスク・他ジョブとのメモリアクセス競合により通信が混み合うことがないため, 全体的に性能をあげられる点である. しかし, 複数あるメモリの管理が難しい問題もある.

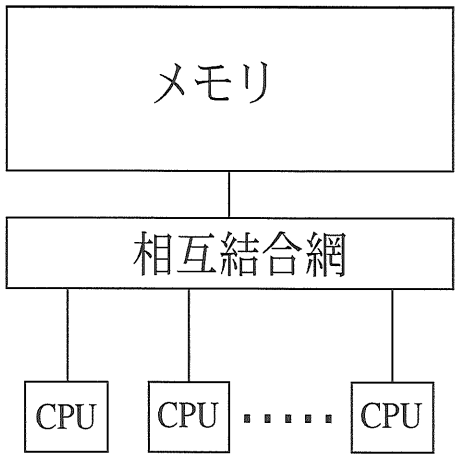

図 4.6: 共有メモリ

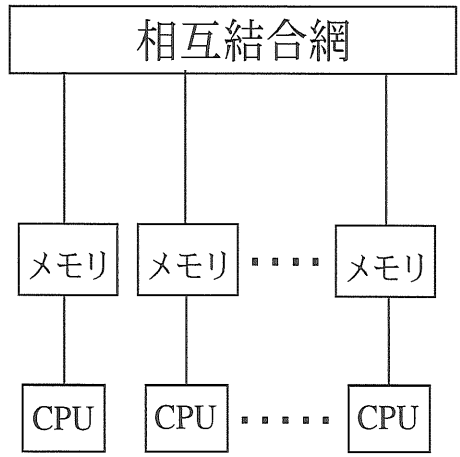

図 4.7: 分散メモリ

そこで, 本研究では遺伝的プログラミングの個体群を並列的に評価計算するためにクラスタマシンを構築した（図 4.8）.

特に, 個体の適応度を評価するためには 2 足歩行運動をシミュレートをする必要があるため, その部分の計算負荷が他の処理部分と比べて大幅に大きい. したがって, 個々の個体の適応度評価をクラスタマシンの各ノードに並列的に分散することによって実験の効率がほぼ線形的に向上した.. 本研究で使用したクラスタマシンの仕様は表 4.2にまとめた.

図 4.8: クラスタマシン

表 4.2: クラスタマシンの仕様

CPU's	533MHz or 433MHz Celeron
Number of PE	64
Memory	128MB/PE
NIC	Fast Ethernet
Switching Hub	100 Base
OS	Redhat Linux 6.2
Library	LAM/MPI

本シミュレーションでは各個体の評価を別々のプロセッサに割り当てるために，次のようなアルゴリズムを用いた．

```
/* assign sub-populations to processes */
/* wpn is number of process and wid is  process ID */
  for ( i=0; i<wpn; i++ ) {
    begin = 0;
    for ( j=0; j<=i; j++ ) {
      if ( j < mpop->pop[0]->size%wpn ) {
        range = (int)(mpop->pop[0]->size/wpn) +1;
      } else {
        range = (int)(mpop->pop[0]->size/wpn);
      }
      if ( j == i ) break;
      begin += range;
    }
    displs[i]  = begin;
    rcounts[i] = range;
  }

/* distribute computation to CPU nodes */
  j=0;
  for ( k = 0; k < mpop->pop[0]->size; ++k ){
    if ( k>=displs[wid] && k < displs[wid]+ rcounts[wid]) {
       evaluate_fitness (mpop,k );
       sbuff_r[j]    = mpop->pop[0]->ind[k].r_fitness;
       sbuff_hits[j] = mpop->pop[0]->ind[k].hits;
      j++;
    }
  }

 /* wait all CPU nodes finish their calculation */
```

```
/*  then communicate results to each other     */
  MPI_Barrier( MPI_COMM_WORLD );

  MPI_Allgatherv(sbuff_r,  rcounts[wid], MPI_DOUBLE,
                 rbuff_r, rcounts, displs, MPI_DOUBLE,
                 MPI_COMM_WORLD);

  MPI_Allgatherv(sbuff_hits, rcounts[wid], MPI_INT,
                 rbuff_hits, rcounts, displs, MPI_INT,
                 MPI_COMM_WORLD);
```

ここでは, evaluate_fitness() の実行を別々のプロセッサで行なうことになる.

4.3 シミュレーション実験および評価

ここでは, 前で述べた神経系モデルと身体力学系, 遺伝的プログラミングシステムを用いて歩行運動のシミュレーションを行ない, その評価を行なう.

まず, 本シミュレーションの初期条件として必要な, 各モデルのパラメータについて述べる. 本研究では歩行開始や停止などの過渡的な歩行運動は想定せず, 定常歩行の実現を目指した. シミュレーション初期条件として, 神経系モデル, 身体力学系モデルのパラメータ, 運動の初期条件パラメータは先行研究 [3] により取得した各モデルのパラメータを参考にして定めた.

4.3.1 身体系モデルにおけるパラメータ

本研究では体系条件の変化は仮定していないので, 成人の体系を想定した.

身体系のパラメータはシミュレーションにおいて基本パラメータとして用いられる. 表 4.3に身体系モデルの慣性パラメータを示す.

Name	Mass[kg]	Moment of inertia[kgm^2] (X, Y, Z)	Link length[m]	Center of gravity[m]
Upper torso	29.482	1.6193, 1.0876, 0.3785	0.333	0.658
Lower torso	8.500	0.0425, 0.0410, 0.0500	0.085	0.170
Right Thigh	6.523	0.1137, 0.1157, 0.022	0.157	0.405
Right Calf	2.685	0.0391, 0.0392, 0.0029	0.176	0.415
Right Foot	0.837	0.0033, 0.0030, 0.0007	0.050	0.150
Left Thigh	6.523	0.1137, 0.1157, 0.022	0.157	0.405
Left Calf	2.685	0.0391, 0.0392, 0.0029	0.176	0.415
Left Foot	0.837	0.0033, 0.0030, 0.0007	0.050	0.150
Right Upper-arm	1.842	0.0133, 0.0132, 0.0022	0.145	0.282
Right Fore-arm	1.513	0.0133, 0.0133, 0.0010	0.120	0.360
Left Upper-arm	1.842	0.0133, 0.0132, 0.0022	0.145	0.282
Left Fore-arm	1.513	0.0133, 0.0133, 0.0010	0.120	0.360

表 4.3: 身体力学系モデルのパラメータ

4.3.2 神経系モデルにおけるパラメータ

神経系のフィードバックネットワーク部分と，他の神経神経振動子からの結合関係に関する情報は先に述べた遺伝的プログラミングにより探索的に決定されるが，その他のパラメータは固定である．これらの値は表 4.4に示す．また，定常入力信号である

Neural Oscillator	τ_i	τ_i'	Neural Oscillator	τ_i	τ_i'
0	0.04307	0.43063	12	0.02153	0.21533
1	0.04307	0.43063	13	0.02153	0.21533
2	0.04307	0.43063	14	0.04307	0.43063
3	0.04307	0.43063	15	0.04307	0.43063
4	0.02153	0.21533	16	0.04307	0.43063
5	0.02153	0.21533	17	0.04307	0.43063
6	0.02153	0.21533	18	0.04307	0.43063
7	0.02153	0.21533	19	0.04307	0.43063
8	0.04307	0.43063	20	0.04307	0.43063
9	0.04307	0.43063	21	0.04307	0.43063
10	0.02153	0.21533	22	0.04307	0.43063
11	0.02153	0.21533	23	0.04307	0.43063

表 4.4: 神経系モデルのパラメータ

u_0 と疲労定数 β は各々 2.1150, 2.5000 とした．

4.3.3 運動の初期条件

シミュレーションの初期運動条件として必要な値は身体力学系では関節の絶対角度，および角速度，腰部関節の原点の並進位置および並進速度である．また神経系では u_n,v_n

の値である．本シミュレーションでは歩行運動は身体力学系と神経系からなる非線形力学系のリミットサイクルとして生成されるので，ある程度初期値に依存しないで安定な周期運動を得ることができると考えられる．しかしながら，その許容範囲は狭く，適切な初期値を与えなければ安定な周期解に到着できず，すぐ転倒してしまう．本シミュレーションに用いた初期条件の値は以下の表 4.5から表 4.7に示す．

関節名	角度 [deg]	角速度 [deg/s]
腰部	6.780	-6.179
胸部	0.387	-5.413
右大腿	-4.520	2.052
右下腿	4.722	5.196
右足	-70.708	24.428
左大腿	0.313	-28.933
左大腿	73.876	-219.547
左足	-50.662	-56.700
右上腕	168.803	-93.445
右前腕	-16.778	50.955
左上腕	183.204	57.589
左前腕	-19.648	-33.391

表 4.5: 関節の初期条件 (角度および角速度)

軸	変位 [m]	速度 [m/s]
x	0.000	1.252
y	0.000	-0.026
z	0.857	0.109

表 4.6: 腰部関節の初期条件 (並進自由度)

神経番号	u_n	v_n	神経番号	u_n	v_n
0	0.458	0.894	12	1.551	1.326
1	-3.541	0.099	13	-9.647	1.955
2	-7.919	0.115	14	0.980	0.286
3	0.578	2.415	15	-1.984	0.140
4	2.574	0.647	16	3.947	2.302
5	-4.850	0.035	17	-14.760	0.577
6	-15.434	1.608	18	-2.754	1.429
7	4.180	1.191	19	0.307	0.704
8	-0.005	0.446	20	-1.192	0.663
9	-1.998	0.666	21	-0.100	1.030
10	1.567	0.592	22	-4.315	1.041
11	-2.485	0.445	23	-4.501	3.472

表 4.7: 神経系の初期条件

4.3.4 数値計算の解法

計算のサンプリング時間は $0.5ms$ とし, 数値積分の方法として最も簡単なオイラー法を用いた. 通常は, より高精度なルンゲクッタ法などが利用されるが, この方式では 1 ステップの計算を行なうのに 4 回の繰り返し計算が必要なため計算時間がかかってしまう. そこで, 本シミュレーションでは計算コストおよびサンプリング時間が短いオイラー法でも十分精度が得られると考え, この手法を採用した.

4.3.5 遺伝的プログラミングのパラメータ

本研究では, 遺伝的プログラミングの実現プラットフォームとして, "lil-gp" システム [116] を修正して用いた. 遺伝的プログラミングシステムで使用した重要なパラメータは表 4.8に示す.

表 4.8: GP Parameters

Fitness	$F = a_1 D + a_2 S + a_3 \mathcal{Z} + a_4 \mathcal{Y}$
Misc.	Population size = 12000 Generation = 100 Depth of tree = 10 Selection = greedy over-selection
Crossover Probability	70%
Reproduction Probability	10%
Mutation Probability	20%
Termination Criterion	After completing 10 walking steps

また, 本研究で使用した遺伝的プログラミングにおける関数ノードとターミナルノードは, 運動力学的な考察により表 4.9のように定義した.

表 4.9に示すように, 本実験で用いたターミナルの数は合計 121 個である. これは通

表 4.9: Terminals and Functions

Terminals	n_i	output of neurons (24)
	$a^{\{r,l\}}_{\{x,y,z\}}$	absolute angles at each side (6)
	$\dot{a}^{\{r,l\}}_{\{x,y,z\}}$	absolute angular velocities at each side (6)
	$r_i^{\{x,y,z\}}$	relative angle at each joint (36)
	$\dot{r}_i^{\{x,y,z\}}$	relative angular velocities at each joint (36)
	$s^{\{r,l\}}$	contact sensory at each side (2)
	$g_{\{x,y,z\}}$	position of center of gravity (3)
	$\dot{g}_{\{x,y,z\}}$	velocity of center of gravity (3)
	c	ephemeral random constant (5)
Functions	$+, -, \times,$ $H(x) = \begin{cases} 1 & (\text{when } x \geq 0) \\ 0 & (\text{otherwise}) \end{cases},$ $K(x) = \begin{cases} x & (\text{when } x \geq 0) \\ 0 & (\text{otherwise}) \end{cases}$	

常の遺伝的プログラミングで用いられるターミナルノードの数よりも非常に多く，探索空間が極めて大きいことを意味している．遺伝的プログラミングにより生成されるプログラムの内部構造の基となる関数ノードとターミナルノードの設定は実験者の事前知識に大きく依存する．

しかし，神経生理学的な研究においても 2 足歩行運動ための神経系構造に関しては未だ充分な知見が得られていないため，望ましい有用なターミナルを選択することは極めて困難である．したがって，現実的には冗長なターミナルノード集合を利用しなければならないが，冗長なターミナルノードはむだな探索を増やして遺伝的プログラミングの探索効率を著しく低下させる．

本研究では，冗長なノード集合から無用なノードを削除するために，ノードの有用性に基づいた適応的な突然変異手法を提案し，記号当てはめ式問題を用いた実験で，提案した手法が有用なノードを効果的に選択し，遺伝的プログラミングの効率を改善することを示した．これに関しては次章で述べる．

4.4 実験結果

2 足歩行運動の評価基準として，式 4.1を最大になるように設定してシミュレーションを行なった．

4.4.1 予備実験 (1 関節のフィードバックネットワークモジュール)

本実験では，提案した手法の有効性を確かめるために，予備実験として人間の歩行運動の重要な役割をする腰の神経振動子におけるフィードバックモジュールの自動生成の実験を行なった． 他の神経振動子の構成は，過去の研究 [3] で研究者による試行錯誤の末に得られた次式のような神経振動子を用いた．予備実験における歩容探索の初期段階では，力学系と神経系のリズムが協調せず，図 5.3のようにすぐに転倒してしまうことが多かった．しかし世代が進むにつれて徐々に進化していき，100 世代後には，

表 4.10: フィードバックネットワークモジュール (式 f_2 ～式 f_{22} は [3] から参考)

$$
\begin{aligned}
*f_0 &= 134.5054\,(r_5^y - r_2^y) + 10.8311\,(\dot{r}_5^y - \dot{r}_2^y) - (n_2 + n_9 - n_3 - n_8) \\
f_2 &= 2.3615\,r_7^y - 1.3880\,r_{11}^y + 0.7623\,r_8^y\,s^r - 1.0420\,s^r - 29.8467\,r_2^y\,s^r - \\
&\quad 11.1777\,\dot{r}_2^y\,s^r - (n_8 - n_9) \\
f_4 &= -1.8263\,r_7^y\,s^l + 1.0680\,r_{12}^y\,s^l + 3.3566\,(s^r - s^r\,s^l\,H(a_x^l - a_x^r)) - (n_2 + n_3) \\
f_6 &= -3.8349\,r_7^y\,s^r + 039.0429\,r_8^y\,s^r + 4.3012\,(\dot{r}_9^y - \dot{r}_8^y)\,s^r - 5.0983\,s^r + \\
&\quad 5.0983\,(1.0 - s^r)/2.0 - (n_2 + n_3) \\
f_8 &= 2.3615\,r_{11}^y - 1.3880\,r_7^y + 0.7623\,r_{12}^y\,s^l - 1.0420\,s^l - 29.8467\,r_2^y\,s^l - \\
&\quad 11.1777\,\dot{r}_2^y\,s^l - (n_2 - n_3) \\
f_{10} &= -1.8263\,r_{11}^y\,s^r + 1.0680\,r_8^y\,s^r + 3.3566\,(s^l - s^r\,s^l\,H(a_x^r - a_x^l)) - (n_8 + n_9) \\
f_{12} &= -3.8349\,r_{11}^y\,s^l - 39.0429\,r_{12}^y\,s^l + 4.3012\,(\dot{r}_{13}^y - \dot{r}_{12}^y)\,s^l - 5.0983\,s^l \\
&\quad +5.0983\,(1.0 - s^l)/2.0 - (n_8 + n_9) \\
f_{14} &= 0.5\,r_{14}^y - 0.5\,r_{16}^y + r_7^y + r_8^y - r_{11}^y - r_{12}^y - (n_2 + n_9 - n_3 - n_8) \\
f_{16} &= -901.2270\,(g_y - a_y^r - 0.8711/20.0)\,(1.0 - s^r)\,K(-r_7^y - r_8^y) - \\
&\quad 232.2408\,r_1^z\,s^r - 27.3127\,\dot{r}_1^z\,s^r + 89.7301\,\dot{a}_y^r\,(1.0 - s^r)\,K(-r_7^y - r_8^y) - \\
&\quad 541.7639\,\dot{g}_y\,(1.0 - s^r)\,K(-r_7^y - r_8^y) \\
f_{18} &= -901.2270\,(a_y^l - g_y - 0.8711/20.0)\,(1.0 - s^l)\,K(-r_{11}^y - r_{12}^y) + \\
&\quad 232.2408\,r_1^z\,s^l + 27.3127\,\dot{r}_1^z\,s^l - 89.7301\,\dot{a}_y^l\,(1.0 - s^l)\,K(-r_{11}^y - r_{12}^y) \\
&\quad +541.7639\,\dot{g}_y\,(1.0 - s^l)\,K(-r_{11}^y - r_{12}^y) \\
f_{20} &= -100.0\,r_3^z - 10.0\,\dot{r}_3^z - 10.0\,r_5^x - \dot{r}_5^x \\
f_{22} &= -10.0\,r_4^z - \dot{r}_4^z - 20.0\,r_0^z + 10.0\,r_7^y + 10.0\,r_8^y - 10.0\,r_{11}^y - 10.0\,r_{12}^y
\end{aligned}
$$

図 5.4のように持続的な歩行が可能となり, 比較的に人間に近い歩容を獲得できた.

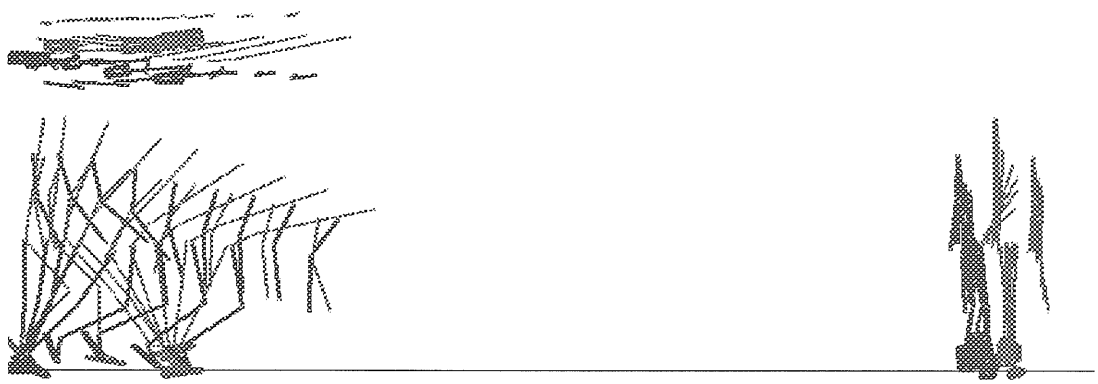

図 4.9: 予備実験: 初期世代での歩容

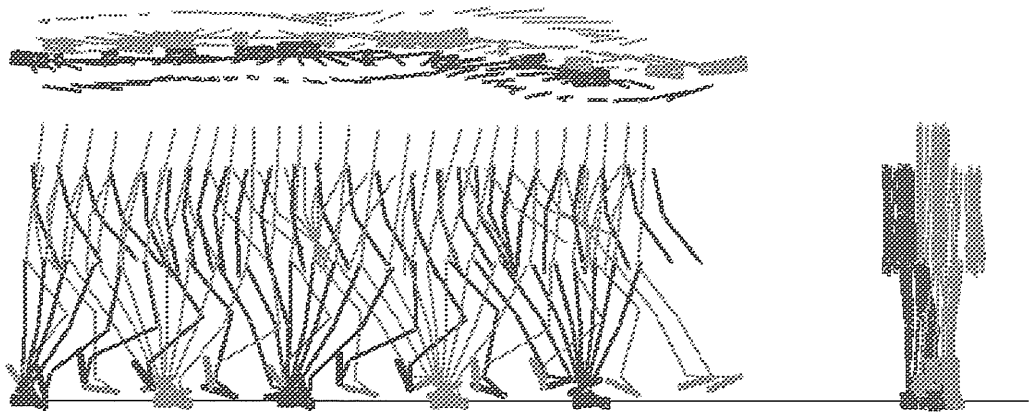

図 4.10: 予備実験:100 世代後での歩容

予備実験から得られた歩行パターンは, 過去の研究で得られた神経振動子による歩行パターンと似ているが, 本研究の手法によって生成された腰の神経振動子の式を分析したところ, 長谷ら [3] によって求められた腰の神経振動子のフィードバックネットワークモジュールの式 (表 4.10の f_0) とかなり異なっていることが分かった. 表 4.11は本シミュレーションにより得られた腰の神経振動子のフィードバックネットワークモジュールである.

