担当教員　小澤正宜先生

提出日　令和 6 年 6 月 10 日

ロボット工学実験

マニピュレータと姿勢制御について

実施日　令和 6 年 5 月 20 日

学籍番号　120158

氏　　名　藤田崇太

共同実験者　原田　古川　牧島

1. 目的

4 年生で学習した車輪型ロボットに任意の移動をさせるための理論を実機体に適用することを通じ，ロボットの動作を理論的に決定することを学ぶ.

また，近年のロボット製作で利用されている ROS(Robot Operating System)の意義と概要を学ぶ.

2. 理論

アームのマニピュレーション(AI により要約)

この実験では、マニピュレータに任意の動作をさせることを目的としている．そのため、マニピュレータの幾何学構造に基づいて各関節の位置を計算する必要がある．順運動学では、与えられた関節角度から手先の位置を求めることができ、逆運動学では、指定した手先の位置から各関節の角度を計算することができる．

1. 順運動学（Robotics P62）

マニピュレータの各関節の角度が決まったときに，手先の位置や方向を計算する方法を順運動学（forward kinematics）と呼ぶ．Fig. 1 において，関節 2 の XY 座標（X2,Y2），関節 3 の座標（X3,Y3），マニピュレータ先端の座標（XE,YE）は以下のように表される．

$X2 = L1\cos\theta 1$

$Y2 = L1\sin\theta 1$

$X3 = X2 + L2\cos(\theta 1+\theta 2)$

$Y3 = Y2 + L2\sin(\theta 1+\theta 2)$

$XE = X3 + L3\cos(\theta 1+\theta 2+\theta 3)$

$YE = Y3 + L3\sin(\theta 1+\theta 2+\theta 3)$

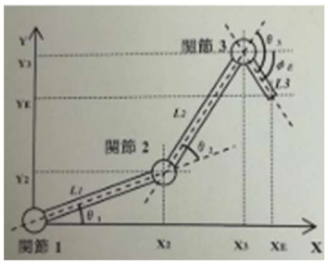

Fig 1 順運動学
(Robotics,P62 引用)

また，リンク 3 の手先の方向は，角度あるいはベクトルで表記できる．リンク 3 の手先方向の X 軸に対する角度を θE とすると，

$\theta E = \theta 1+\theta 2+\theta 3$ となる．

この幾何学的記述は理解しやすい反面，自由度の増加，3 次元空間への拡張を考えると複雑になりすぎて現実的ではないため，一般的には，座標変換行列を用いた表記をする．

2. 逆運動学（Robotics P63）

逆運動学は順運動学の逆の計算であり，手先の位置と姿勢から各関節の角度を求める方法である．

今回は逆運動学の実験に間に合わなかったため，以下に図と式にて関係を表す．

実際の慶安では，まず関節 3 の位置 X3 と Y3 を計算する．

X3＝XE－L3cos ϕ P

Y3＝YE－L3sin ϕ P

したがって，X3 と Y3 を元に θ1 と θ2 が計算できれば，θ3 も求めることが出来る．基本となるのは Fig. 2 に示した三角形の辺の長さと角度の関係である．図において，頂点 ER から底辺 L に下ろした垂直の長さを S とすると，下記の関係がある．

$l = l_1 + l_2$

$S2 = L_1^2 - l_1^2 = L_2^2 - l_2^2 = L_2^2 - (l - l_1)^2$

よって，

$L_1^2 - l_1^2 = L_2^2 - l^2 + 2l * l_1 - l_1^2$

$\therefore\ l_1 = \frac{L_1^2 - L_2^2 + l^2}{2l}$

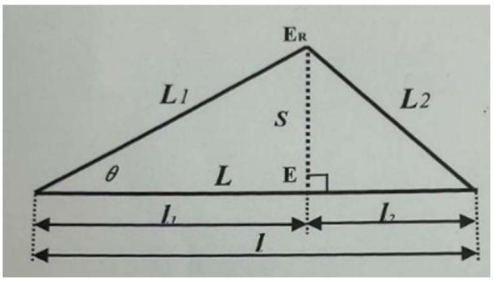

Fig 2 三角形の辺の長さを角度の関係(Robotics,P64 引用)

この式に，Fig. 10 のパラメータを当てはめて，

$$l = \sqrt{X_3^2 + Y_3^2}$$

Fig. 11 の関係を Fig. 12 に適応して，OE＝l1 より

$\theta_1 = \angle WOX - \theta_1'$

$$= \tan^{-1}\left(\frac{Y_3}{X_3}\right) - \theta_1$$

$$= \tan^{-1}\left(\frac{Y_3}{X_3}\right) - cos^{-1}\left(\frac{L_1^2 - L_2^2 + (X_3^2 + Y_3^2)}{2L_1 * \sqrt{X_3^2 + Y_3^2}}\right)$$

