Online Trajectory Planning for a PUMA Robot

Chul-Goo Kang[1,#]

1 School of Mechanical Engineering, Konkuk University, Hwayang-Dong, Gwangjin-Gu, Seoul, South Korea, 143-701
Corresponding Author / E-mail: cgkang@konkuk.ac.kr, TEL: +82-2-447-2142, FAX: +82-2-447-5886

KEYWORDS: Trajectory planning, Orientation, Quaternion, Kinematic singularity, PUMA robot

Robotic applications, such as automatic fish cutting, require online trajectory planning because the material properties of the object, such as the bone or flesh conditions, are not known in advance. Different trajectories are required when the material properties vary. An effective online trajectory-planning algorithm is proposed using quaternions to determine the position and orientation of a robot manipulator with a spherical wrist. Quaternions are free of representation singularities and permit computationally efficient orientation interpolations. To prevent singular configurations, the exact locations of the kinematic singularities of the PUMA 560 manipulator are derived and geometrically illustrated when a forearm offset exists and the third link length is not zero.

Manuscript received: November 10, 2006 / Accepted: May 23, 2007

1. Introduction

The goal of robotic trajectory planning is to send motion commands to the robot control system at specific points in time to ensure the execution of a given manipulator task. A trajectory can be planned in joint space and/or operational (Cartesian) space. However, it is more convenient to use an operational space trajectory since the manipulator tasks are easier to specify; joint space coordinates are neither orthogonal nor able to separate the position from orientation.[1]

Trajectory planning in operational space for a robot manipulator is generally composed of planning the position and orientation of the end effector. For some applications, such as tool path generation in surface grinding and polishing, position planning is sufficient and orientation planning is not required.[2] However, for many robotic applications, such as fish cutting, grinding and deburring operations, and assembly automation, both the position and orientation of the end effector must be planned together. In applications such as automatic fish cutting, trajectories cannot be planned off-line before the task starts because the trajectory requirements may change during the actual processing. For example, a different trajectory is immediately required if a bone is reached (bone positions are not known in advance) or the body is deformed.

Accurate motion tracking is usually achieved by designing good control strategies.[3,4] However, good trajectory planning is required before the controller is designed to achieve good tracking performance.

Angeles and Akhras[5] developed an off-line procedure to plan the position and orientation of Cartesian trajectories in robot manipulators with 3-degrees-of-freedom (DOF) spherical wrists. Wu and Jou[6] proposed a geometric approach to plan the orientations of a robot end effector along a prescribed translational path using natural local coordinates and a rotation matrix. Other studies have used orientation interpolation using quaternions[7] for computer and space applications.[8,9] Quaternions have also been introduced for the resolved rate and acceleration control of manipulators[10] and other dynamic applications.[11] Quaternions describe an object, providing planning orientations that are computationally efficient in Cartesian space and free of the singularity gimbal lock problem[11] that exists in Euler angle representations.

The effectiveness of trajectory planning is related to the inverse kinematics algorithm, singularity problems, computational load, and Cartesian space trajectory generation. Efficient closed-form solutions for the inverse kinematics of the PUMA robot have been developed by several researchers.[12–14] In addition to finding inverse kinematic solutions, it is also important to select an appropriate solution from several available solutions, especially when flip and no-flip solutions are available at consecutive sampling times of a robotic task.

Singular configurations should be avoided through trajectory planning whenever possible, either by avoiding the actual singular configurations or by using joint space interpolation. A PUMA manipulator has three types of kinematic singularities, consisting of shoulder, elbow, and wrist singularities.[12, 13] Since most studies in the literature[15, 16] assume that no shoulder or elbow offset exists and that the length of the third link is zero, it follows that theoretical singular configurations are generally different from real ones encountered in an actual PUMA 560 robot.

In this paper, the exact singular configurations of a PUMA 560 manipulator is determined when a shoulder offset exists, and the three types of singular configurations are geometrically illustrated. An effective online trajectory-planning algorithm for both the position and orientation of a robot manipulator is then presented using quaternions for orientation planning. The orientation of the end effector is specified by three orthogonal unit vectors of the tool frame that are expressed in the Cartesian world coordinate frame.

The outline of this paper is as follows. Section 2 describes a new inverse kinematic solution for a PUMA 560 robot using a different home configuration and different coordinate frames. In Section 3, exact singular configurations of the PUMA 560 robot are derived when an elbow offset exists. An orientation planning technique using quaternions is presented in Section 4, and an online trajectory planning technique for an automatic fish-cutting process is proposed in Section 5. The conclusions are presented in Section 6.