表 4.11: 本シミュレーションで得られた腰部のフィードバックネットワークモジュール

$$
\begin{aligned}
f_0 \;=\; & ((\dot{g}_x + \dot{r}^y_5 + H(H(n_{21}))((\dot{r}^y_{11} - r^x_{12}) - \dot{r}^x_{12} r^x_{12})(\dot{r}^x_8 + r^y_{11}) + r^x_{14} - r^z_9 - \\
& H(H(n_{14}) - H(H(n_{14})))(\dot{g}_x + \dot{r}^y_5)) + ((\dot{r}^x_8 + r^y_{11} + H(H(n_{21})) \\
& ((\dot{r}^z_2 + r^x_7 - H(n_{14}))(\dot{r}^z_2 + r^x_7)(\dot{r}^z_2 + r^x_7)) + r^x_{14} - r^z_9 - H(r^x_{12})) \\
& (\dot{r}^y_{11} - r^x_{12} - \dot{r}^x_{12} r^x_{12})((\dot{r}^x_8 + r^y_{11})(\dot{r}^z_2 + r^x_7))\dot{r}^x_{12} r^x_{12})) + \\
& (\dot{g}_x + \dot{r}^y_5 + H(H(n_{21}))((\dot{r}^y_{11} - r^x_{12} - \dot{r}^x_{12} r^x_{12}) r^x_{12}) + (r^x_{14} - r^z_9 - H(H(n_{14})) - \\
& H(H(n_{14})))(\dot{g}_x + \dot{r}^y_5))(\dot{r}^x_8 + r^y_{11} + H(H(n_{14})) \\
& ((\dot{r}^y_{11} - r^x_{12} - (\dot{r}^y_{11} - r^x_{12} - \dot{r}^x_{12} r^x_{12})) \\
& ((\dot{r}^x_8 + r^y_{11})(r^y_{16} - 15.32862))) + H(x)H(x) r^x_{12}) - H(n_{14})
\end{aligned}
$$

本シミュレーションの予備実験の結果から，本探索手法が歩行運動を発生する神経系構造の獲得手法として有望であることが確認できた．

4.4.2 本実験 (12 関節のフィードバックネットワークモジュール)

本実験では，すべての神経振動子におけるフィードバックネットワークモジュール (グローバルフィードバックネットワーク) を探索するため，運動力学的な考察により，歩行運動で最も大きく運動するのは関節の屈曲伸展の運動であることから，ターミナルノードを表 4.12のように合計 32 個まで減らしてシミュレーションを行なった．

図 5.6は本実験でのシミュレーションにより得られた歩容の様子を示している．

図 4.12は，シミュレーションから得られた大腿，下腿，足の各関節の角速度，図 4.13は各々の角度の時間変化を示している．これらの図からわかるように，比較的安定した周期的リズムで各関節が運動していることがわかる．図 4.14はシミュレーションから得られた大腿，下腿，足の位相状態を示している．横軸はそれぞれの関節の角度，縦軸は

表 4.12: Terminals

$a_{\{y\}}^{\{r,l\}}$	absolute angles at each side (2)
$\dot{a}_{\{y\}}^{\{r,l\}}$	absolute angular velocities at each side (2)
$r_i^{\{y\}}$	relative angle at each joint (8)
$\dot{r}_i^{\{y\}}$	relative angular velocities at each joint (8)
$s^{\{r,l\}}$	contact sensory at each side (2)
$gs_{1,2,3}$	global states sensory (3)
$g_{\{y\}}$	position of center of gravity (1)
$\dot{g}_{\{y\}}$	velocity of center of gravity (1)
c	ephemeral random constant (5)

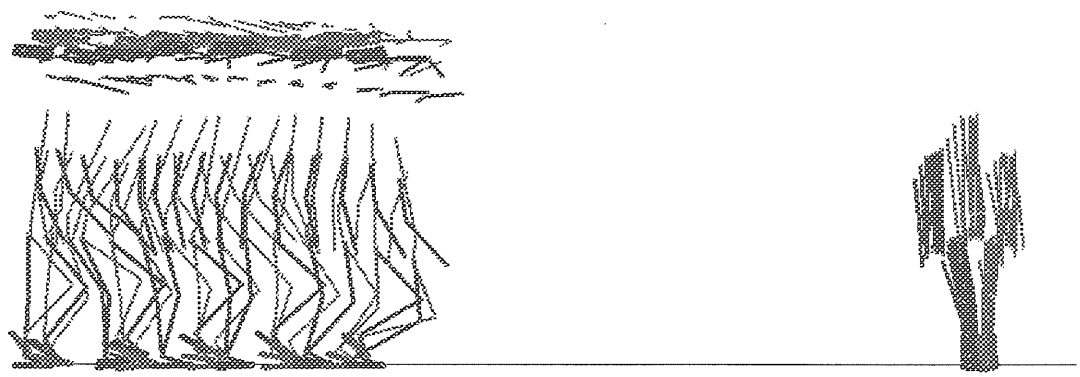

図 4.11: 本実験:100 世代後での歩容 (グローバルフィードバックネットワーク)

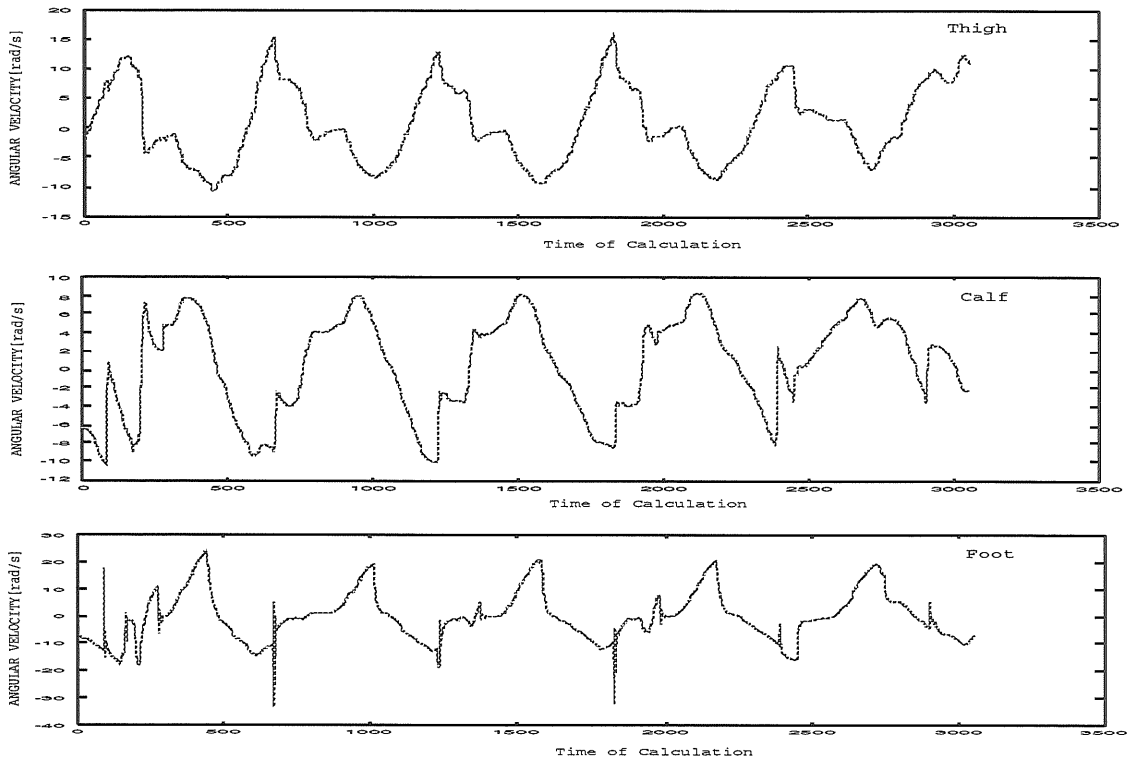

図 4.12: Angular Velocity of Thigh, Calf and Foot

角速度を表している．この位相図は矢状面内の歩行運動の安定性を示している．本実験での歩行運動は力学系と神経系からなる非線形系のリミットサイクルとして生成されるので, その許容範囲は狭く, 本実験では安定な周期解に到達できず, 図 5.6のように 10 歩程度歩いた後, 転倒してしまった．しかしながら, 本実験結果が示すように, 一部リズミカルな歩行運動パターンの生成には成功しており, 今後の改良により, より安定した歩行の実現が可能であると考えれる.

本実験で歩容の獲得過程における適応度の推移の様子を図 4.15に示めしたものである．図中の値は遺伝的プログラミングおいて最も適応度が高かった個体のグループに関する値である．図 4.15のように, 世代が進むにつれて適応度がほぼ単調に増加しており, 探索が有効に行なわれたことが分かる.

次式は本システムにより生成されたグローバルフィードバックネットワークの例である.

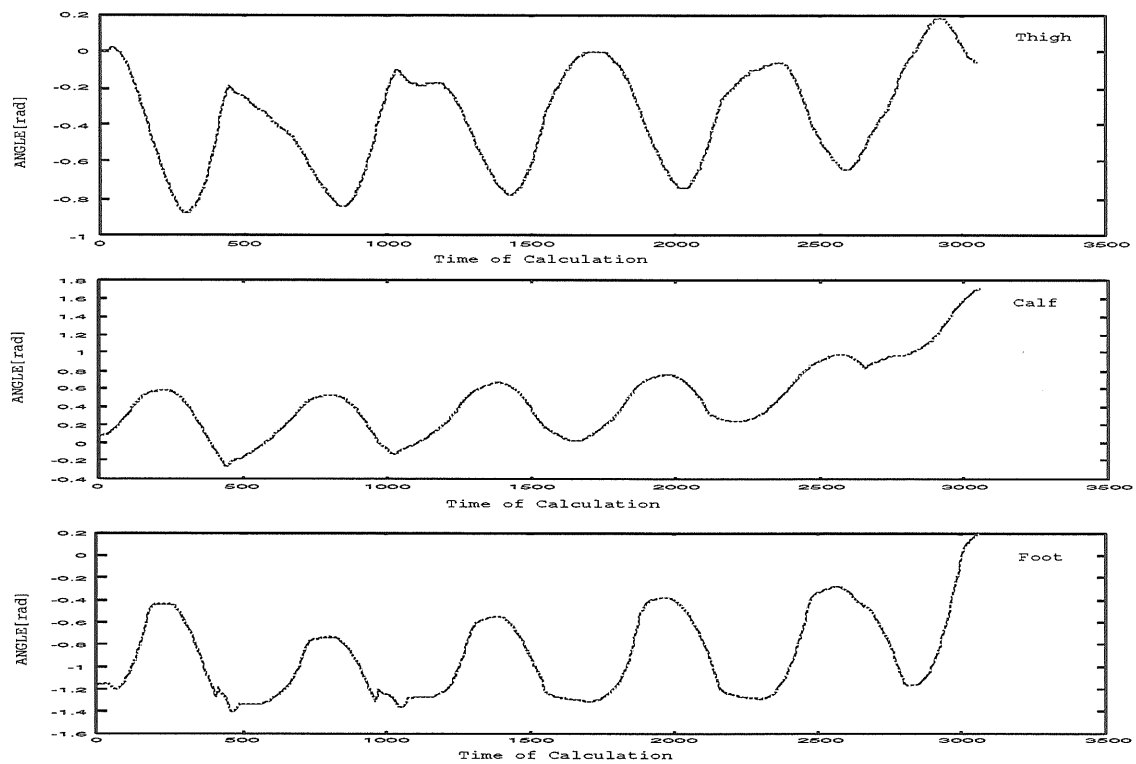

図 4.13: Angle of Thigh, Calf and Foot

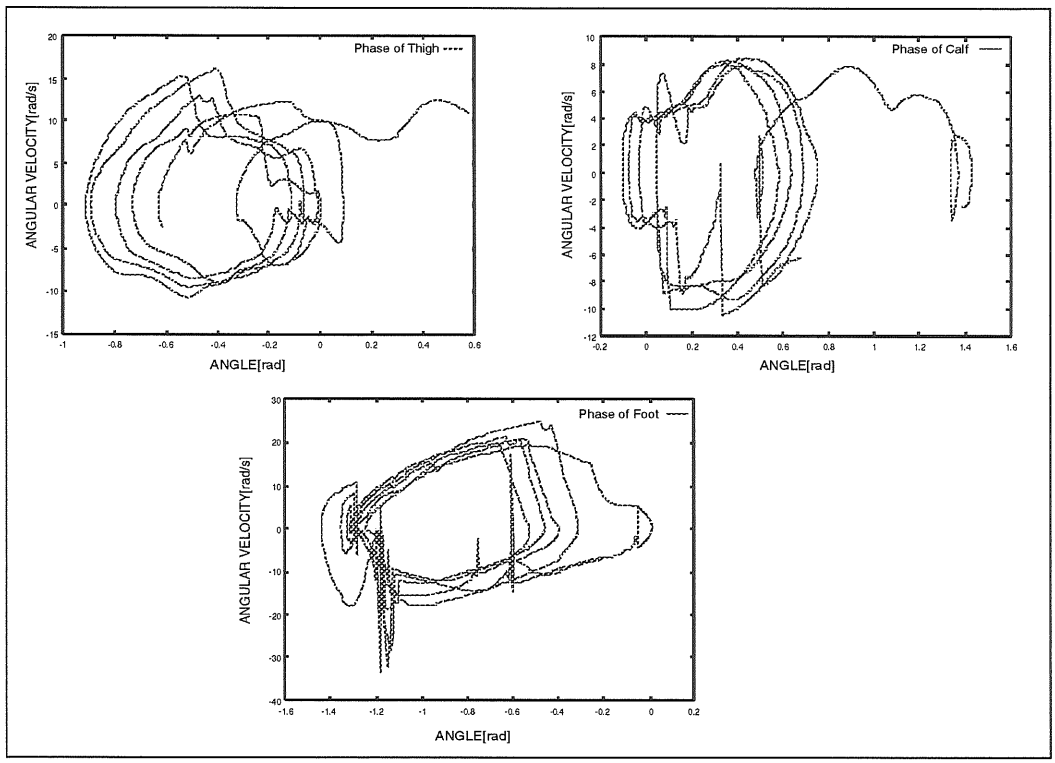

図 4.14: Phase of Thigh, Calf and Foot

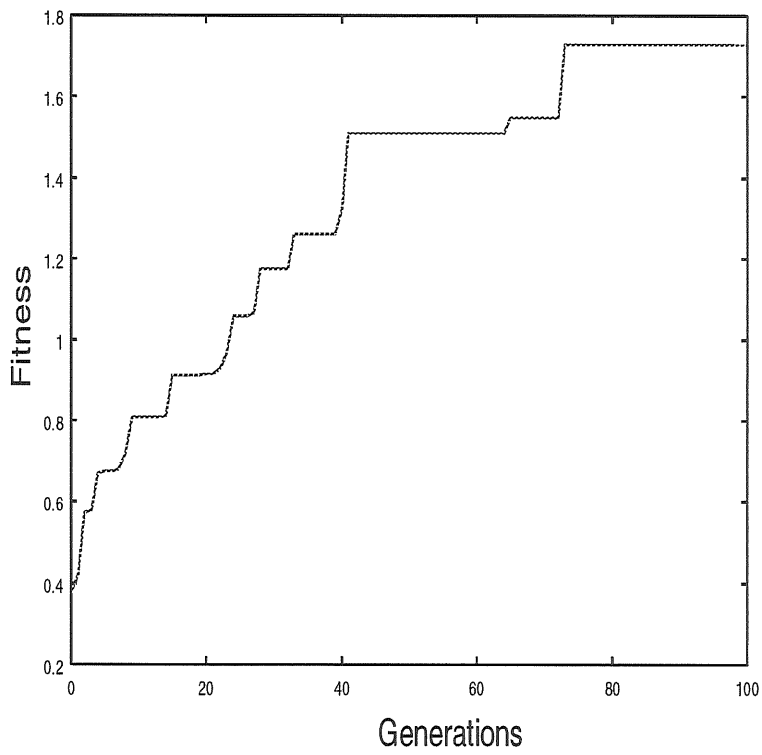

図 4.15: Transition of Walking Pattern Fitness

表 4.13: 本シミュレーションで得られたグローバルフィードバックネットワークモジュール

$$
\begin{aligned}
f_0 &= K(K((-0.29749 + ((K(s^r(1.0 - s^l)))(K(s^r)))) - (s^r(1.0 - s^l)))) \\
f_2 &= ((((K(r_7^y)) - (H(K(0.17454)))) - (((0.39401 - 8.34390) - ((s^l s^r) \\
&\quad (s^l(1.0 - s^r))))((K((K(0.17454)) + (0.39401 - 8.34390))) \\
&\quad (K((PI)(s^l))))))(K(s^r))) - (((K(((H(\dot{r}_9^y)) - (((\dot{r}_{11}^y) - (r_2^y)) + \\
&\quad ((\dot{r}_9^y) + (\dot{a}_y^r)))) - ((H(H(s^l(1.0 - s^r))))(K(s^l(1.0 - s^r))))) - \\
&\quad ((K(((r_9^y) - (\dot{r}_{13}^y)) - (0.39401 - 8.34390))) + \\
&\quad (((H(H(98.77023 \times 330.51065))) - (((s^l s^r)(s^l(1.0 - s^r))) \\
&\quad (r_7^y)))((K(92.95238(s^r))) - (K(r_{12}^y))))) + (((K((((PI)(r_5^y)) + \\
&\quad ((1 - s^r) + 80.63203)) - (K(K(\dot{r}_{13}^y))))) - (H((((\dot{r}_8^y) - 0.85720) + \\
&\quad ((((s^l)(\dot{r}_{11}^y))(H(s^l(1.0 - s^r))))18.43823)) - (K(0.17454))))) - \\
&\quad ((H(0.38148))((K(0.39401 + (0.39401 - 8.34390))) \\
&\quad ((((PI)(s^l))(K(r_{13}^y))) - (\dot{r}_7^y)))))) \\
f_4 &= (K((((s^r(1.0 - s^l)) + (((\dot{r}_{12}^y)(PI)) + ((r_7^y)(s^r(1.0 - s^l))))) + \\
&\quad (\dot{r}_{13}^y)) - (s^l(1.0 - s^r)))) \\
f_6 &= (H((\dot{r}_{13}^y)3.85593))
\end{aligned}
$$

