Modulobe: 物理シミュレーションによる仮想生物構築環境

江渡　浩一郎[†]　川崎　禎紀[††,†]
渡辺　訓章[†]　前川　峻志[†††,†]

モジュールを組み合わせることによって仮想生物を作ることができる物理シミュレーション環境 *Modulobe* を開発した．ユーザはモジュールを組み合わせて骨格を作り，モジュール間のリンクに動きを指示することによって，容易に仮想生物を実現できる．実世界の物理法則を忠実にシミュレートすることによって，仮想生物はリアルな生物の動きを示す．本論文では，システムの概要と実装，得られた知見について述べる．

Modulobe: Constructing Virtual Creatures in a Physical Simulation Environment

KOUICHIROU ETO,[†] YOSHINORI KAWASAKI,[††,†] KUNIAKI WATANABE[†]
and TAKASHI MAEKAWA[†††,†]

We developed a physical simulation environment, *Modulobe* system, which allows the user to create virtual creatures easily. The user can create a body structure by connecting modules and set movement of the virtual creature by simply specifying changes of angle of hinges. The system simulates the physical laws of the real world, so as to the creatures move like real creatures. In this paper, we describe the basic ideas, implementations details, and experiences of the system.

1. はじめに

レゴ・ブロックのような，基本的な形を組み合わせて任意の形を作りあげることができるおもちゃは，広く普及している．ブロックの組み合わせで形を作り上げる仕組みには様々な種類が存在しており，作り上げられる形も多様である．

しかし，作り上げられた形状に対して動きを追加しようとする場合には，形状を作り上げるのに比べて簡単ではない．例えば，レゴ・テクニック・シリーズや，フィッシャー・テクニックなどには，モータやギアなどの部品が存在しており，それらを組み合わせることによって任意の動きを実現することができる．しかし，モータによる運動は一定方向に一定速度で回転するのみであり，その力をギアなどを介して任意の動きを作り上げることは習熟を要する．自動車のような回転運動による動きは容易に再現できるが，生物の歩行のような動きを再現することは困難である．一般に生物においては，それぞれの関節毎に筋肉がついており，その筋肉を収縮・弛緩させることによって歩行などの動作を行う．このような生物特有の動きは，モータとギアの組み合わせでは容易には再現できない．

コンピュータ上に構築された仮想世界においては，実世界の制約をうける必要はない．再現が困難だった生物のような動きも，仮想世界であれば筋肉と同じ仕組みで再現できるはずである．このような発想の元，物理法則をリアルに再現する物理シミュレーション環境を構築し，単純な部品を組み合わせることによって容易に生物のような動きを実現することができるシステム *Modulobe* を開発した．ユーザは，部品を接続することによって様々な形を作り，部品間の接続部分に動きを指示することによって，生物のような動きを実現することができる．

2. Modulobe

Modulobe は，編集モードと再生モードの大きく二つにわかれている．編集モードでは，ユーザは仮想生

† 独立行政法人 産業技術総合研究所 情報技術研究部門 情報流デザイングループ
Fluid Information Design Group, Information Technology Research Institute, National Institute of Advanced Industrial Science and Technology (AIST)
†† 東京大学大学院 情報理工学系研究科 コンピュータ科学専攻
Department of Computer Science, Graduate School of Information Science and Technology, The University of Tokyo
††† 多摩美術大学 美術学部 情報デザイン学科
Department of Information Design, Faculty of Art and Design, Tama Art University

