［研究報告］

視覚的ロボット動作プログラミングによる操作性の向上*

長谷川　辰雄**、南幅　留男**

電子機器・機械部品の設計現場では３次元ＣＡＤが急速に浸透し始めている。その理由は設計段階で問題点を抽出できるためと言われている。この３次元ＣＡＤを利用したロボットシミュレータは、従来の煩雑なロボット動作のティーチング（教示）作業を容易化した。しかし、３次元ＣＡＤの操作が困難であるため、特定の人しか使えないという問題が深刻化している。本報告書では、高額で操作習得に時間を要する従来のロボットシミュレータに対して、必要な最小限の機能に限定することで低価格化し、コンピュータのヒューマンインターフェイスを活用することで操作性を向上した内容について述べる。動作教示には、音声入力や力感覚による操作を可能とした特徴がある。
キーワード：視覚的，コンピュータグラフィックス，低コスト

Visible motion programming of carrier robot for human interface.

HASEGAWA Tatsuo and MINAMIHABA Tomeo

Recently, 3D-CAD is rapidly used in the design field of electronic equipment and machine part. It is known that 3D-CAD can extract the problem in the early design phase. In additon to this reason, it can be seen that the 3D-CAD facilitated the complicated design task in conventional technique. However, the problem exists that it's too expensive, and only the expert user can use it. We suggest the minimum CAD which limited function, not the multifunctional CAD which requires the time for the use. We developed the software that trajectory control is possible with computer graphics. This software has features of voice input and kinesthetic sense, network remote manipulation, and real robot drive.
keywords: visible, computer graphics, low cost

１　緒　　言

製造ラインに使われるロボットや半導体搬送ロボットの位置決め制御は，ティーチングボックス(教示装置)を用いて,実際に動作させながら位置の順序を決定する. このとき, 実ロボットは低速で動作するため, 動作軌跡を設定するには多大の時間を要する．また，位置決めの設定方法にも熟練を必要とする．メーカごとに異なる操作方法は，操作者に負担となっている．一方，ＣＧを用いたロボットの位置決め制御は，オフライン・ティーチングとして提案されているが、そのＣＧの作成に時間やコストがかかることが問題となっている. 本研究では, これらの問題を解決するために, コンピュータ・グラフィックス（ＣＧ）を用いた視覚的に分かりやすい操作方法で位置決めが直感的に可能なソフトウェアを開発した．また，音声指示による操作や, 力感覚による操作, ネットワークによる遠隔操作, ステレオ立体表示の機能により操作性を向上した．ＣＧ作成に時間を要する問題では，数値入力のみで簡易的に作成することで作成時間を短縮化する手法を提案した．これはロボットアームの幾何形状を線画で描画するために高速にＣＧを作成することが可能となる．しかし，実際の幾何形状とは異なるため干渉チェックはできないという問題点を抱えている．開発した視覚的ロボット動作プログラミング・システムはティーチングの高速化を目的としているため，それ以外の機能をできるだけ削除し，シンプルな機能構成となるように設計した.

２　実験方法

２－１ ロボットモデル

ロボットを数学モデルで記述すると，通常の３次元空間の(x，y，z)座標に対して，アームの縮尺wを考慮した(wx，wy，wz，w)で表現され，ベクトルvは式１で表される.

$$v=[x,y,z,w]^T \quad T \text{ は転置} \quad \cdots\cdots\cdots\cdots \text{式１}$$

３次元空間における座標系の位置と姿勢は, (x,y,z)方向の単位ベクトル(i,j,k)を用いて式２のように表現する．ここで，縮尺を等倍とするためにwを１とする.

$$T=\begin{bmatrix} i_x & j_x & k_x & o_x \\ i_y & j_y & k_y & o_y \\ i_z & j_z & k_z & o_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdots\cdots\cdots \text{式２}$$

ただし, $o=[o_x,o_y,o_z,1]^t$は基準となる原点を表す.

