International Journal of Advanced Mechanical Engineering.
ISSN 2250-3234 Volume 4, Number 2 (2014), pp. 145-150

http://www.ripublication.com/ijame.htm

Analysis of Puma-560 and Legged Robotic Configurations Using MATLAB

Manish Paliwal* and Yatheshth Anand**

**Student, School of Mechanical Engineering, SMVDU, Katra, Jammu, J&K.*
***School of Mechanical Engineering, SMVDU, Katra, Jammu, J&K.*

Abstract

This paper will deal with the precise modelling and analysis of an industrial robot (Puma560) and a legged robot. These two configurations are analyzed because the PUMA 560 is the widely used robot in the industry and legged motion is far superior to the wheeled locomotion. The modelling and analyze is done by using a very powerful simulator called the MATLAB (Robotic Toolbox). The programme analyses the time and motion sequences, the dynamics and control (with and without motion) for the given robotic configurations. Framework for dynamic modelling is proposed with the concept of kinematic modules (forward and inverse), where each module is taken as a set of serially connected links. Translation of robot's base and end effecter and the modelling of robots arm in different orientations are also considered. The results are graphical represented and analyzed as the plots between the different robotic parameters like the link's joint angle, time and the motion of the end effecter i.e. the roll, pitch, and yaw, for various poses (positions/orientations).

Keywords: Modeling and analysis; Puma 560; Legged Robot; MATLAB (Robotic Toolbox); Time and Motion Sequence; Dynamics; Kinematics; Orientation.

1. Introduction

Nowadays robots accompany people in everyday life and have taken over some of their daily procedures. A large family of robot manufacturing equipment exists, supplying the motion required in manufacturing processes such as arc welding, spray painting, assembly, pick and place, cutting, polishing, milling, drilling, etc. A

manipulation task is usually given in terms of a desired end-effecter trajectory. Since the manipulator is controlled by joint servos, a mapping from the task space to the joint space is required. Trajectory planning converts a description of a desired motion to a trajectory defining the time sequence of intermediate configurations of the arm between the origin and the final destination. An effective approach to the motion control problem for robotic manipulators is the so-called kinematic control. This is based on an inverse kinematic transformation which feeds the reference values corresponding to an assigned end-effecter trajectory to the joint servos. In this work, a six axis 6-R robot system for a "pick and place" operation and a legged robot will be designed and developed using MATLAB/robotics toolbox simultaneously as shown in Fig. 1 and Fig 2

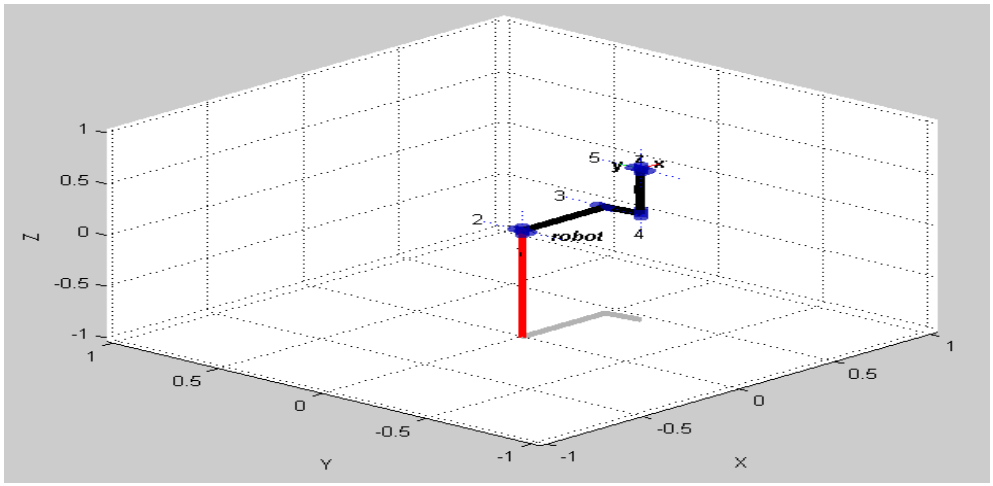

Fig. 1: Puma560.