Copyright (c) 2007 by KSPE

2. The Inverse Kinematics Problem of a PUMA Manipulator

The inverse kinematics problem of a manipulator is used to determine the joint variables in terms of a given position and the orientation of the end effector. The solutions for the inverse kinematics of a PUMA manipulator have been derived by Elgazzar[12], Paul and Zhang[13], and Cote *et al.*[14] The stretched-up home configuration adopted by the PUMA manufacturer[17] and Elgazzar[12] is generally not desirable because it contains an elbow singularity as well as a wrist singularity. Paul and Zhang[13] selected an elbow-up home configuration, but an elbow-down home configuration is more appropriate for manipulator tasks such as fish cutting and component assembly since they are mostly carried out using elbow-down configurations. If the home configuration is defined as elbow-up, the end-effector has to move across the elbow singularity to perform elbow-down tasks, which is not desirable.

Therefore, the home configuration and coordinate frames for a PUMA 560 manipulator were defined (Fig. 1) following the Denavit–Hartenberg convention, whose parameters are given in Table 1. The PUMA 560 robot is kinematically designed to decouple the position and orientation determination processes using a spherical wrist, which simplifies the kinematics problem.

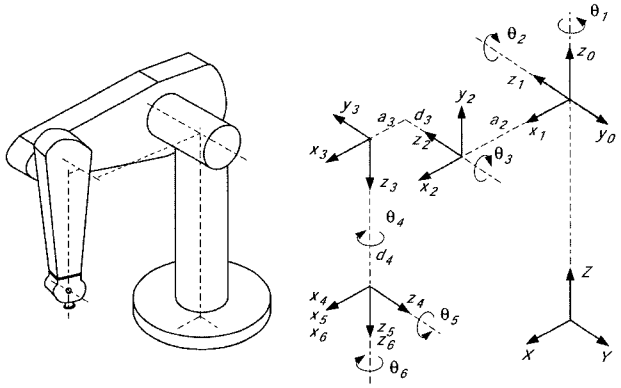

Fig. 1 Home configuration and coordinate frames for the PUMA 560 manipulator

Table 1 Denavit–Hartenberg parameters of the PUMA 560 manipulator

Link	θ_i (rad)	d_i (m)	a_i (m)	α_i (deg)
1	θ_1	0	0	90
2	θ_2	0	0.4318	0
3	θ_3	0.1254	0.0203	90
4	θ_4	0.4318	0	90
5	θ_5	0	0	−90
6	θ_6	0	0	0

A PUMA 560 manipulator has eight different inverse kinematic solutions if the wrist center (the origin of the $x_6y_6z_6$ frame) is within the workspace and the configuration does not have singularities. Let the position of the wrist center with respect to the base frame $x_0y_0z_0$ be $[p_x, p_y, p_z]^T$ and the orientation be

$$\begin{bmatrix} \cdot & o_x & a_x \\ \cdot & o_y & a_y \\ \cdot & o_z & a_z \end{bmatrix}$$

The Paul and Zhang[13] solutions for joint variables corresponding to the end effector position and orientation can then be modified for the present home configuration as follows:

$$\theta_1 = \begin{cases} \text{Atan2}\dfrac{p_y}{p_x} + \sin^{-1}\dfrac{d_3}{\sqrt{p_x^2+p_y^2}} & \text{(Right arm solution)} \\ \text{Atan2}\dfrac{p_y}{p_x} + \sin^{-1}\dfrac{d_3}{\sqrt{p_x^2+p_y^2}} + \pi & \text{(Left arm solution)} \end{cases}$$

$$\theta_2 = \begin{cases} \text{Atan2}\dfrac{p_z}{p_xc_1+p_ys_1} + \cos^{-1}\dfrac{a_2^2-d_4^2-a_3^2+(p_xc_1+p_ys_1)^2+p_z^2}{2a_2\sqrt{(p_xc_1+p_ys_1)^2+p_z^2}} \\ \text{Atan2}\dfrac{p_z}{p_xc_1+p_ys_1} - \cos^{-1}\dfrac{a_2^2-d_4^2-a_3^2+(p_xc_1+p_ys_1)^2+p_z^2}{2a_2\sqrt{(p_xc_1+p_ys_1)^2+p_z^2}} \end{cases}$$