＊インターネットを利用した低コストの遠隔制御機械システムの開発（第３報）（基盤的先導的技術研究推進事業）
＊＊電子機械部

この４×4行列を用いると，例えばＺ軸を回転軸として点 $p[x,y,z,1]^T$ が角度θだけ回転して $p'[x',y',z',1]^T$ になった場合を考えるとその変換は式４で表現される．

$$p' = \begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} \cos q & -\sin q & 0 & 0 \\ \sin q & \cos q & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \quad \cdots\cdots\cdots \text{式４}$$

これをグラフィック記述言語のOpenGLで記述すると *p'=glRotatef(angle，0，0，1)* となる．

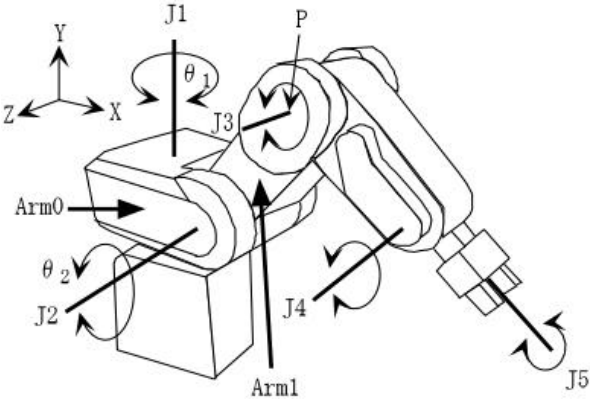

図１　ロボットの回転軸

図１はロボットの各回転軸を示す．絶対座標系（ワールド座標系）でロボットのリンク機構を実現する場合，軸J1を角度θ_1回転する行列を T_1 とし，軸J2を角度θ_2回転する行列を T_2 とすると，Arm1の先端位置pは T_1T_2 pで求めることができる．このとき，はじめに T_2 pを計算し，その結果を T_1 で変換する必要がある．この場合の注意点は，実際の動作の逆順に変換をしなければならないことである．具体的にArm1の位置ベクトル A_1 は T_2 変換→T_1 変換の順で計算を行う．これは，それぞれの変換行列が絶対座標系に対して作用することに起因する．これを関数で表すと式５の合成関数で記述することができる．Arm0とArm1の位置ベクトルを A_1, A_2 とすると

$$A_1 = f(q_1) \qquad A_2 = g(A_1, q_2)$$
$$g \circ f = g(f(q_1), q_2) \quad \cdots\cdots \text{式５}$$

このように絶対座標における座標計算は，ロボットを構成するアームの数が増えるほど先端位置の計算が複雑になり計算量が増える[1]．これに対して，座標系自体を回転することでリンク機構の回転変換を順序化したものがローカル座標系である．回転対象となるアームとその座標系を一緒に回転するため，各アームの姿勢は，実際に駆動する順番に回転行列を掛けることで求めることができる．このようにローカル座標系はシーケンシャル（動作順）に指定できるため，プログラミングが簡単化できる．しかし，各軸ごとに異なる座標系を持ち，回転するたびに座標系が変化するため，例えば，先端位置の移動距離などの絶対的な位置の把握ができない．ローカル座標はワールド座標に対する変換行列 ${}^WL$ で定義されるため，その逆行列 $({}^WL)^{-1}$ を求めることでローカル座標をワールド座標へ変換し，絶対位置を取得することができる．本開発では各回転軸の変換にローカル座標系を用い，先端位置（ハンド部分）の計算にワールド座標系を用いている．回転変換を容易にする目的で，対象となるアームをいったん原点に平行移動してから回転させ，また元の場所へ平行移動で戻す処理を行った．この処理について軸J2の回転変換を例にとって説明する．Arm1の回転軸のベクトル座標を(a,b,c)とすると，原点の平行移動行列 T_0，軸J2回りの回転行列 R_{J2}，元位置への平行移動 T_1 は式６で定義できる．

$$T_0 = \begin{bmatrix} 1 & 0 & 0 & -a \\ 0 & 1 & 0 & -b \\ 0 & 0 & 1 & -c \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad R_{J2} = \begin{bmatrix} \cos q_2 & -\sin q_2 & 0 & 0 \\ \sin q_2 & \cos q_2 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad T_1 = \begin{bmatrix} 1 & 0 & 0 & a \\ 0 & 1 & 0 & b \\ 0 & 0 & 1 & c \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$\cdots\cdots\cdots$ 式６

Arm1の位置ベクトル A_1 は $T_0R_{J2}T_1A_1$ で求まる．これをOpenGLで記述すると次のようになる．

```
glTranslatef(-a,-b,-c);
glRotatef(θ,0,0,1);
glTranslatef(a,b,c);
```

