ROS を用いたロボットアームによる物体の操りに関する研究

M2019SC013 竹下薫
指導教員：中島明

1 はじめに

1.1 ロボットアーム

ロボットアームとは，その名の通り人の腕のような形をした ロボットである．現在，ロボットアームは工場などにおいて，単純な反復作業や人間が行うには危険な作業を代わりに行うことができるロボットである．ロボットアームを用いて，物体を自由に操ることが可能になれば，工場などの現場だけでなく，我々の生活の中でも活躍できると考えられる．物体の操りは様々な種類があり，物体を完全に拘束する場合や部分的に拘束する場合，物体の投擲やキャッチすることなど多岐にわたり存在する．そのような物体の操りに関して，段階を踏んで操り制御システムの構築を目標としている．

1.2 研究計画

はじめに，カメラを用いて物体がどこにあるかを認識し，ロボットアームを用いて物体を追従，把持を行い，目標位置に運ぶ動作を作成する．

次に，ロボットアームを用いて物体を投擲や空中でキャッチするなど，高度なマニピュレーションのシミュレーションを作成する．

次に，シミュレーションを行ったものを実機に反映し，障害物を実際に避けることが可能であるか検証する．また，シミュレーションと実機でどの程度の誤差があるのかを検証する．

次に，モーションキャプチャを用いることで，より人間に近い動作を追求し，投擲動作など様々なマニピュレーションの作成に取り組む．

最後に，実機を用いて物体の把持をはじめ，様々な実験を行うことを考えている．また，可能であれば，周りの実験環境に応じて適切な動作を行う制御を可能にしたいと考えている．

また，本研究はロボットソフトウェアプラットフォーム ROS を用いており，3 次元動力学シミュレータである Gazebo を用いた．Gazebo はシミュレーション上に障害物やテーブルを作成することができ，より現実的な条件下で物体の把持や物体の操り動作のシミュレーションを作成することが可能である．

2 ROS

2.1 概要

著書 [1] に基づき，ROS について解説する．ROS とは，「ROS はオペレーティングシステム (OS) から通常，提供されるサービス，例えばハードウェアの抽象化，低レベルのデバイス制御，共通して利用される機能の実装，プロセス間のメッセージ交換，パッケージ管理に加えて，複数のコンピュータ間におけるコードの取得，ビルド，記述，実行のためのツールやライブラリを提供する」とされている．言うなれば，ROS は研究者が取り組みたい内容に適した開発環境を提供しており，ロボットアプリケーションプログラムを効率よく開発できるものである．

2.2 ノード

ROS にはノードと呼ばれる ROS 内で実行される最小のプロセスが存在する．ROS では目的ごとにそれぞれノードが作成されており，センサドライバや障害物検出などの目的ごとに細分化してノードが作成される．

2.3 パッケージ

パッケージとは，プログラムの集合であり，ROS ソフトウェアの基本単位である．ROS のアプリケーションプログラムはパッケージ単位で開発されており，その中には少なくとも 1 つはノードが含まれている，または他パッケージのノードで使用される設定ファイルが含まれている．現在使用している ROS Kinetic では 1600 個以上の公式 ROS パッケージが公開されている．

2.4 ROS wiki

ROS についての基本的な説明や ROS が提供するパッケージ，機能の情報は ROS wiki[4] が提供している．そのほか，パッケージで使用されるパラメータ，著作者，ライセンス，チュートリアルなどが記載されている．

2.5 リポジトリ

公開パッケージの場合，各パッケージの Wiki にはパッケージのリポジトリが明記される．各リポジトリには，URL が割り当てられ，パッケージのソースコードが管理されている．svn や git などのソースコードマネジメントシステムを用いると，パッケージのダウンロードやアップロードを利用することが出来る．現在公開されている ROS パッケージのほとんどは GitHub によって管理されている．各パッケージに関する質問などはそのパッケージを管理しているリポジトリの問い合わせ窓口から問い合わせることができる．

また，ROS を用いることの利点として，システムや制御などのユーザーが興味のある分野の ROS ソフトウェアが公開されている点や豊富な開発ツールが存在する点である．近年，卓上ロボットアームは 3D プリンタの登場により，安価で制作できるものとなり，近い将来，我々の生活の中にも存在するだろう．その際，ロボットモデルのサンプ

