Web3D キャラクタエージェント 構築アプリケーションの開発

Development of Web3D Character Agent System and Construction Application

佐々木 優理
Yuhri SASAKI

有限会社 アントラッド
（〒630-8101 奈良県奈良市青山４丁目４番地１２３ E-mail: you-ri@nbn.ne.jp ）

ABSTRACT. This paper investigates new approaches to using Web3D as an introduction for home pages and aims to expand its use for this purpose. At present applications to produce such contents require specific knowledge of existing 3D structural tools and this has prevented widespread use by individual programmers. Based on the above consideration, this paper will endeavor to find both a Web3D system, which can be used as a Character Agent and applications that can easily design 3D character animations.

１．背景

Web3D はその将来を羨望されながら、実際の利用は思ったよりはるかに少なく、企業などで利用されているものの個人一般には深く浸透していない。その原因として考えられるのは、利用範囲の狭さや開発ツールの技術的敷居の高さなどが上げられる。製作された Web3D コンテンツはブラウザ上のフレームという枠内で制限され、アートや広告手段としてしか利用できない。これでは趣味で３Ｄモデリングしている人か一部の企業しかその利用価値を見出すことは難しい。これ以上の普及を目指すには別のアプローチが必要である。

また、普及にはそのコンテンツを作成するための構築アプリケーションも重要である。三次元コンンピュータグラフィックスを製作する多種多様なソフトが既に存在するが、どれも扱うには専門的な知識を必要とし、アニメーションとなると原理や数式を理解していなと使いこなせないものがほとんどである。もっといろんな人に使ってもらうためには、これらの数学的な機構を内部に閉じ込め、直感的なインターフェイスを前面に出したツールが必要となる。

２．目的

本プロジェクトは Web3D の新しい活用方法として、３Ｄで表現されたキャラクタをホームページの案内役として活躍させるシステムを提案するものである。３Ｄキャラクタを HTML のフレーム制限から開放し、ウェブページの上を自由に歩き回れるようにする。また、ただのマスコットとして画面に配置するだけではなく、キャラクタにサイトを案内させ、サイト閲覧者の操作に随時反応できるようになれば、このシステムはより実用的になる。

2D と 3D との違いは座標軸が１つ増えただけではなく、計算によりイメージを自動生成するという点も重要である。キャラクタエージェントは多種多様な表現を必要とするため、ビットマップイメージだけでは表現に限界がある。

開発者は本研究をはじめる前から 3D アニメーションツール Mikoto を開発、公開していた（図１）。今回、これに新しい機能を付加することで、目的のアニメーション構築アプリケーションを開発する。Mikoto は３Ｄアニメーション専用ソフトであり、形状モデルを作成することはできない。そのため、形状モデリングに関してはすでに存在する別のアプリケーションを利用する。重要なのは、3D キャラクタ、コンテンツを製作するにあたり、作業しやすい環境をいかにして構築するかという点である。本研究では形状モデルにボーンを仕組み、アニメーションを編集するまでの仕様についても思案する。

本プロジェクトの最終的な目標は、キャラクタエージェントとして利用できる独自の Web3D システムと、その構築アプリケーション Mikoto を普及させることにある。本論文はその研究内容を書き記したものである。

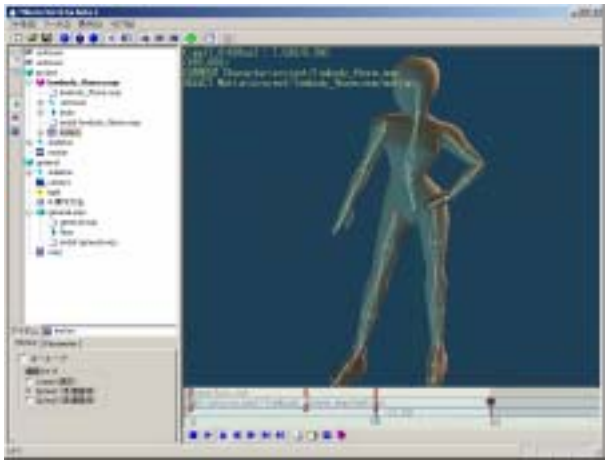

図１　Mikoto の実行画面

３．成果の概要

本プロジェクトでは、主に以下の作業を行った。

１）Web3D キャラクタエージェントシステムの開発
２）再利用性の高いキャラクタセットアップシステムの開発
３）バネモデルを基とするインバースキネマティクスの研究（FlexIK）
４）Python による機能拡張システムの開発