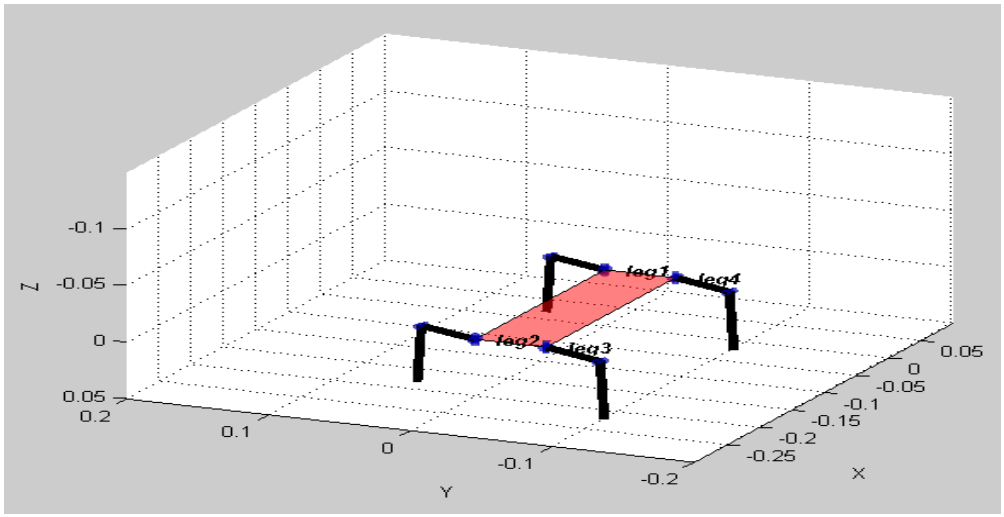

Fig. 2: Legged robot.

2. Robots Kinematics and Dynamics

2.1 Forward Kinematics

The forward kinematics problem is concerned with the relationship between the individual joints of the robot manipulator and the position and orientation of the tool or end-effector. Stated more formally, the forward kinematics problem is to determine the position and orientation of the end-effector, given the values for the joint variables of the robot. The joint variables are the angles between the links in the case of revolute or rotational joints, and the link extension in the case of prismatic or sliding joints. The objective of forward kinematic analysis is to determine the cumulative effect of the entire set of joint variables. For example consider a 3 DOF manipulator as shown in figure 3.

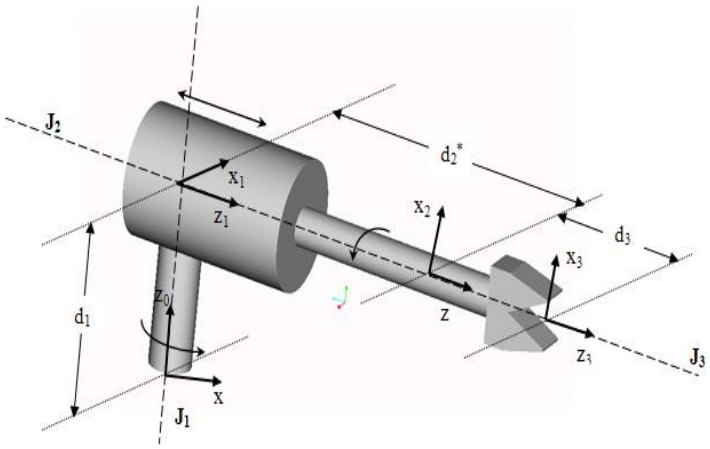

Fig. 3: 3 DOF manipulator.

Table 1: DH Parameters.

Links	Joints	a_i	α_i	d_i	θ_i
1	J_1	d_1	0	0	θ_1
2	J_2	d_2	0	0	θ_2
3	J_3	d_3	0	0	θ_3

where the joints' axes are assigned based on the Denavit-Hartenberg representation (Table1).