ルや ROS のソフトウェアを組み合わせることで，簡単に汎用性の高いロボットを自作することができるのも ROS の利点である．また，ROS はセンサモデルのライブラリが豊富であるため，卓上ロボットを用いて日常生活のサポートや人間とロボットが協働して作業を行うのに適していると言えるだろう．

また，ROS は異機種デバイス間の通信をサポートしている．利用する OS の種類，プログラミング言語によらず，ROS のインストールや動作を確認することがで ROS のノード間の通信は容易に実現可能である．これは，Ubuntu が搭載された PC で制御されるロボットを他の OS の PC から確認やロボットに命令を送ることが可能であることを示している．(図 1 参照)

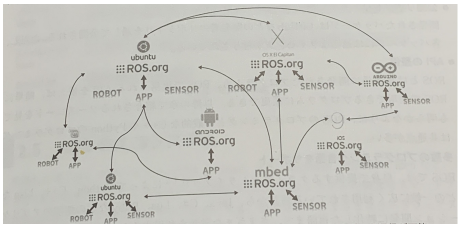

図 1　異なるハードウェア，OS 間で通信可能な一例

2.6　可視化ツール

ROS には，豊富な種類の可視化ツールが存在する．本研究で用いた可視化ツールの 1 つは，Gazebo である．Gazebo は ROS によって動作するロボットのための 3D モデルのバーチャルシミュレータである．Gazebo は ROS と親和性が高く，ROS のロボットモデル表現である URDF に Gazebo 用の要素やプラグインの情報を追記するだけで，Gazebo 上でロボットが使用可能となる．Gazebo に標準で用意されている機能やプラグインで CRANE-X7 の大半の動作がシミュレート可能となる．また，Gazebo では，標準の環境モデル，障害物の設置や重力の有無など様々なシミュレーション条件を設定することができるうえ，図 2 のように机や物体などを作成することでより現実的なシミュレーションを行うことが可能である．

2 つ目は，rviz（図 3 参照）である．rviz は，ROS アプリケーションの 3D 視覚化ツールであり，ロボットモデルのビューを提供し，ロボットセンサからのセンサ情報をキャプチャして、キャプチャしたデータを再生できる．カメラ，レーザー，3D/2D デバイスからの写真やポイントクラウドを含むデータを表示できる点が Gazebo との違いである．

最後に，rqt である．rqt は，Qt をベースとした ROS 専用 GUI 開発フレームワークとツールである．状態表示などの様々な機能がプラグインとして提供されている．rqt を用いることで簡易的に動作しているノードやメッセージの流れを確認することが出来る．

2.7　運動学について

ROS 内での運動学計算は MoveIt!を用いることで計算を行った．MoveIt!は，マニピュレータの動作制御のための統合ライブラリである．モーションプランニングのための高速な順運動学計算，マニピュレーションのための高度なアルゴリズム，ロボットアームの制御，動力学など多様な機能を提供しているマニピュレータ用のフレームワークである．

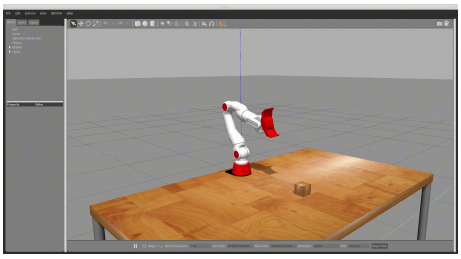

図 2　Gazebo

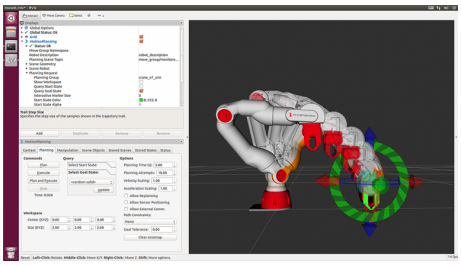

図 3　rviz

3　ロボットアームの制御法

著書 [1] に基づいて，ロボットアームの制御法について記述する．ロボットアームの制御法は，関節空間での制御と作業空間での制御に分けることが出来る．

3.1　関節空間での制御について

関節空間での制御は，各関節の回転角度を指令し，制御する．各関節の回転角度に応じたロボットアームの先端の座標が順運動学計算により求められる．目標の手先位置，姿勢が与えられた際，その与えられた指令を実現するための関節変異を算出するには逆運動学計算を用いる．