尚、２）３）４）は Mikoto 上で実装する機能である。

外部モデラーに Metasequoia、外部スクリプトに Python を利用する。対応ＯＳは Windows98/2000/XP のみで、InternetExplorer と NetScape 上で動作する。各アプリケーションの関係は（図２）のようになる。

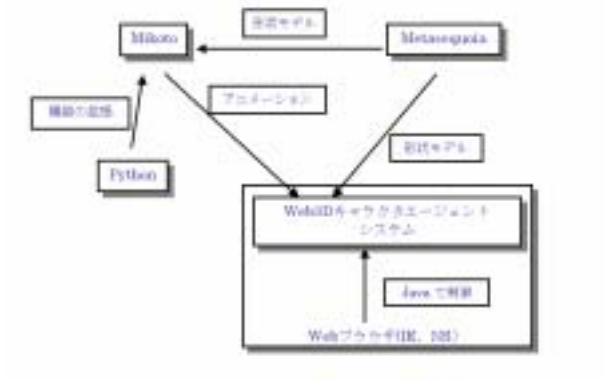

図２　各アプリーションの関係

４．Web3D キャラクタエージェントシステムの開発

（１）　概要

MicrosoftWord 等でヘルプ機能を利用すると登場するイルカのことをキャラクタエージェントと称するが、Web ブラウザ上に描画された３Ｄキャラクタでそれと同様の表現ができるシステムを作成する。

Web3D をキャラクタエージェントとして利用するためには、ブラウザ内を自由に移動することができなければならない。そのためにはブラウザ上で高速に 3D 計算、描画する必要がある。Java3D ではその部分に問題があるため、ActiveX と DirectX8 を利用する。実際の 3D エンジン部分は Mikoto で実装しているものをそのまま利用することで、Mikoto 上でデザインしたビジュアルとアニメーションを完全に再現することができる。

（２）　擬似透過処理

キャラクタエージェントとして描画するには、ブラウザから割り当てられたフレームに描かれたキャラクタと背景とを重ね合わせる処理が必要である。
InternetExplorer などのブラウザはフレームの透過処理は非サポートであるため、自力で透過処理を行う必要がある。しかし、ブラウザに描画される背景となるビットマップを取得する手段がない。そこで、ブラウザのビューウインドをサブクラス化し、PAINT メッセージを乗っ取ることで背景のビットマップを取得することにした。
PAINT メッセージ発生時、ディスプレイデバイスに背景にするべき画像が描画されるので、背景データとして保存。すぐさま３Ｄキャラクタと重ねてフレームに描画することで擬似的な透過処理を行う。

（３）　結果

背景の更新が頻繁に発生した場合、点滅が激しく表示が芳しくない状態になる場合があるものの、背景が頻繁に更新されない限り問題はない。Web3D の新しい活用方法として提案していきたい（図３）。

図３　サンプルコンテンツにおける Web3D キャラクタエージェントシステムの実行画面

５．再利用性の高いキャラクタセットアップシステムの開発

（１）概要

このシステムは Mikoto 上に実装した。Mikoto には形状モデリングに相当する機能はないので、外部から形状データを持ってくる必要がある。Mikoto は Metasequoia

を形状モデリングの外部アプリケーションとして利用する。Metasequoia はフリーウェア版とシェアウェア版があり、プロアマと問わず幅広く利用されている形状モデリングツールである。

問題はボーンやそれに対する影響力の設定方法である。Mikoto にそれらの機能を付加すると、影響力やボーン情報を保持したまま形状データを再編集することはできない。このことは、データの再利用を難しくさせる。モデルやボーン、その影響力を同じジオメトリデータ内に保存することが望ましい。各情報をレイヤー分けすることで再利用性が高いシステムを構築することが可能である。