$$\theta_3 = \text{Atan2}\frac{(p_xc_1+p_ys_1)c_2+p_zs_2-a_2}{(p_xc_1+p_ys_1)s_2-p_zc_2} - \text{Atan2}\frac{a_3}{d_4}$$

$$\theta_4 = \begin{cases} \text{Atan2}\dfrac{a_xs_1-a_yc_1}{(a_xc_1+a_ys_1)c_{23}+a_zs_{23}}, & \text{if } s_5>0 \\ \text{Atan2}\dfrac{-a_xs_1+a_yc_1}{-(a_xc_1+a_ys_1)c_{23}-a_zs_{23}}, & \text{if } s_5<0 \\ \text{indeterminate}, & \text{if } s_5=0 \end{cases}$$

$$\theta_5 = \text{Atan2}\frac{-\{(a_xc_1+a_ys_1)c_{23}+a_zs_{23}\}c_4-(a_xs_1-a_yc_1)s_4}{(a_xc_1+a_ys_1)s_{23}-a_zc_{23}}$$

$$\theta_6 = \text{Atan2}\frac{-[\{(o_xc_1+o_ys_1)c_{23}+o_zs_{23}\}c_4+(o_xs_1-o_yc_1)s_4]c_5-\{(o_xc_1+o_ys_1)s_{23}-o_zc_{23}\}s_5}{(o_xs_1-o_yc_1)c_4-\{(o_xc_1+o_ys_1)c_{23}+o_zs_{23}\}s_4}$$

where Atan2 is the arctangent function of two arguments that returns values from $-\pi$ to π, $c_i = \cos\theta_i$, $s_i = \sin\theta_i$, $c_{23} = \cos(\theta_2+\theta_3)$, and $s_{23} = \sin(\theta_2+\theta_3)$. The two solutions to θ_2 correspond to elbow-up and elbow-down configurations, and the two solutions to θ_4 correspond to flip and no-flip solutions. If $\theta_5 = 0$, an infinite number of solutions exist ($\theta_6 = -\theta_4$ for any θ_4) for which the two joint axes of Joints 4 and 6 are collinear.

One can avoid shoulder and elbow singularities when planning a trajectory for a specific manipulator task by identifying those singular configurations in advance. However, wrist singularities must be tolerated since they can occur anywhere inside the workspace. If the motion trajectory is planned across a wrist singularity ($\theta_5 = 0$), the selection between two solutions for θ_4 is important because a wrong choice could lead to an angle jump of π radians or an angle discontinuity in the consecutive joint angles of θ_4. Even if the solution of θ_4 depends on the joint angle θ_5, the following simple logic is used to choose a continuous solution θ_4 instead of using θ_5:

$$\textit{If } |\theta_4(t_{i-1}) - \theta_4(t_i)| \approx \pi \textit{, select the other solution } \theta_4$$

Two consecutive joint angles θ_4 cannot be a large angle such as π radians in a smooth operation. Therefore, if one encounters a large angle jump at θ_4 during the trajectory planning, this is interpreted as an inappropriate θ_4 solution since one passes through a wrist singularity $\theta_5 = 0$. Instead, the other θ_4 solution is selected to obtain a smooth operation. In other words, if the difference between the previous angle and the present angle of θ_4 is approximately π, the other solution for θ_4 is selected as the present joint angle θ_4.

3. Analysis of the Kinematic Singularities of the PUMA Manipulator

The kinematic singularities of a manipulator are defined as the configurations at which the Jacobian $\boldsymbol{J}(\mathbf{q})$ is rank-deficient and consequently the mobility of the manipulator is reduced. Kinematic

singularities may also have other characteristics, such as an infinite number of inverse kinematic solutions.[18] The Jacobian of a manipulator defines the mapping $[\dot{\mathbf{p}}^T \omega^T]^T = \boldsymbol{J}(\mathbf{q})\,\dot{\mathbf{q}}$ between the joint velocity vector $\dot{\mathbf{q}}$ and the end effector velocity vector $[\dot{\mathbf{p}}^T \omega^T]^T$, which is expressed with respect to the fixed base frame $x_0 y_0 z_0$.