The general transformation matrix $^{i-1}_{i}\mathbf{T}$ for a single link can be obtained as follows.

$$ {}^{i-1}_{i}T = \begin{bmatrix} \cos\theta_i & -\sin\theta_i & 0 & a_{i-1} \\ \sin\theta_i \cos\alpha_{i-1} & \cos\theta_i \cos\alpha_{i-1} & -\sin\alpha_i & -\sin\alpha_{i-1} d_i \\ \sin\theta_i \sin\alpha_{i-1} & \cos\theta_i \sin\alpha_{i-1} & \cos\alpha_{i-1} & \cos\alpha_{i-1} d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

For example a 3DOF manipulator is considered then the transformation of manipulator is given by:

$$ {}^{0}_{3}T = {}^{0}_{1}T(q_1) * {}^{1}_{2}T(q_2) * {}^{2}_{3}T(q_3) $$

2.2 Inverse Kinematics

The inverse kinematics problem of the serial manipulators has been studied for many decades. It is needed in the control of manipulators. Solving the inverse kinematics is computationally expansive and generally takes a very long time in the real time control of manipulators. Tasks to be performed by a manipulator are in the Cartesian space, whereas actuators work in joint space. However, joint space is represented by joint angles. The conversion of the position and orientation of a manipulator end-effector from Cartesian space to joint space is called as inverse kinematics problem. There are two solutions approaches namely, geometric and algebraic used for deriving the inverse kinematics solution, analytically. Let's start with geometric approach.

2.3 Dynamics

In this topic we consider the dynamics and control of a serial-link manipulator. Each link is supported by a reaction force and torque from the preceding link, and is subject to its own weight as well as the as the reaction forces and torques from the links that it supports.

Equation of motion

Consider the motor which actuates the j^{th} revolute joint of a serial-link manipulator. From Figure **3** we recall that joint *j* connects link $j-1$ to link j. The motor exerts a torque that causes the outward link, j, to rotationally accelerate but it also exerts a reaction torque on the inward link $j-1$. Gravity acting on the outward links j to N exert a weight force, and rotating links also exert gyroscopic forces on each.

$$ Q = M(q)\ddot{q} + C(q,\dot{q})\dot{q} + F(\dot{q}) + G(q) + J(q)^{T} g $$

where q, $\dot{q}$, and $\ddot{q}$ are respectively the vector of generalized joint coordinates, velocities and accelerations, M is the joint-space inertia matrix, C is the Coriolis and centripetal coupling matrix, F is the friction force, G is the gravity loading, and Q is the vector of generalized actuator forces associated with the generalized coordinates q. The last term gives the joint forces due to a wrench g applied at the end effector and J is the manipulator Jacobian.

3. Results

Within the Toolbox we represent a robot link with a Link object which is created by

$$ L(i) = \text{Link}([\alpha_i, d_i, a_i, \theta_i, \beta_i]) ; \qquad (2.1) $$

Where β_i indicates whether the joint is revolute ($\beta_i = 0$) or prismatic ($\beta_i = 0$). If not specified a revolute joint is assumed.

3. 1 Puma 560

The Puma 560 as an example of the class of all-revolute six-axis robot manipulators. We create all six links by the above script, eq. 2.1. According to the DH parameters. And then linking all the links with function called **SerialLink** which create a seriallink object puma 560.

Puma560= SerialLink(L , 'name' , 'puma560');

We can see the modelled object Puma560, Fig. 1. Let the configuration be eg. q_x = ***[0,0,0,0,0,0]*** and ***Puma560.plot(q_x);*** will plot the robot for the entered configuration. The forward kinematics and inverse kinematics are computed using ***fkine, ikine*** method.

Consider the end-effector moving between two Cartesian poses ***T1, T2***. And the initial and final joint coordinate vector associated with these poses are ***q1, q2*** which is computed using ***ikine*** method.

And we require the motion to occur over a time period of ***t*** steps.