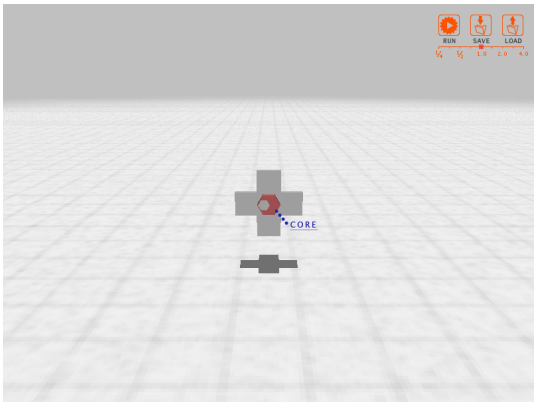

図 1　Modulobe 起動時の画面

図 2　システムメニュー

物の骨格を作り，その動作を指定できる．再生モードでは，編集モードで作成した仮想生物を実際に動かしてみることができる．

図 1 にシステム起動時の画面を示す．システム起動時は編集モードとなっており、右上のシステムメニュー (図 2) から，RUN ボタンによって編集モードと再生モードを切り替えられる．また，LOAD ボタン，SAVE ボタンによって仮想生物のデータ読込，保存を行う．

2.1　編集モード

仮想生物は，モジュールと呼ばれる部品によって構成されている．一番最初の初期状態では，コアという起点となるモジュールだけが存在する．コアに対してモジュールを接続していくことで，仮想生物の骨格を作っていく．全体としては，コアを起点としてモジュールが分岐する木構造となる．

モジュールにはシャフトとリンクの 2 種類がある．シャフトは 4 方向に新しいモジュールを接続できる，分岐構造を形成可能なモジュールである．コアはこのシャフトの特殊形である．リンクは 2 方向に接続可能で，中心で曲げられるモジュールであり，同時に動力にもなる．このリンクの角度を設定して変化させることによって，動きを与えることができる．

初期状態ではコアが選択されており，その周囲には選択用のガイドが 4 方向に表示されている．このガイドのいずれかをクリックして選択すると，その場所に新しいモジュールを接続することができる．シャフトやリンクの中心をクリックするとそのモジュールが選

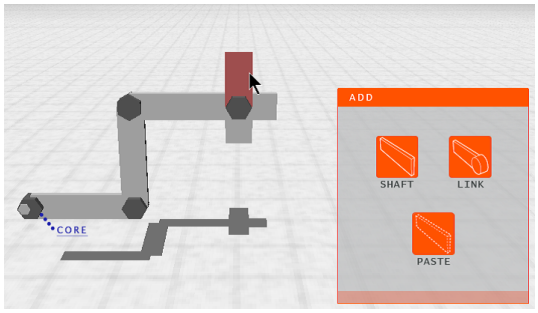

図 3　ガイド選択時のメニュー

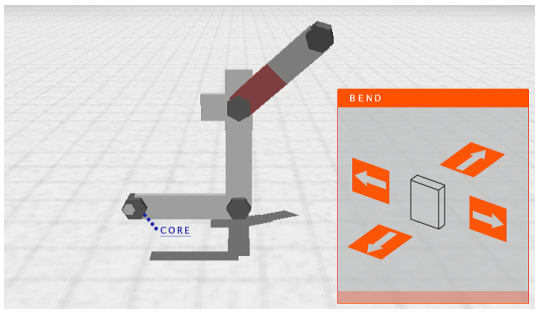

図 4　モジュール間連結部分選択時のメニュー

択され，モジュールが接続されていない箇所にはガイドが表示される．続いてガイドを選択すると，コアの場合と同様に，その場所にモジュールを接続することができる (図 3)．このように，コアから始めてモジュールを接続していくことで，仮想生物全体の形状を構築していく．

それぞれのモジュール間の連結部分を選択すると，接続角度を変更できる (図 4)．接続角度は，回転方向には隣接する接続との距離が 45 度より小さくならない範囲で，折り曲げ方向には ±45 度まで変化させることができる．角度変化は 15 度刻みである．このように，奥行きのある三次元形状を作成することができるようになっている．また，カメラコントロールとして，左ドラッグで視線方向変更，ホイール回転でズームイン・ズームアウトを行うことができる．カメラの位置は，選択しているモジュールが常に画面の中心となるように自動的に調整される．