と求めることができる．同様に

$\theta_2 = \theta_1' + \theta_2'$

$$= cos^{-1}\left(\frac{L_1^2 - L_2^2 + (X_3^2 + Y_3^2)}{2L_1 * \sqrt{X_3^2 + Y_3^2}}\right) + cos^{-1}\left(\frac{L_2^2 - L_1^2 + (X_3^2 + Y_3^2)}{2L_2 * \sqrt{X_3^2 + Y_3^2}}\right)$$

と求められる．

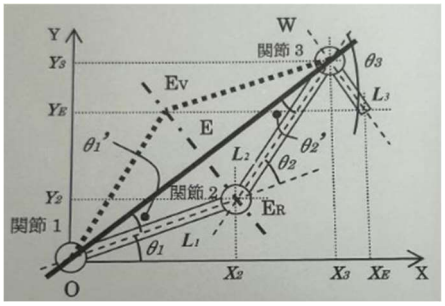

Fig 3 逆運動学

(Robotics,P63 引用)

また，

$$\theta_3 = \varphi_P - \theta_1 - \theta_2$$

が得られる．

3. PTP（Point to Point）制御（Robotics P86）

PTP 制御とは，飛び石を連続して飛んでいくようなイメージで，描くべき軌道の重要な点をいくつか指定し，その間を移動するように制御する方法である．マニピュレータのエンドエフェクタの場合，エンドエフェクタの位置をいくつか指定することが考えられる．一番単純な軌道は初期地点と最終地点を直線で 結ぶ軌道であるが，障害物を回避する必要がある場合には Fig. 4（a）のよう に，いくつかの点を結ぶことで障害物を回避していく．

しかし，エンドエフェクタの位置だけを指定するのでは，完全には障害物を回避することはできない．例えば，Fig. 4（b）のように，マニピュレータの「肘」が 障害物に衝突してしまう可能性もある．

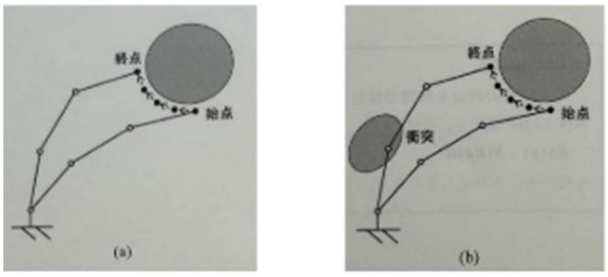

Fig. 4 PTP 制御

3. 実験機材

本実験では，waffle-pi に装備されている Open MANIPULATOR-X を使用する．このマニピュレータは 4 自由度の関節と 1 自由度のグリッパを有している．マニピュレータの各部寸法を Fig. 5 に示す.

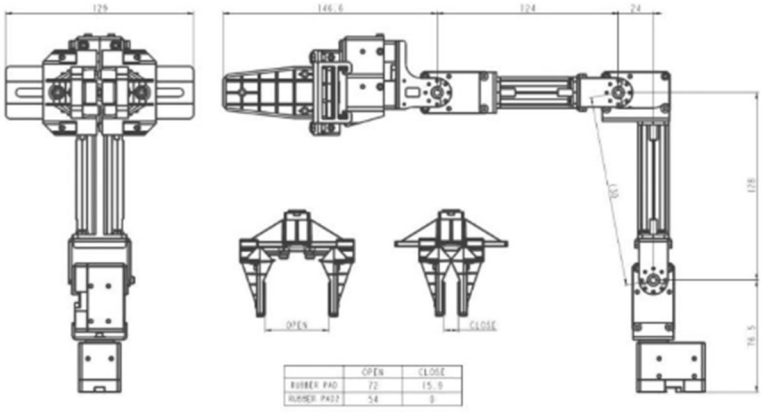

Fig. 5 Open MANIPULATOR-X 関節部寸法

4. 実験手順

実験は以下の手順で行う.