$$
\begin{aligned}
f_8 \;=\;& (((((K(r_{11}^y)) - (H(K(0.17454)))) - (((0.39401 - 8.34390) - ((s^l s^r) \\
& (s^r(1.0 - s^l))))((K((K(0.17454)) + (0.39401 - 8.34390))) \\
& (K((PI)(s^r))))))(K(s^l))) - (((K(((H(\dot{r}_{13}^y)) - (((\dot{r}_7^y) - (r_2^y)) + \\
& ((\dot{r}_{13}^y) + (\dot{a}_l^y)))) - ((H(H(s^r(1.0 - s^l))))(K(s^r(1.0 - s^l))))) - \\
& ((K(((r_{13}^y) - (\dot{r}_9^y)) - (0.39401 - 8.34390))) + \\
& (((H(H(98.77023 \times 330.51065))) - (((s^l s^r)(s^r(1.0 - s^l))) \\
& (r_{11}^y)))((K(92.95238(s^l))) - (K(r_8^y))))) + (((K((((PI)(r_5^y)) + \\
& (((10. - s^l)) + 80.63203)) - (K(K(\dot{r}_9^y)))) - (H((((\dot{r}_{12}^y) - 0.85720) + \\
& ((((s^r)(\dot{r}_7^y))(H(s^r(1.0 - s^l))))18.43823)) - (K(0.17454)))) - \\
& ((H(0.38148))((K(0.39401 + (0.39401 - 8.34390))) \\
& ((((PI)(s^r))(K(r_9^y))) - (\dot{r}_{11}^y)))))))
\end{aligned}
$$

$$
\begin{aligned}
f_{10} \;=\;& (K((((s^l(1.0 - s^r)) + (((\dot{r}_8^y)(PI)) + ((r_{11}^y)(s^l(1.0 - s^r))))) + \\
& (\dot{r}_9^y)) - (s^r(1.0 - s^l))))
\end{aligned}
$$

$$
f_{12} \;=\; (H((\dot{r}_9^y)3.85593))
$$

$$
\begin{aligned}
f_{14} \;=\;& (((H(((K((H((a_r^y) - 0.46037)) - -0.30068)) - (K((r_{11}^y) \\
& (s^l(1.0 - s^r)))))(\dot{r}_7^y)))(a_y^l)) - (s^l(1.0 - s^r)))
\end{aligned}
$$

$$
\begin{aligned}
f_{16} \;=\;& (((H(s^l)) - (K(((H(H(H(H(r_{13}^y)))))((K(a_y^r)) - (0.35471 + \\
& (K((s^l s^r)(s^l(1.0 - s^r)))))))(((s^l)(H(s^l)))((PI) + ((((s^r) + \\
& ((a_y^r))(K((s^r) + (\dot{r}_{12}^y))))(r_5^y))))))) + (((((1.0 - s^l))(s^l(1.0 - s^r))) + \\
& (H(1.0 - s^r))) + ((H(H(((H(K(-0.31001 + (1.0 - s^r)))) \\
& (77.11329 - ((1.0 - s^r)))) - (H(K(a_y^r)))))) + (((K(r_9^y)) - \\
& (((H(309.04040)) + ((s^r(1.0 - s^l)) + 55.43011))(K(K(s^l)))))
\end{aligned}
$$

$$
\begin{aligned}
& (H(((36.37975 \times 0.46265)(((s^l(1.0 - s^r))((r_8^y) - (\dot{r}_2^y)))\\
& (216.18030(s^l)))) - (H(K(a_y^r)))))))))\\
f_{18} \;=\; & (((H(s^r)) - (K(((H(H(H(H(r_9^y)))))((K(a_y^l)) - (0.35471 +\\
& (K((s^l s^r)(s^r(1.0 - s^l)))))))((s^r)(H(s^r)))((PI) + ((((s^l) +\\
& (a_y^l))(K((s^l) + (\dot{r}_8^y))))(r_5^y))))))) + ((((1.0 - s^r)(s^r(1.0 - s^l))) +\\
& (H((1.0 - s^l)))) + ((H(H(((H(K(-0.31001 + ((1.0 - s^l)))))\\
& (77.11329 - ((1.0 - s^l)))) - (H(K(a_y^l)))))) + (((K(r_{13}^y)) -\\
& (((H(309.04040)) + ((s^l(1.0 - s^r)) + 55.43011))(K(K(s^r)))))\\
& (H(((36.37975 \times 0.46265)(((s^r(1.0 - s^l))((r_{12}^y) - (\dot{r}_2^y)))\\
& (216.18030(s^r)))) - (H(K(a_y^l)))))))))\\
f_{20} \;=\; & ((r_7^y)((K((H(\dot{a}_y^r)) - (H((a_y^l) - 0.00986))))(H(PI))))\\
f_{22} \;=\; & ((((s^r(1.0 - s^l)) - 89.92130) - (43.75387(((((a_y^l) - (a_y^r)) -\\
& ((a_y^l) - (s^l)))(H(H((1.0 - s^l))))) - (H(s^r))))) + (H(((s^l s^r)\\
& -0.59105)((s^l s^r) + (r_7^y)))))
\end{aligned}
$$

4.4.3 運動評価基準関数の考察

本研究で用いた歩行運動の評価基準関数 4.1は転倒するまでの歩行距離, 歩数, 身体の横ずれ, そして体重心の高さの変化となる重み付き線形和である. この運動評価基準関数として運動の合目的性を仮定し, これを式で表したものであるが, 歩行における運動評価基準は必ずしも適当だとは言えない. ここで評価関数の重み係数に関する比較検討実験をおこなった. 図 4.16, 図 4.17, 図 4.18はそれぞれ評価関数に適応した歩容を遺伝的プログラミングにより生成したものである. それぞれの結果は適応度が最も高かった個体に関するものである.

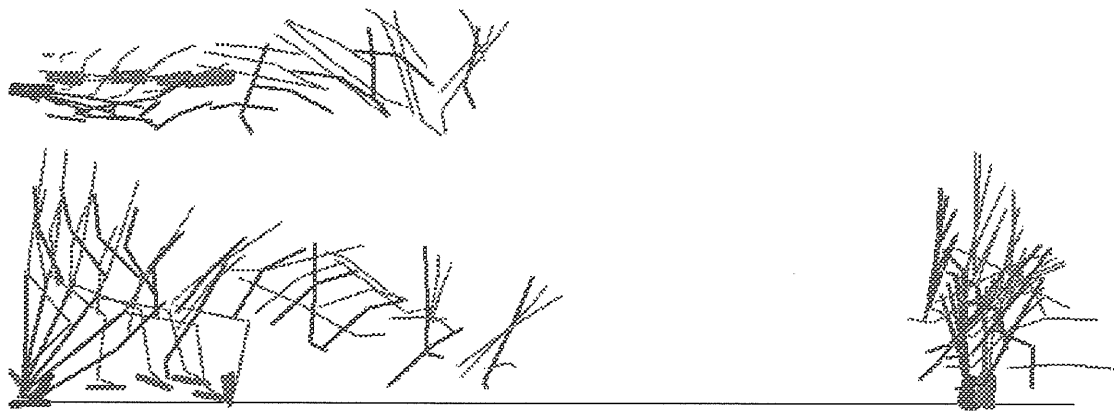

図 4.16: 運動評価基準関数：例 (I=0.1D+0.01S)

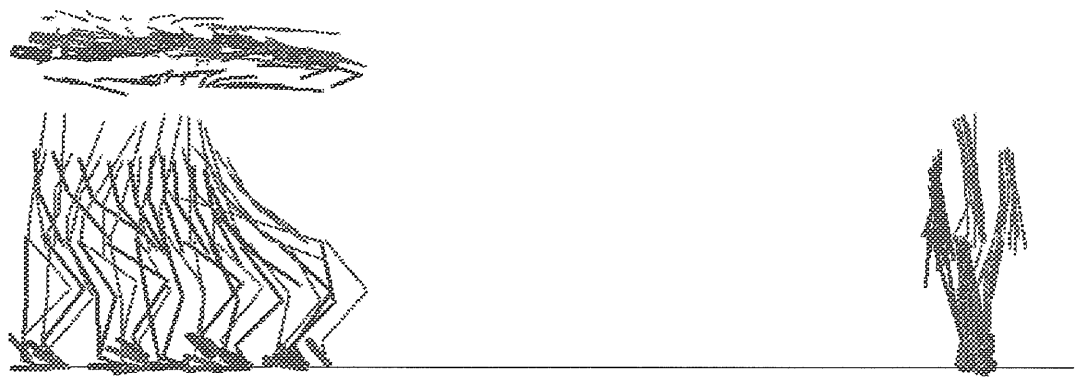

図 4.17: 運動評価基準関数：例 (I=0.1D+0.1S)

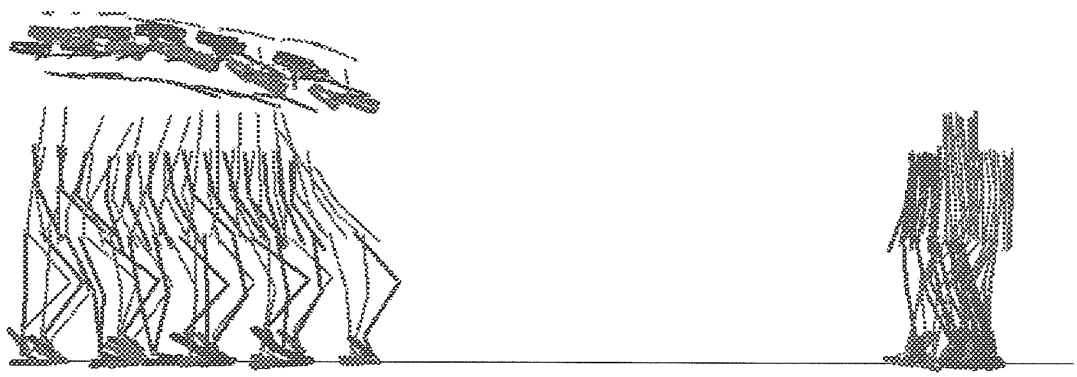

図 4.18: 運動評価基準関数：例 (I=0.18D+0.1S)

運動の評価基準で用いた歩行距離と歩数は密接な関係を持っている. たとえば, 図 4.16のように歩行距離が歩数よりも重み係数を 10 倍程度重くした場合, 進化の方向がなるべく歩行距離を伸ばす方に行くことによって, 途中でジャンプするのように歩行運動が表れた. そして, 図 4.17と図 4.18のように歩行距離の重み係数の割合を下げると足踏みのような歩行運動が生成された.

本実験では歩行距離と歩数などの比較的に簡単な評価関数を用いて, ある程度妥当な歩容を生成することは可能であったが, 歩行における運動評価基準としては必ずしも適当と言えない. 今後の本研究の課題として, 歩行の周期や歩幅など人間の歩行運動に対して精密な生体力学的な分析を行うことで, より有効な評価関数を定める必要がある.

第 5 章

遺伝的プログラミングの探索効率の改善

遺伝的プログラミングにおける染色体表現は, 関数ノードと終端ノードの組合せによって構成された木構造であり, 実行可能なプログラムを表現する. 木を構成するノードは, 対象問題領域の解を表現するために適当に設計された固定の記号集合である. 遺伝的プログラミングは, それらのノードで構成される階層的なプログラムの形やサイズをダイナミックに変えることにより, 構成可能なプログラムの空間の中で, より良いプログラムを適応的・確率的に探索する手法である. 遺伝的プログラミングを用いて与えられた問題を効率的に解決するためには, 木構造を構成する各ノードを適切に設計することが重要であり, その効果的な設計には対象とする問題に関する十分な事前知識が必要とされる. 遺伝的プログラミングを適用しようとしている問題領域に関する事前知識が不足している場合には, 遺伝的プログラミングの適用を繰り返すことにより, 問題解決のために有用なノードを試行錯誤的に選ぶ必要がある. これは非常に時間のかかる面倒なプロセスであり, 問題領域に関する十分な事前知識を持たない現実的問題に遺伝的プログラミングを応用する際に解決すべき大きな障害の一つになる.

一般に, 遺伝的プログラミングで種々の問題を解く状況において, 必要最小限のノードをあらかじめ最適に決定することは非常に困難である. そのため, 遺伝的プログラミングの適用に当たっては, ノードを冗長に設計し, 解探索を行うプログラム空間を大きく設定する方が解の見逃しを防ぐという点でより現実的である. しかしながら, それ

により, 選択されたノード集合には無用なノードが多く含まれることになり, 遺伝的プログラミングの探索効率は拡張した仮説空間でむだな探索を行うことで低下してしまう. 遺伝的プログラミングの性能を改善するためには無関係なノードを取り除いて, 有用なノードだけを自動的に抽出することが望ましい.

遺伝的プログラミングにおいて適切なノードを決定する過程は, 他の機械学習パラダイムにおける特徴選択, あるいは特徴部分集合選択 [33, 79, 86] と同様の意味を持つ. 機械学習においては, 学習する概念に無関係な特徴が学習アルゴリズムに与えられていると, 概念の学習に必要な訓練事例数が爆発的に増大することが知られている. 特徴選択は, 与えられた特徴集合の中から, 概念の学習に有用な少数の特徴を識別・選択するものであり, 機械学習分野では活発に研究されている.

しかしながら, 遺伝的プログラミングにおける無関係なノードの問題は, 現在まであまり研究の対象とされてこなかった. 遺伝的プログラミングの進化過程においては, 評価の低いプログラムは自然と淘汰され, 集団内のプログラムにおける冗長なノードの割合は徐々に減少して行く [83] とされていた. しかしながら, 実際には真に有用な特徴量の規定が困難な現実の複雑な問題 (たとえば, 株価の変動予測問題など) に対して遺伝的プログラミングを適用する際には, 冗長なノードの存在によってもたらされる遺伝的プログラミングによる探索の非効率さは深刻な問題になる.

本研究では, 遺伝的プログラミングにおいて有用なノードの選択を加速化するための新しいアプローチに関して述べる. まず, 次章では機械学習における特徴選択の研究との関係について述べる. その後, 遺伝的プログラミングの進化過程で無用なノードを削除する新しいアプローチに関して詳しく述べる. 最後に, 我々が提案したアプローチの有効性を確かめるために無用なノードを含む 2 つの記号当てはめ式問題の実験を行ない, 提案した方法をこれらの問題に適用した結果に対する詳しい分析を行う.

5.1 機械学習における特徴選択

機械学習を困難にする要因の一つに, 特徴空間の次元数の問題がある. 一般に, 機械学習を行う場合, 利用者は訓練事例を記述するための特徴を増やしすぎる傾向にある. これは, 事例を記述する特徴の数を増やせば, それだけ学習に有効な情報が多く手に入り, 学習率が上昇すると期待できることによるものである. しかしながら, 事例記述のための特徴の数を増やせば増やすほど, 学習すべき概念とは無関係な特徴が混入する可能性も増大する. また, 特徴空間の次元の増大は探索空間の拡大を生じ, 適切な概念の獲得に必要な計算量を増加させる結果 (次元の呪い) に陥る. 一般的に, 概念学習に必要な訓練サンプル数は, 特徴数のとともに指数的に増えて行くのに対して, 特徴量の増大と共にデータ収集のコストは高くなるので, 実際に得られる訓練サンプル数は限られている場合が多い. したがって, 不用意に特徴空間の次元数を増やせば, かえって機械学習の性能は低下してしまう. このことから, 事前に適切に特徴選択を行い, 事例記述に用いる特徴数を削減することは機械学習の重要な課題の一つである.

特徴量の評価選択に関しては, これまでにもパターン認識や統計などの分野で研究が行われている. これまで機械学習方式においては特徴選択のために主として2つのアプローチが提案されてきた [33]. 一つは明示的なヒューリスティック探索による特徴部分集合選択法, もう一つは特徴重み付けによる特徴選択法である.

ヒューリスティック探索アプローチでは, 探索空間の各状態を可能な特徴の組み合わせの部分集合で記述する. ヒューリスティック探索アプローチには, 特徴部分集合に対する探索方向の観点から, 順方向選択手法と逆方向排除手法が提案されている. さらに, 学習プロセスとは独立した評価基準を用いて, 学習に対して事前に適切な特徴部分集合を選択するフィルター手法や, 学習プロセス自体を用いて最適な特徴部分集合の評価選択を行なうラッパー手法という分類も可能である. フィルタ手法では適切な帰納バイアスの設計が重要であるが, John らによって提案されたラッパー方法 [82] では, 属性集合の候補 (利用可能属性集合の部分集合) を, 学習アルゴリズムそのものを用

いて評価することを繰り返すことによって, 最適な属性集合を求められる. 属性集合の候補は, その属性のみから記述されたデータを学習アルゴリズムに入力し, 学習結果の精度を交差解析することによって評価される.

特徴重み付け法による選択では, 学習過程において学習目標 (概念) と特徴との関連深さの度合を反映する重み値が各々の特徴に割り当てられる. 一般的に, ヒューリスティック探索アプローチは, その探索結果が人間が理解できることが重要であったり, 結果が他のプログラムの入力になる場合には自然な手法である. 一方, 重み付けアプローチはオンラインの段階的な特徴選択が簡単に実現できるため, 純粋に学習性能のみを考慮した時, 重み付けアプローチが有利であることが多い. 一般に長時間にわたる遺伝的プログラミングの問題解決過程を考慮すると, 遺伝的プログラミングにおいて有用なノードを選択するには効率的なオンライン学習の方が好ましい. したがって, 本研究では遺伝的プログラミングにおけるノード選択方法として, ノードの重み付けに基づく選択手法を用いることにした. 次章では, 遺伝的プログラミングの進化過程において無用なノードを削除するアプローチに関して詳しく説明する.

5.2 遺伝的プログラミングにおけるノード選択

遺伝的プログラミングにおいて, ノード集合の設定は解の探索性能を大きく左右する最も重要な要素である. しかしながら, 生成されるべき望ましいプログラムにとって冗長, あるいは無関係なノードが存在することがもたらす問題は, 未だ研究者による十分な議論が行われていない. Koza は, 遺伝的プログラミングの基本的な枠組をまとめた最初の本 [81] の中で, 遺伝的プログラミングの進化過程においてはランダムな突然変異によってツリー集団中の冗長なノードは徐々に減少して行くので, 無関係なノードの存在は一般的に遺伝的プログラミングの探索効率を悪化させるだけで, 遺伝的プログラミングによる問題解決にとって深刻な問題ではないと述べている. しかしながら, 現実の問題においては, ノードとして利用可能な情報の中に, 問題解決に関連する

可能性はあるが, その関連性が不明確なものが数多く存在する. このような現実の問題に対して遺伝的プログラミングを適用する際には, 冗長なノードによって生じる非効率性は深刻な問題になる. 本章では, 遺伝的プログラミングを用いた問題解決過程において, 有用なノードの選択を加速化するためのノードの重み付けに基づいた新しい突然変異手法について述べる.