各モジュールの中心を選択すると，そのモジュールに対応したメニューが表示され，編集できる．シャフトを選択した場合は，切り取り (CUT)，複製 (COPY)，削除 (DELETE) が行える (図 5)．カットやコピーした内容は，ガイドに対してペーストすることができる．一旦カットしてから，モジュールを一部削除して，そこにペーストするといった使い方ができる．コアを選

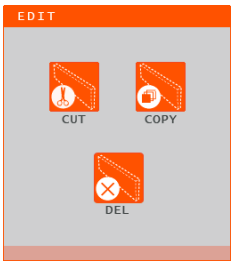

図 5　シャフト選択時のメニュー

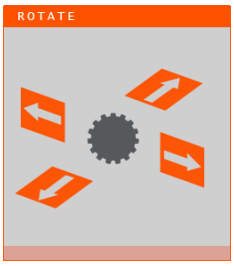

図 6　コア選択時のメニュー

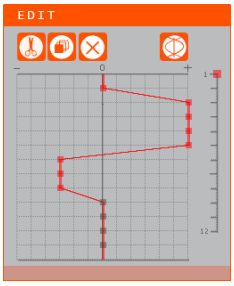

図 7　リンク選択時のメニュー

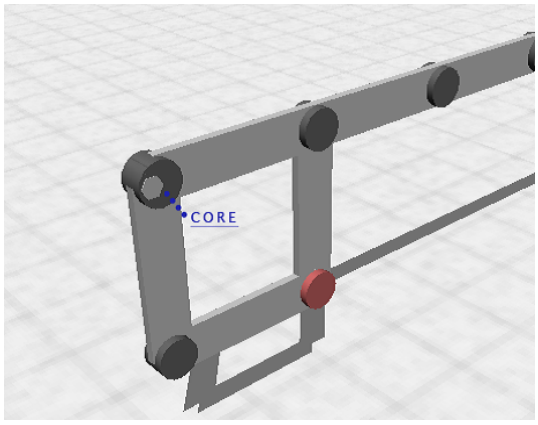

図 8　重なっていないモジュールを選択した場合

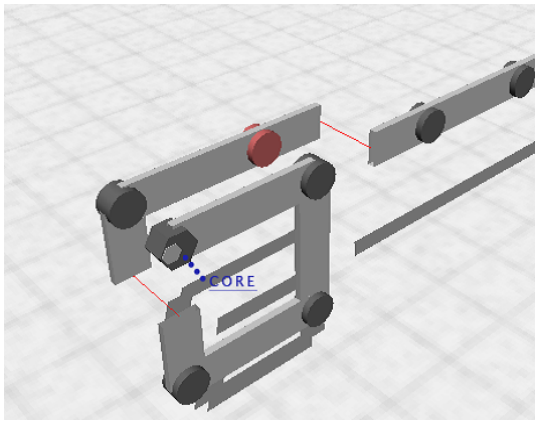

図 9　重なっているモジュールを選択した場合

択した場合は，仮想生物全体の回転を行える (図 6). リンクを選択した場合は，シャフトと同様の切り取り，複製，削除のほかに，角度の制御ができる．

リンク中心部分の曲がる方向は一方向だけであり，リンクモジュールのある平面に垂直な軸を中心に曲げられる．リンクを選択すると，そのリンクをどのように曲げるかを設定するグラフが表示される (図 7)．このグラフの縦軸は時間軸で，横軸はそのリンクの回転角度である．15 度刻みとなっており，±90 度まで曲げることができる．時間軸は縦方向につながっており，下まできたら，また上から繰り返す．右にある縦線と点は，角度変化の周期を示すものである．何も指定しないと，1 周が 1 秒とすばやく動き，点を下の方にもっていくと変化の周期が長くなり，ゆっくりした動きになる．ここで指定する角度はあくまで目標角度であり，実行時には角度が目標と一致させる方向に力が加えられる．そのため，高速に目標角度を変化させたり，重量のある物体を重力に逆らって回転させようとしたりした場合には，目標角度に達しないことがある．

