次世代型モーションキャプチャシステムの製作

４年A組　武田 優生
指導教員　米田 隆恒

１．要約

私は、独自に開発したハードウェアと Kinect を用いることにより、従来のものに比べて、非常に安価なモーションキャプチャシステムを製作することに成功した。

キーワード　　モーションキャプチャ、物理エンジン、Kinect、ジャイロセンサ、画像処理、OpenNI、OpenGL、Bullet、3DCG、ウェアラブルコンピューティング

２．研究の背景

モーションキャプチャとは、実物の人物や物体の動きをデジタル化して記録する技術である。現在この技術は、映画の CG 作成や、スポーツ選手の身体の動きなどのデータを収集するために使われている。

一般的に用いられている「光学式モーションキャプチャ」は、精度が高く、よりリアルな CG 作成には欠かせない。しかし、複数のカメラやマーカ、専用のスタジオを用いて、人物や物体の動きを記録するため、動きが制限されてしまうだけでなく、モーションキャプチャを行う環境を用意するために、数千万円単位のコストがかかるという問題がある。

光学式モーションキャプチャの次に用いられている「機械式モーションキャプチャ」は、専用の装置を身につけてモーションキャプチャを行うため、光学式モーションキャプチャに比べ、非常に明るい場所や広い範囲を必要とする動きといった過酷な環境下でもモーションキャプチャを行うことができる。しかし、これもまた、専用の装置を身につけるため、装置の重さなどによって動きが制限されてしまうという問題を抱えている。

図１　光学式モーションキャプチャ
(Wikipedia より抜粋)

ところで、モーションキャプチャという技術は映画の CG 作成と行った特殊な用途だけではなく、身近なものの操作に活用できるのではないかと考えられる。例えば、スマートフォンの普及と共に高機能化した「マルチタッチ機能」である。

図２　マルチタッチ機能を搭載した
スマートフォン(iPhone4)

本来マルチタッチ機能は、複数の指の動きや位置情報を、タッチパネルを用いて取得することで、直感的な操作を可能にするものである。しかし、タッチパネルを用いて取得するため、２次元空間の位置や動きを取得することしかできない。また、公共の場に設置する場合、不特定多数の人が物理的に触れるため、衛生面での不安がある。

そこで、複数のカメラやマーカ、専用のスタジオを用意することなく、高精度なモーションキャプチャを実現することはできないかと私は考えた。

３．本システムの概要

本システムは、Microsoft 社製ゲーム機 XBox360 のコントローラーである Kinect と自作ハードウェアによって構成されている。

Kinect には、640*480 の画像を 30fps で出力できる RGB カメラと、その画像に対するカメラと物体の深度を取得する赤外線深度センサ、ユーザーの声を取得することができるマルチアレイマイクロフォン、そしてこれらの信号を処理するプロセッサが搭載されている。

図３　Kinect Microsoft 社製
XBox360 コントローラ

図４　自作ハードウェア

また、自作ハードウェアには、２軸コンパスセンサと、２軸ジャイロセンサの２つのセンサ、これら２つのセンサの信号をワイヤレスでパソコンへと送信する XBee モジュールが搭載されている。

４．研究内容

（１）研究項目

［研究１］ OpenGL を用いて、3D オブジェクトを表示する。

［研究２］ Kinect から画像と深度センサーの値を取得する。

［研究３］ Kinect を使い、ボーントラッキングを行う。

［研究４］ 回転運動の精度向上のため自作ハードウェアを製作する。

［研究５］ 物理エンジンと組み合わせて、仮想空間の物体と干渉させる。

（２）研究内容

［研究１］ OpenGL を用いて、3D オブジェクトを表示する。

OpenGL とは、Silicon Graphics 社(SGI)が中心となって開発した 2D/3D 用グラフィックスのためのプログラムインターフェイスである。この OpenGL の動作確認をするため、立方体を表示することにした。立方体を表示するために必要な関数は、OpenGL にあらかじめ用意されているため、比較的簡単に動作確認をすることができた。

OpenGL を使用して立方体を表示するには、最低でも３つの関数が必要となる。まずはじめに、glLightfv で光源の位置を指定し、glMaterialfv という関数で、オブジェクトに対する光源の反射具合を設定する。最後に、glutSolidCube(size)を実行することで、立方体を表示させることができる。