（2）影響力の自動算出

今までのボーンに対する各頂点の影響力設定は、ウェイトマッピングという手法で行うが定番であった。この手法は各頂点にボーンの重みを手作業で割り当てる方法で、対象モデルの形状が変更されると、再設定を余儀なくされた。この問題を解消するには、その影響力を箱状のプリミティブで定義し、その内側に変形対象になるモデルを内包するように配置する（図４－１）。この箱状のプリミティブと内包する頂点の距離から重みを算出することで､ウェイトマップ編集作業の省略が可能になる。再利用性が高く、セットアップ作業にかかる時間を大幅に短縮することができる（図４－２）。

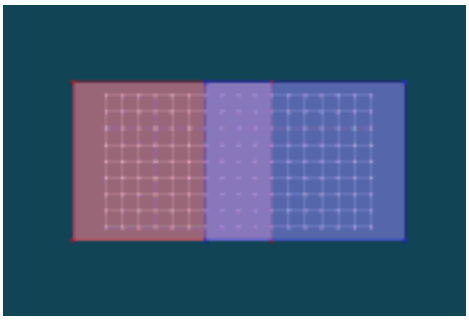

図４－１ 影響力を定義するプリミティブ

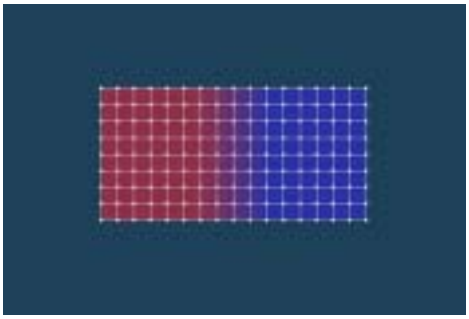

図４－２ 影響力を自動算出した結果

（3）姿勢補間によるボーン変形

３Ｄキャラクタの姿勢を制御する際、ボーンとよばれる骨格を使ったモデル変形を使う方法が一般的である。各頂点に設定されたボーンに対する影響力から局所変形を実現する。問題となるのは、曲げた内側にできる、つぶれる現象である（図５－１）。ボーン間の角度に対し影響力から位置をブレンドするのではく、姿勢をブレンドすることで、つぶれる現象を防ぐことができる。球形状に変形するので、２対の連続接続されたボーン変形の場合に限り、形が内側に潰れることがない。（図５－２）

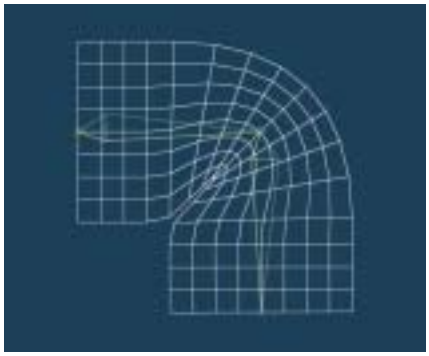

図５－１ 線形補間によるボーン変形

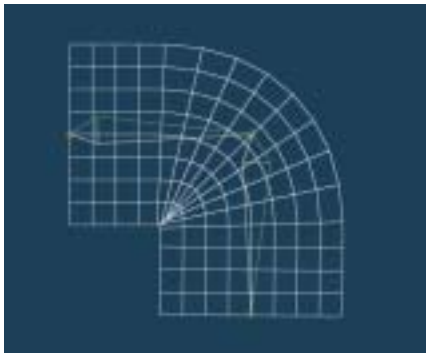

図５－２ 姿勢補間によるボーン変形

（4）結果

ボーン構造とボーンによる局所変形の影響力情報をジオメトリデータとして持つことで、モデルデータの変更しても再設定する必要がないまま対応できる。それらの情報は形状モデルと別レイヤーで保存されているため、別モデルへの再利用も少ない手順で適用することができる。

ボーンに関するすべての情報はモデラー側で設定するため、一度形状データを出力し、Mikoto に読み込ませないことにはデータの確認ができない。このことは、細かい調整作業の際にモデラーで出力したデータを Mikoto でインポート、確認するという作業を何回も繰り返すことになる。