5.2.1 ノードの重み付け方法

機械学習における一般的な特徴重み付け方法では, 概念の学習に有用な特徴の重みは学習過程の中で段階的に増加される. 我々は, 遺伝的プログラミングにおいて有用なノードを抽出するために, 遺伝的プログラミングの処理過程で個体 (プログラム) の適合度評価が行われる際に, より適合度の高いプログラム中で用いられているノードに対して, その重みを増加させる処理を反復的に繰り返して, 各ノードの重みの更新を行う. 提案したノードの重み付けの方法では, ノードの重みは以下の式を用いて更新される.

$$W_n(g) = \sum_{i \in S_g} (fit(i) * freq_n(i)) + W_n(g-1)$$

ただし,

$W_n(g)$: 世代 g でのノード n の重み

$fit(i)$: 個体 i の適応度 (非負の値)

$freq_n(i)$: 個体 i におけるノード n の出現回数

S_g: 世代 g における上位 10% の適応度を持つ個体の集合

適合度の高い (集団全体の上位 10%) プログラムに含まれているノードには, そのプログラムの質 (適合度) とプログラム中のノードの出現頻度にしたがって重みが加えられる. このようにして, ノードがプログラムの適合度に与える影響の大きさに応じて, ノードの有用性が繰り返し評価され, ノードの重みの値が更新される.

5.2.2 適応的突然変異

一般的に, 遺伝的プログラミングにおける突然変異処理は, 個体中からランダムに突然変異点を選択することから始まる. この突然変異点は, プログラム中の関数ノードでも終端ノードでも良い. 一般的なランダムな突然変異では, 選択した突然変異点以下のプログラムに含まれるサブツリーを削除し, 新しくランダムに生成されたサブツリーをその突然変異点に付け足す. 従来の遺伝的プログラミングでは, 通常すべてのノードに対して, 同一の確率で突然変異点が選択される. 我々は無用なノードを削除するために新しい突然変異操作を用いることにし, それを適応的突然変異と呼ぶ. 提案した適応的突然変異では, ノードの重みに応じて突然変異点が選択される. すなわち, 重みの小さなノードほど高い確率で突然変異点として選択される. そして選択されたノード以下のサブツリーはプログラムから削除される. したがって, 遺伝的プログラミングの進化過程において, ある程度の割合で突然変異が起きる場合には, 適応的突然変異によって無用なノードがプログラムに含まれる可能性が徐々に低くなっていく. 適応的突然変異の概念図を図 5.1に示す. 我々が提案した適応的突然変異は以下の 4 ステップから成る. 最初の 2 ステップは有用なノードの選択に関係し, 残る 2 ステップはサブツリーの削除と生成過程である. 各ステップに関して, 以下に詳しく説明する.

1. **ノード候補の抽出**

   問題解決にとって有用なノードと無用なノードを識別するために, ノードの重みに基づいてノード集合を 2 つのカテゴリにクラスタリングする. クラスタリングを行うに際して, まずノード重みに正規化を行い, 更に平滑化のために値域 ($[0, 1.0]$) を 10 等分に分割する. そして分割された各領域に対して, 個々のノード重みの値に応じて, 以下の式に基づき各領域におけるノードの分布を世代毎に更新する.

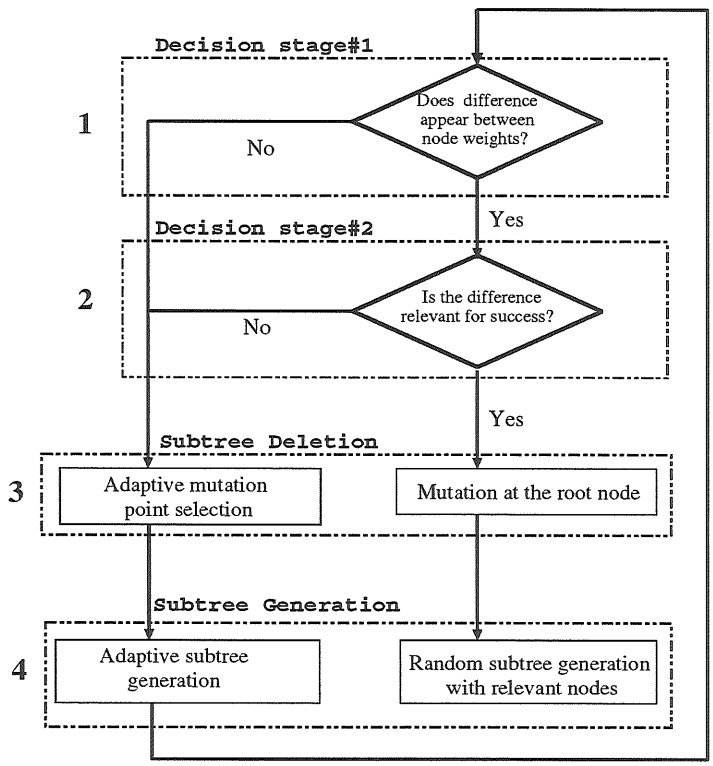

図 5.1: 適応的突然変異

$$H(i) = \begin{cases} +0.5 & (i = \lfloor 10\frac{W_n(g)}{\sum_{n=1}^{N} W_n(g)} \rfloor \text{のとき}) \\ +0.2 & (i = \lfloor 10\frac{W_n(g)}{\sum_{n=1}^{N} W_n(g)} \rfloor \pm 1 \text{のとき}) \\ +0.1 & (i = \lfloor 10\frac{W_n(g)}{\sum_{n=1}^{N} W_n(g)} \rfloor \pm 2 \text{のとき}) \end{cases}$$

ただし,

i: 領域インデックス (0 から 9)

N: ノード集合のサイズ

$W_n(g)$: 世代 g でのノード n の重み

$H(i)$: 領域 i におけるノードの累積分布

上の式において, $+0.1, 0.2, 0.5$ などの数値や $i = \lfloor 10 \times W_n(g) / \sum_{n=1}^{N} W_n(g) \rfloor \pm 2$ 等の条件は, ノード分布の形状を制御するために適当に定められたものである.

これらの値や条件を変えることにより，ノード識別のタイミングを調整することが可能である．このようにして世代ごとにノード重みの分布に対する累積ヒストグラムを求めると，世代を経るに従い累積ヒストグラムに双峰性が現れる．我々は，そうした双峰性の出現を，ノードの重み値を用いて有用なノードと無用なノードの識別を行うのに個体集合が十分に進化した兆候であり，ノードにはその有効性を判別するための証拠が重みとして十分に蓄積されたと考える．

上記のクラスタリングのアルゴリズムを簡単な例を用いて説明する．たとえば，第1世代での10個のノードに対してその重みを次の通りと考える：$W_1(1) = 3$, $W_2(1) = 2$, $W_3(1) = 4$, $W_4(1) = 5$, $W_5(1) = 6$, $W_6(1) = 7$, $W_7(1) = 14$, $W_8(1) = 26$, $W_9(1) = 0$, $W_{10}(1) = 6$．その場合，ノードの重みの合計 $\sum_{n=1}^{N} W_n(g)$ は 73 になる．さらに各ノードにおいて，$W_n(g)/\sum_{n=1}^{N} W_n(g)$ を求めると次のようになる．

$$W_1(1)/\sum_{n=1}^{N} W_n(1) = 0.041 \ (3/73)$$

$$W_2(1)/\sum_{n=1}^{N} W_n(1) = 0.027 \ (2/73)$$

$$W_3(1)/\sum_{n=1}^{N} W_n(1) = 0.053 \ (4/73)$$

$$W_4(1)/\sum_{n=1}^{N} W_n(1) = 0.068 \ (5/73)$$

$$W_5(1)/\sum_{n=1}^{N} W_n(1) = 0.082 \ (6/73)$$

$$W_6(1)/\sum_{n=1}^{N} W_n(1) = 0.095 \ (7/73)$$

$$W_7(1)/\sum_{n=1}^{N} W_n(1) = 0.191 \ (14/73)$$

$$W_8(1)/\sum_{n=1}^{N} W_n(1) = 0.356 \ (26/73)$$

$$W_9(1)/\sum_{n=1}^{N} W_n(1) = 0.000 \ (0/73)$$

$$W_{10}(1)/\sum_{n=1}^{N} W_n(1) = 0.082 \ (6/73)$$

したがって，

$$\lfloor 10 \times W_1(1)/\sum_{n=1}^{N} W_n(1) \rfloor = \lfloor 0.41 \rfloor = 0$$

$$\lfloor 10 \times W_2(1)/\sum_{n=1}^{N} W_n(1) \rfloor = \lfloor 0.27 \rfloor = 0$$

$$\lfloor 10 \times W_3(1)/\sum_{n=1}^{N} W_n(1) \rfloor = \lfloor 0.53 \rfloor = 0$$

$$\lfloor 10 \times W_4(1)/\sum_{n=1}^{N} W_n(1) \rfloor = \lfloor 0.68 \rfloor = 0$$

$$\lfloor 10 \times W_5(1)/\sum_{n=1}^{N} W_n(1) \rfloor = \lfloor 0.82 \rfloor = 0$$

$$\lfloor 10 \times W_6(1)/\sum_{n=1}^{N} W_n(1) \rfloor = \lfloor 0.95 \rfloor = 0$$

$$\lfloor 10 \times W_7(1)/\sum_{n=1}^{N} W_n(1) \rfloor = \lfloor 1.91 \rfloor = 1$$

$$\lfloor 10 \times W_8(1)/\sum_{n=1}^{N} W_n(1) \rfloor = \lfloor 3.56 \rfloor = 3$$

$$\lfloor 10 \times W_9(1)/\sum_{n=1}^{N} W_n(1) \rfloor = \lfloor 0.00 \rfloor = 0$$

$$\lfloor 10 \times W_{10}(1)/\sum_{n=1}^{N} W_n(1) \rfloor = \lfloor 0.82 \rfloor = 0$$

が得られる.

以上の結果から, 式 $H(i)$ を用いて各領域におけるノード重みの分布を求めると次のようになる.

$$\begin{aligned} H(0) &= 0.5 \times 8 + 0.2 = 4.2 \\ H(1) &= 0.2 \times 8 + 0.5 + 0.1 = 2.2 \\ H(2) &= 0.1 \times 8 + 0.2 + 0.2 = 1.2 \\ H(3) &= 0.1 + 0.5 = 0.6 \\ H(4) &= 0.2 \\ H(5) &= 0.1 \end{aligned}$$

さらに, たとえば第 9 世代においてノードの重みが以下のように変化したと考える : $W_1(9) = 13$, $W_2(9) = 32$, $W_3(9) = 574$, $W_4(9) = 14$, $W_5(9) = 602$, $W_6(9) = 14$, $W_7(9) = 29$, $W_8(9) = 570$, $W_9(9) = 6$, $W_{10}(9) = 43$. これら

の値に基づき, 上の同様に $H(i)$ を求めると, $H(0) = 3.5, H(1) = 1.7, H(2) = 1.3, H(3) = 1.5, H(4) = 0.6, H(5) = 0.3$ となる.

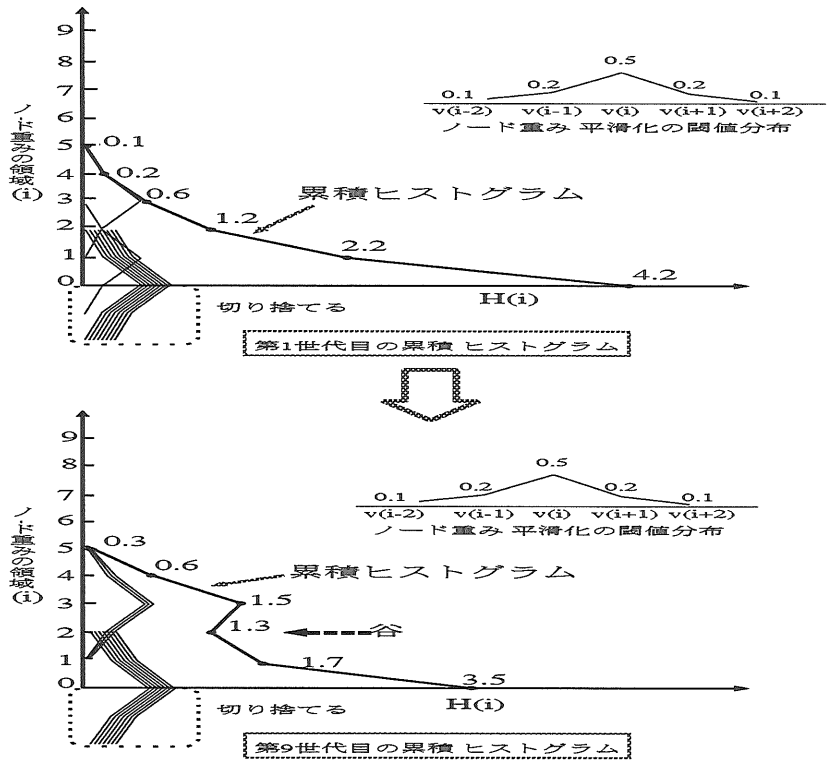

図 5.2: クラスタリング手法

このようにして各世代ごとに求められるノード重みの分布に対する累積ヒストグラムを図示すると, 図 5.2のようにある世代の累積ヒストグラムからはグラフに双峰性が現れ, ノードをその重みの大きさに応じて 2 つのグループにクラスタリングする際の指標として用いることができる.

2. **ノードの有用性の判定**

上で選ばれた大きな重みを持つカテゴリに属するノードが, 本当に問題解決のために有用なノードかどうかを判断する. そのために, 適合度の高いプログラムの中に, 選択されたノード候補が実際に利用されているかどうかを調査する. もし, 大きな重みを持つカテゴリに属するノード集合と, 適合度の最も高い上位 1% の

プログラム集団に含まれているノード集合が一致した時, そのノード集合は有用なノードとして判定される. この判定は, 遺伝的プログラミングにおける進化過程では, 十分に進化した適合度の高いプログラムには有用なノードだけが含まれている, という仮定に基づいている.

3. **サブツリーの削除**

突然変異操作では, まず突然変異点を決定し, 突然変異点の以下のサブツリーを削除する. 提案した適応的突然変異の場合, 有用なノードが見つかったかどうかによって, 以下の 2 つのタイプの突然変異点選択方法から該当するものが適用される.

- **有用なノードがまだ見つかっていない場合：**

  ノードの重みに反比例して突然変異点が選択される.

- **有用なノードが見つかった場合：**

  全てのツリーのルートを突然変異点として選択する. これは有用なノードだけで全集団を再初期化するためである. この処理は遺伝的プログラミングのプロセスにおいて, 1 度だけ行われる.

4. **サブツリーの生成**

突然変異点が選択された後, 新しいサブツリーを生成し, 突然変異点に挿入する. 適応的突然変異では, 有用なノードが見つかったかどうかによって 2 つのタイプのサブツリー生成方法がある.

- **有用なノードがまだ見つかっていない場合：**

  ノード重みの比例して選択したノードを用いてサブツリーを生成する.

- **有用なノードが見つかった場合：**

  有用なノードだけを用いてサブツリーを生成する.

有用なノードが定められた後は, 適応的突然変異は従来の遺伝的プログラミングのランダム突然変異と全く同様な操作を行う. ただし, 有用なノードが選択された後の進化過程は, 選択された有用なノードのみを用いて続けられる.

5.3 実験

提案した適応的突然変異手法の有効性を検証するために, 記号当てはめ問題を用いて実験を行った. 記号当てはめ問題はある未知関数 f に対する入力を $\{x_1, x_2, \cdots, x_n\}$, 出力を $y = f(x_1, x_2, \cdots, x_n)$ とし, 遺伝的プログラミングは, いくつかの入力値と出力値のペアを与えられて, f を同定, もしくは近似する式を与えられた関数と終端記号の組合せにより生成する.

本実験では, 記号当てはめ対象として 1 変数の関数と 3 変数の非線形関数の 2 種類を用意し, 各々の問題に対して全く問題とは無関係なノードを含む 33 個の終端ノードをあらかじめ遺伝的プログラミングに与えて実験を行った. 本実験では, 通常の遺伝的プログラミングの応用において, より重要な問題になる終端ノードに注目して実験を行ったが, 本研究で提案した手法は, 無用な関数ノードの削除にも全く同様に応用することができる.

本実験で実際に用いた記号当てはめ対象である 2 つの未知関数式は, 以下のとおりである :

- $f_1(x_1, \ldots, x_{33}) = x_1^3 + x_1^2 + x_1$
- $f_2(x_1, \ldots, x_{33}) = \sin(x_4) + \sin(2x_4) + \sin(3x_4) + \sin(4x_4) + \sin(x_5) + \sin(2x_5) + \sin(3x_5) + \sin(x_{13}) + \sin(2x_{13})$

各未知関数に対して, 各変数の値を [-1. 0, 1. 0] の範囲でランダムに選択した 200 組の訓練データを用意し, そのデータを遺伝的プログラミングに与えることで記号当てはめを行う. 本実験で用いた遺伝的プログラミングのパラメータは, 表 5.1に示すとお

りである. 本実験の実装は, 汎用的な遺伝的プログラミングツールである lil- 遺伝的プログラミング 1. 0 [116] を一部修正して行った.

5.3.1 単純な記号当てはめ式問題

単純な記号当てはめ問題として, Koza [81] の中で, 冗長な無効終端ノードの遺伝的プログラミングに対する影響を調べるために用いられたものと同じ以下の式を用いた.

$$y = x_1^3 + x_1^2 + x_1$$

この問題では, 用意された終端ノード集合に含まれる終端ノードのうち, 実際に式に用いられる有効終端ノードは 1 個だけで, 他の 32 個の終端ノードは記号当てはめすべき対象の式とは無関係な値を持つ. 各終端ノードは独立で, その値は $[-1.0, 1.0]$ の範囲でランダムに選択される. この実験で用いたその他のパラメータは表 5.1に示している. 本研究で示した結果は, ランダムシードを変更し, 100 回実験を試行した平均結果である.

多くの無効ノードが含まれる問題に対して遺伝的プログラミングを適用する際に, 適応的突然変異方法が有効ノードの選択に効果的であるか否かを調べるために, 本実験では次のような三つのタイプの実験を行い, その結果を比較した.

1. 無効ノードを含まない標準 GP(Standard GP(1))

   この実験は有効終端ノード x_1 だけが与えられたランダム突然変異の操作を行う標準的な遺伝的プログラミングを用いた実験である.

2. 無効ノードを含む標準 GP(Standard GP(2))

   この実験は有効終端ノード x_1 に無効終端ノード $x_2 \ldots x_{33}$ の 32 個を加えた終端ノード集合を用いたランダム突然変異の操作を行う標準的な遺伝的プログラミングを用いた実験である.

表 5.1: 遺伝的プログラミングパラメータ

Objective	Evolve a function that fits the data points of the fitness cases
Terminal set	$x_1 \ldots x_{33}$
Function set	$+, -, *, \%, \sin, \cos, \exp, rlog$
Fitness cases	The given sample of 200 data points $\{x_1(i), \cdots, x_{33}(i)\}$ each terminal's value is in the interval $[-1, +1]$, (1) $y = x_1^3 + x_1^2 + x_1$ (2) $y(i) = \sin(x_4)+\sin(2x_4)+\sin(3x_4)+\sin(4x_4)+\sin(x_5)+\sin(2x_5)+\sin(3x_5)+\sin(x_{13})+\sin(2x_{13})$
Raw fitness	The sum, taken over the 200 fitness cases, of the absolute value of difference between value of the dependent variable produced by the S-expression and the target value y_i of the dependent variable.
Hits	Number of fitness cases for which the value of the dependent variable produced bye the S-expression comes within 0.01 of the target value of the dependent variable.
Parameters	Population size = 2000, Generation = 200 for the function (1), Generation = 1000 for the function (2).
Crossover Probability	80%
Reproduction Probability	10%
Mutation Probability	10%
Success Predict	An S-expression scores 200 hits

3. 無効ノードを含む適応的 GP(Adaptive GP)

この実験は有効終端ノード x_1 に無効終端ノード x_2〜x_{33} の 32 個を加えた終端ノード集合で適応的突然変異の操作を行う遺伝的プログラミングを用いた実験である.