同様にして各軸の回転が記述できる．

２－２ 構造体とリンク構造

ロボットの先端位置座標を順番に記憶させるために，構造体とリストアルゴリズム[2]を用いた．図２に構造体とリスト構造に示す．構造体とは複数のデータをグループ化する手法であり，リストアルゴリズムは連続するデータの編集（挿入，削除）を効率よく行う手法である．図２の順番１は開始点の構造体を示しており，座標，回転軸，ポインタなどのデータから構成される．リストアルゴリズムは，次に続くデータの接続関係を示すポインタを定義し，そのポインタを順番に検索する手法である．本開発では，「行き」と「帰り」の検索を可能とするために，前後のポインタを定義する双方向リスト構造を用いた．このリスト構造を順番にたどっていくことでアニメーション表示を実現した．

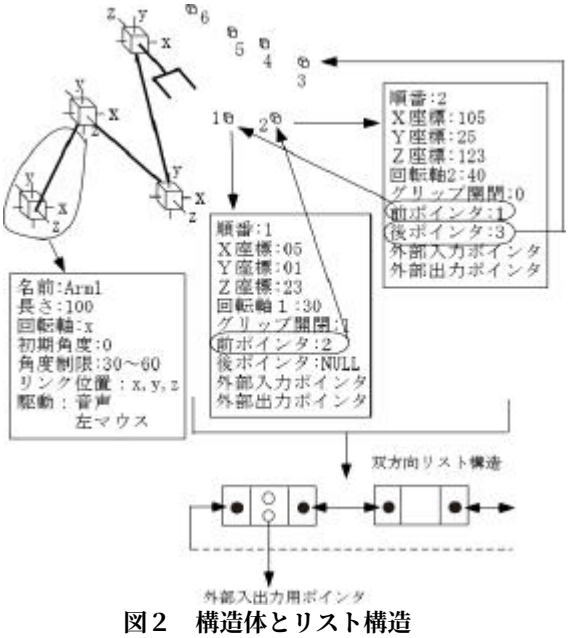

図２　構造体とリスト構造

先端位置（ハンド部分）の計算にワールド座標を利用することは前に述べたが，多関節ロボットのようにアーム数が増えると計算量が増える問題がある．そこで，本研究では，先端座標を行列の積で求めるのではなく，対象の３次元幾何データを平面へ投影する投影行列とその逆行列で３次元座標と平面座標の相互変換によって先端位置を把握している．

２－３ ＣＧプログラミング

ロボットシミュレータを作る場合，各関節ごとに動きが設定できなければならない．グラフィックを記述するOpenGL言語[3)]では，各関節の回転を行列スタックという機能で実現する．スタックとは，積み重ねたデータの最上段に対して処理を行う構造である．積み重ねる処理を「プッシュ」，データを取る処理を「ポップ」と呼び，OpenGLではglPushMatrix(void)とglPopMatrix(void)で記述する．ここで問題になるのは，行列の積で変換した後の先端座標の取得である．OpenGLでは各回転軸をマウスやキーボードで対話的に操作し，表示が終了するとそれまでの行列スタックは破棄される．変換行列スタックが無ければ，先端座標を取得することはできない．そこで，本研究では，描画に関する全ての情報を表示せずに，プログラムで取得する“フィードバック機能”を用いた．この機能によって，変換行列スタックごとに先端座標を配列へ保存することができる．この配列には先端位置を判別するための情報（トークン）を同時に保存できるため，プログラムから先端位置の検索が容易となる．この一連の操作をOpenGLで記述すると以下のようになり，gluUnProject関数でワールド座標(wx,wy,wz)を求めることができる．

```
glFeedbackBuffer(1024,GL_3D_COLOR);
glRenderMode(GL_FEEDBACK);
glRotatef(θ,0,0,1);
glRenderMode(GL_RENDER);
Find_xyz_by_Token(GL_POINT_TOKEN);
glGetDoublev(GL_PROJECTION_MATRIX, Pmat);
glGetDoublev(GL_MODELVIEW_MATRIX, Mmat);
glGetIntegerv(GL_VIEWPORT, Vport);
gluUnProject(x,y,z,Mmat,Pmat,Vport,&wx,&wy,&wz);
```

