James Dyson Foundation Undergraduate Bursary 2020/2021

Project report
Conformal geometric algebra for robot kinematics

Zach Lambert

1 Overview of robot manipulators

1.1 Forward and inverse kinematics

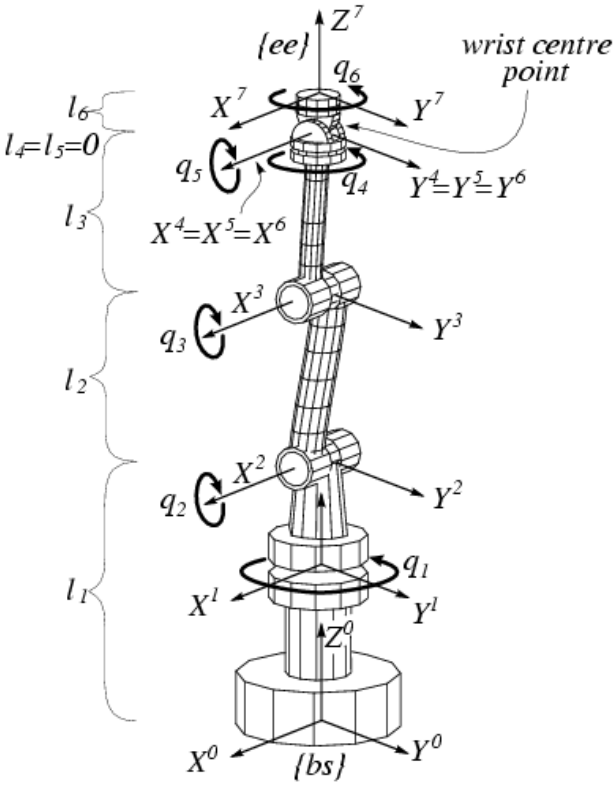
A robot manipulator is a robot capable of moving an **end-effector** in 3D space. This end-effector might be a gripper or a tool.

Such a manipulator consists of a number of **links** connected by **joints**. The joints connect links, such that by setting the joint positions (the angles of the joints), you configure the state of the robot.

With this you can control the position and orientation of the end-effector, collectively called it's **pose**.

With robot manipulators, we have two problems we want to solve:

- **Forward kinematics:** Given a set of joint positions, what is the pose of the end-effector.
- **Inverse kinematics:** Given a *desired* end-effector pose, what are the *required* joint positions.



Solving the forward kinematics is required for visualisation of the robot state, which aids control.

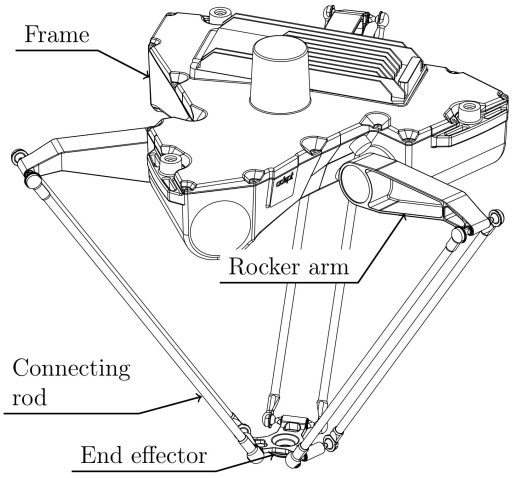
Solving the inverse kinematics is required for controlling the end effector to move to where we want.

1.2 Types of robot manipulator

The robot shown above is an example of a **serial robot**, where the links form a serial chain.

The figure on the right shows a **delta robot**. This is classified as a **parallel robot**, as opposed to a serial robot. This is because it has three different serial chains which meet at the end effector.

Depending on the type of robot, the solution for the forward and inverse kinematics is very different.



1.3 Solving the kinematics

Solving the kinematics of the serial robot requires linear algebra. The forward kinematics is straightforward, but the inverse kinematics requires using numerical methods.

