作業手順の直感的表現による生産ラインの効率改善に関する研究

Research on the Improvement of Assembly Processes using Intuitive Expression of Human Motion

福谷武司・新見浩司
Takeshi Fukutani and Kouji Niimi

機械素材研究所　計測制御科

近年、安価な３Ｄセンサの普及とそれを操作するオープンソースライブラリを用いて、今までに無かった３Ｄデータ取得や３Ｄ表現が安価に実現できるようになってきている。特に、グラフィックカードに搭載されているＧＰＵを用いた高速並列処理演算を活用することにより、作業者の動作をリアルタイムに取得することが現実味を帯びてきた。本報告では、これを実現するプログラム手法とこれを生かした作業者の動作リアルタイムキャプチャ手法の開発方法と、その動作について述べる。

In recent years, it has become easier to use low-cost 3D sensors and open source libraries to manipulate them to create 3D point cloud data of real objects and expressions never seen before. Especially, using a GPU(Graphic Processing Unit) on a Graphic Board, high-speed parallel processing can show the work of a factory worker in an assembly process in realtime. This report describes several ways to develop such programs for accomplishing this.

1. はじめに

従来、３次元動作キャプチャは、動作対象物に多くのマーカーを貼付し、それを特徴点として画像処理を行い、解析していく場合が大多数を占めていた。マーカーを用いない手法はキャプチャ点が必然的に多くなり、リアルタイム処理はハードウェア的に厳しかった。しかし、近年ＨＰＣ（High Performance Computing）の１手法として、グラフィックカードに搭載された画像表示のためのプロセッサ（ＧＰＵ）による、高速並列処理演算プログラム手法がＣＰＵのクロック周波数上昇による高速化が頭打ちになるに連れて脚光を浴びており、これを生かしてリアルタイムな処理を実現できる可能性が現実味を帯びてきた。

この手法の代表的なものとして、CUDA というNVIDIA 社よりリリースされている、同社グラフィックカード向けの開発環境を利用したものがある。この手法では様々な方式が利用可能であるが、それぞれ利用シーンに応じて選定することが一般的である。

本報告では、このグラフィックカードを使った高速並列処理演算プログラム手法を実現するための３通りの方法について、順にそれぞれの環境構築とこれらを利用したプログラム例を挙げながら説明するとともに、その中の１つの手法を用いた作業者の動作リアルタイムキャプチャを実施したので報告する。

2. 開発ハードウェアとプログラム開発環境

2.1 開発用ハードウェア

作業者の動作のリアルタイムキャプチャを実現するためのハードウェアとして、表１に示す開発環境を構築した。

表１　開発環境構成

PC	CPU	Intel Xeon E5-2690(2.9GHz)
	メモリ	DDR3-1333 SDRAM 64GB
	OS	Windows 7 64bit
ｸﾞﾗﾌｨｯｸｶｰﾄﾞ	名称	NVIDIA Quadro 6000
	ｺｱ数	448
	ｸﾛｯｸ	1150MHz
	ﾒﾓﾘ	384bit GDDR5 SDRAM 6GB
GPU 開発環境	ｺﾝﾊﾟｲﾗ	C++ｺﾝﾊﾟｲﾗ(Visual Studio 2010 for Professional 同梱)
	IDE	Nsight for VS
	CUDA	CUDA 4.2
ＯＳ		Windows 7 Professional
３Ｄセンサ		Microsoft Kinect (version 1)

2.2 プログラム開発環境

３Ｄセンサを扱うためのプログラム開発環境を構築した。プログラム開発に当たり以下に示す３通りの手法を試行した。

（１）点群処理ライブラリ PCL による方法

これは最も早い時期から使われている手法であり、C/C++言語主体のやや難解なプログラムとなっている。その一方でメモリ管理をプログラマ

の意図した通りに扱えることでハードウェアの能力をより発揮できるプログラム手法であり、実行速度を追求するのであればこの手法で行うのが最も有利である。

（2）Kinect for Windows SDK の利用

これは Microsoft がゲーム機の付属品に過ぎなかったKinectをPCでも制御できるように改変し、無償の３Ｄセンサ周辺ライブラリと共にリリースしたものである。（1）より１年程度遅れて発表されたものの、VisualStudio を使っての開発となっているため、リファレンスやコードヒントが充実しており、プログラミングが比較的容易になっている。