3.2　作業空間での制御について

作業空間での制御はロボットアーム先端の座標を指令することで，各関節の回転角度を制御する．作業空間上でのロボットアームの手先の状態は位置 (Position) と姿勢 (Orientation) で表現される．3 次元空間に置かれた物体の位置，姿勢は 6 つの未知数を決定する必要があり，6 つの方程式があれば解を一意に求めることが出来る．これは，6 つの関節を持つロボットアームであれば，作業空間上のどのような位置，姿勢で物体が置かれていても，ロボット

アームが届く距離であれば物体を把持することが可能であることを示している．ロボットアームの目標の先端座標を実現するための各関節の回転角度は，逆運動学で求めることが出来る．また，ロボットが伸びきった姿勢や、2 つ以上の軸が一直線上に並んだ時場合の姿勢では、ロボットの先端を動かすことができない方向があり，逆行列計算の解が存在せず，そのような状態を特異点という．

3.3　運動学的冗長性

[2] に基づき，冗長について解説する．作業空間の次数よりもロボットアームが持つ運動の自由度を持っている場合，ロボットアームは運動学的冗長性を持つという．運動学的冗長性を持つということはロボットアーム先端の位置，姿勢を変化させることなく，ロボットアーム全体の姿勢変化が可能となる．

3.4　クォータニオン（四元数）

著書 [3] に基づいて，クォータニオンについて記述する．クォータニオンとは，複素数を拡張したようなもので，姿勢表現の際に扱われる．クォータニオンによる姿勢表現はロール，ピッチ，ヨー角，ZYZ-オイラー角による姿勢表現とは異なり，表現上の特異点が存在せず，姿勢間の補完が容易といった利点が存在する．

4　実験機器説明

本研究で用いた機器について説明する．

はじめに，本研究で使用したカメラは，物体のリアルタイムの座標データを取得するために Optitrack 社製のモーションキャプチャカメラ「FLEX13」(図 4) を 3 台使用した．FLEX13 は，高解像度イメージセンサーで様々なシーンに対応することができ，素早い動作もキャプチャ可能なモーションキャプチャカメラである．物体の位置を認識するためのモーションキャプチャのマーカーとして図 5 のマーカーを用いた．

つぎに，本研究で使用したロボットアームは，アールティ社製のロボットアーム CRANE-X7(図 6) を用いた．CRANE-X7 は，7 軸構成で自由度の高さを活かした障害物の回避計画や、人の腕のように柔軟な動作を行うことが可能なロボットアームである．手先の作業空間の次数は 3.2 節で解説したように，最大 6 であるため，7 軸構成である CRANE-X7 は運動学的冗長性を持っているロボットアームである．自由度が多くなるにつれて特異点は増えるが，このロボットアームは自由度が 7 あるため，特異点となるような姿勢を避ける軌道生成が期待される．

図 4　FLEX13

図 5　モーションキャプチャで用いるマーカー

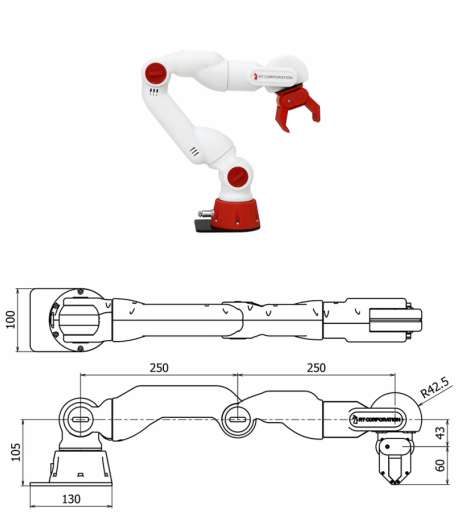

図 6　CRANE-X7

5　実装

5.1　ROS とモーションキャプチャの連携について

本研究では，VRPN ライブラリ用のクライアントノードとして「vrpn client ros」を用いた．このパッケージは，モーションキャプチャソフトウェアである motive によって取得した座標データを ROS 側の PC で受信するために利用される．このパッケージを利用することで，図 7 のようにリアルタイムで物体の位置と姿勢を取得し続けることが出来る．

また，カメラの配置及びデータの取得構成図は図 8 の通りである．

5.2　実験環境説明

本研究の実験環境を説明する．Optitrack 社のモーションキャプチャカメラ 3 台，モーションキャプチャソフトウェア Motive を実行する PC が 1 台，および ROS を動作させる PC が 1 台，ロボットアーム CRANE-X7 で構成される．本研究では ROS を動作させるプラットフォームとして，ROS の公式サポートしている Ubuntu を用いた．