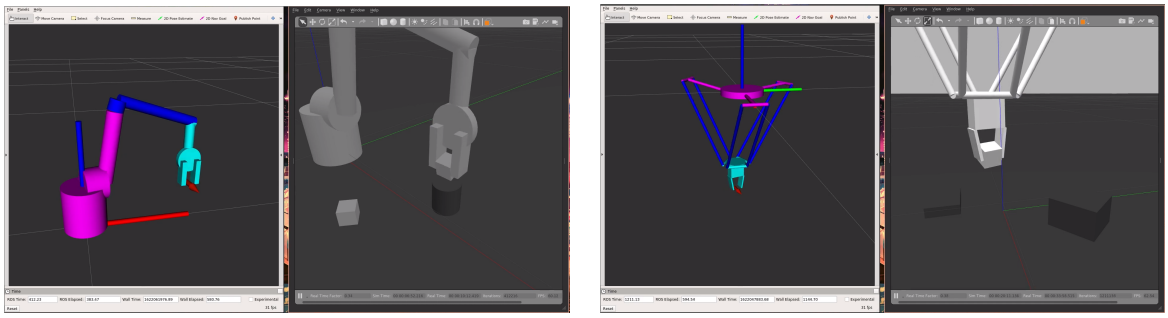
Solving the kinematics of the delta robot required solving intersection problems. Both the forward and inverse kinematics have algebraic solutions.

In both cases, a type of maths called **conformal geometric algebra** was *also* used to solve the problems, and shown to be a viable alternative method to conventional methods.

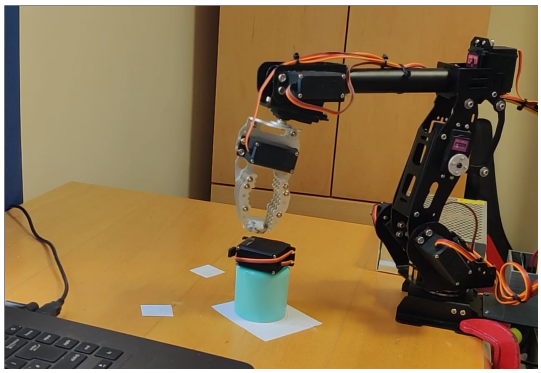
2 Controlling simulated and real robots

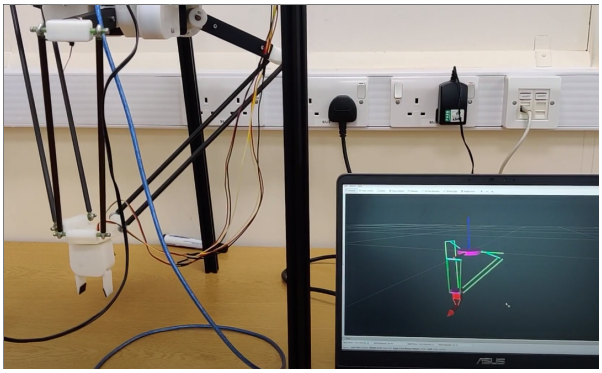
These forward and inverse kinematics solutions were used in the control of simulated and real robots. Software was written using the **C++** programming languages and used the **robotics operating system (ROS)** for organising the software.

The **Gazebo** robotics simulator was used for simulation, and two robot kits used for controlling real robots.



Above shows screenshots of the two robots being simulated. Within each screenshot, a visualisation of the robot state is shown on the left, and the simulation environment shown on the right.





Above shows photos of two robot kits being controlled. Again, visualisations of the robot state was visible, as shown in the delta robot photo.

For both the simulated and real robots, each robot could be manually controlled using an Xbox controller.

While manually controlling, the user could record a set of waypoints which the robot could then replay motion between. This would allow it to be setup to perform tasks, such as picking up objects and placing them somewhere else.

3 Project conclusions and future work

The main objective of the project was to demonstrate that conformal geometric algebra could be used for robot kinematics and applied to the control of real robots, which the project was able to do.

The motivation behind using conformal geometric algebra over conventional methods is that it provides a consistent and simpler method of solving geometric problems, so could be well suited to robotics. The drawbacks come from the decreased computational efficiency.

Future work could look at different types of robot, or investigate some more complex control methods, such as incorporating computer vision (which can also be solved with conformal geometric algebra).