（3）AirKinect と Away3D の組み合わせ

これは Adobe の Flash 用スクリプト言語 ActionScript3.0 のネイティブ拡張機能を使った AirKinect というライブラリによって Kinect を使う方法である。また、Away3D は Adobe が GPU 技術を利用した Stage3D という３D 表現技術を適用したライブラリである。この組み合わせを利用すると、演算速度がやや遅くなるものの、実現したい機能を手早く作成することが可能である。

2.2.1 点群処理ライブラリ PCL による方法

最初の試みとして、点群処理のリアルタイム処理を空間認識に反映させながら、作業者の動作リアルタイムキャプチャを行う。この手法は単なる形状キャプチャになるが、最も基本的手法として重要である。

表２　PCL(Point Cloud Library)開発環境構成

使用ソフト	PCL 1.6.0 all in one
構成ライブラリ	Boost 1.50.0
	Eigen 3.0.5
	FLANN 1.7.1
	VTK 5.8
	Qt 4.8.1
	QtSDK 1.2.1
	QHULL 6.2.0.1385
	CMake 2.8.10.2
ﾀﾞｳﾝﾛｰﾄﾞﾂｰﾙ	TortoiseSVN 1.7.11
3D ｾﾝｻ用ﾄﾞﾗｲﾊﾞ	OpenNI 1.5.4
	avin2/SensorKinect 5.1.0.25

高速並列演算を点群処理に利用するためのライブラリとして PCL(Point Cloud Library＝「点群処理ライブラリ」の意)というものがある。このライブラリとこれを構成するサブライブラリの一覧を表２に示す。

この手法は、本来は UNIX 系 OS で行う GCC(GNU Compiler Collection)C/C++コンパイラにおいて使うライブラリ用の構成がなされているが、これを Windows 環境にてビルドするためのツールCMake を用いてソースを変換することにより、Visual Studio の C/C++言語用ソリューションファイル（拡張子　*.sln）を生成した上で実現する。