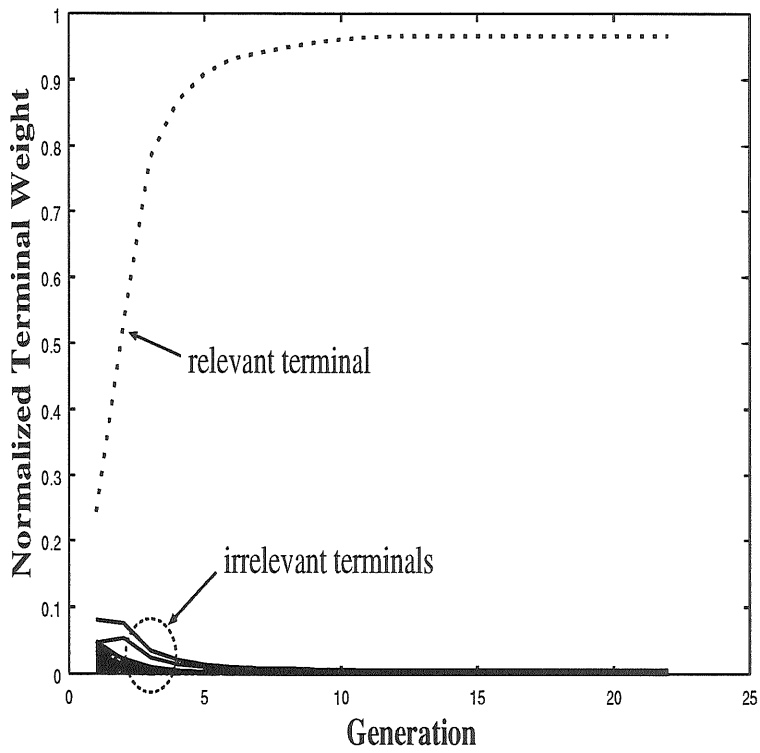

図 5.3: 簡単な記号当てはめ式の問題に対する終端ノード重みの変化

図 5.3は適応的 GP の実験において世代に伴う終端ノードの重みの変化を示したものである. このグラフでは 33 個の終端ノードの重みの総和が 1. 0 になるよう正規化されている. このグラフから, 適応的 GP は無効終端ノードを含む 33 個の終端ノード集合の中から, 比較的早い世代 (実際には 8 世代目) で有効終端ノードを抽出できたことがわかる. すなわち, 適応的突然変異におけるノード選択圧力は, ノード集合に多数の無効ノードが含まれている際に, 進化の方向性に対して大きな影響を与えることが可能であると考えられる. 図 5.4は, 上で言及した 3 タイプの遺伝的プログラミング

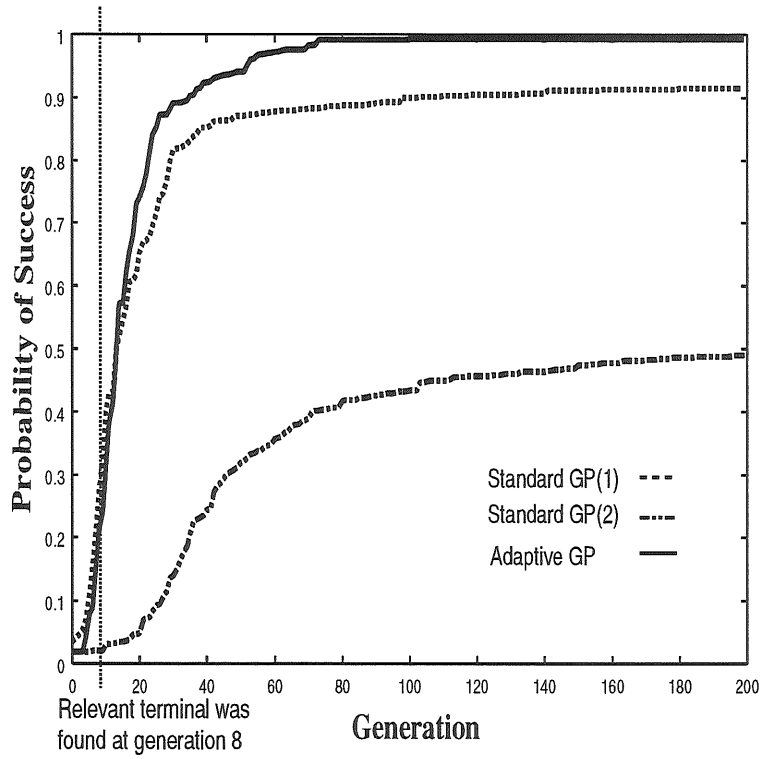

図 5.4: 簡単な記号当てはめ式問題における成功率

実験において解の成功確率を示している．このグラフから，適応的 GP は有効ノードを選択する前までは，有効ノードだけ含まれている標準 GP(1) よりも解探索効率の点で劣っているが，有効ノード集合が選択された後では，適応的 GP の性能は標準 GP(1) に匹敵することがわかる．また，適応的 GP や標準 GP(1) と標準 GP(2) の結果を比較すると，多くの無効ノードを含む標準 GP(2) の結果がかなり悪いことが分かる．

有効ノードが 1 つだけである簡単な記号当てはめ式問題の実験からは，適応的突然変異手法が効果的に有効ノードを見つけて，遺伝的プログラミングの性能向上に大きく寄与することがわかった．しかし，有効ノードが複数ある場合，適応的突然変異手法が全ての有効ノードを効率的に選択できるかどうかはこの実験では確認できていない．そこで我々は，有効終端ノードを 3 つ持つより複雑な記号当てはめ式の問題に対して実験を行った．

5.3.2 複雑な記号当てはめ式問題

2 番目の実験として，次のような式を未知関数とするより困難な記号当てはめ問題に対して，上と同様の実験を行った．

$$\begin{aligned} y \;=\; & \sin(x_4) + \sin(2x_4) + \sin(3x_4) + \sin(4x_4) + \\ & \sin(x_5) + \sin(2x_5) + \sin(3x_5) + \\ & \sin(x_{13}) + \sin(2x_{13}) \end{aligned}$$

この問題での終端ノードの値の設定と遺伝的プログラミングパラメータは，以前の簡単な記号当てはめ問題と同様である．本問題では，与えられた終端ノード集合に含まれる 33 個の終端ノードの内，有効終端ノードは x_4, x_5, x_{13} の 3 個である．更に，3 個の有効終端ノードは，個々の変数値の関数値に対する影響度が異なるため，無効終端ノードと有効終端ノードを区別するのが困難である．我々は，この問題でも前記と同様の 3 タイプの実験に対して，ランダムシードを変えて 100 回つづ実験を行った後，その平均

結果を比較した. 図 5.5は, 適応的 GP による世代に対する終端ノードの重みを変化を

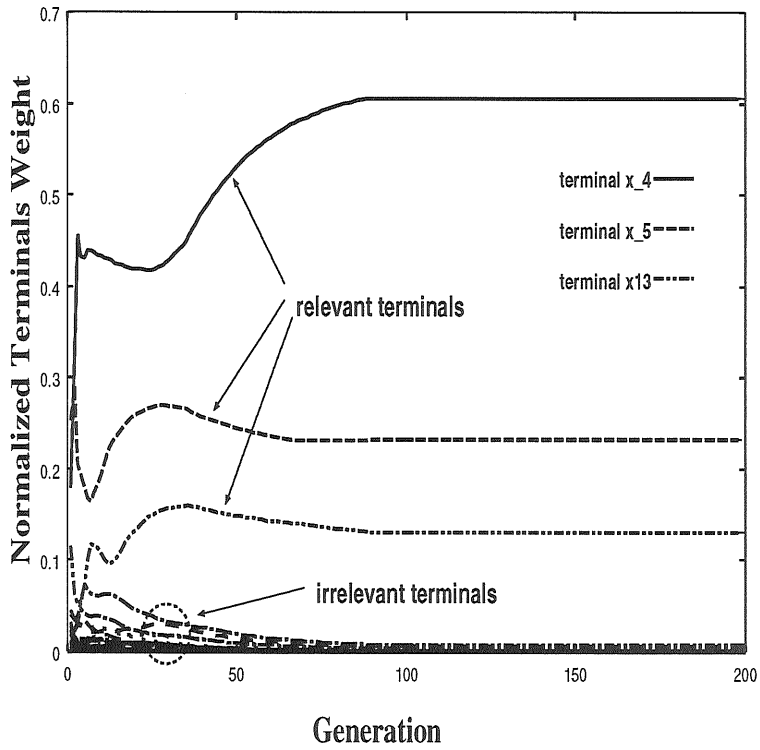

図 5.5: 複雑な記号当てはめ式の問題に対する終端ノード重みの変化

示している. このグラフから, 有効終端ノードと無効終端ノードの重みの差が世代が経つに連れ, 明らかになることがわかる. 本実験では, 適応的 GP が約 90 世代で有効終端ノードのすべてを選択するのに成功することが確認された. 図 5.6は, 3 タイプの遺伝的プログラミングによる解の成功確率を示している. 無効ノードを含まない標準 GP(1) での成功確率を比べることにより, 本問題が前記の単純な記号当てはめ問題よりも難しいことをがわかる. このグラフから, 初期段階では適応的 GP は無効終端ノードを含む標準 GP(2) と同程度の性能であるが, 世代が進むにつれて, 適応的 GP の性能が改善され, 有効終端ノードだけを含む標準 GP(1) の性能に近づいていくことがわかる. この結果から, 適応的 GP で用いられた適応的突然変異方法が多数の冗長なノードから複数の有用なノードを見つけるのに有効であることが確認された.

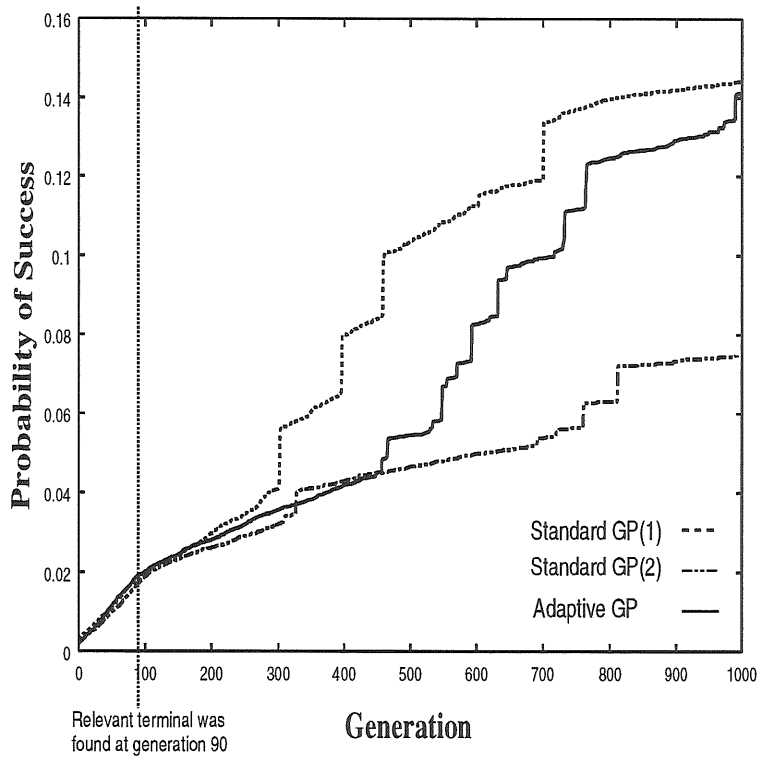

図 5.6: 複雑な記号当てはめ式問題における成功率

5.4 議論

遺伝的プログラミングにおいて, 世代の進展に伴って集団内の個々の木のコードサイズが肥大化することにより探索空間が拡大し, その処理効率が悪化することが知られており, bloat 現象 [32] などと呼ばれている. 近年ではそうした事態を防ぐために, 不要なノードあるいは部分木の除去, 発生の抑制をはかることに関して, 多くの研究事例が存在する（たとえば, 除去バイアスの研究 [101] など）. しかしながら, これらの研究で取り上げられている問題は, 本研究の取り扱っているのは別の冗長性に関する問題である. そこでは, 個々のノードの特質としてではなく, 他のノードや部分木との相対位置関係により, ノードの冗長性が決定される. そして, そのような関係性を維持したノード（部分木）が増殖することによる木の肥大現象を bloat と呼び, それを防ぐための交叉手法などの研究が盛んに行われている. すなわち, ノードの特質が問題解決に無関係であるか否かに関わらず, 他のノードや部分木との相対的位置関係により, そのノードが木の評価値に影響を与えない冗長な存在になり得ることが bloat 現象の要因の一つである. 本研究では, 解決すべき問題と個々のノードとの関係を, 他のノードや部分木との相対的位置関係とは独立に評価し, 問題解決に本質的に関係しないノードを, 遺伝的プログラミングの処理過程においてできるだけ早い段階で, 遺伝的操作の対象から外すことにより, 遺伝的プログラミングにおける探索効率の改善を図ることを意図しており, bloat に関する研究とは立場を異にする. しかしながら, 本研究で得られた研究成果は, bloat に関する研究によって得られた効率改善手法と相反するものではなく, むしろ併用することにより, 遺伝的プログラミングの更なる効率改善が可能になるものと考えられる.

遺伝的プログラミングの効率化という点において他の関連深い研究としては, ADF (Automatic Defined Functions) [83] や MA(Module Acquisition) [78] などのような自動関数生成に関する研究がある. これらは有用な部分木を切り出して破壊的な交叉の影響を受けないように保護することにより, そのような有用な部分木が集団全体に

広く使われることを促進する手法である. これは機械学習の研究においては, 個々の特徴量を組合わせて, 概念獲得により有効な特徴量を新たに生成する特徴構築に近い技術と考えることができ, 本研究の成果を組合わせて適用することにより, より適切なモジュールの生成が可能になると考えられる.

従来の遺伝的プログラミングの研究, 応用においては解決すべき問題と無関係なノードが多少木に含まれていても, それらは突然変異により自然に減少していくものであり, あまり深刻に対処する必要がないと考えられており, それが遺伝的プログラミングを利用する利点の一つであるとされていた. 更に, そのような考えを支持する応用結果もいくつか発表されている [61]. それに対して, 本研究は無関係なノードの存在が遺伝的プログラミングの効率を著しく損なうことがあり, そのような場合でも我々の提案する手法を適用することにより, 遺伝的プログラミングの効率の効果的な改善が可能であることを実験的に示したもので, 従来の遺伝的プログラミングの効率化に関する研究とは, 問題意識や手法の点で大きく異なるものである.

また本来, 遺伝的プログラミングにおける不要なノード, あるいは不要な部分木の定義は, 本研究中の実験で扱われているように明確である場合は少なく, ある世代までは全く使われていなかったノード, あるいは部分木が, その世代以降では重要な役割を果たすものとして使用される場合も多々あり, 初めから”不要である”ことが自明の終端記号を除去することにどの程度の意味があるかも意見がわかれるところと考えられるが, 本研究ではノードの重みの更新の式 ($W_n(g)$) において, ノードの重みは木の適応度をベースに計算されているので, 早い世代において木の適応度が低い段階でのノードの重みは大きく更新されず, ある程度世代を経て, 重要なノードが揃い始め, 木の適応度が高まった段階からノードの重みの更新も加速される. 有効なノードを正しく選択することができるかどうかは, どの時点でノードの有効性を最終的に判断するかというタイミングに大きく関わっくると考えられるが, 本研究では, 大きな重みを持つカテゴリに属するノード集合と, 適合度の最も高い上位 1% のプログラム集団に含まれているノード集合が完全に一致した時に初めて, そのノード集合を有用なノード

として判定することにより，ノードの有効性の判断を保守的に行っている．それにより，上記のような場合でも本手法により正しく有効なノードの選択を行うことが可能であると考えられる．

第 6 章

結論

本研究では 2 足歩行運動生成のための神経系構造を自動的に獲得する計算機シミュレーション手法の開発を行なった．進化的計算手法の 1 つである遺伝的プログラミングを用いて，人為的な試行錯誤を行なうことなく 2 足歩行運動を実現するための神経系構造を獲得することができた．通常の制御手法では，歩行生成のための目標軌道が存在し，かつその軌道は 1 周期性，つまり左右の足の運びは常に同一である．これに対して，本シミュレーションでは歩行の周期性を前提とせず，実際現れた歩行パターンでは左右の足の運びはまちまちでだった．これは，力学的条件に応じて多様な歩行パターンを生み出す能力を示唆し，生物の歩行に類似しうるものとして考えられる．Taga らの研究と主な違いは (1) 身体力学モデルが 3 次元であること，(2) 自動的神経系の構造を決定したことである．しかし、力学系の同調あるいはリミットサイクルといった特性は共通に現れた．多賀らの神経系モデルのような試行錯誤的に定めていた過去の研究に比べ，本研究のアプローチは人為的な努力を大幅に削減することができ，さらに様々な体系の運動パターンを発生する際にも柔軟に応用することができると考られる．

また，神経系の構造の探索ために単純な運動評価基準を用いたにも関わらず，ある程度の歩行運動を生成することができた．これは，本研究で開発手法が自律的な神経系構造を決定するのに有効であるが確認できた。

本研究の応用として，身近かなもので，ロボットの制御系の改良などが考えられる．そ

の他にも, 本手法を応用すれば, 様々な運動に適応した神経系構造を持つ自律的な運動発生機構を簡単に構築することができるため, 障害動作の回復予測やスポーツ訓練効果予測などの様々な分野への神経筋骨格モデルの応用が期待できる.

さらに, 本研究では遺伝的プログラミングの解探索性能を高めるために, 冗長なノード, 無用なノードが含まれているノード集合から, 問題解決に有効なノード集合を決定するための新しい手法として適応的突然変異手法を提案した. 本手法では, 遺伝的プログラミングによる解探索過程において, オンラインで問題解決に有効なノード集合を自動的に獲得し, 人間の専門家が持つ問題領域に関する知識を前提としない. 本研究では, 冗長なノード集合が与えられた記号当てはめ問題を用いて, 提案した手法が世代の進展と共に適切な有効ノードを選択し, 標準的な遺伝的プログラミングよりも解品質や効率の点で高い性能を示すことを確認した.

6.1 今後の課題

安定的かつ生体力学的に妥当な歩容を発生するためには, 適切な神経系の構造を決定しなければならない. そのためには, 合目的な運動評価基準を定める必要がある. 本シミュレーションで用いた運動評価基準 (式 4.1) は合目的な運動評価基準とは言えない. 本研究の今後の課題として, (1) 生体力学的特性を考慮し、合目的な運動評価基準を定めることである. 今現在, 考えているものとしてエネルギー効率の最大化と運動の滑らかさの最大化などがある. また (2) 遺伝的プログラミングのノードの設計が難しい本歩行シミュレーションに場合に対しても, 本研究で提案した適応的ターミナル選択手法を 2 足歩行運動生成システムにも適用することである.