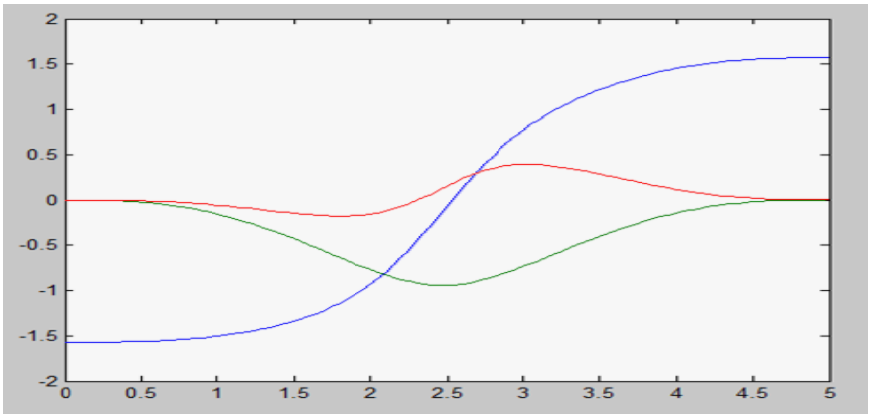

Fig. 4: (Roll,pitch,yaw) vs time.

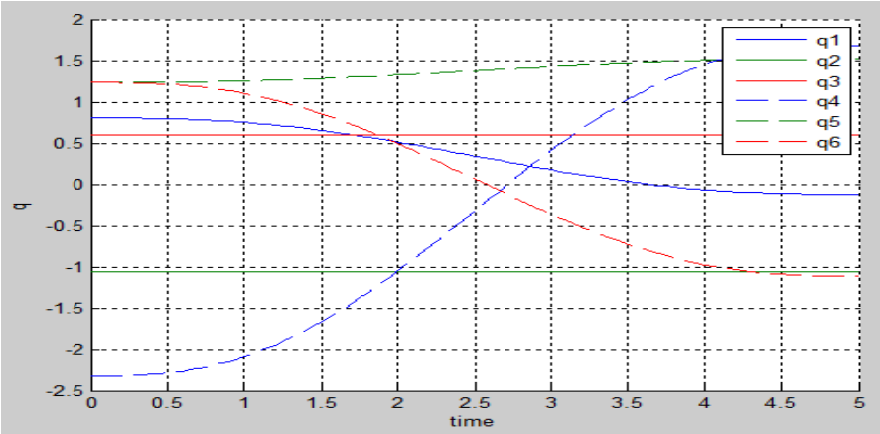

Fig. 5: Joint angles vs time.

3.2 Legged Robot

First we create a 3 link manipulator which is as shown in Fig6 and then we create a patch as shown in Fig.7 after creating patch the 3 link manipulator is translated at the edge of patch. Width and length of patch is taken in this example is 0.5, 2 mm respectively. Final model shown in Fig. 2 and then a loop is created as

```
for i=1:4
legs(i).plot(qcycle(1,:),plotopt)
end
```

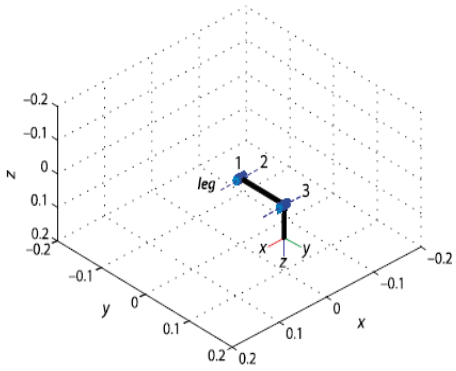

Fig. 6: Leg(1)

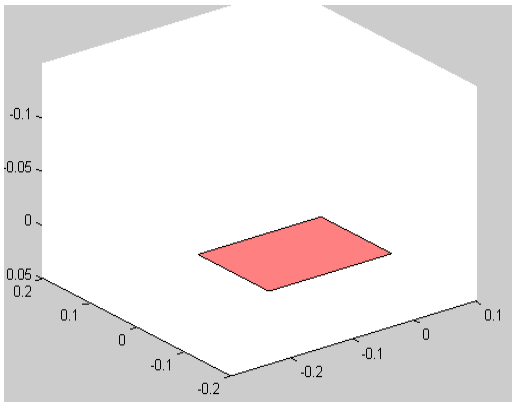

Fig. 7: Patch

References

[1] Mahmoud Gouasmi1,et.al(2012),Algeria Structural Mechanics Research Laboratory, Mechanical Engineering Department, Blida University, Algeria pp- 2-3

[2] Peter Corke(2011)Faculty of Built Environment and Engineering,School of Engineering Systems,Queensland University of Technology (QUT) pp - 142-145 , 164-168.