具体的には図１のような手順で行う。

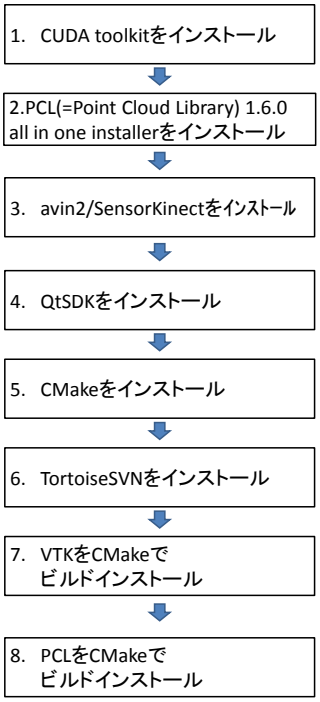

図１　PCL 開発環境作成手順

こうして作成した開発環境から、図２のように点群をキャプチャした上レンダリング処理をした形状を取得するプログラムが作成できた。

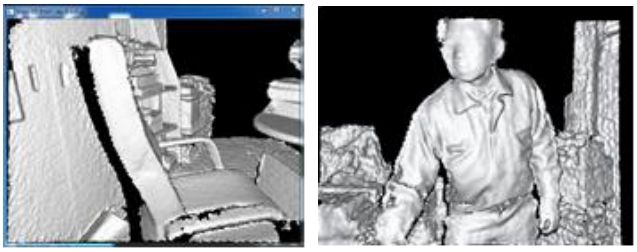

図２　ライブラリプログラム動作例
（左：什器キャプチャ、右：人物キャプチャ）

2.2.2 Kinect for Windows SDK の利用による方法

Kinect for Windows SDK を使う方法であるが、Microsoft の Kinect 開発者向けのページからインストーラ付きのソフトウェアをダウンロードすることで完結する。Microsoft 側で高機能な関数群が多数用意されており、これを C 言語、C#および Visual Basic で、GUI 画面構成をしながらプログラムを作っていくことができる。また、チュートリアルやユーザー相互の情報交換サイトなどを利用しながら豊富な情報と図３に示すプログラム紹介ページにて、様々なサンプルプログラムを試しながら作成したい機能を構築することが可能である。この手法によるプログラムの動作例を図４に示す。

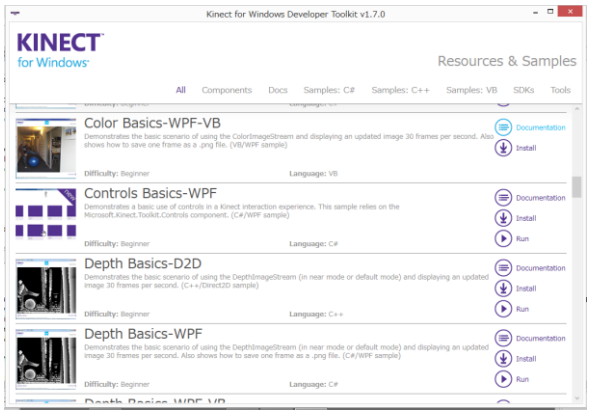

図３　Kinect for windows
サンプルプログラム紹介ページ

図４　Kinect for Windows のサンプルプログラム動作例
（左：顔認識、右：骨格認識）

2.2.3 AirKinect と Away3D の組み合わせ

通常ＧＰＵライブラリはハードウェアと強く結びついているため、基本的には C/C++言語のようなメモリ領域を扱うことのできる言語で行うのが一般的である。しかし、C/C++言語はその扱いが非常に複雑であり、プログラム上のバグを生じさせやすく、そのバグも見つけにくい。近年、GPU ライブラリをスクリプト言語である ActionScript3.0 で扱う Kinect センサ用ライブラリ AirKinect や GPU 技術によって 3D 空間上の物体を表現するライブラリ Away3D が登場しており、これらを利用することにより、実行効率は多少落ちるものの、比較的早期に目的のプログラムを作成することができる。

そして、これらのライブラリを利用するためには、図５に示すように、Adobe 社の Flash Builder を利用してプログラムすることが必要である。Flash Builder の機能として「*.ane」というネイティブ拡張ファイルを取り込む設定がある。

図５　Adobe Flash Builder 開発環境
（ソフトウェア起動ページと
AcrionScript3.0 プログラム画面）

ネイティブ拡張とは、C/C++言語で作成されたライブラリをあたかも AcriontScript3.0 のライブラリであるかのように取り込む Flash Builder 内のコンパイラの機能である。

この手法によるプログラムの動作例を図６に示す。

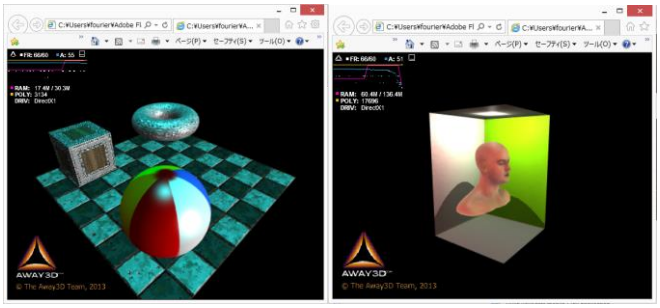

図６　Away3D 動作例
（高速３Ｄレンダリング処理）

3.　動作のリアルタイムキャプチャ

作業者の動作のリアルタイムキャプチャを実現するために基礎となる２つのプログラムを作成した。１つは３Ｄ空間に空間線を描くプログラム。そしてもう１つは3Dセンサ(kinect)から常時PCに対してなされる入力座標データを得るためのプログラムである。これの基本になっているのは「2.2.3 AirKinect と Away3D の組み合わせ」である。