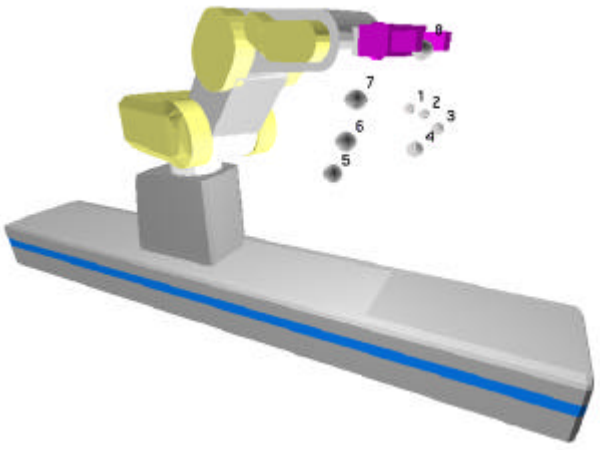

図３　視覚的動作教示

２－４　音声入力と力感覚伝達

音声入力はIBM社製のソフトウェアViaVoiceSDKで開発を行った．日本語音声入力の仕様を以下に示す．

・任意の発音定義により自由に言葉の設定が可能
・数値の認識により角度値を指　定可能
・回転軸ごとに音声指　示を行う

ここでの問題は，ViaVoiceSDKがVisualC++を用いたウィンドウアプリケーションを前提としたプログラムであり，OpenGLのウィンドウシステムと異なっている点である．OpenGLのウィンドウシステムはauxライブラリを使用しており，ViaVoiceSDKで必要とするウィンドウハンドルを取得することができない．そこで，本研究ではViaVoiceSDK用にダミーウィンドウを作成してウィンドウハンドルを取得する工夫をした．一方，力感覚伝達のシステム開発ではMicrosoft社製のForceFeedBackProというジョイスティックを使用した．DCサーボモーターが2個内蔵されており，X軸，Y軸を別々に駆動することで力を伝達する仕組みであり，１秒間に500回の頻度でモーターの力を制御している．コントロールプログラムはForceFeedback API5.0と呼ばれるライブラリを用いた．具体的にはDirectInput関数で，初期化および力感覚パラメータを構造体を使って設定した．500Hzで駆動する力伝達プログラムの作成は，実行時間を保証するリアルタイム技術必要となり非常に困難である．本開発では普及OSであるWindowsを採用したためリアルタイム処理ができなかった．OSが処理時間を自動決定するが，簡易的に力伝達を使用する場合は十分目的を達成する．本開発では，ロボットの動作範囲を超えると手に振動を与える機能として実現した．

３　実験結果

図３はＣＧによってロボットの動作を視覚的に教示した結果である．球形は先端位置を表し，教示の順番に自動採番され，ハンドの開閉も教示することができた．入力方法の実験ではマウス，キーボード，音声の組み合わせで教示を行うことができた．力感覚伝達の実験では，動作範囲を越える指示を与えたときに，手に振動を伝えることができた．教示結果を確認するために，教示順のアニメー

図４　実ロボットとロボットシミュレータ

ション表示を行うことができた. これによって, 双方向リストアルゴリズムの確証ができた．図４は本研究で使用したロボットの三菱電機社製のムーブマスターEX とロボットシミュレータの外観を示している．ムーブマスターEXはシリアル通信によるコマンドプログラミングが可能であるため，簡単な指定で実ロボットを動作させることができる．このコマンドをプログラムで参照可能なテーブルとして作成し，OpenGL で作成した動作情報をパラメータとして，実ロボットの動作コマンド列を自動生成している. 図３の視覚的動作教示をの結果をもとに, 実ロボットは教示された通りの動作を実行した.

４　考　　察

ロボットモデルは手作業で作成しなければならず，ロボットの幾何形状の作成に多大の時間を要した．また，実用化のためには周辺設備のモデル化が必要となりさらに時間を要する．この幾何形状の作成時間を短縮するために，ロボットモデルの単純化を提案した（特願 2000-23067）．図５にロボットモデルの簡易化の仕組みを示す．ロボットを構成する最小単位をアームの長さと回転軸で定義する．これを単位アームと呼び線画で描画するし，順番に接続してロボットモデルを構築する．単位アームの操作には，入力装置（マウス，キーボード，音声入力，力感覚ジョイスティック）を自由に割り当てることができる．周辺モデルも同様に立方体で簡略表現する．このとき，周辺モデルの座標はロボットの基準座標（ロボット原点）からの距離を座標値(x,y,z)与える．ロボットモデルは作業スペースに複数台設置し，連携動作を教示することができる．図６は複数ロボットの連携教示の概要図である．連携教示をシミュレーションするために，双方向リンク構造体にシーケンサ命令が記述できるポインタの追加を検討している．また，制御点に対してラダー図記述ができるように検討している.

５　結　　論

開発したロボットモデルの動作を実証することができた．この結果，ロボットモデル作成の時間短縮方法について提案できた．コンピュータグラフィックスを利用して視覚的に設計・製造を支援することは，今後，急速に進展していくと考えられる．しかし，本来解決したい問題は，高機能なＣＡＤでなくても対策可能と考える．本研究で提案する簡略化手法は，低コストでロボット教示の作業効率を向上することが可能である．また，ロボット以外にも，加工装置や計測機器などの制御・管理にも応用化が可能である.