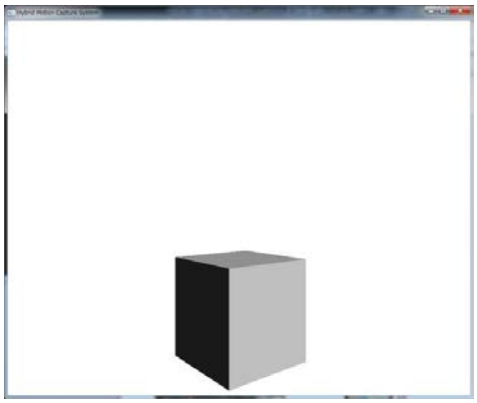

図５ OpenGL で立方体を表示する

次に、テクスチャのある 3D オブジェクトを表示することを試みた。しかし、OpenGL 自体に 3D オブジェクトを読み込んで表示する機能がないため、GLmetaseq というモデルローダを用いてメタセコイアという 3DCG ソフトのデータを読み込んだ。

ところが、このモデルローダでは、テクスチャとして貼りつけて使用する画像が、ファイル容量の大きい bmp ファイルと tga ファイルしか読み込むことができない。ファイル容量が大きいと、性能の良いコンピュータでしかプログラムを実行することができないという問題や、実行したとしても、画面を書き換える回数が少なくなってしまい、動作がカクカクしてしまうという現象が発生してしまう。

そこで、libpng という PNG デコーダをプログラムに組み込んだ。Libpng とは、PNG 方式で圧縮された画像を展開するためのライブラリで、PNG 公式のリファレンスライブラリであるため、PNG の仕様のほぼすべてをサポートしている。

このライブラリを使用すると、ファイル容量の少ない PNG ファイルをテクスチャに使うことができる。

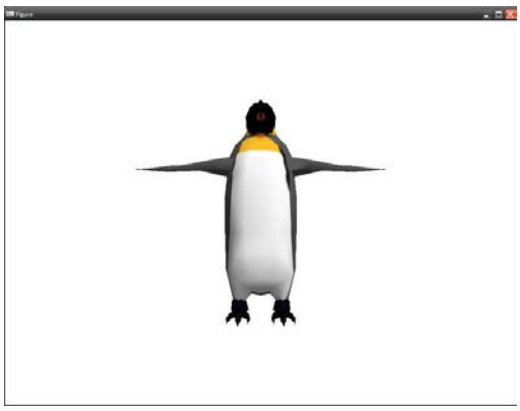

図６ PNG 読み込みに対応した
モデルローダで
ペンギンを 3D 表示している様子

［研究２］ Kinect から画像と深度センサーの値を取得する。

Kinect は、開発コードネームが Project Natal と呼ばれ、コントローラーを用いずにゲームの操作を行うゲームシステムを確立するために開発されてきた。実際の用途としては、ジェスチャーや音声認識を行い、直感的にゲームを操作するといったものである。

この Kinect の出力端子は USB であるため、パソコンに接続して制御することで、モーションキャプチャシステムに転用することができる。また、Microsoft 側も転用することを容認しており、Kinect の USB 接続はあえて暗号化していないという。

図７　Kinect と接続できる USB ケーブル

Kinect の制御には、OpenNI という Natural Interaction のためのフレームワークを用いて、ソフトウェアから制御する。Microsoft 社に Kinect の深度センサを提供している、PrimeSense 社や OpenCV を提供している WillowGarage 社らが立ち上げた OpenNI.org で提供されているオープンソースのライブラリである。

OpenNI には、Kinect に接続を行うデバイス部とそのデータの画像解析を行う、ミドルウェア部の２つを統合して扱うインターフェイスになっている。そのため、OpenNI をプロジェクトに追加し、Kinect に接続すると、Kinect からカメラの画像と赤外線深度センサからのデータを取得することができる。

赤外線カメラの画像と赤外線深度センサを重ねて表示すると、図８のように表示することができる。

図８　OpenNI でカメラと赤外線深度センサのデータを重ねて表示

また、ミドルウェア部の一部は、Prime Sensor NITE と呼ばれるプログラムをサポートしており、これのミドルウェアを使用すると、Kinect の画像データの中から背景を差分し、人物だけをトラッキングすることができる。