For the PUMA 560 manipulator, the Jacobian $\boldsymbol{J}(\mathbf{q})$ is given by

$$J(\mathbf{q}) = \begin{bmatrix} j_{11} & -c_1(a_2s_2 + a_3s_{23} - d_4c_{23}) & j_{13} \\ j_{21} & -s_1(a_2s_2 + a_3s_{23} - d_4c_{23}) & j_{23} \\ 0 & a_2c_2 + a_3c_{23} + d_4s_{23} & j_{33} \\ 0 & s_1 & s_1 \\ 0 & -c_1 & -c_1 \\ 1 & 0 & 0 \end{bmatrix}$$

$$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ c_1s_{23} & c_1c_{23}s_4 - s_1c_4 & -c_1c_{23}c_4s_5 - s_1s_4s_5 + c_1s_{23}c_5 \\ s_1s_{23} & s_1c_{23}s_4 + c_1c_4 & -s_1c_{23}c_4s_5 + c_1s_4s_5 + s_1s_{23}c_5 \\ -c_{23} & s_{23}s_4 & -s_{23}c_4s_5 - c_{23}c_5 \end{bmatrix}$$

where

$$\begin{aligned} j_{11} &= -s_1(a_2c_2 + a_3c_{23} + d_4s_{23}) + d_3c_1 \\ j_{21} &= c_1(a_2c_2 + a_3c_{23} + d_4s_{23}) + d_3s_1 \\ j_{13} &= -c_1s_2(a_3c_3 + d_4s_3) - c_1c_2(a_3s_3 - d_4c_3) \\ j_{23} &= -s_1s_2(a_3c_3 + d_4s_3) - s_1c_2(a_3s_3 - d_4c_3) \\ j_{33} &= c_2(a_3c_3 + d_4s_3) - s_2(a_3s_3 - d_4c_3). \end{aligned}$$

The rank deficiency of $\boldsymbol{J}(\mathbf{q})$ for this manipulator implies $\det \boldsymbol{J}(\mathbf{q}) = 0$, which after extensive mathematical manipulation gives the following three equations:

$$a_2c_2 + a_3c_{23} + d_4s_{23} = 0 \quad (1)$$

$$a_3s_3 - d_4c_3 = 0 \quad (2)$$

$$s_5 = 0 \quad (3)$$

Equation (1) corresponds to the shoulder singularities of the PUMA manipulator. Examples of singular shoulder configurations are shown geometrically in Fig. 2. These singularities arise when the x_1 coordinate of the wrist center o_4 is zero for any trunk rotation θ_1. They appear as elbow-up or elbow down configurations, where the velocity component in the z_1 direction cannot be generated, thus creating a degenerate z_1 direction. If an offset d_3 equals zero, these singular configurations will have an infinite number of solutions to the inverse kinematics for which the wrist center intersects the axis of the trunk rotation z_0.

Equation (2) represents elbow singularities because it originates from the motion of the elbow joint θ_3. These singularities occur when the wrist center is extended (but not fully extended) or retracted (but not fully retracted), as shown in Fig. 3. The exact locations of these singularities are configurations with $\theta_3 = 87.308°$ or $-92.692°$ when the forearm offset d_4 = 0.4318 m and the link length $a_3 = 0.0203$ m. However, the retracted configuration does not occur within the workspace due to the motion limitation of Joint 3. The degenerate direction of these singularities is x_1. Singular configurations given in previous studies[15, 16] are different from those derived in this paper and encountered in actual practice because a shoulder and elbow offset exist and the length of the third link is not zero.

Equation (3) corresponds to wrist singularities, which arise whenever the joint axes z_3 and z_5 are collinear. When this singularity is encountered, the mobility is reduced and the number of solutions to the inverse kinematics is infinite. Shoulder and elbow singularities can be avoided in the trajectory planning by adjusting the robot size and/or selecting the manipulator working region carefully, but wrist singularities can be encountered anywhere inside the manipulator workspace; as a consequence, special care must be taken when planning the trajectory of the end effector.