謝辞

本研究を進めるにあたり、常に適切な御指導を賜りました独立行政法人産業技術総合研究所知能システム部門分散システムデザイングループ主任研究員及び筑波大学連携大学院工学研究科併任教官宮下和雄助教授, 筑波大学大学院工学研究科電子・情報専攻西原清一教授に心から感謝の意を表します. また適宜幾多の御助言を賜りました独立行政法人産業技術総合研究所長谷和徳主任研究員, 筑波大学大学院工学研究科電子・情報専攻福井幸男教授に深く感謝致します. 論文の執筆にあたっては独立行政法人産業技術総合研究所知能システム部門分散システムデザイングループ黒川治久主任研究員, 神村明哉研究員に感謝致します.

参考文献

[1] 五十嵐越朗, 馬飼享: “二足歩行運動における両脚支持相の力学解析と制御”. 日本ロボット学会誌, Vol 7(3), pp 30-38.

[2] 古荘純次, 山田誠: “角運動量を考慮した 2 足歩行ロボットの動的制御”. 計測自動制御学会論文誌, Vol 22(4), pp 451-458.

[3] 長谷和徳, 山崎信寿: “3 次元全身神経筋骨格モデルによる 2 足歩行のシミュレーション”. 日本機械学会 シンポジウム講演論文誌, No.98-31.

[4] 湯浅, 伊藤. : “分岐現象を用いた多様なパターンを生成する自律分散システム, 計測自動制御学会論文集, 27-11, pp. 1307-1314, 1991

[5] Manu Ahluwalia, Larry Bell, and Terence C. Fogarty. Co-evolving functions in genetic programming: A comparison in ADF selection strategies. In John R. Koza, Kalyanmoy Deb, Marco Dorigo, David B. Fogel, Max Garzon, Hitoshi Iba, and Rick L. Riolo, editors, *Genetic Programming 1997: Proceedings of the Second Annual Conference*, pp. 3–8, Stanford University, CA, USA, 13-16 July 1997. Morgan Kaufmann.

[6] E. Alba, J. F. Aldana, and J. M. Troya. Genetic algorithms as heuristics for optimizing ann design. In R. F. Albrecht, C. R. Reeves, and N. C. Steele, editors, *Proceedings of the International Conference on Artificial Neural Nets and Genetic Algorithms*, pp. 683–690. Springer-Verlag, 1993.

[7] Enrique Alba, Carlos Cotta, and Jose J. Troyo. Type-constrained genetic programming for rule-base definition in fuzzy logic controllers. In John R. Koza, David E. Goldberg, David B. Fogel, and Rick L. Riolo, editors, *Genetic Programming 1996: Proceedings of the First Annual Conference*, pp. 255–260, Stanford University, CA, USA, 28–31 July 1996. MIT Press.

[8] David Alderson. Toward a technique for cooperative network design using evolutionary methods. In John R. Koza, editor, *Genetic Algorithms and Genetic Programming at Stanford 1999*, pp. 1–10. Stanford Bookstore, Stanford, California, 94305-3079 USA, 15 March 1999.

[9] Ricardo Aler, Daniel Borrajo, and Pedro Isasi. Genetic programming and deductive-inductive learning: A multistrategy approach. In Jude Shavlik, editor, *Proceedings of the Fifteenth International Conference on Machine Learning, ICML'98*, pp. 10–18, Madison, Wisconsin, USA, July 1998. Morgan Kaufmann.

[10] Lee Altenberg. Emergent phenomena in genetic programming. In A. V. Sebald and L. J. Fogel, editors, *Evolutionary Programming — Proceedings of the Third Annual Conference*, pp. 233–241. World Scientific Publishing, 1994.

[11] Lee Altenberg. Evolving better representations through selective genome growth. In *Proceedings of the 1st IEEE Conference on Evolutionary Computation*, Vol. 1, pp. 182–187, Orlando, Florida, USA, 27-29 June 1994. IEEE.

[12] Lee Altenberg. Genome growth and the evolution of the genotype-phenotype map. In Wolfgang Banzhaf and Frank H. Eeckman, editors, *Evolution as a Computational Process*, pp. 205–259. Springer-Verlag, Berlin, Germany, 1995.

[13] Luis F. Alvarez and Vassili V. Toropov. Application of genetic programming to the choice of a structure of global approximations. In John R. Koza, editor, *Late Breaking Papers at the Genetic Programming 1998 Conference*, University of Wisconsin, Madison, Wisconsin, USA, 22-25 July 1998. Stanford University Bookstore.

[14] Bjorn Andersson, Per Svensson, Peter Nordin, and Mats Nordahl. Reactive and memory based genetic programming for robot control. In Riccardo Poli, Peter Nordin, William B. Langdon, and Terence C. Fogarty, editors, *Genetic Programming, Proceedings of EuroGP'99*, 第1598巻 of *LNCS*, pp. 161–172, Goteborg, Sweden, 26-27 May 1999. Springer-Verlag.

[15] Bjorn Andersson, Per Svensson, Peter Nordin, and Mats Nordahl. On-line evolution of control for a four-legged robot using genetic programming. In Stefano Cagnoni, Riccardo Poli, George D. Smith, David Corne, Martin Oates, Emma Hart, Pier Luca Lanzi, Egbert Jan Willem, Yun Li, Ben Paechter, and Terence C. Fogarty, editors, *Real-World Applications of Evolutionary Computing*, Vol. 1803 of *LNCS*, pp. 319–326, Edinburgh, 17 April 2000. Springer-Verlag.

[16] David Andre and John R. Koza. Parallel genetic programming on a network of transputers. In Justinian P. Rosca, editor, *Proceedings of the Workshop on Genetic Programming: From Theory to Real-World Applications*, pp. 111–120, Tahoe City, California, USA, 9 July 1995.

[17] David Andre and John R. Koza. Parallel genetic programming: A scalable implementation using the transputer network architecture. In Peter J. Angeline and K. E. Kinnear, Jr. , editors, *Advances in Genetic Programming 2*, chapter 16, pp. 317–338. MIT Press, Cambridge, MA, USA, 1996.

[18] P. J. Angeline. Morphogenic evolutionary computations: Introduction, issues and examples. In John Robert McDonnell, Robert G. Reynolds, and David B. Fogel, editors, *Evolutionary Programming IV: The Fourth Annual Conference on Evolutionary Programming*, pp. 387–401. MIT Press, 1995.

[19] P. J. Angeline and J. B. Pollack. Evolutionary module acquisition. In D. Fogel and W. Atmar, editors, *Proceedings of the Second Annual Conference on Evolutionary Programming*, pp. 154–163, La Jolla, CA, USA, 25-26 February 1993.

[20] Peter J. Angeline. Adaptive and self-adaptive evolutionary computations. In Marimuthu Palaniswami and Yianni Attikiouzel, editors, *Computational Intelligence: A Dynamic Systems Perspective*, pp. 152–163. IEEE Press, 1995.

[21] Peter J. Angeline. Evolution revolution: An introduction to the special track on genetic and evolutionary programming. *IEEE Expert*, Vol. 10, No. 3, pp. 6–10, June 1995. Guest editor's introduction.

[22] Peter J. Angeline. An investigation into the sensitivity of genetic programming to the frequency of leaf selection during subtree crossover. In John R. Koza, David E. Goldberg, David B. Fogel, and Rick L. Riolo, editors, *Genetic Programming 1996: Proceedings of the First Annual Conference*, pp. 21–29, Stanford University, CA, USA, 28–31 July 1996. MIT Press.

[23] M. L. Audu. and D. T Davy. "The influence of muscle model complexity in musculoskeletal motion modeling." *Transaction of the ASME Journal of Biomechanical Engineering.* 107, 147-157.

[24] Peter J. Angeline and K. E. Kinnear, Jr. , editors. *Advances in Genetic Programming 2.* MIT Press, Cambridge, MA, USA, 1996.

[25] Peter J. Angeline and Jordan B. Pollack. Competitive environments evolve better solutions for complex tasks. In Stephanie Forrest, editor, *Proceedings of the 5th International Conference on Genetic Algorithms, ICGA-93*, pp. 264–270, University of Illinois at Urbana-Champaign, 17-21 July 1993. Morgan Kaufmann.

[26] Peter J. Angeline, Gregory M. Saunders, and Jordan B. Pollack. An evolutionary algorithm that constructs recurrent neural networks. *IEEE Transactions on Neural Networks*, Vol. 5, pp. 54–64, 1994.

[27] Scott Austin. Genetic neurosynthesis. In *Proceedings of AIAA Aerospace VIII*, Baltimore, MD, October 1991.

[28] J. S. Bay, H. Hemani :Modeling of Neural Papptern Generator with Coupled Nonlinear Oscillators, *IEEE Trans. on Biomedical Engineering,* BME-34-4,pp 297-306, (1987).

[29] Randall D. Beer and John C. Gallagher. Evolving dynamical neural networks for adaptive behavior. *Adaptive Behavior*, Vol. 1, No. 1, 1992.

[30] Richard K. Belew and John McInerney. Using the genetic algorithm to wire feed-forward networks. Technical abstract, University of California, San Diego, La Jolla, CA, May 1989. Submitted to Neural Information Processing Systems 1989.

[31] Hamid Benbrahim and Judy A. Franklin. Biped dynamics walking using reinforcement learning. *Robotics and Autonomous Systems*, Vol. 22, pp. 283–302, 1997.

[32] Blickle, T. and Thiele, L. : Genetic Programming and Redundancy, *Genetic Algorithms in Framework of Evolutionary Computation (Workshop at KI-94)*, pp. 33–38 (1994).

[33] Blum, A. L. and Langey, P. : Selection of Relevant Features and Examples in Machine Learning, *Artificial Intelligence*, Vol. 97, pp. 245–271 (1997).

[34] H. Braun and J. Weisbrod. Evolving neural feedforward networks. In R. F. Albrecht, C. R. Reeves, and N. C Steele, editors, *Proceedings of the International Conference on Artificial Neural Networks and Genetic Algorithms*, pp. 25–32. Springer-Verlag, 1993.

[35] B. Bril and A. Ledebt. Head coordination as a means to assist sensory integration in learning to walk. *Neuroscience and Biobehavioral Reviews*, Vol. 22, No. 4, pp. 555–563, 1998.

[36] B. Calancie, B. Needhamshropshire, P. Jacobs, K. Willer, G. Zych, and B. A. Green. Involuntary stepping after chronic spinal-cord injury — evidence for a central rhythm generator for locomotion in man. *Brain*, Vol. 117, No. 5, pp. 1143–1159, 1994.

[37] Calancie B, Broton JG, Klose KJ, Traad M, Difini J and Ayyar DR : "Evidence that alterations in presynaptic inhibition contribute to segmental hypo- and hyperexcitability after spinal cord injury in man. "*EEG Clin Neurophysiol* 89:177-186.

[38] M. Cao and A. Kawamura. A design method of neural oscillatory networks for generation of humanoid biped walking patterns. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pp. 2357–2362, Piscataway, 1998. IEEE Computer Society.

[39] Maureen Caudill. Evolutionary neural networks. *AI Expert*, pp. 28–33, March 1991.

[40] M. Y. Cheng and C. S. Lin. Genetic algorithm for control design of biped locomotion. *Journal of Robotic Systems*, Vol. 14, No. 5, pp. 365–373, 1997.

[41] C. Chevallereau, A. M. Formal'sky, and B. Perrin. Low energy cost reference trajectories for a biped robot. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pp. 1398–1404, Piscataway, 1998. IEEE Computer Society.

[42] J. J. Collins and S. A. Richmond. Hard-wired central pattern generators for quadrupedal locomotion. *Biological Cybernetics*, Vol. 71, No. 5, pp. 375–385, 1994.

[43] G. S. Cymbalyuk, R. M. Borisyuk, U. Mueller-Wilm, and H. Cruse. Oscillatory network controlling six-legged locomotion. optimization of model parameters. *Neural Networks*, Vol. 11, No. (7-8), pp. 1449–1460, 1998.

[44] Hugo de Garis. Walker, a genetically programmed, time dependent, neural net which teaches a pair of sticks to walk. Technical report, Center for AI, George Mason Univ, Virginia, 1989.

[45] Hugo de Garis. Brain building with gennets. *Proceedings of INNC-90*, Vol. 2, pp. 1036–1039, 1990.

[46] Hugo de Garis. *Genetic Programming: Building Nanobrains with Genetically Programmed Neural Network Modules.* CADEPS AI Research Unit, Universitye Libre de Bruxelles, CP 194/7, B-1050 Brussels, Belgium, 1990.

[47] Hugo de Garis. Genetic programming: Evolution of a time dependent neural network module which teaches a pair of stick legs to walk. In *Proceedings of the 9th European Conference on Artificial Intelligence*, pp. 204–206, Stockholm, Sweden, AUG 6-10 1990.

[48] Hugo de Garis. Exploring gennet behaviors using genetic programming to explore qualitatively new behaviors in recurrent neural networks. In *Proceedings of the International Joint Conference on Neural Networks*, pp. III–547 – III–552. IJCNN-92, 1992.

[49] Hugo de Garis. Steerable gennets: the genetic programming of steerable behaviors in gennets. In F. J. Varela and P. Bourgine, editors, *Toward a Practice of Autonomous Systems. Proceedings of the First European Conference on Artificial Life*, Cambridge, MA, USA, 1992. MIT Press.

[50] Hugo de Garis. Circuits of production rule gennets. the genetic programming of artificial nervous systems. In *Artificial Neural Nets and Genetic Algorithms*, pp. 699–705. ANNGA93, 1993.

[51] Hugo de Garis. Incremental evolution of neural networks: Genetic programming in incremental steps. In *Proceedings of the World Congress on Neural Networks*, pp. II447 – II450. WCNN93, 1993.

[52] Garis, H. : "Genetic Programming Evolutionary Approaches to Multistrategy Learning", in *Proc of the Society of Instrument and Control Engineers (Japan)*, 21-40 (1993).

[53] M. R. Dimitrijevic, Y. Gerasimenko, and M. M. Pinter. Evidence for a spinal central pattern generator in humans. *Annals of the New York Academy of Sciences*, Vol. 860, pp. 360–376, 1998.

[54] J. Duysens and H. Van de Crommert. Neural control of locomotion; part 1. the central pattern generator from cats to humans. *Gait & Posture*, Vol. 7, pp. 131–141, 1998.

[55] John G. Elias. Genetic generation of connection patterns for a dynamic artificial neural network. In *Proceedings of the Conference on Combinations of Genetic Algorithms and Neural Networks*, pp. 38–54, 1992.

[56] B. Ermentrout and N. Kopell. Learning of phase lags in coupled neural oscillators. *Neural Computation*, Vol. 6, No. 2, pp. 225–241, 1994.

[57] D. Floreano and F. Mondada. Automatic creation of an autonomous agent: Genetic evolution of a neural-network driven robot. In *Proceedings of the Conference on Simulation of Adaptive Behavior*, 1994.

[58] David B. Fogel. Using evolutionary programming to create neural networks that are capable of playing tic-tac-toe. In *Proceedings of the International Conference on Neural Networks*, 1993.

[59] Y. Fujimoto and A. Kawamura. Simulation of an autonomous biped walking robot including environmental force interaction. *IEEE Robotics & Automation Magazine*, Vol. 5, No. 2, pp. 33–42, 1998.

[60] T. Fukuda, Y. Komata and T. Arakawa: "Recurrent Neural Network with Self-adaptive GAs for Biped Locomotion Robot," *Proc. of the 1997 Int'l Conference on Neural Networks*, Vol. 3 of 4, pp1710-1715 (1997)

[61] Gilbert, R. , Goodacre, R. , Shann, B. , Taylor, J. , Rowland, J. and Kell, D. : Genetic Programming-Based Variable Selection for High-Dimensional Data, *Genetic Programming 1998: Proceedings of the Third Annual Conference*, Morgan Kaufmann, pp. 109–115 (1998).

[62] S. Gracovetsky,: "An hypothesis for the role of the spine in human locomotion: A challenge to current thinking", *Journal of Biomedical Engineering.* , Vol 7, pp 205-216

[63] Frederic Gruau. Genetic synthesis of modular neural networks. In *Proceedings of the Fifth International Conference on Genetic Algorithms*, 1993.

[64] Frederic Gruau. Genetic micro programming of neural networks. In Kim Kinnear, editor, *Advances in Genetic Programming.* MIT Press, 1994.

[65] P. J. B. Hancock. Optimising parameters in neural net simulations by genetic algorithm. In *Mini-Symposium on Neural Network Computation.* Rank Prize Funds, Broadway: unpublished, 1989.

[66] P. J. B. Hancock. *Coding Strategies for Genetic Algorithms and Neural Nets.* PhD thesis, Department of Computing Science and Mathematics, University of Stirling, 1992.

[67] S. A. Harp, T. Samad, and A. Guha. Towards the genetic synthesis of neural networks. In J. D. Schaffer, editor, *Proceedings of the Third International Conference on Genetic Algorithms*, pp. 360–369. Morgan Kaufmann, 1989.

[68] K. Hase and N. Yamazaki. Computational evolution of human bipedal walking by a neuro-musculo-skeletal model. In *Proceedings of the Third International Symposium on Artificial Life and Robotics*, pp. 174–177, 1998.

[69] Hase, K. and Yamazaki, N. : "Computational evolution of human bipedal walking by a neuro-musculo-skeletal model", *Artificial Life and Robotics*, 3, 133-138 (1999).

[70] K-U. Hőffgen, H. P. Siemon, and A Ultsch. Genetic improvement of feedforward nets for approximating functions. In H-P. Schwefel and R. Mănner, editors, *Proceedings of the Conference on Parallel Problem Solving from Nature*, pp. 302–306. Lecture notes in Computer Science 496, Springer Verlag, 1991.

[71] Jianjuen Hu, Jerry Pratt, and Gill Pratt. Adaptive dynamic control of a bipedal walking robot with radial basis function neural networks. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 400–405, Piscataway, NJ, 1998. IEEE Computer Society.

[72] Iba, H. : " Evolving multiple agents by genetic programming", in *Advances in Genetic Programming 3*, 447-466 (1999).

[73] Akira Ito and Hiroyuki Yano. The emergence of cooperation in a society of autonomous agents – the prisoner's dilemma game under the disclosure of contract histories –. In Victor Lesser, editor, *ICMAS-95 Proceedings First International Conference on Multi-Agent Systems*, pp. 201–208, San Francisco, California, USA, 12–14 June 1995. AAAI Press/MIT Press.

[74] Ito, S. , Yuasa, H. , and Luo, Z, Ito, K. : "Generation of Locomotion Patterns according to the Energy Consumption", in *Proc. of 12th. Annual Conf. of Robotics Society of Japan*, 1159-1160 (1994).

[75] Satoshi Ito, Hideo Yuasa, Zhi wei Luo, Masami Ito, and Dai Yanagihara. A mathematical model of adaptive behavior in quadruped locomotion. *Biological Cybernetics*, Vol. 78, pp. 337–347, 1998.

[76] Kimura, S., Yano, M. and Shimuzu,H. : "A self-organizing model of walking patterns of insects." *Biological Cybernetics*. 69, pp. 183-193

[77] Kimura, H. , Sakurama, K. and Akiyama, S. : "Dynamic Walking and Running of the Quadrupled Using Neural Oscillator," in *Proc. of IROS98*, 50-57 (1998).