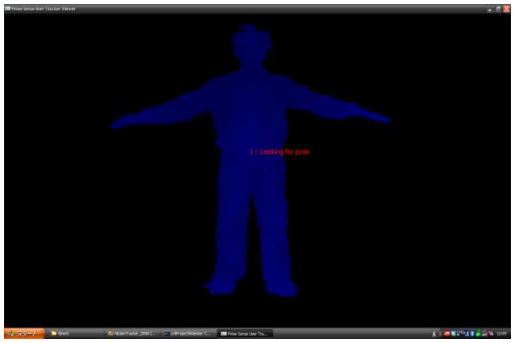

図９　NITE と Kinect を使い、人物トラッキングを行っている様子

［研究３］ Kinect を使い、ボーントラッキングを行う。

Prime Sensor NITE は、スケルトン処理と呼ばれる骨格モデルを表示するボーントラッキングも行うことができる。

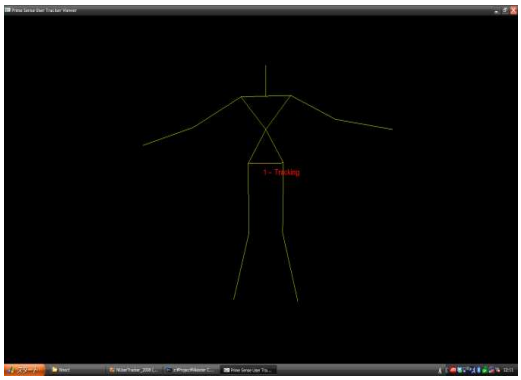

図 10　Prime Sensor NITE を使い、ボーントラッキングを行っている様子

これらを組み合わせることにより、人物をトラッキングしながら骨格モデルを表示することができた。

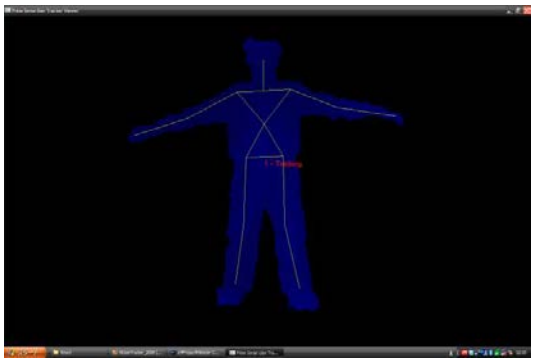

図 11　人物をトラッキングしながら、ボーンを表示している様子

［研究４］ 自作ハードウェアと組み合わせて、精度を向上させる。

研究３により、Kinect を用いてボーントラッキングを行うことができた。

しかし、Kinect は深度センサを用いて、ユーザーを認識するため、その性質上、奥行きの変化が少ない運動、つまり回転運動に対してはうまく検知することができなかった。この動きは、タッチパネルとして使う場合には特に必要のない動きだが、遠隔会議システムなどを作成するときには必要となる。

そこで、自作のハードウェアを使い、Kinect のボーントラッキングの回転運動補正を行った。

自作のハードウェアには前述した通り、２軸のコンパスセンサと２軸のジャイロセンサの２つのセンサを搭載している。

２軸のコンパスセンサは、ヨー角の回転(Z 軸の回転)を取得するために用いられる。このセンサとして、Honeywell International 社製２軸コンパスセンサ HMC6352 を採用した。

図 12　HMC6352 搭載 Sparkfun 社製２軸コンパスモジュール

このセンサの特徴は、I2C インターフェイスを用いて、2 本のデータ線で現在の角度を、北向きを 0°とした絶対的な回転角を出力することである。分解能は 0.1°と、モーションキャプチャを行うには十分な精度である。これにより、ヨー角の角度は正確に取得できる。

しかし、このコンパスセンサは水平な平面上で使われることを想定しているため、許容範囲を超えて大きく傾いた状態で測定すると誤差が非常に大きくなってしまう。そのため、ロール角の回転(X 軸の回転)、ピッチ角の回転(Y 軸の回転)の取得には使うことができない。

そこで、2 軸のジャイロセンサを用いて、ロール角、ピッチ角の回転運動を取得することにした。今回、このセンサとして村田製作所の圧電振動ジャイロセンサ「ジャイロスター」を採用した。

図 13　村田製作所の圧電振動ジャイロセンサ「ジャイロスター」を搭載した秋月電子通商の圧電振動ジャイロモジュール

しかし、ジャイロセンサであるため、特性上、相対的な値しか出力することができない。また、絶対的な回転角を出力するわけではないため、使用しているうちに誤差が大きくなってしまう問題や、 圧電振動ジャイロセンサゆえに、ドリフトと呼ばれる基準点が外部の振動によりずれてしまう現象が起こり、正確な回転角が取得できなくなってしまうという問題がある。

そこで、このセンサを補正する演算を追加し、2 軸ジャイロセンサだけでもロール角の回転とピッチ角の回転を取得できるようにした。

次に、これら 2 つのセンサのデータを PC へワイヤレスで送信するために、XBee モジュールを搭載した。XBee とは、IEEE802.15.4 に準拠した 2.4GHz 帯無線通信モジュールで、115,200bps の転送速度で PC と通信することができる。日本での TELEC の適応を受けており、マイコンと接続することで比較的簡単に無線通信をすることができる。XBee にはさまざまなシリーズがあるが、今回は、「XBee Series1 Chip Antenna Model」を採用している。