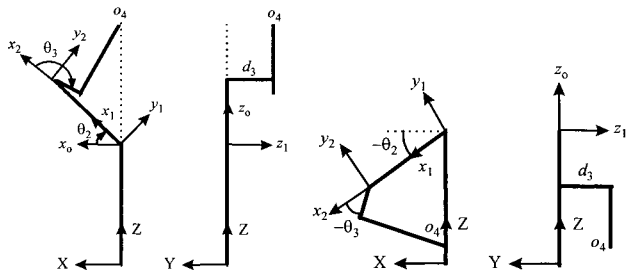

Fig. 2 Shoulder singularities of the PUMA 560 manipulator: (a) elbow-up and (b) elbow-down configurations

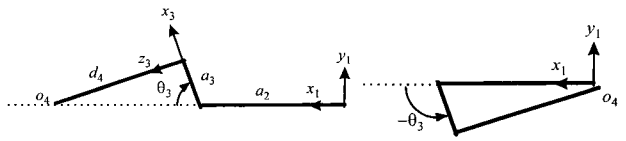

Fig. 3 Elbow singularities of the PUMA 560 manipulator: (a) extended and (b) retracted configurations

4. Orientation Planning Using Quaternions

When planning a rotation that reorients the end effector from one orientation to another, Euler angle, Bryant angle, or roll-pitch-yaw angle representations are not appropriate if intermediate orientations are required. Euler angle representations also suffer from gimbal lock or representation singularities.[11] When planning intermediate orientations as well as the final orientation, an axis-angle representation $(\mathbf{u}, \theta)$ is more intuitive than angle representations from a physical point of view. However, this representation also has a drawback when the rotation angle θ equals 0 or π, for which the axis $\mathbf{u}$ is not defined. Quaternion representations are free of these singularities and computationally efficient. Quaternions are relatively well-known and have been successfully used for computer graphics and space technology.

The quaternion $\mathbf{q}$ is a four-parameter representation of a rotation using three independent parameters as follows:

$$\mathbf{q} = \begin{bmatrix} q_0 \\ q_1 \\ q_2 \\ q_3 \end{bmatrix} = \begin{bmatrix} \cos(\theta/2) \\ -\mathbf{u}\sin(\theta/2) \end{bmatrix}, q_0^2 + q_1^2 + q_2^2 + q_3^2 = 1 \quad (4)$$

where the unit vector $\mathbf{u}$ is the eigenvector of the rotation matrix R (or the axis of rotation corresponding to the rotation matrix R), and θ is the rotation angle about $\mathbf{u}$. One coordinate frame is rotated with respect to the other instead of considering the rotation of rigid objects. The relationship between the quaternion $\mathbf{q}$ and the rotation matrix R in component form is[8]

$$\begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} = \begin{bmatrix} q_0^2 + q_1^2 - q_2^2 - q_3^2 & 2(q_1q_2 - q_0q_3) & 2(q_1q_3 + q_0q_2) \\ 2(q_1q_2 + q_0q_3) & q_0^2 - q_1^2 + q_2^2 - q_3^2 & 2(q_2q_3 - q_0q_1) \\ 2(q_1q_3 - q_0q_2) & 2(q_2q_3 + q_0q_1) & q_0^2 - q_1^2 - q_2^2 + q_3^2 \end{bmatrix} \quad (5)$$

Suppose the original orientation of the end effector is R_1^0 and the final orientation is R_2^0 in Cartesian space. Then the rotation matrix between the two orientation is $R = (R_1^0)^T R_2^0$, and the axis of rotation $\mathbf{u}$ and the rotation angle θ_{12} are obtained from the quaternion relationship given by Eq. (5) (refer to the algorithm given in Ref. 8). The angular position θ and angular velocity ω at each sampling time can be computed using, for example, the trapezoidal velocity profile shown in Fig. 4(a). Then the orientation in Cartesian space at the current sampling time can be obtained from R_1^0 postmultiplied by the current rotation matrix. If the angle change is too small to reach the plateau of the trapezoidal velocity profile, then the triangular profile shown in Fig. 4(b) will be followed.

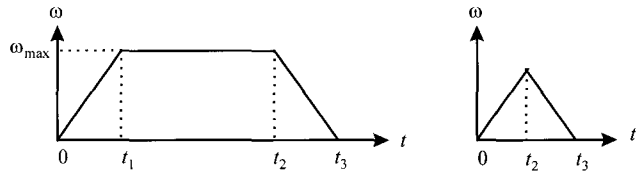

Fig. 4 Trapezoidal angular velocity profile for (a) a large angle and (b) a small angle

5. Trajectory Generation for a Robot Manipulator

Although the manipulator joint coordinates fully specify the position and orientation of the end effector, they are not suitable as a working coordinate system because they are not orthogonal or able to separate position from orientation. In this section, trajectory planning in Cartesian space is considered for which path motions are planned instead of point-to-point motions. The Cartesian space trajectory is then converted to a joint space trajectory that is used as motion commands for the joint servos.