また，グラフ上の点をクリックすることで，目標角度の有無を切り替えることができる．目標角度無しに設定した場合には，その時刻では特定の角度に近づけるための力は働かなくなる．全時刻で目標角度無しに設定すると，受動的にのみ角度が変化するフリーリンクを作ることができる．例えば，図 7 の場合は，1 ステップ後に 90 度曲がり，その 4 ステップ後に反対側に 45 度曲がり，さらに 3 ステップ後には目標角度無しにすると設定している．

このグラフを左右反転する，つまり角度変化の符号を入れ替えることもできる．一般に生物が歩く際には，おおまかに右足と左足が逆の動きをする．そのような動きを作る場合には，まず生物の体の片方を作ってからそれをカット&ペーストしてもう片方の体を作り，コピーされたリンクのグラフを左右反転することにより，左右が逆の動きをする仮想生物を容易に作ることができる．

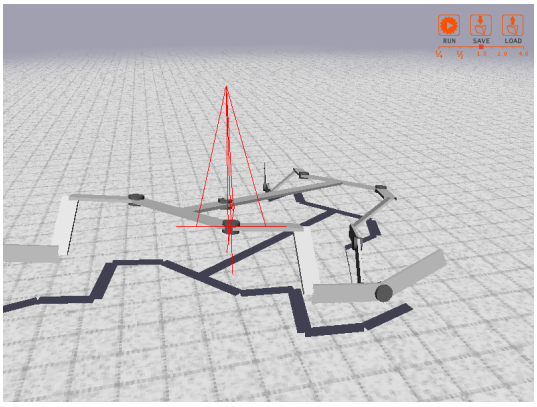

図 10　垂直方向に力を加えた状態

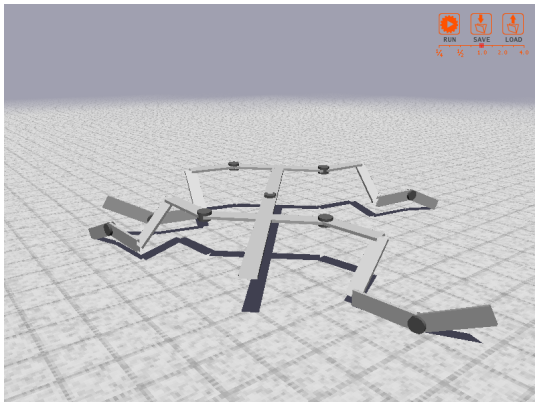

図 11　四本足で歩く仮想生物

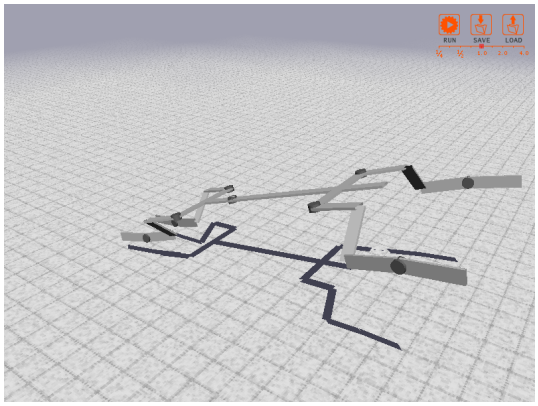

図 12　ジャンプして前に進む仮想生物

図 8 の様に一部のモジュールが重なったり，密集してしまうことがある．その場合には，重なっているモジュールを選択すると，図 9 のように見かけの位置をずらして表示され，適切なモジュールを選択することができるようになる．このとき，本来接続されているモジュール同士は赤いラインで繋がれている．コアから遠いほうのモジュールの位置がずれ，ずれた後に他のモジュールと重ならないように空いてる側に広がる．このような表示によって，モジュールを複数重ねて，見た目の上では接続しているような形状が容易に作成できる．