[78] Kenneth E. Kinnear, J. : Alternatives in automatic function definition: A comparison of performance, *Advances in Genetic Programming* (Kenneth E. Kinnear, J. (ed.)), MIT Press, pp. 119–141 (1994).

[79] Koller, D. and Sahami, M. : Toward optimal feature selection, *Proc. 13th International Conference on Machine Learning*, San Francisco, ICML, CA: Morgan Kaufmann, pp. 284–292 (1996).

[80] Koza, J. R. and Rice, J. P. : "Genetic Generation of Both the Weights and Architecture for a Neural Network", In *Proc. of the IEEE International Joint Conference on Neural Networks*, 2, 397-404 (1991)

[81] Koza, J. : *Genetic Programming: On the Programming of Computers by Means of Natural Selection*, The MIT Press, Cambridge, Massachusetts (1992).

[82] John, G. , Kohavi, R. and K, P. : Irrelevant features and the subset selection problem, *In Machine Learning: Proceedings of the Eleventh International Conference (ICML-94)*, New Brunswick, ICML, NJ: Morgan Kaufmann, pp. 121–129 (1994).

[83] Koza, J. : *Genetic Programming II: Automatic Discovery of Reusable Programs*, The MIT Press, Cambridge, Massachusetts (1994).

[84] Ming-Yi Lay. Application of genetic programming in analyzing multiple steady states of dynamical systems. In *Proceedings of the 1994 IEEE World Congress on Computational Intelligence*, pp. 333–336b, Orlando, Florida, USA, 27-29 June 1994. IEEE Press.

[85] Alain Leroux, J. Fung, and H. Barbeau. Adaptation of the walking pattern to uphill walking in normal and spinal-cord injured subjects. *Experimental Brain Research*, Vol. 126, No. 3, pp. 359–368, 1999.

[86] Liu, H. and Setiono, R. : A probabilistic approach to feature selection - A filter solution, *In 13th International Conference on Machine Learning (ICML'96)*, Bari, Italy, ICML, NJ: Morgan Kaufmann, pp. 319–327 (1996).

[87] Sean Luke and Lee Spector. Evolving teamwork and coordination with genetic programming. In John R. Koza, David E. Goldberg, David B. Fogel, and Rick L. Riolo, editors, *Genetic Programming 1996: Proceedings of the First Annual Conference*, pp. 150–156, Stanford University, CA, USA, 28–31 July 1996. MIT Press.

[88] Meyer,J. : "Evolutionary Approaches to Neural Control in Mobile Robots", in *Proc. of the IEEE International Conference on Systems, Man and Cybernetics* (1998).

[89] Miyashita,K. : "Job-Shop Scheduling with Genetic Programming", *Proc. of the Genetic and Evolutionary Computation Conference (GECCO)* , pp. 505-512, 2000.

[90] Sanjay Mishra. Control of biped locomotion using oscillators. Master's thesis, University of Maryland, 1993. Also available as Institute for Systems Research technical report

[91] Seiichi Miyakoshi, Gentaro Taga, Yasuo Kuniyoshi, and Akihiko Nagakubo. Three dimensional bipedal stepping motion using neural oscillators — towards humanoid motion in the real world. In *Proceedings of IEEE/RSJ International*

Conference on Intelligent Robots and Systems, pp. 84–89, Piscataway, NJ, 1998. IEEE Computer Society.

[92] Fumio Miyazaki and Suguru Arimoto. A control theoretic study of dynamical biped locomotion. *J. Dynamic Systems, Meas. and Control, ASME Trans*, Vol. 102, pp. 233–239, 1980.

[93] U. Mullerwilm. A neuron-like network with the ability to learn coordinated movement patterns. *Biological Cybernetics*, Vol. 68, No. 6, pp. 519–526, 1993.

[94] Jun Nishii. A learning model for oscillatory networks. *Neural Networks*, Vol. 11, pp. 249–257, 1998.

[95] Ok, S. , Miyashita, K. , and Nishihara, S. : "Improving Performance of GP by Adaptive Terminal Selection", in *Proc. of PRICAI 2000*, 435-445 (2000).

[96] Ok, S. , Miyashita, K. and Hase, K. : Evolving Bipedal Locomotion With Genetic Programming — A Preliminary Report —, *submitted for publication* (2001).

[97] M. H. Raibert: "Hopping in leg systems-Modeling ans simulation for the two-dimensional one-leg case. " *IEEE Transcations on Systems, Man and Cybernetics.* SMC-14(3), pp 451-463

[98] Stefan Schaal, Dagmar Sternad, and Christopher G. Atkeson. One-handed juggling: A dynamical approach to a rhythmic movement task. *Journal of Motor Behavior*, Vol. 28, No. 2, pp. 165–183, 1996.

[99] S. V. Shastri. A biologically consistent model of legged locomotion gaits. *Biological Cybernetics*, Vol. 76, No. 6, pp. 429–440, 1997.

[100] C. -L. Shih and W. A. Gruver. Control of a biped robot in the double-support phase. *IEEE Transactions on Systems, Man, and Cybernetics*, Vol. 22, No. 4, pp. 729–735, 1992.

[101] Soule, T. and Foster, J. A. : Removal Bias: a New Cause of Code Growth in Tree Based Evolutionary Programming, *Proc. of IEEE International Conf. on Evolutionary Computaion 98*, IEEE, pp. 181–186 (1998).

[102] Taga, G. , Yamaguchi, Y. and Shimizu, H. : "Self-organized control of bipedal locomotion by neural oscillators in unpredictable environment", *Biological Cybernetics*, 65, 147-159 (1991).

[103] Taga. G,: "A model of the neuro-musculo-skeletal system for human locomotion. I. Emergence of basic gait", *Biological Cybernetics*, 73, 97-111, (1995).

[104] Taga. G,: "A model of the neuro-musculo-skeletal system for human locomotion. II. Real-time adaptability under various constraints" *Biological Cybernetics*, 73, 113-121 (1995).

[105] Gentaro Taga. A model of the neuro-musculo-skeletal system for anticipatory adjustment of human locomotion during obstacle avoidance. *Biological Cybernetics*, Vol. 78, pp. 9–17, 1998.

[106] Tom Wadden and Orjan Ekeberg. A neuro-mechanical model of legged locomotion: single leg control. *Biological Cybernetics*, Vol. 79, pp. 161–173, 1998.

[107] M.W. Walker and D.E.Orin. "Efficient dynamic computer simulation of robotic mechanisms." *Transactions of the ASME Journal of Dynamic System, Measurement, and Control* 104, 205-211.

[108] Jill Whitall and Graham E. Caldwell. Coordination of symmetrical and asymmetrical human gait. *Journal of Motor Behavior*, Vol. 24, No. 4, pp. 339–353, 1992.

[109] William F. Punch. How effective are multiple populations in genetic programming. In John R. Koza, Wolfgang Banzhaf, Kumar Chellapilla, Kalyanmoy Deb, Marco Dorigo, David B. Fogel, Max H. Garzon, David E. Goldberg, Hitoshi Iba, and Rick Riolo, editors, *Genetic Programming 1998: Proceedings of the Third Annual Conference*, pp. 308–313, University of Wisconsin, Madison, Wisconsin, USA, 22-25 July 1998. Morgan Kaufmann.

[110] Satoshi Yamada, Akira Watanabe, and Michio Nakashima. Hybrid reinforcement learning and its application to biped robot control. In Michael I. Jordan, Michael J. Kearns, and Sara A. Solla, editors, *Advances in Neural Information Processing Systems*, Vol. 10. The MIT Press, 1998.

[111] Jin'ichi Yamaguchi, Sadatoshi Inoue, Daisule Nishino, and Atsuo Takanishi. Development of a bipedal humanoid robot having antagonisitc driven joints and three DOF trunk. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 96–101, Piscataway, NJ, 1998. IEEE Computer Society.

[112] J. F. Yang, M. J. Stephens, and R. Vishram. Infant stepping: a method to study the sensory control of human walking. *Journal of Physiology — London*, Vol. 507, No. 3, pp. 927–937, 1998.

[113] Yao, X. : " Evolving Artificial Neural Networks", in *Proc. of the IEEE*, 87(9) 1423-1447 (1999).

[114] Kai Zhao and Jue Wang. Multi-robot cooperation and competition with genetic programming. In Riccardo Poli, Wolfgang Banzhaf, William B. Langdon, Julian F. Miller, Peter Nordin, and Terence C. Fogarty, editors, *Genetic Programming, Proceedings of EuroGP'2000*, Vol. 1802 of *LNCS*, pp. 349–360, Edinburgh, 15-16 April 2000. Springer-Verlag.

[115] T. Zielinska. Coupled oscillators utilised as gait rhythm generators of a two-legged walking machine. *Biological Cybernetics*, Vol. 74, No. 3, pp. 263–273, 1996.

[116] Michigan State University: *lil-gp 1. 0 User's Manual*, Cambridge, MA, USA (1995).

付録 A

運動方程式の構築

A.1 順動力学の解法

各リンクの運動を記述するために運動方程式を構築する．本モデルのように複数のリンクが存在する場合，運動方程式を閉じた形で解くのは効率的でない，そこで本研究のような分岐構造を持つ任意のリンクモデルを扱うことのできる汎用の解法を利用することを考える．

ここではまず基本となるリンク直鎖リンク構造を持つ系に対する運動方程式について考える．これは図 A.1のように分岐構造を持たず，1 軸の回転関節のみからなり，基本座標に固定された土台を持っている 3 次元リンク構造である．

このモデルについて運動方程式を構築した後に，本モデルに適用できるように改良を行なう．この系のリンク数を p とし，運動を記述する状態変数として，ここでは図 A.1に示すような各リンクの絶対角度 $q_i(i = 1, 2, \cdots, p)$ を考える．通常，状態変数として，相対的な関節角度を用いるが，ここでは状態運動状態の記述の見通しがよい絶対角度を用いた．また，関節受動要素特性を含めた各関節に作用するトータルなモーメントを関節駆動モーメント n_i として定義する．この直鎖リンク系の運動方程式は一般的に次式のように記述することができる．

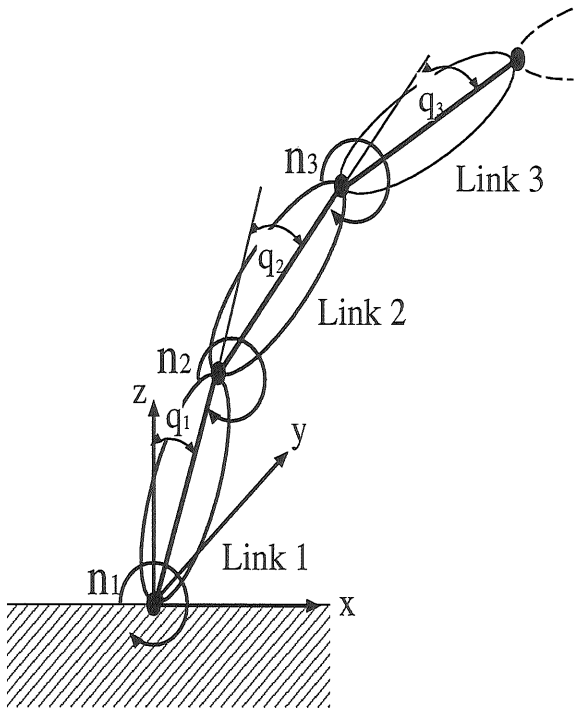

図 A.1: 直鎖リンク構造

$$\mathbf{J}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{c}(\dot{\mathbf{q}}, \mathbf{q}) + \mathbf{d}(\mathbf{q}, \mathbf{g}) = \mathbf{n} \tag{A.1}$$

ここで $\mathbf{q}$ は系の状態変数となる $p \times 1$ の列ベクトルで表されるリンクの絶対角度である. $\mathbf{J}(\mathbf{q})$ は $p \times p$ の慣性力に関する項,$\mathbf{c}(\dot{\mathbf{q}}, \mathbf{q})$ は $p \times 1$ の遠心力, コリオリ力に関する項,$\mathbf{d}(\mathbf{q}, \mathbf{g})$ は $p \times 1$ の重力の項, $\mathbf{g}$ は 2×1 の重力加速度ベクトル, $\mathbf{n}$ は $p \times 1$ の関節駆動モーメントである.

この運動方程式について初期状態と関節駆動モーメント $\mathbf{n}$ が与えられた時のリンク系の運動, すなわち角度 $\mathbf{q}(t)$, 角速度 $\dot{\mathbf{q}}(t)$, 角加速度 $\ddot{\mathbf{q}}(t)$ を求める問題を順動力学問題といい, その逆で角度, 角速度, 角加速度が与えられた時の関節駆動モーメントを求める問題逆運動学問題と呼ぶ. 本研究では関節駆動モーメントを神経系から得られる筋張力によって生じる関節モーメントとして与えて運動状態を求める順動力学問題を解くことになる. まず, 式 A.1を整理すると次式が得られる.

$$\begin{aligned} \ddot{\mathbf{q}} &= \mathbf{H}^{-1}(\mathbf{n} - \mathbf{b}) \\ \mathbf{H} &= \mathbf{J}(\mathbf{q}) \\ \mathbf{b} &= \mathbf{c}(\dot{\mathbf{q}}, \mathbf{q}) + \mathbf{d}(\mathbf{q}, \mathbf{g}) \end{aligned} \tag{A.2}$$

ここで関節駆動モーメント $\mathbf{n}$ が与えられたとき, この微分方程式を何らかの数値計算解法を用いて解くことで, リンク系の運動を求めることができる. このときの問題は $\mathbf{H}(p \times p)$ とベクトル $\mathbf{b}(p \times 1)$ をいかに効率良く計算するかということである. 本研究では以下に示す Walker ら [107] によって提案された方式を用いたこととした. この方法では運動方程式の逆動力学問題の解法を基本アルゴリズムとして用いている. 基本アルゴリズムは慣性モーメント行列の逆行列を計算することで, 順動力学計算を行なうことができる. いま角度 $\mathbf{q}(t)$, $\dot{\mathbf{q}}(t)$ 速度, 重力 $\mathbf{g}$ が与えられており, 角加速度のみをすべて 0

$$\ddot{\mathbf{q}}(\mathbf{t}) = \mathbf{0} \tag{A.3}$$

とし, 関節駆動モーメントを未知数として運動方程式の逆動力学問題を解くことを考える. すると次式のように慣性項が除去され, 得られる関節駆動モーメントによりベクトル $\mathbf{b}$ を求めることができる.

$$\begin{aligned} \mathbf{n} &= \mathbf{J}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{c}(\dot{\mathbf{q}}, \mathbf{q}) + \mathbf{d}(\mathbf{q}, \mathbf{g}) \\ &= \mathbf{c}(\dot{\mathbf{q}}, \mathbf{q}) + \mathbf{d}(\mathbf{q}, \mathbf{g}) \\ &= \mathbf{b} \end{aligned} \tag{A.4}$$

次に角度 $\mathbf{q}(t)$ はそのままで, 他の値を以下のように設定する.

$$\begin{aligned} \mathbf{g} &= \mathbf{0} \\ \dot{\mathbf{q}}(t) &= \mathbf{0} \\ \ddot{\mathbf{q}}(t) &= \mathbf{e}_i (i = i, 2, \cdots, p) \end{aligned} \tag{A.5}$$

$\mathbf{e}_i$ は i 行が 1 で他はすべて 0 となっている列ベクトルである. そして先と同様に運動方程式を逆動力学問題を解き, 関節駆動モーメントを求めると, 今度は $\mathbf{c}(\dot{\mathbf{q}}, \mathbf{q})$ や $\mathbf{d}(\mathbf{q}, \mathbf{g})$ がすべて除去され, 行列 $\mathbf{H}$ の第 i 列 $\mathbf{h}_i$ を関節駆動モーメント $\mathbf{n}$ により求めることができる.

$$\begin{aligned} \mathbf{n} &= \mathbf{J}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{c}(\dot{\mathbf{q}}, \mathbf{q}) + \mathbf{d}(\mathbf{q}, \mathbf{g}) \\ &= \mathbf{J}(\mathbf{q})\mathbf{e}_i \\ &= \mathbf{h}_i \end{aligned} \tag{A.6}$$

したがって, i を i から p まで変更し, 同様に計算を繰り返すことで慣性項 $\mathbf{H}$ を求めることができる. 以上のように運動方程式の逆運動力学問題を解く操作を繰り返すことにより, 順動力学問題を解くことができるようになる

A.2 逆動力学問題の基本解法

逆動力学問題の解法には大きく分けて, 力の釣り合いに基づいたニュートン・オイラー法と一般化座標を用いたラングランジュ法とが知られている. 本研究では力学的

な見通しの立てやすいニュートン・オイラー法に基づいた方法を採用した.

ここではまず基本となる図 A.2の直鎖リンク構造についての解法を考える.