図 14　XBee Series1 Chip Antenna Model

また、自作ハードウェア側からのデータを受信できる距離を長くするため、PC 側には 10cm 程度の長さを持つアンテナを外付けにした「XBee Series1 U.FL Antenna Model」を使用した。

図 15　XBee Series1 U.FL Antenna Model

外付けアンテナには、TELEC(財団法人テレコムエンジニアリングセンター)の認証が通っている、10cm 程度の U.FL コネクタ接続のアンテナを使用した。

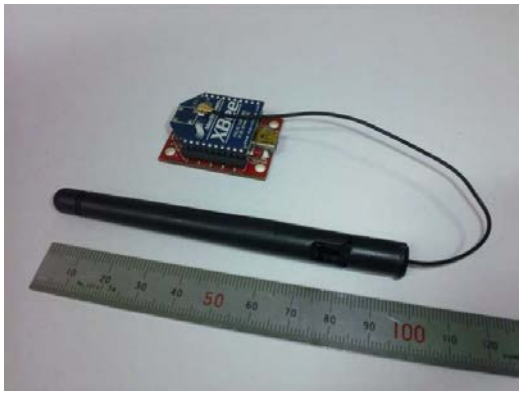

図 16　外付けアンテナ

これにより、センサ側の子機 XBee では省スペースなチップアンテナを採用しながら、より長い転送距離を稼ぐことができた。

また、転送距離を延ばすためには、外付けの U.FL アンテナを垂直に立てる必要がある。そこで、設置を容易にするため、専用のケースの図面を CAD で作成し、CNC フライス(コンピュータ数値制御フライス盤)を用いて 2mm のアクリル板から、6 枚のアクリル板を削りだし、組み立てた。

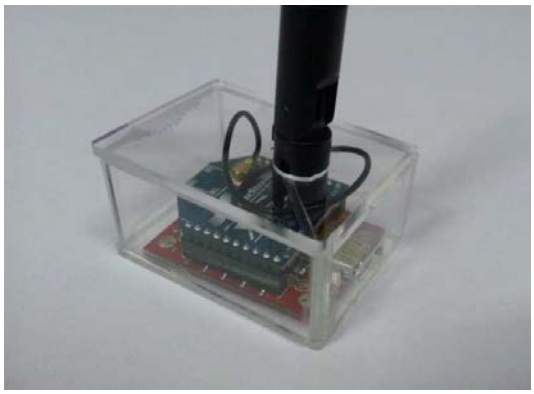

図 17　PC 側 XBee 専用ケース

今回の用途では、PC 側にセンサからのデータを受信するホスト用の XBee が 1 台なのに対し、センサ側の XBee が複数存在するため、スタートポロジ(ブロードキャスト)方式でデータを転送している。これにより、PC 側の XBee でセンサ側の XBee を一括管理でき、子機側からのデータを一括受信することに成功した。

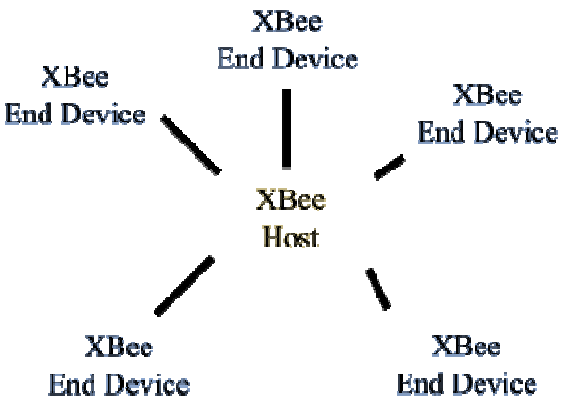

図 18　スタートポロジ
(ブロードキャスト)方式

さらに、この複数ある XBee に対して、ハードウェアの固有の名前と ID を振り分けて登録するソフトウェアを Visual Basic を用いて作成した。

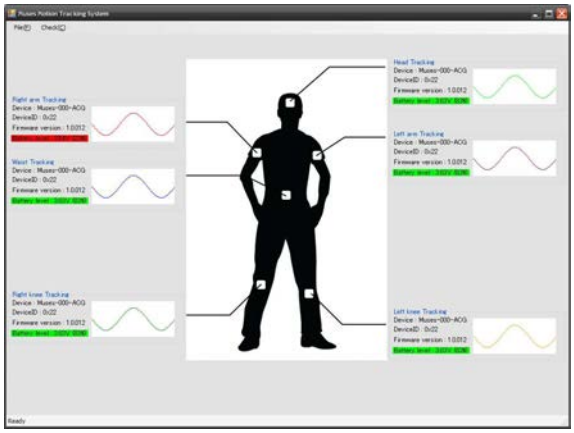

図 19　自作ハードウェア登録ソフト

このソフトウェアを使うことで、設置した自作ハードウェアの固有の名前と ID を登録し、CSV ファイルとして書き出すことができる。この情報をもとに、Kinect とハードウェアを組み合わせる。