文　　献

1) 長谷川辰雄 , 多田三郎:インターネットを利用した低コストの遠隔制御機械システム , 工技セ報告５ pp.20-21，1998
2) 近藤嘉雪:アルゴリズムとデータ構造 , ソフトバンク ,1992
3) Jackie Neider，Tom Davis，Mason Woo: OpenGL Proguramming Guide, アジソンウェスレイ ,1996

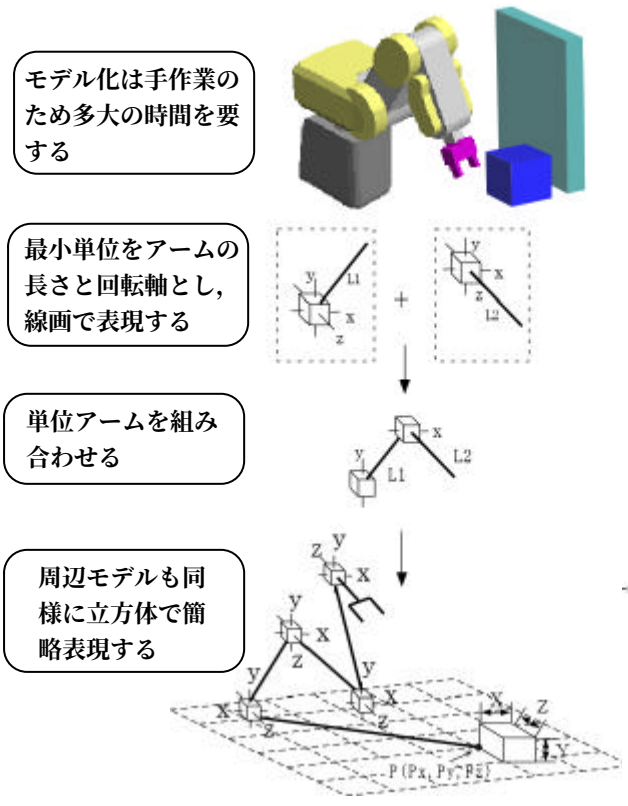

図５　モデル作成の簡略化

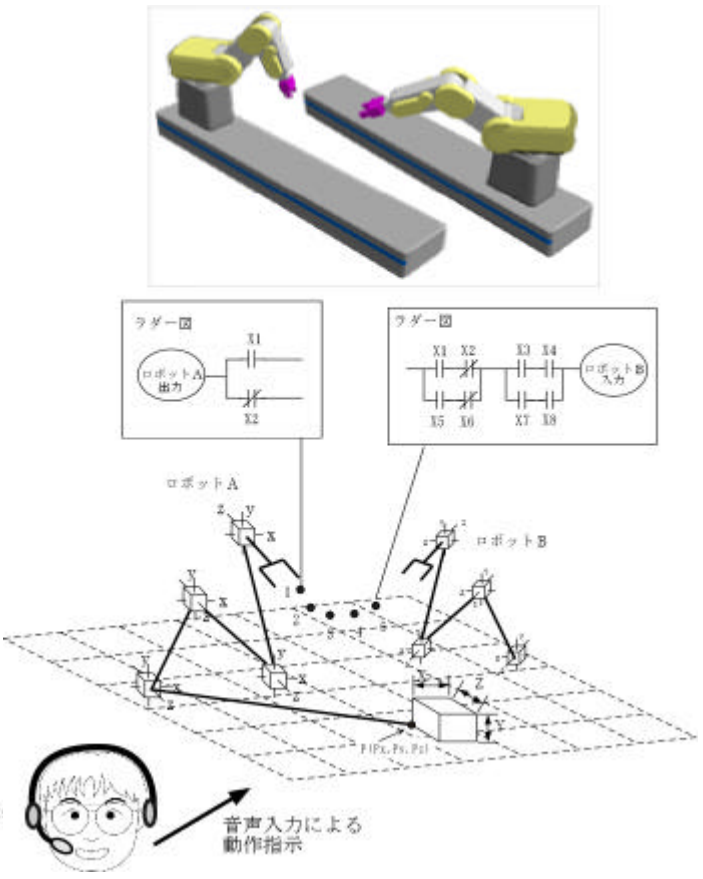

図６　複数ロボットの連携教示シミュレーション