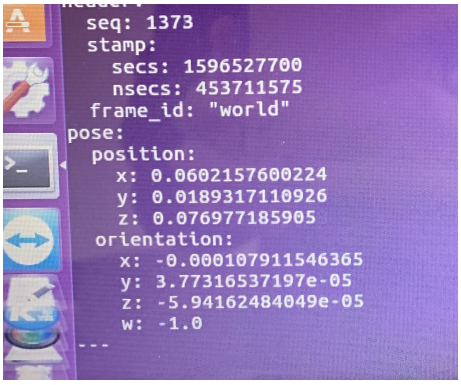

図 7　motive から受信したデータ

本研究で使用した実験環境について表 1 の通りである．

5.3　実験目的

本研究では，平面上で物体が移動してしまった際にもロボットアームの手先を物体に追従させ続けロボットアームの届く範囲内であれば，物体の姿勢に応じて適切な角度で把持することを目的とした．また，図 9 のように一定のキーを押すまでループし続けることで，物体を追従し続けるプログラムを作成した．モーションキャプチャを用いることで，リアルタイム上での物体の位置，姿勢を取得し続けることを実現した．また，図 10，11 のように物体が同じ位置にあった場合でも姿勢が異なる場合，把持する角度は異なる．

表 1　実験環境一覧

ROS を動作させる PC	Ubuntu 16.04
ROS	Kinetic
Gazebo	7.0.0
Motive を動作させる PC	Windows 8
Motive	2.0.1

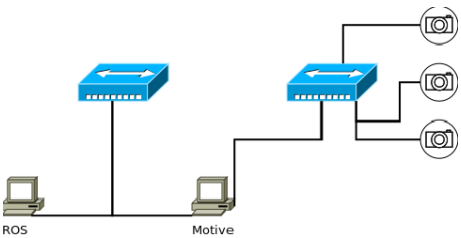

図 8　実験機器の構成図

6　おわりに

本研究では，モーションキャプチャを用いることで，リアルタイム上の物体の位置，姿勢を受信し続け，ロボットアームの手先を物体に追従，把持を行った．今後の課題としては，以下の 2 点である．

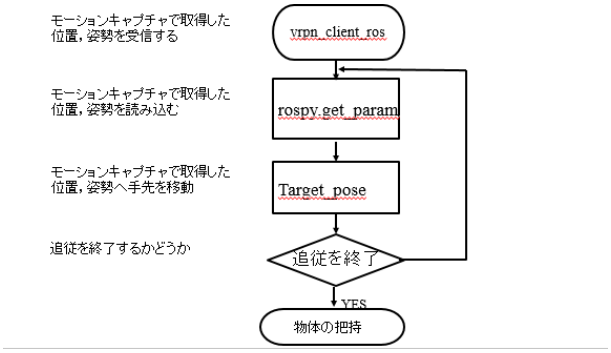

図 9　追従のアルゴリズム

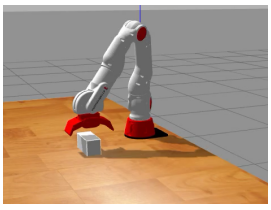

図 10　yaw ＝ 0.0

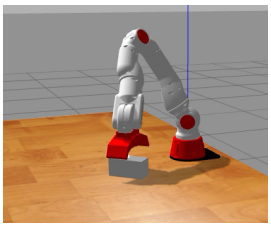

図 11　yaw =0.8

1. ロボットアームとモーションキャプチャを用いた操り動作は把持だけでなく，ジャグリングやなど様々なものがあるため，他の操り動作についての制御手法の提案や物体の形状や大きさに応じて把持する力を変える手法等が課題である．
2. 本論文ではモーションキャプチャを用いて物体の追従，把持を行うだけにとどまったため，CRANE-X7 の特徴である運動時に冗長自由度を持つ利点を生かすことが出来なかった．そのため，今後は障害物があった際に把持が行えるかどうかの提案が課題である．

参考文献

[1] 倉爪亮，表允晢，ROS ロボットプログラミングバイブル，オーム社，2018.

[2] 杉本貴大，「運動学的冗長性を有するマニピュレータの特性解析と冗長自由度を最大限活用する運動制御」，2011

[3] 細田耕，「実践ロボット制御」，オーム社，2019.

[4] http://wiki.ros.org/ja