ACME Robotics Path Planner for Manipulators

Chang-Hong Chen (117397857) , Sparsh Jaiswal (117433968)

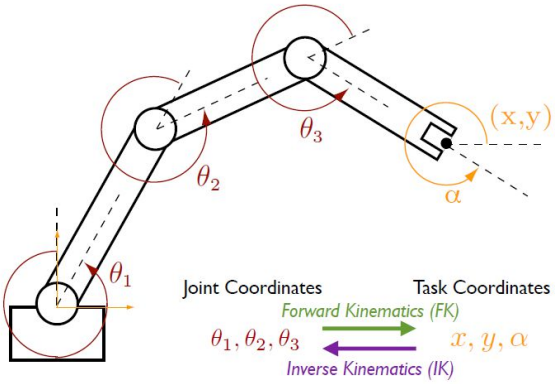
Abstract - Path Planers are a key component in the operation of manipulators for tasks such as material handling, machine tending etc. With the improvements of sensing technologies and computer vision, manipulators are becoming smarter and require dynamic path planning as opposed to doing fixed/pre-programmed repetitive tasks that they have been doing so far. This component aims to do just that. Enable manipulators to plan its path on the fly based on inputs from a vision system and a target location.

1. Introduction

As we move forward, collaborative robots are gaining popularity. For robots to be work in collaboration with humans, they need to be able to act/react with the sensor inputs and plan /re-plan its trajectories and speed of operation autonomously rather than just stop if an obstacle is detected or operate at a constant speed otherwise.To enable this autonomous planning on the fly, we propose a path planning module that generalizes the path planning algorithm for manipulators of various degrees of freedoms. Since we aim to cater to different types of manipulators an object oriented approach will be taken for the implementation.

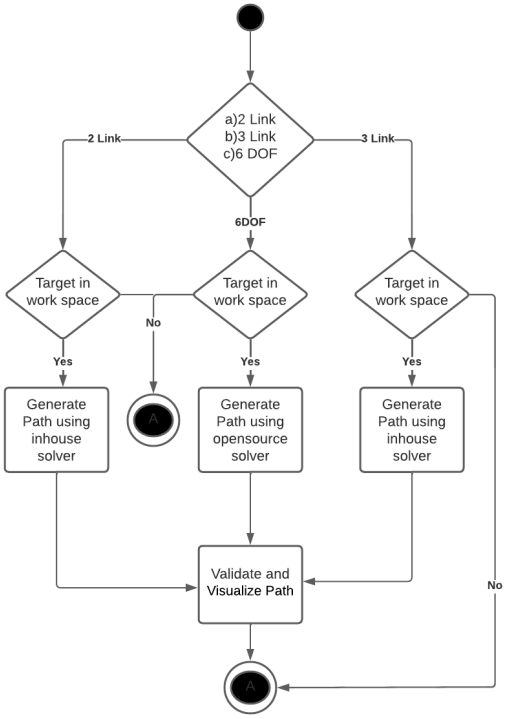
2. System Implementation

The module will be developed in C++ and AIP will be incorporated for implementation. AIP allows us to maintain the quality of the work and keep a check on the resources being used for development by tracking the progress and changes closely. The functionality of the component is to take in the target position of the end effector and return an optimal planned path



Algorithm

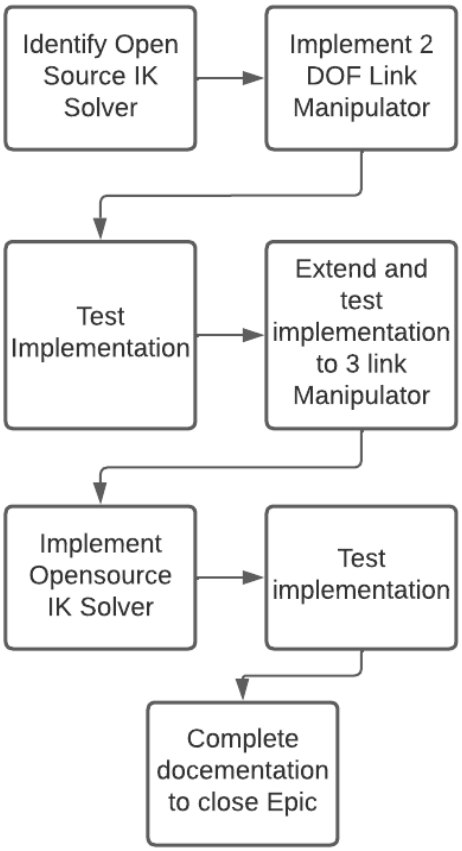
A sampling based graph search algorithm will be implemented such as A*, Dijkstra, RRT or RRT*. At first the solver will be developed in house for link manipulators following that an external Inverse kinematic solver will be used for planning for higher degree of freedom manipulators .



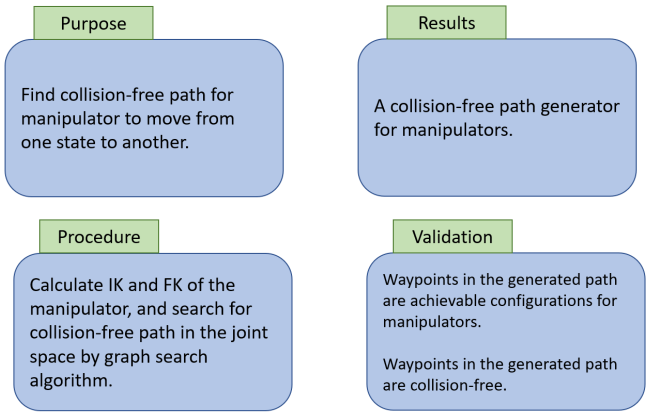
The main risk of using an open source library for the IK solver can be the compatibility of the solver with our implementation. To mitigate such a risk, before starting the development 2 solvers will be selected and development proceed in such a manner that pivoting from one solver to the second selected solver can be done with minimal code changes. The identified open source solvers and libraries use MIT licenses.

3. Team

The realization of this software will be done by a team of 2. The team will take a test driven approach where in the first sprint one member will play the role of the driver in the first sprint and in the second sprint that member will be the navigator. Each sprint will be a week long. The preliminary flow of task for development are as follows:



4. Quad Chart



5. References

[1]https://github.com/TheComet/ik
[2] Software Engineering - The Current Practice
[3]https://disigns.wordpress.com/portfolio/solving-inverse-kinematics/
[4]https://moveit.ros.org/documentation/concepts/

6. Video

https://youtu.be/pWCyieuHHsQ