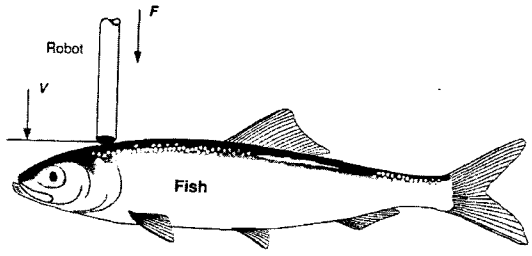

Fig. 5 Fish-cutting automation

For robotic applications, such as automatic fish cutting shown in Fig. 5, it is not always feasible to generate the trajectory in advance or off-line because a different operation may be required due to changes in the environment (*e.g.*, encountering a bone). In such applications, motion commands for the present instant in time must be generated online according to the present environment conditions. Online trajectory planning also avoids computer memory problems that sometimes occur when all the joint commands for the entire path are computed in advance. To save on the computational time required for the trajectory generation, the motion commands are generated at longer sampling times and at a higher level of the hierarchical control structure. Then they are interpolated in the joint space to obtain the fine motion commands at the joint servo control level. In other words, a control structure with multiple sampling rates is desirable.

A manipulator trajectory in Cartesian space was planned using path motions, not point-to-point motions. The procedure used to generate online motion commands was as follows. First an appropriate tool frame (or tool tip) motion of the robot manipulator with respect to the world coordinate frame *XYX* was generated in some manner, for example, using the trapezoidal velocity profiles as shown in Fig. 6, where one profile is for the linear velocity while the other profile is for the angular velocity. Since the position and orientation planning can be considered separately (but not independently) due to the spherical wrist, the maximum linear and angular accelerations are used so that the position and orientation planning might end at different instants in time.

The tool frame motion is generated in two steps, consisting of pre-calculation of the trajectory parameters, such as the instants in time of trapezoidal velocity profiles and acceleration components, and online path generation at each sampling time. The task conditions are checked in the second step. If a change occurs in the conditions, a different trajectory is immediately planned in Cartesian space.

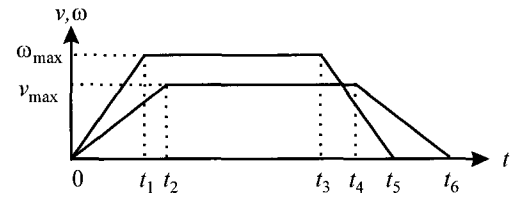

Fig. 6 Position and orientation trajectory planning in Cartesian space using trapezoidal velocity profiles

The tool tip motion (*i.e.*, position and velocity) commands for the position and orientation at a specific sampling time are first converted into motion commands for the wrist center, and then into motion commands with respect to the robot base frame instead of the world coordinate frame. Finally, they are converted into either joint coordinate commands using the inverse kinematics algorithm and/or joint velocity commands using the Jacobian inverse J^{-1}. The damped least-squares Jacobian inverse is adopted instead of the Jacobian inverse in the neighborhood of kinematic singularities and at the singularities themselves.[19]

Using the above technique, a trajectory can be planned through a singular wrist configuration. The fine joint motion commands at each sampling time of the joint servo control are obtained by interpolating the joint motion commands at the joint servo control level. The procedure can be summarized as follows:

```
prepare_path_planning( ... )
interrupt_service_routine( )
    Cartesian_motion_command( ... )
    wrist_center_motion( ... )
    transform_to_robot_base_frame( ... )
    inverse_kinematics( ... )
    inverse_velocity_kinematics( ... )
```

The above algorithm was implemented using ANSI C and applied successfully to an automated fish-cutting process using a PUMA 560 robot manipulator. The C code structure for the path planning is shown in Fig. 7. The functions *PTP* (*P1*, *P2*), *StraightLine* (*P1*, *P2*, *vi*, *vf*), *SemiCircleUP* (*P1*, *P2*), *VerticalDownCut* (*P1*, *P2*, *P3*, *vi*), and *HorizontalCut* (*P1*, *P2*, *vi*, *vf*) were used to generate each segment of the fish-cutting path. The entire C code consisted of 2300 lines.