[研究５]　物理エンジンと組み合わせて、仮想空間の物体と干渉させる。

Kinect と自作ハードウェアを用いてモーションキャプチャを行うことで、仮想空間上の物体に触れることができるが、それらには、現実世界のような物理法則が働かないため現実味に欠ける。

そこで、物理エンジンを搭載することを考えた。仮想空間の物体ごとに、衝突判定や引力、反発係数、摩擦係数といったさまざまな係数を計算することで、現実世界の物理法則を働かせることができる。

物理エンジンとして、AMD が中心となって開発しているオープンソースな物理演算エンジンの”Bullet”を使用した。Bullet は物理演算エンジンとして最適化されており、非常に高速に物理演算を行うことができる。

さまざまな係数を指定して Bullet でシミュレーションを行うとその結果が出力される。その出力された情報をもとに、OpenGL でオブジェクトを描画すれば、物体はまるで現実空間での動きかのように振る舞うことができる。

５．考察

今回の研究で、Kinect を用いて全身の動きを取得することができたが、[研究３]、[研究４]、[研究５]において、いくつかの問題点が挙げられる。

[研究３の問題点]

Prime Sensor NITE を用いてボーントラッキングを行ったが、激しい動きをするとボーントラッキングから外れてしまう。

これは、Kinect のカメラの更新周期が 30fps(秒間 30 枚)のため、高速な動きに対応できないからである。

そこで、自作ハードウェアを改良して高速な動きは自作ハードウェア側で取得し、Kinect のボーントラッキングに補正をかけることで改善ができるのではないかと考えている。

[研究４の問題点]

今回は、自作ハードウェアを汎用ユニバーサル基板上で作成したため、装置が大型化してしまった。また、コンパスセンサの出力はデジタル、ジャイロセンサの出力はアナログと、デジタル出力とアナログ出力が混在してしまい、アナログ出力であるジャイロセンサにノイズが乗ってしまうときがあった。

次回、ハードウェアを作成するときにはプリント基板を発注し、より小型化する。、使用する部品を再検討し、ノイズが乗りにくいように改善したいと考えている。

[研究５の問題点]

今回は、物理エンジンに Bullet を用いたが、これだけでは条件を指定するための係数が不足する場合があった。また、物理エンジンが非常に CPU パワーを必要とするため、最新の PC でしかまともに動作させることができなかった。

今後、改良していくにつれて、Bullet を徐々に廃止し、CPU パワーをあまり使わない物理エンジンを自分で作成し、実際に使用していきたいと考えている。

６．今後の課題

物理エンジンを搭載することで、仮想空間がより現実に近い世界になった。

これらの技術に加え、私が今までに研究してきた母音認識システムを追加すれば、仮想空間上に表示した人物のモデルのモーフィング(口パク)を行うことができると考えられる。

これらの技術をすべて使うことができれば、高性能な遠隔会議システムを作成することができるのではないかと考えている。

今までの遠隔会議システムは、Web カメラからの画像とマイクからの音声をたよりに会議を行っていたが、このシステムが完成すれば、仮想空間上で実際に会議を行うことができる。さらに、身振り手振りを Kinect が検出し、話す内容に応じて口が動けば、より相手に意思が伝わりやすいのではないかと考えている。

また、現実のオブジェクトを、あらかじめ Kinect を用いて 3D オブジェクト化し、RFID(近距離無線個体識別技術)を使い登録しておけば、現実のオブジェクトを画面に近づけるだけで仮想空間上に表示することができるだろう。

これらをふまえて、今後は自作ハードウェアの小型化と会議システムの開発を行っていきたい。

７．参考文献

[1] OpenNI - OpenNI.org
http://www.openni.org/
[2] MSDN ライブラリ - Microsoft japan
http://msdn.microsoft.com/ja-jp/library/ms123401.aspx

８．謝辞

今回の研究にあたり、様々なアドバイスをいただいた顧問の米田先生、サイエンス研究会統括顧問の川口先生にはご指導、助言をいただきました。さらに、本校サイエンス研究会物理班の先輩方、メンバーにも多大な協力をいただきました。お世話になった方々に、この場をお借りして深く御礼申し上げます。