以上のように，シャフトとリンクの 2 種類のモジュールを接続し，リンクの動きを設定するだけで，生物のように動く複雑な形状を簡単に作成できるようになっている．

2.2　再生モード

再生モードでは，仮想生物が物理シミュレーションを基に動作するのを眺めることができる．平面は無限に広がり，壁や環境の変化は存在していない．カメラコントロールは編集モードと同様に，左ドラッグで視線方向変更，ホイール回転でズームイン・ズームアウトとなっている．カメラの位置は，画面の中心が仮想生物の重心に位置するように自動的にコントロールされる．

任意のモジュールに力を加えることもできる．まず，左クリックで力を加えるモジュールを選択し，左ドラッグをすると，ドラッグ速度に比例した大きさの水平方向の力が発生する．同様に，モジュールを選択してからホイールを回転させると，回転量に比例した大きさの垂直方向の力が発生する (図 10).

右上のシステムメニューにある速度指定バーから，物体の動作速度を指定することができる．動作速度の変化によって，物体の動きがどのように変化するかを実験してみることができる．

3.　様々な仮想生物

本節では，Modulobe を用いて作られた仮想生物から，いくつかの興味深い実例を紹介する．

3.1　四本足で移動する仮想生物

図 11 は，右前足と左後ろ足，左前足と右後ろ足とを連動させ，足を上げながら前にもっていき，足を下げながら後にもっていくという動作を繰り返すことによって前進する仮想生物である．骨格はほとんど同じであるが，左右の足を同時に動かし，前後の足で違う動きを繰り返すことによって，ジャンプしながら前進する仮想生物 (図 12) を作ることもできる．

3.2　地面の摩擦を利用して前進する仮想生物

図 13 は，蛇のように這うことによって前進する仮想生物である．地面との間の摩擦力は，接地する角度によって摩擦力の働く方向が異なるため，このように角度を変化させながら左右に移動するだけで，前進する方向に力が発生する．同様に，図 14 に，多数の細かい足が生えており摩擦力によって前進する仮想生物