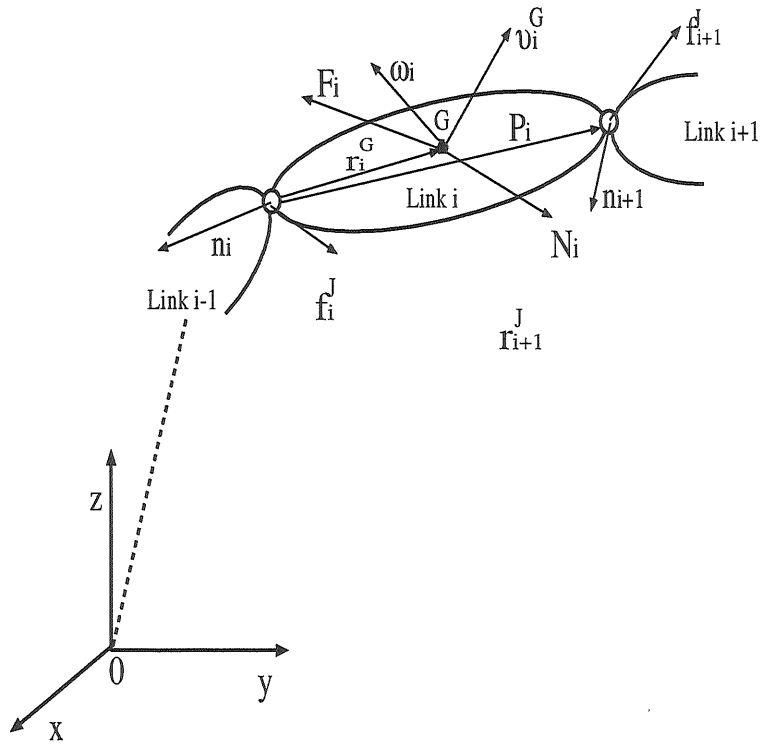

図 A.2: 直鎖リンクの物理量

ある任意のベクトルを回転軸として, 座標系を回転させることを考えると図 A.3のように, 回転軸として単位ベクトル

$$\mathbf{r} = (r_x, r_y, r_z)^T, \quad \|r\| = 1 \tag{A.7}$$

をとる, r 軸回りの回転を直接考える代わりに, まず r 軸を z 軸に一致させるように, x 軸まわりに α,y 軸回りに $-\beta$ の回転を行ない, その後 r 軸 (z 軸) 回りに q の回転を行なう. そして, r 軸をもとの位置にもどす.

$$\mathbf{R}_{r,q} = \mathbf{R}_{x,-\alpha} \bullet \mathbf{R}_{y,\beta} \bullet \mathbf{R}_{z,q} \bullet \mathbf{R}_{y,-\beta} \bullet \mathbf{R}_{x,\alpha} \tag{A.8}$$

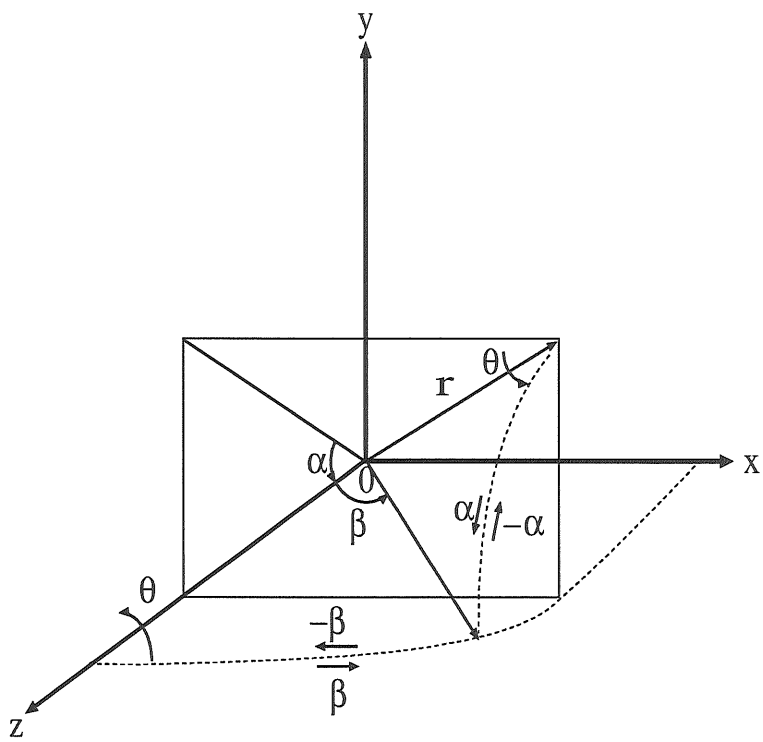

図 A.3: 任意軸回りの回転

$$
= \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\alpha & \sin\alpha \\ 0 & -\sin\alpha & \cos\alpha \end{bmatrix} \begin{bmatrix} \cos\beta & 0 & \sin\beta \\ 0 & 1 & 0 \\ -\sin\beta & 0 & \cos\beta \end{bmatrix} \begin{bmatrix} \cos q & -\sin q & 0 \\ \sin q & \cos q & 0 \\ 0 & 0 & 1 \end{bmatrix}
$$

$$
\times \begin{bmatrix} \cos\beta & 0 & -\sin\beta \\ 0 & 1 & 0 \\ \sin\beta & 0 & \cos\beta \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\alpha & -\sin\alpha \\ 0 & \sin\alpha & \cos\alpha \end{bmatrix}
$$

ここで,

$$
\sin\alpha = r_x/\sqrt{r_y^2 + r_z^2} \quad \cos\alpha = r_z/\sqrt{r_y^2 + r_z^2}
$$

$$
\sin\beta = r_x \qquad \cos\beta = \sqrt{r_y^2 + r_z^2}
$$

を代入して, 整理すると

$$
\mathbf{R}_{r,q} = \begin{bmatrix} r_x^2(1-\cos q) + \cos q) & r_x r_y(1-\cos q) - r_z \sin q) & r_x r_z(1-\cos q) + r_y \sin q) \\ r_x r_y(1-\cos q) + r_z \sin q) & r_y^2(1-\cos q) + \cos q) & r_y r_z(1-\cos q) - r_x \sin q) \\ r_x r_z(1-\cos q) - r_y \sin q) & r_y r_z(1-\cos q) + r_x \sin q) & r_z^2(1-\cos q) + \cos q) \end{bmatrix}
$$

となる

直鎖リンクに作用するベクトルや直鎖リンクの物理量の定義を図 A.2および表 A.1に示す.

リンク番号は土台より順に $0, 1, \cdots, p$ とする. また各リンクの重心位置は2つの関節点を結ぶ線分上に位置するものと仮定した. 計算は次の2つの手順から成っている. まず始めには各リンクの並進加速度を最下位リンクから先端リンクまで順次求めていく正順計算を行う. 次に得られた並進加速度を用い, 先端リンクまで最下位リンクまで逆方向に各リンクに関する運動方程式を求めていき, 各関節に作用するモーメントを計算する逆順計算を行う.

表 A.1: 物理量の定義

記号	意味
i	リンク番号
m_i	リンク質量
I_i	リンクの慣性モーメント
q_i	リンクの相対角度
$\mathbf{P}_i$	リンク i で位置ベクトル
$\mathbf{r}_i^G$	リンク i での重心位置ベクトル
$\mathbf{f}_i^J$	関節反力
$\mathbf{n}_i$	関節駆動モーメント
$\mathbf{R}_i$	リンク i での回転行列
$\mathbf{F}_i$	リンク i での慣性力
$\mathbf{N}_i$	リンク i での慣性モーメント
ω_i	リンク i での回転速度
$\dot{\omega}_i$	リンク i での回転加速度
v_i	リンク i での並進速度
$\dot{v}_i$	リンク i での並進加速度
$\mathbf{e}$	$\mathbf{e} = [0, 0, 1]^T$

A.2.1 正順計算

各リンクの並進加速度を求める正順計算は以下の漸化的な手順より成り立っている.

- 土台（第 0 リンク）の速度, 加速度を漸化式の初期条件として以下のように設定する

$$\begin{aligned} \dot{\mathbf{q}}_0 &= \mathbf{0} \\ \ddot{\mathbf{q}}_0 &= \mathbf{0} \\ \ddot{\mathbf{r}}_0^J &= \mathbf{g} \end{aligned} \tag{A.9}$$

ここで土台に重力加速度 $\mathbf{g}$ を作用するように仮定することによってすべてのリンク関して重力加速度の影響を考慮することができる

リンク $i-1$ とリンク i が回転関節で結合されている場合, リンク間の角速度の関係は

$$\omega_i = \mathbf{R}_{i-1}\omega_{i-1} + \mathbf{e}\dot{q}_i \tag{A.10}$$

の繰り返し式で表わされる. 式 A.10を微分すると, 角加速度の関係が得られる.

$$\dot{\omega}_i = \mathbf{R}_{i-1}\dot{\omega}_{i-1} + \mathbf{R}_{i-1}\omega_{i-1} \times \mathbf{e}\dot{q}_i + \mathbf{e}\ddot{q}_i \tag{A.11}$$

一方, 並進速度は

$$\upsilon_i = \mathbf{R}_{i-1}(\upsilon_{i-1} + \dot{\omega}_{i-1}\mathbf{R}_{i-1}\omega_{i-1} \times \mathbf{P}_i) \tag{A.12}$$

として与えられる. したがって, 上式を微分すれば, 次式のような並進加速度の関係が得れあれる.

$$\dot{\upsilon}_i = \mathbf{R}_{i-1}(\upsilon_{i-1} + \dot{\omega}_{i-1} \times \mathbf{P}_i) + \omega_{i-1} \times (\omega_{i-1} \times \mathbf{P}_i) \tag{A.13}$$

また, 重心並進加速度は回転角加速度および並進加速度を使って次式のように求めることができる.

$$\dot{v}_{Gi} = v_i + \dot{\omega}_i \times \mathbf{r}_{Gi} + \omega_i \times (\omega_i \times \mathbf{r}_{Gi}) \tag{A.14}$$

そして, リンク i に重心に作用する角加速度と並進加速度が求まったので運動方程式を各リンクについて次式のように得ることができる.

$$\begin{aligned} \mathbf{F}_i &= m_i \dot{v}_{Gi} \\ \mathbf{N}_i &= \mathbf{I}_i \dot{\omega}_i + \omega_i \times \mathbf{I}_i \omega_i \end{aligned} \tag{A.15}$$

ここで, $\mathbf{F}_i$, $\mathbf{N}_i$ はリンク i の慣性力, 慣性モーメントである.

A.2.2 逆順計算

関節駆動モーメントと求める逆順計算は以下の漸化的な手順により成り立っている.

- 先端リンクの関節反力, 関節駆動モーメントを漸化式初期条件として以下のように設定する.

$$\begin{aligned} \mathbf{f}_{p+1}^J &= \mathbf{0} \\ \mathbf{n}_{p+1} &= \mathbf{0} \end{aligned} \tag{A.16}$$

- $\mathbf{f}_{i+1}^J$ と $\mathbf{n}_{i+1}$ を既知としたとき, $\mathbf{f}_i^J$ と $\mathbf{n}_i$ は以下の力とモーメントの釣り合い式により求められる.

$$\begin{aligned} \mathbf{f}_i^J &= \mathbf{R}_{i+1} \mathbf{f}_{i+1}^J + \mathbf{F}_i \\ \mathbf{n}_i &= \mathbf{R}_{i+1} \mathbf{N}_i + \mathbf{P}_{i+1} \times \mathbf{R}_{i+1} \mathbf{f}_{i+1} + \mathbf{r}_{Gi} \times \mathbf{F}_i \end{aligned} \tag{A.17}$$

- 初期条件式（式 A.16）のもとで漸化式（式 A.17）を $i = p$ から $i = p$ まで計算することで全てのリンクに関する $\mathbf{f}_i^J$ と $\mathbf{n}_i$ が計算できる. なお, これまでの関

節駆動モーメントは関節の受動要素特性を含めたトータルなモーメントを考えていた．したがってこの影響を除いた分のモーメント，すなわち生体の場合では筋張力によって生じるモーメント $\mathbf{n}_i^F$ は次式により求めることができる

$$\begin{aligned} \mathbf{n}_i^F &= \mathbf{n}_i - passive(\Delta q_i, \Delta \dot{q}_i) \\ \Delta q_i &= q_i - q_{i-1} \end{aligned} \tag{A.18}$$

A.2.3 身体モデルへの適応

以上の式はロボットアームのように空間に固定された直鎖構造をもつリンクモデルに関する運動方程式の解法であった．これに対して本モデルでは空間内の移動を行う並進の移動成分が存在し，さらに四肢のように分岐構造を持つ．また水平面モデルと矢状面モデルとが相互作用するようにもなっている．そのため前述のアルゴリズムに関して以下のような改良を加えた．

- 並進移動成分の考慮:

  先の連鎖リンクモデルではロボットアームのように土台に固定された構造を考えていた．これに対して本研究の身体モデルは歩行運動に対象としているため，リンク系に固定点は存在せず，リンク系自身が並進の自由度を持つ．そこで，図 A.4に示すように土台と最下位リンクが 3 自由度持つ仮想的な並進駆動の関節により接続されていると仮定する．そのため正順計算における加速度の初期値として，重力加速度のほかに並進駆動により生じる加速度を加えて計算を行うようにする．またリンク系全体の状態変数としてもこの並進自由度の 2 変数が加えられる．さらに逆動力学問題の解法においては，この並進駆動関節の仮想駆動力を関節駆動モーメントと同様に考慮する．なお，実際の身体モデルでは腰椎節のリンク座標系の原点の並進移動成分を並進の状態変数として用いている．

- 外力の考慮:

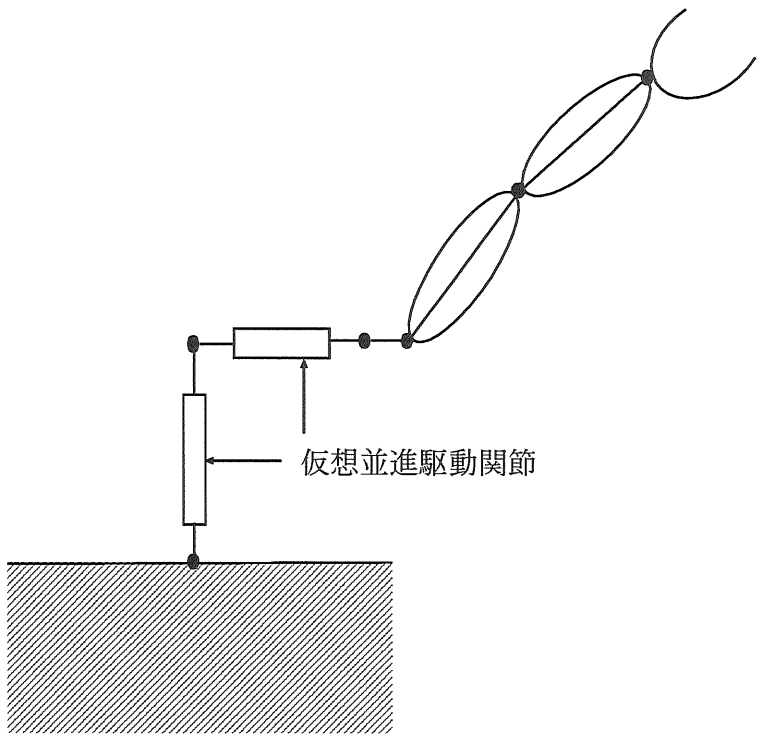

図 A.4: 仮想並進駆動関節

先の直鎖リンクでは系の外部から作用する力は重力のみであったが, 歩行などでは床からの反力などの外力が作用する. したがって, これも考慮できるようにする.

- 分岐構造の考慮:

身体モデルでは四肢の節を表すリンクが分岐構造となるため, 正順, 逆順計算の漸化計算の方法に工夫が必要となる. 基本的には分岐構造は有するモデルを複数の直鎖リンク群の集合体として扱うようにする. すなわち本モデルにおいて図 A.5のように矢状面モデルを四肢と体幹の合計 5 つのリンク群に分割して考える. 図は正順計算を示しており, 逆順計算ではこの逆の順序になる. 正順計算においては, それぞれの直鎖リンク群の最下位リンクの原点の位置および加速度をそのリンクが接続しているリンクの運動状態を求め, これを漸化式の初期値とする. また逆順計算ではそれぞれの直鎖リンクの最下位リンクから得られた関節反力, 関節駆動モーメントを最下位が接続するリンクに作用する外力および外モーメントとして扱うようにして同様な計算を行う.

A.2.4 関節受動要素関数

各関節受動要素関数は文献 [23] を参考し, 次式のよに定義した.

$$\begin{aligned} passive_i(\Delta q_i, \Delta \dot{q}_i) &= -k_{i1}^J \exp\{k_{i2}^J(\Delta q_i - \overline{\Delta q_i} + k_{i3}^J)\} \\ &\quad + k_{i4}^J \exp\{-k_{i5}^J(k_{i6}^J + \Delta q_i - \overline{\Delta q_i})\} - c_i^J \Delta \dot{q}_i \quad \text{(A.19)} \end{aligned}$$

ここで k_{il}^J から k_{i6}^J は係数であり, $\overline{\Delta q_i}$ は自然立位状態における関節角度を表す基準関節角度である. また c_i^J 粘性係数である. この式の第 1 項と第 2 項とが弾性受動特性を表す関数である.

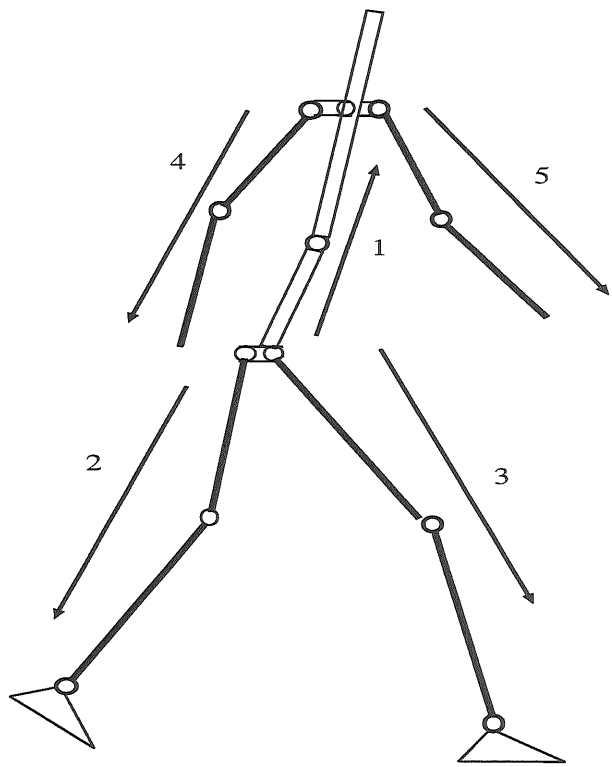

図 A.5: 分岐構造を持つ場合の計算順序

A.2.5 床面モデル

床面モデルは次式のような線形の粘弾性体により表した.

$$f^E_{x,k} = \begin{cases} -k^E(x^E_k - x^{E0}_k) - c^E\dot{x}^E_k & (x^E_k \leq 0) \\ 0 & (x^E_k > 0) \end{cases}$$

$$f^E_{y,k} = \begin{cases} -k^E(y^E_k - y^{E0}_k) - c^E\dot{y}^E_k & (z^E_k \leq 0) \\ 0 & (y^E_k > 0) \end{cases}$$

$$f^E_{z,k} = \begin{cases} -k^E z^E_k - c^E \max_0(-\dot{z}^E_k) & (z^E_k \leq 0) \\ 0 & (z^E_k > 0) \end{cases}$$

ここで, $f^E_{x,k}$, $f^E_{y,k}$, $f^E_{z,k}$ は第 k 番目の床反力の x, y, z 成分, k^E は弾性係数（$k^E = 6000N/m$）, c^E は粘性係数（$c^E = 600N/m$）, x^E_k, yk^E, zk^E はカカト点や中足点の座標値, すなわち, 第 k 番目の床反力作用位置の座標値である. また, x^E_k, y^E_k は第 k 床反力作用位置が床面に接した瞬間の x, y の座標値である. 鉛直方向の粘弾項については鉛直方向の速度が正のとき, すなわち足が床面より離れていくときは作用しないようにした.

業績リスト

【学術雑誌】

[1] 玉 秀列、宮下和雄、西原清一:適応的ノード選択による遺伝的プログラミングの効率改善、情報処理学会論文誌, Vol. 42, NO. 7, 2001

【査読付き国際会議】

[1] Sooyol OK, Kazuo Miyashita, Seiichi Nishihara: Improving Performance of GP by Adaptive Terminal Selection, The Sixth Pacific Rim International Conference on Artificial Intelligence(PRICAI'2000) pp. 435-445, 2000.

[2] Sooyol OK, Kazuo Miyashita, Kazunori Hase: Evolving Bipedal Locomotion with Genetic Programming, Congress on Evolutionary Computation(CEC'2001), pp. 1025-1032, 2001.

【修士論文】

Human Hand Deformation Using Inverse Kinematics and NURBS, 1988.

【国内会議等】

[1] 玉 秀列、宮下和雄、長谷和徳、西原清一:遺伝的プログラミングによる3次元仮想人間の歩行生成、第 59 回情報処理学会全国大会講演論文集(2)、pp.161-162.

[2] 玉 秀列、宮下和雄、西原清一:適応ターミナル選択による遺伝子プログラミングの性能向上、電子情報通信学会技術研究報告、「人工知能と知識処理」研究会報告、Vol.100 No.88 pp.45-52.

[3] 玉 秀列、宮下和雄、長谷和徳:GP を用いた神経振動子モデルに基づく2足歩行運動の制御、第 15 回人工知能学会全国大会講演論文, CDROM.

以上