1. 機材のリセット(turtlebot と操作用 PC の時刻同期)

1. waffle-pi の raspberry pi に LAN ケーブルを接続する
2. waffle-pi の電源を投入する
3. 操作用 PC で端末を立ち上げ(端末 1 とする), SSH(Secure Shell)接続で waffle-pi に接続する
   アカウント, パスワードは機体に記載されている
   「ssh {アカウント名}@{機体の IP アドレス}」を実行すること.
   例： ssh pi@192.168.0.100
4. 別の端末を立ち上げる(端末 2 とする)
5. 端末 1, 2 それぞれで以下を実行し, 時刻を整合させる
   ntpdate ntp.nict.jp
6. 端末 1, 2 それぞれで以下を実行し, 時刻が整合したことを確認する
   ntpdate
7. waffle-pi は sudo reboot を実行, 操作用 PC は GUI 上から再起動する

2. 実験準備

1. waffle-pi の電源を投入する
2. 操作用 PC で端末を立ち上げ(端末 1 とする), roscore を実行する
3. 別の端末を立ち上げ(端末 2 とする), SSH(Secure Shell)接続で waffle-pi に接続する
4. waffle-pi に接続した端末 2 で以下を実行する
   「roslaunch turtlebot3_bringup turtlebot3_robot.launch」
5. さらに別の端末を起動し(端末 3 とする), マニピ動作用ノードを実行する.
   「roslaunch turtlebot3_manipulation_bringup turtlebot3_manipulation_bringup.launch」
6. さらに別の端末を起動し(端末 4 とする), マニピ姿勢計算用ノードを実行する.
7. 「roslaunch turtlebot3_manipulation_moveit_config move_group.launch」

8. さらに別の端末を起動し(端末 5 とする)，マニピ動作指示用ノードを実行する(Fig. 6)

Fig. 6 マニピュレータ動作指示画面

「roslaunch turtlebot3_manipulation_gui turtlebot3_manipulation_gui.launch」

9. ペンホルダーを使ってペンを把持させる. このときペンの先端位置を計測しておくこと.
10. 配布された用紙を，テープを使用してボードに取り付ける.

3. 直交座標指定による図形の描画

「Task space」タブを有効にし，座標指示で次の図形を描画せよ

1.縦線→パラメータを Fig. 7 に示す. 描画した図形を Fig. 8 に示す.

2.横軸→描画した図形を Fig. 8 に示す

4. 角度指定による図形の描画

「Joint space」タブを有効にし，関節角度指示で次の図形を描画せよ

1.縦線→描画した図形を Fig. 8 に示す

2.横軸→時間が足らずできなかった

5. 逆運動学による図形の描画

逆運動学で任意の手先位置を実現する各関節の角度を算出し，次の図形を描画せよ

1.縦線→時間が足りずできなかった

2.三角形→時間が足りずできなかった

6. Additional Work

手先位置指示と角度指示を組み合わせ，本マニピュレータにおける特異姿勢と動かすことができない方向を探せ

→時間が足りずできなかった.

(AI)

手順

1.ヤコビアン行列の導出

- ヤコビアン行列（J）は，手先の速度と関節速度の関係を示す．具体的には，手先の位置や角度の変化を関節の角度の変化として表現する.

2.特異点の検出

ヤコビアン行列 ¥(J¥) の行列式がゼロになるポイントを見つける．行列式がゼロのとき，その姿勢が特異点であり，特定の方向に動かせない状態になる.

具体例

3 自由度ロボットアーム

ヤコビアン行列（J）の例

1. 手先の位置（x, y, z）と関節角度（θ_1，θ_2，θ_3）の関係を定義.
2. ヤコビアン行列を計算し，各関節角度における行列式を求める.

特異点の検出方法

・行列式がゼロになる関節角度を見つける．この角度でロボットは特定の方向に動かせない.

結果の解釈

・特異点: ヤコビアン行列の行列式がゼロになる姿勢.

・動かせない方向: 特異点でのロボットの自由度が失われ，動かせなくなる方向.

以上の手順を使って，マニピュレータの特異姿勢と動かせない方向を特定できる.

3. 実験結果および考察

結果：3.1 の縦線は Fig. 7 に示すパラメータまで微調整を繰り返した.

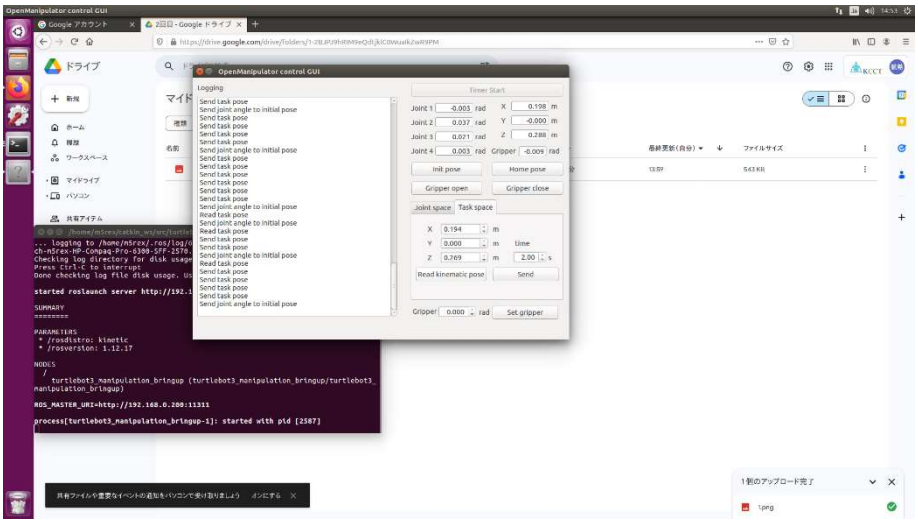

Fig. 7 パラメータ調整結果(縦線)

横線も Fig. 16 のようにパラメータを調整して描画した. 3.1 から 4.1 までの描画した結果を Fig. 17 に示す.