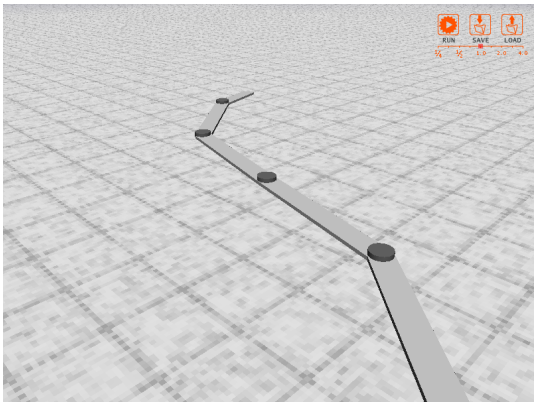

図 13　這うことによって前進する仮想生物

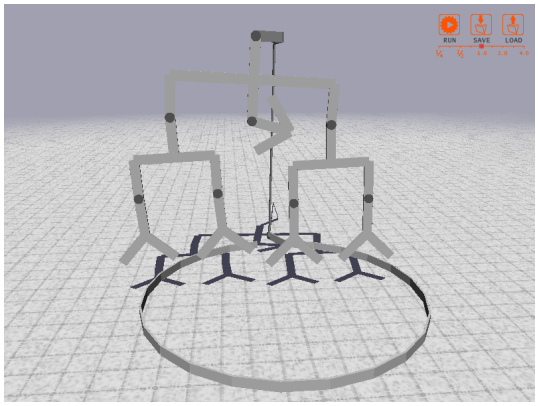

図 15　モビールのシミュレーション

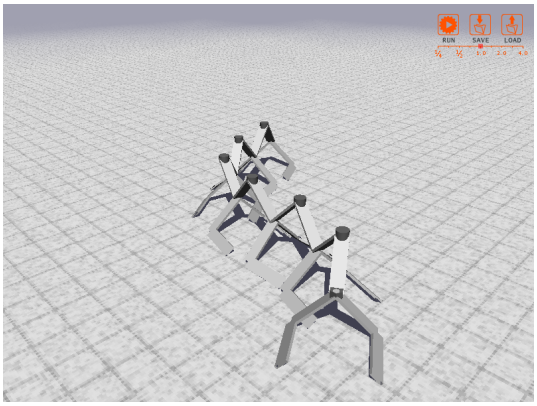

図 14　細かい足で這いながら前進する仮想生物

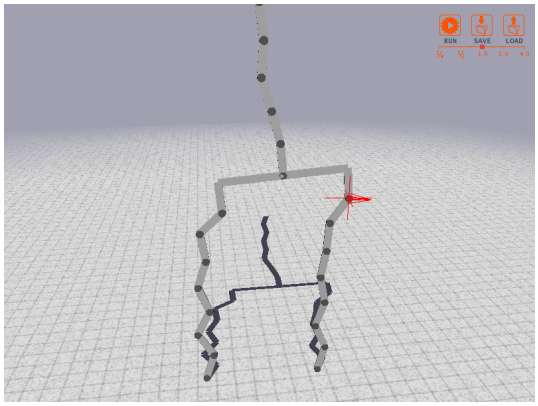

図 16　不安定なバランスで立っているモデル

を示す．モジュールの接地する面によって摩擦係数が異なるため，こちらの方が素早く前進するようになる．

3.3　物理法則のシミュレーション

Modulobe は，仮想生物のシミュレーションだけではなく，通常の物理法則によるモデルのシミュレーションに用いることもできる．図 15 は，モビールの動作原理を再現したモデルである．中央部についている下方向を向いた矢印が左右に振れることで振り子の振動が伝わり全体が揺れ動く．図 16 に示すモデルは，最初はバランスを保つことによって二本の足で静止しているが，少しでも力を加えると，崩れてしまう．

4.　実　　装

本システムは，Visual C++環境にて DirectX 9 を用い，実装されている．システムはおおまかに，レンダラ，UI，物理シミュレーションエンジンの三つの部分に分かれている．レンダラは，与えられたモジュールの座標を元に画面表示を行う．UI は，表示モードと編集モードにおけるユーザ・インタフェースを担当する．物理シミュレーションエンジンは物理法則の計算を行う部分であり，本システムの核となる部分である．

4.1　物理シミュレーションエンジン

通常のバネモデルによる物理シミュレーションの場合は，画面の再描画に合わせて計算を行うため，フレーム間の時間間隔はフレームレート (一般に一秒間に 60 回程度) に依存する．この場合は，物体はフレーム間で振動を繰り返すため，物体はゴムのような柔らかい材質でできているように表現される．それに対し，本システムの物理シミュレーションエンジンでは，内部的にはバネモデルを用いているが，一回の計算における変化時間 (Δt) を短い時間間隔 (1/4800 秒) に設定し，多くの計算 (一秒間に 4800 回) を繰り返すことによって，物体は振動をしていないように表現され，剛体としての見た目を実現している．このように短時間に物理計算を行う必要から，フレーム毎の計算をできるだけ簡略化している．計算は，リンクで接続されたブロック毎に一括して行われる．ブロック間のリンクの接続角度を固定する力は，通常はリンク個所に力が発生するが，本システムでは各々のブロックの重心に対して復元力が働くこととしている．このように，フ

レーム毎の物理計算を，単純な計算を多く繰り返す仕組みにしているため，将来的には並列計算による高速化が可能な構成となっている．

衝突判定は，物体と地面との間のみ行なっている．地面との衝突における反発力は，ペナルティ法[1] のみを用いて計算を行った場合，地面との間で振動が発生し，不安定な動作を示す場合がある．本システムでは，ペナルティ法を中心にインパクトベース法を部分的に併用することで，物体の振動を抑えている．