The vertical and horizontal cutting motion elements provide the fish-cutting operations. During the vertical cutting motion, the end effector moves downward and cuts the fish vertically. When the effector touches the flesh of the fish during the downward motion, its speed is reduced as specified and maintained until it reaches a bone, whereupon it stops with the maximum deceleration. This motion generation element requires flesh and bone hitting information, which must be generated online. The horizontal cutting motion cuts the fish horizontally. This motion element has the capability to start or end with either a zero velocity or the maximum cutting velocity, as specified. Whenever a bone is touched, the end effector moves one

step upward and then continues its horizontal motion.

```
#include <stdio.h>
#include "pumapath.h"

void main()
{
    double H[4][4] = { 1.0, 0.0, 0.0, 0.4521,
                       0.0,-1.0, 0.0,-0.1254,
                       0.0, 0.0,-1.0, 0.24003-TOOL_LENGTH,
                       0.0, 0.0, 0.0, 1.0};          // Home pose

    double A[4][4] = { 0.0, 1.0, 0.0, 0.0,
                       1.0, 0.0, 0.0, 0.25,
                       0.0, 0.0,-1.0, 0.3,
                       0.0, 0.0, 0.0, 1.0};          // Start pose
     ........

    PTP(H, A);
    VerticalDownCut(A, Ab, Ah, 0); // 0 means zero initial vel.
    StraightLine(Ab, A, 0, 1);/*0,1 means zero initial velocity and
            nonzero final velocity, i.e., maintain max. velocity */

    B[2][3] = A[2][3];    // actual z
    SemiCircleUp(A, B);

    Bb[1][3] = B[1][3];   // actual y
    VerticalDownCut(B, Bb, Bh, 1); // 1 means nonzero initial vel.
    StraightLine(Bb, B, 0, 1);
.......

    HorizontalCut(Q, Dh, 0, 1); // initial= zero, final_vel=nonzero
    HorizontalCut(Dh, Ch, 1, 1); // initial= zero, final_vel=nonzero
.......

    PTP(A, H);
}
```

Fig. 7 C code structure for path planning

The function *PTP (P1, P2)* in Fig. 7 generates six joint angle and angular velocity commands at each sampling time, corresponding to the position and orientation change between two configurations in Cartesian space using a trapezoidal velocity profile. The function *StraightLine (P1, P2, vi, vf)* is similar to *PTP (P1, P2)* except that it maintains the same orientation in Cartesian space during the straight-line motion. The variables *vi* and *vf* are the initial and final velocity flags: $vi = 0$ specifies start with a zero velocity, $vi = 1$ specifies start with the maximum velocity, $vf = 0$ specifies end with a zero velocity, and $vf = 1$ specifies end with the maximum velocity. For example, *StraightLine (P1, P2, 0, 1)* will accelerate the end effector from a zero velocity to the maximum velocity with the prescribed acceleration, and maintain the maximum velocity to the end point *P2*.

The function *SemiCircleUp (P1, P2)* generates an upward semicircle trajectory in the vertical plane between two Cartesian positions, *P1* and *P2*. The trajectory keeps the same orientation in Cartesian space at the maximum speed from *P1* to *P2*. The function *VerticalDownCut (P1, P2, P3, vi)* generates a motion command to cut the fish horizontally. In this function, the main direction of motion is in the downward z direction; $vi = 0$ specifies that the motion starts from a zero velocity at *P1*, while $vi = 1$ specifies that the motion starts with the maximum velocity. If flesh is touched in the downward motion, the speed decelerates to the speed reducing rate multiplied by the maximum velocity, which is maintained until bone is touched, whereupon the end effector stops and its position is returned to *P3*. The position *P2* is an assumed bone position. If bone is not encountered before the effector reaches *P2*, it will stop at *P2*.

The function *HorizontalCut (P1, P2, vi, vf)* generates a motion command to cut fish horizontally. The main direction of the motion is in the horizontal x or y directions. The flags *vi* and *vf* have the same meaning as in *StraightLine*(...) except that the maximum velocity is the maximum cutting speed, which is the maximum velocity multiplied by the speed reducing rate defined in 'pumapath.h.' Whenever a bone is hit during the horizontal motion, the function generates a one step upward motion in the positive z direction at the maximum cutting speed for one sampling period.