3.1 空間線を描画するためのプログラム

動線軌跡を描画するためには、次のような流れでプログラムを記述する。

（１）描画用３次元空間の設定

描画用３次元空間を設定する。Vector3D 型による３Ｄ座標を使って、空間座標プロット用３次元グリッドを作成する。

（２）３次元空間描画点の生成／更新

乱数を使って空間描画点の座標を決定する。１つ目の点と２つ目の点を線で結ぶ。線の描画は SegmentSet 関数を用いる。

３Ｄ軌跡を描画するプログラムの動作例を図７に示す。

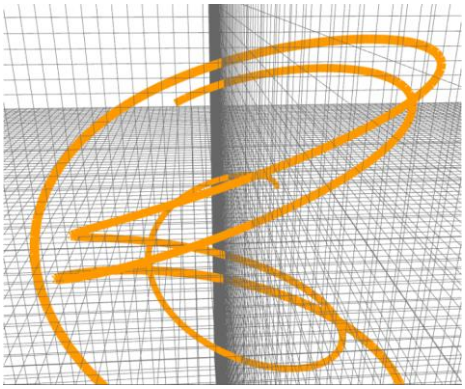

図７　プログラムによる３Ｄ軌跡描画

3.2 ３Ｄセンサ(kinect)の入力データを得るためのプログラム

ActionScript3.0 で記述できる kinect を利用するためのプログラムは、AirKinect というものが最もよく使われる。Kinect センサを利用するためには図８のような流れでプログラム記述を行う。

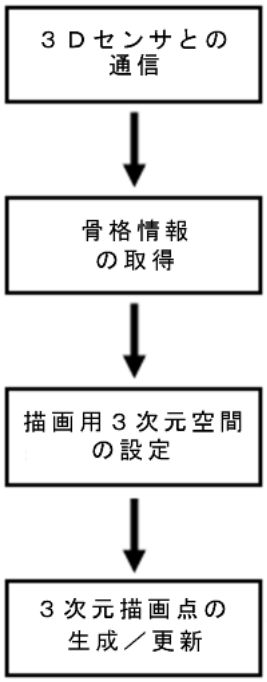

図８　３D センサ(kinect)の入力を得るためのプログラムの流れ

（１）３Ｄセンサとの通信

３Ｄセンサを扱う Kinect クラスを使って変数を生成することで、３Ｄセンサの情報を取得することができる。

（２）骨格座標の取得

３Ｄセンサの情報から得られた座標情報は Vector3D 型（３次元空間表現用の型）の変数によってプログラム上において利用が可能である。また、この点群座標を用いて、人間の形をしているもの（数十センチの幹的な物体から５方向に形状が突出している形状）は人間の骨格としてリアルタイムに認識がされるようなライブラリとして用意されており、これを利用する。