地面との間の摩擦力は，物体の重さによる地面への圧力と物体の速度と摩擦係数を掛けた値である．摩擦係数は，各モジュールの当たっている個所に応じて，異なった計算を行っている．モジュールの角の部分は摩擦が大きく，平らな部分では摩擦が少なくなるようにすることによって，安定した足のような構造やスキーのようにすべる構造を同時に実現できるようにしている．

5. 議　　論

本システムの設計時における議論について述べる．

モジュールの基本形状

モジュールの基本形状をどのような形状とするかについて議論があった．レゴ・ブロックのような立体的なブロックの場合は，ボリュームのある形状を表現するには都合がいいが，関節によって曲がるような形は表現しづらい．同様に，平面的なパネルの組み合わせも，関節による動きを表現しづらい．生物のような動きを表現するのに最適な形という観点から，細長い棒状の形を基本形状とした．丸い棒や断面が正方形の棒の場合には向いている方向がわからなくなってしまうため，細長い板のような形とした．

現在のシステムでは，モジュールの形状はこの基本形状一種類のみであり，変形させる手段は提供していない．物理シミュレーション上では異なる形状のモジュールを同時に扱えるが，編集時の混乱を避けるために，モジュールの形状は一種類のみとした．

二次元形状の物体

本システムでは，物体の形状として三次元形状を作ることができるようになっているが，三次元形状による物体の作成は二次元形状と比べて習熟を要する．三次元形状の物体を作る前に，二次元形状の物体で実験をしてみることができると，有用であると考えられる．

編集モードの最初の状態では，物体を横から見る視点となっている．この状態で前後方向の曲げを行わずに物体を作ると，二次元形状の物体を作ることになる．現実世界では，二次元形状の物体を地面に垂直に置くと，わずかな傾きを元にその物体は倒れる．しかし，本システムでは物体を垂直方向に置いた場合には，あえて前後に倒れないようにした．このような工夫によって，二次元形状による物体と三次元形状による物体とを，同じシステム内でシームレスに扱えるようにした．

仮想生物を掴む方法

表示モードにおいて，仮想生物を掴んで動かす操作方法についての議論があった．現在のシステムでは，一旦クリックすることによってモジュールを指定してから，そこに力を与えるという二段階の操作を行うこととしている．これに対して，マウスでモジュールをドラッグして，直接移動先を指示するようにした方が直感的ではないかと考えられる．しかし，この場合には，モジュールをどちらかにひっぱるとその力で物体は動き，物体の移動に応じてカメラも移動し，それによってまた物体はポインタの方にひっぱられるというループが発生してしまう．これにより，物体は一定方向に無限に力を加えられ続ける状態となる．これは意図する操作とは異なる．モジュールを掴んでいる時だけカメラの重心への追随をやめることもできるが，実際には物体そのものが空間上で移動しているため，モジュールを掴んだ瞬間に物体を停止させる方向に強くひっぱることになる．これもまた掴んで動かすという目的からはかけはなれた操作となる．

また，通常カメラは俯瞰位置から物体を見下ろしている場合が多い．その状態でモジュールの移動先を直接指定すると，水平方向への移動を指示しているのか，垂直方向への移動を指示しているのかが定まらない．カメラからの視線に直交する方向への移動を指示したことにすると，斜め上や斜め横への移動を指示することとなり，意図しない方向への移動を指示する結果となる．現在の操作方法では，マウスの移動で地平面に対して水平方向に，ホイル操作で地平面に垂直方向に力を加えるようにして，この曖昧さを回避している．

6. 関 連 研 究

本節では Modulobe の関連研究について述べる．

6.1 物理シミュレーション環境

Soda 社の Ed Burton は，バネモデルによる仮想生物を作り，登録することができるシステム *Sodaplay*[2] をインターネット上で公開している．他のユーザが作成した様々な仮想生物を，ブラウザ上で動かして見ることができる．仮想生物はバネモデルによる柔らかい接続構造となっているため，剛体による動きを表現することはできない．