６．バネモデルを基とするインバースキネマティクスの研究

（1）概要

３Ｄアニメーションは専門的な数式が必要となり、直感的にデザインすることができない現状では普及を促すことは難しい。あたかも実際に手を使って人形を操っているように操作できるシステムとユーザインターフェイスを開発することを目指す。

３Ｄ編集ソフトのでは骨格モデルを使い､腰から手足、頭にかけて座標系での親子関係が構築され、親から順に

姿勢を決めることで全体の姿勢が確定する。もし手足の先を固定して場合などの要求がでた場合、子から親に向かって順次姿勢を確定する、インバースキネマティクスの概念と取り込む必要がある。

（2）バネモデルを基とするインバースキネマティクスの研究（FlexIK）

そこで、バネモデルを利用した新しいアプローチによるインバースキネマティクスを研究、開発することにした。この手法では自由に任意の部分を固定することができ、複雑な拘束条件でも理想的の姿勢を取ることができる。バネが安定するまで内部ループを何回も計算することで目的の姿勢を算出する。しかし、この方法論では多くの問題がある。

1）発散現象
2）内部ループ数による計算量増大
3）精度の問題
などが上げられる。

各ポイントの速度パラメータを排除し、各拘束エレメント単位で理想的な位置を決め付ける。平均位置を算出し、すべての拘束条件を満たす姿勢へ徐々に近づけるといった手法をとることで発散現象の問題を解決する。

精度を限りなく上げることは可能だが、それに応じて安定の条件が厳しくなり、内部ループ数が増す。よって計算量が増大し、動作が鈍くなるといった不具合が発生する。見た目で問題ない精度と快適な速度を保つ、適した値を設定することにより、実用上ほとんど問題ない結果を導くことに成功した。

（3）フォワードキネマティクスとインバースキネマティクス

FlexIK 技術を利用すれば、各ポイントを引っ張ることでポイント間の距離を一定に保ったまま最も変化の少ない姿勢を作ることができる。しかし、一つのポイントに作用した力はボーンによって接続している全てのポイントに力が伝達されるため、姿勢を保ったまま平行移動させることができない。フォワードキネマティクスだけでは直感的に姿勢をとることはできないが、インバースキネマティクスだけでは姿勢が変化しやすく、整った姿勢を保持したまま操作することが難しい。この問題を解消するにはユーザ自身が部位やアニメーションの種類によって、フォワードキネマティクスとインバースキネマティクスを明示的に切り替える必要がある。

また、固定するポイントを決めることで、インバースキネマティクスとフォワードキネマティクスのうち、どれで制御するか自動的に切り替える仕組みを作らなければならない。例えば、両足のみ固定したい場合などは、腿と脛のボーンはインバースキネマティクスで制御されるべきである。これは、固定した点から親にさかのぼってその間にあるボーンをインバースキネマティクスで制御することで解決する。また、手腕部分はインバースキネマティクスで制御したいが、胴体部分はフォワードキネマティクスで制御したい場合などがある。その場合、インバースキネマティクス制御により、子から伝わる力をカットする役割を持つ IK チェイン切断フラグを用意することで解決する。

（4）ユーザインターフェイス

インターフェイスはその機能の価値や評価を大きく左右する。機能は優れていてもインターフェイスが未熟だと多くの人に利用してもらうことができない。最短の手順で姿勢の編集ができるようにデザインすることが重要である。

インポート直後はすべてのボーンはフォワードキネマティクスで制御された状態で初期化される。初期状態で固定化されているのはルートに位置するポイントのみである（図６－１）。通常、フォワードキネマティクスで制御されているボーンを左クリック、そのままドラッグすることでそのボーンを回転させることができる。また、ルートに位置するポイントをドラッグすると姿勢そのままに全体が移動する。任意のポイントを固定したい場合、固定したいボーンをもしくはポイントを左クリックし、左ボタンをホールドしたまま右クリックすることで関連するポイントをワールド座標上で固定化する。固定化されたポイントから IK チェイン切断フラグが立っているポイントを持つ親ボーンまでがインバースキネマティクスによる生成制御の対象となる(図６－３)。インバースキネマティクスで制御されているポイントは左ドラッグにより調整することができる。