Fig. 8 描画結果

3.1, 3.2 の図形が途中で点ができているのは，動く過程で紙を強く押さえつけてしまうためだと考える．これは座標指示の制御内容による問題で，特定の時間内でエンドエフェクタの座標のみを移動させるため，各軸のモータは順序関係なく動いてしまう．そのため，座標指示のみできれいな図形を描くには常に紙とペン先の距離が一定に保てるように非常に短い座標間で点を取り続ける必要があると考える．

また，4.1 の図形が 3.1，3.2 の図形と比べて綺麗に描けているのは角度指示のプログラムが要因であると考えられる．角度指示のプログラムは座標指示と比べて各軸が毎秒何度動くかを制御できる．これにより図形の描画が綺麗であると考える．

5 について，今回使用するマニピュレータで逆運動学を求めるのは非常に困難であった．なぜなら Fig. 5 に示すように，モータ軸から節が伸びておらず少し折れ曲がった位置から次のモータ軸が伸びているような部分が多かったためである．実験中に行った逆行列の途中までの計算結果を Fig. 9 に示す．

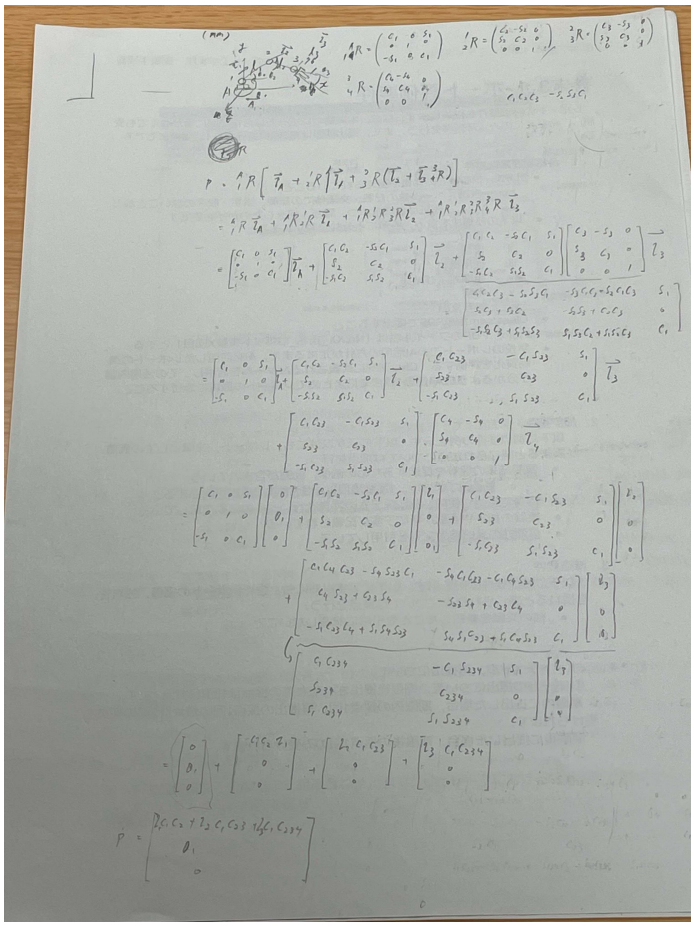

Fig. 9 逆運動学(計算途中)

この計算は前述したモータ軸が中途半端な部分でつながっていることを一度考慮せずに行っている．理由としてはこの計算が完璧に現実で起こると考えられなかったため，実際の近似値としてだすための計算とする方がよいと考えたからだ．

参考文献

1. 日本機械学会, “Robotics”, 丸善出版株式会社, (2011) , P62~64,P86　2024 年 6 月 9 日(日)閲覧
2. 2 週目 配付資料「マニピュレータの運動学・逆運動学」 2024 年 6 月 9 日(日)閲覧
3. M5R_No.33_森山拓海_ロボット実験 2 回目(レポート)　　2024 年 6 月 9 日(日)閲覧

[AI による要約]

理論の「アームのマニピュレーション」を AI により要約した.