Falco は，Sodaplay の基本的な特徴をそのまま三次元に拡張したシステム *Springs World 3D*[3] を開発した．Sodaplay とは異なり編集環境は内蔵していない．三次元において安定した形を作るためには物体内部の接続構造を増やす必要があり，自由な形態を作ることが難しくなっている．

Waddoups は，様々な形態を作り，動かすことができるシステム *Juice*[4] を開発した．物理シミュレーションエンジンとして，*Open Dynamics Engine* (ODE)[5] を使用している．システム内でモデルを編集することもできるが，操作は直感的ではなくわかりづらい．

1999 年に SCEI より発売された PlayStation 用のゲーム「**パネキット**」[6] では，様々な役割を持った正方形のパネルを組み合わせることによって，地上を走る車，海を進む船，空を飛ぶ飛行機などの様々な乗り物を作ることができる．主に乗り物のような動きに特化しており，生物のような動きを実現することは難しい．

6.2 遺伝的アルゴリズムによる進化

Sims は，物理法則の制約の中で遺伝的アルゴリズムによって仮想生物が自動的に進化するシステム *Evolving Virtual Creatures*[7] を開発した．地上を歩く，ジャンプする，水中を泳ぐなどの動作を行う仮想生物が，進化によって自動的に生成された．仮想生物間の競争によって進化を促進させる実験[8] も行なわれた．

Wagner は，Sodaplay における仮想生物を遺伝的アルゴリズムによって自動進化させるシステム *Wodka*[9] を開発した．人間が作るよりも素早く動く仮想生物を，自動的に生成することができるようになった．

Modulobe では，ユーザが手動で仮想生物を制作することとしており，遺伝的アルゴリズムによる自動進化は今後の課題である．

6.3 物理的な機構による生物モデル

オランダのアーティスト Theo Jansen は，物理的な機構のみを用い，風を動力として生物のように動くロボット *strandbeest*[10] を作っている．

Raffle らは動的なブロックを組み合わせて生物のような動きを作りだすことができるシステム *Topobo*[11] を開発した．動的なブロックは，一度動かした動きを記録し，その動きを繰り返すことができる．親ブロックから複数の子ブロックに動きを伝播させ，同時に動かすこともできる．

Modulobe は，仮想世界において生物のような動きを実現するシステムであるが，現実世界における動きのシミュレーションのために使うこともできると考えられる．

7. おわりに

単純なモジュールを組み合わせることによって，仮想生物を作ることができる物理シミューレション環境 Modulobe を開発した．静的なモジュールの組み合わせで骨格を作り，動的なモジュールに動きを指示することによって，容易に生物のような動きを実現することができる．実際に本システムを用いて，様々な動きを示す仮想生物を作り出すことができた．今後は，より多様な生物環境を実現できるようなシステムへと発展させていきたい．

参考文献

1) Bourg, D. M.: *Physics for Game Developers*, O'Reilly (2001).
2) Burton, E.: Sodaplay. http://www.sodaplay.com/
3) Falco, M.: Springs World 3D. http://www.sodaplaycentral.com/features/sw3d2/
4) Waddoups, N.: Juice. http://www.natew.com/juice/
5) Smith, R.: Open Dynamics Engine. http://ode.org/
6) SCEI: パネキット (1999).
7) Sims, K.: Evolving Virtual Creatures, *Computer Graphics (Siggraph '94) Annual Conference Proceedings*, pp.15–22 (1994).
8) Sims, K.: Evolving 3D Morphology and Behavior by Competition, *Artificial Life IV Proceedings*, pp. 353–372 (1994).
9) Wagner, W.: Wodka. http://wodka.sourceforge.net/
10) Jansen, T.: strandbeest. http://www.strandbeest.com/
11) Raffle, H., Parkes, A. and Ishii, H.: Topobo: A Constructive Assembly with Kinetic Memory, *Proceedings of CHI 2004*, pp.647–654 (2004).