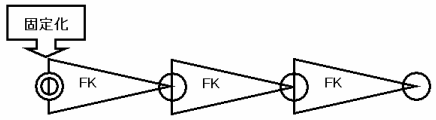

図６－１ インポート直後の状態

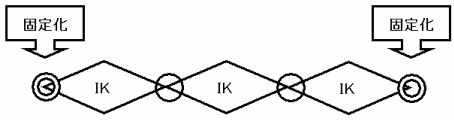

図６－２ エンドポイントを固定化した状態

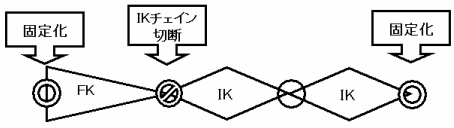

図６－３ IK チェイン切断フラグを立てた状態

（4） 結果

足元を固定したまま、腰を落とすことに成功した（図7）。しかし、インターフェイスの設計にはまだ問題があり、全キーフレームを対象にした一括操作などの機能を充実することで実用性を高める必要がある。また、角

度制限に相当する機能がないため、操作中に不自然な姿勢になることが多々ある。角度制限が実現すれば、理想的な姿勢制御システムに発展する可能性があると実感した。

図７ FlexIK の動作結果
左）足元を固定化した状態
右）腰にあるポイントを移動操作した後の姿勢

７．Python による機能拡張システムの開発

（１） 概要

個人でＣＧソフトを開発するのは作業量的に無理がある。一時はユーザサイドで自由に拡張できるようにするため、オープンソース化や SDK を配布して解決することも考えたが、コアシステムの仕様が決定していないうえに隠蔽したいコードもあり断念した。そこで Python を Mikoto のスクリプト言語として利用することで、上記問題を解決し、ユーザ自ら Mikoto の機能を拡張できる機能を提案する。

（２） 実行時型情報の生成

Mikoto には多くのクラスがあり、それら全てを外部に出すには多くの作業量を必要とする。また、クラスの仕様を変更するたびに Python 側のクラス定義も変更したのでは実用的ではない。Ｃ＋＋で定義したクラスの情報を実行時に判断することができれば、Python クラスとＣ＋＋クラスの定義を同期させることが可能である。

Ｃ＋＋標準の実行時方情報（RTTI）ではクラス名を取得できるものの、メンバ名とその引数の型を判断する手段がない。マクロやテンプレートを使用して自力でクラスのメンバ情報をデータ化する必要がある。C++で定義されたクラスのメンバとメソッドの情報を実行時に構築する。

（３）結果

スクリプト機能を実装することにより、数式によるボーンの制御や、ユーザ定義によるインポート、エクスポートなど利用できる。また、Mikoto の機能をユーザレベルで拡張できる。このことは初心者だけではなく、本格的に利用することを思慮したコアユースへのアプローチになる。

８．まとめ

本プロジェクトでは Web3D キャラクタエージェントとして利用可能なシステムとその構築アプリケーションの開発を行った。機能は制限されているものの、本研究で開発した一部の機能は既にホームページで公開中である。

Web3D キャラクタエージェントの普及の努力を行うと同時に、構築アプリケーションの普及も目指す。基本的にフリーソフトとして公開することで、幅広く利用してもらえるようにする。

Mikoto に関する新しい機能はまだ開発中であるが、本研究での成果を元に完成度の高め、実用的な機能に仕上げたものから順次公開していく予定である。また、FlexIK の研究、開発をさらに進める。開発結果は開発者のホームページで順次公開していく。
http://www3.nbn.ne.jp/~you-ri/

９．参加企業及び機関

有限会社アントラッド

１０．参考文献

［1］Eric Lengyel：ゲームプログラミングのための３Ｄグラフィックス数学, ボーンデジタル, 2002
［2］Mark Lutz ： Python プログラミング, オーム, 1998
［3］Adam Denning ： ActiveX Controls Inside Out, アスキー, 1997