（３）描画用３次元空間の設定

描画用３次元空間を設定する。Vector3D 型による３Ｄ座標を使って、（２）で得られた人間の骨

格座標をプロットする。

（4） 3次元描画点の生成／更新

人間の骨格の動きにあわせて、(2)の情報を更新する。これにより、手先の情報を逐次的に取得していくことができる。

以上のような流れで作成したプログラム出力結果を図９に示す。

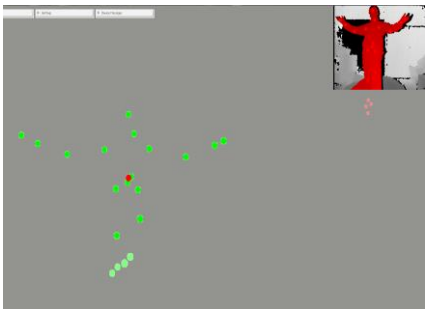

図９　AirKinect による骨格情報取得

3.1、3.2 のプログラムを合わせると作業者の動作をリアルタイムに取得することができる。

これを利用した作業者が基板検査動作をする際の手先動線キャプチャの様子と取得データを図１０と図１１に示す。

4. 考察

このように、作業者のキャプチャができるようになったが、現場に投入するにはまだいくつか課題がある。それは次のようなものである。

（1）複数人数対応

現状では複数人数対応ができていない。２人目が入ると１人目と２人目のデータを混同するような動作があり、２つのデータの峻別ができない。対策としては、画角に入ってきた作業者にＩＤを付与するなどソフトウェアにて対応できる可能性が高いので、ライブラリを詳細解析して複数人数対応もしくは、２人目を排除するなどの仕様にまとめることで対応できると考える。

図１０　キャプチャ対象の作業者動作
（基板検査工程）

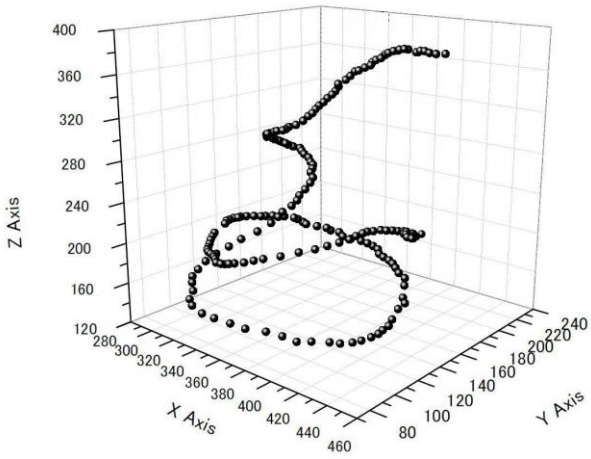

図１１　取得データ

（2）近接距離限界

３Ｄセンサのデータ取得のための近接側の限界距離が 60cm であり、少し離れたところから取得しなければならない。一定以上のスペースが必要である。これは、ハードウェアの限界であるので、導入時には設置スペースの確保のための適切なレイアウトを行う必要がある。

図１２　ジェスチャー入力と外部機器の ON/OFF 動作

（3）作業範囲の広さ

作業範囲が広い場合３Ｄセンサでとらえられる範囲を超えてしまうため、取得できない場合がある。複数の３Ｄセンサを利用した事例を参考に広範囲に対応した３Ｄセンサの活用方法の開発を進めることが望ましい。

しかし、本研究により３Ｄセンサと高速並列処理演算を用いることで得られた成果としては、図１１のような作業者の手先の動線軌跡のリアルタイムキャプチャによる動線軌跡の描画にとどまらず、図１２に示すように、パソコン画面に対するジェスチャー入力やランプなど外部機器の ON/OFF をスイッチに変わって行えることも確認できた。以上のことから作業手順の直感的表現による生産ラインの効率改善につながるツールとして利用できると考えている。

5. おわりに

演算量の多い３次元空間処理に関して、高速並列処理演算の活用や対応ライブラリを利用することで、リアルタイムに作業者動作キャプチャができることがわかった。高速並列処理演算や３Ｄセンサに関しては、現在進行形で大きな発展を見せている分野であり、より利便性の高い製品もリリースされ続けていることから、その動きを注視し、県内企業に役立つ技術として展開させ普及を図りたいと考えている。

文献

1) KINECT センサープログラミング、中村薫著、秀和システム

2) CUDA 高速 GPU プログラミング入門、小山田耕二監修、岡田賢治著、秀和システム

3) Channel 9 Coding4Fun
http://channel9.msdn.com/coding4fun