6. Conclusions

The exact locations and degenerate directions of the shoulder and elbow singularities of the PUMA 560 manipulator were demonstrated analytically and geometrically when the forearm offset d_4 and link length a_3 had actual values and were not set to zero. The shoulder singularities arose when the x_1 coordinate of the wrist center was zero, regardless of the trunk rotation. The degenerate direction of these singular configurations was z_1. Elbow singularities occurred when the wrist center was extended with the elbow joint variable $\theta_3 = 87.308°$ (not fully extended) or when the wrist center was retracted with the elbow joint variable $\theta_3 = -92.692°$ (not fully retracted). The degenerate direction of these singularities was x_1. The exact locations and degenerate directions of the shoulder and elbow singularities are important since they must be avoided when planning the trajectory of the PUMA manipulator whenever possible.

An online trajectory generation algorithm for a robot manipulator was proposed using quaternions for orientation trajectory planning. The proposed algorithm allowed the planning of a different trajectory immediately whenever an environmental condition change was identified. Orientation planning using quaternions was successfully and efficiently implemented in an automatic fish-cutting process.

ACKNOWLEDGMENTS

The author gratefully acknowledges partial financial support from the Seoul R&BD Support Program(e-Printing Cluster Project: 2-C). Thanks also to Professor Clarence de Silva at the University of British Columbia in Vancouver, Canada, for his valuable comments and discussions.

REFERENCES

1. Paul, R. P., "Manipulator Cartesian path control," IEEE Transactions on Systems, Man, and Cybernetics, Vol. 9, No. 11, pp. 702–711, 1979.
2. Kim, H. T., Yang, H. J. and Kim, S. C., "Control method for the tool path in aspherical surface grinding and polishing," International Journal of Precision Engineering and Manufacturing, Vol. 7, No. 4, pp. 51–56, 2006.
3. Yang, C., "Control strategy to reduce tracking error by impulsive torques at the joint," International Journal of Precision Engineering and Manufacturing, Vol. 6, No. 2, pp. 61–71, 2005.
4. Sung, Y. G. and Lee, K. T., "An adaptive tracking controller for vibration reduction of flexible manipulator," International Journal of Precision Engineering and Manufacturing, Vol. 7, No. 3, pp. 51–55, 2006.
5. Angeles, J. and Akhras, R., "Cartesian trajectory planning for 3 DOF spherical wrists," Proceedings of the IEEE Robotics and Automation Conference, pp. 68–74, 1988.
6. Wu, C. H. and Jou, C. C., "Planning and control of robot orientational path," IEEE Transactions on Systems, Man, and Cybernetics, Vol. 19, No. 5, pp. 1234–1241, 1989.
7. Bennett, M. K., "Affine and Projective Geometry," John Wiley & Sons, Inc., p. 72, 1995.
8. Shoemaker, K., "Animating rotation with quaternion curves," Computer Graphics (Proc. Siggraph 85), Vol. 19, No. 3, pp. 245–254, 1985.
9. Pletincks, D., "The use of quaternions for animation, modelling and rendering," New Trends in Computer Graphics (Proc. CG International '88, Springer–Verlag), pp. 45–53, 1988.
10. Yuan, J. S. C., "Closed-loop manipulator control using quaternion feedback," IEEE Journal of Robotics and Automation, Vol. 4, No. 4, pp. 434–440, 1988.

11. Wittenburg, J., "Dynamics of Systems of Rigid Bodies," B.G. Teubner, 1977.

12. Elgazzar, S., "Efficient kinematic transformations for the PUMA 560 robot," IEEE Journal of Robotics and Automation, Vol. 1, pp. 142–151, 1985.

13. Paul, R. P. and Zhang, H., "Computationally efficient kinematics for manipulators with spherical wrists based on the homogeneous transformation representation," International Journal of Robotics Research, Vol. 5, pp. 32–44, 1986.

14. Cote, J., Gosselin, C. M. and Laurendeau, D., "Generalized inverse kinematic functions for the Puma manipulators," IEEE Transactions on Robotics and Automation, Vol. 11, No. 3, pp. 404–408, 1995.

15. Sciavicco, L. and Siciliano, B., "Modeling and Control of Robot Manipulators," McGraw–Hill, 1996.

16. Chiaverini, S. and Egeland, O., "A solution to the singularity problem for six-joint manipulators," Proceedings of the IEEE Robotics and Automation Conference, Vol. 1, pp. 644–649, 1990.

17. Unimation Inc., "Unimate PUMA Mark II Robot 500 Series Equipment Manual," Unimation Inc., 1985.

18. Spong, M. W., Hutchinson, S. and Vidyasagar, M., "Robot Dynamics and Control," Wiley, 2006.

19. Chiaverini, S., Sciavicco, L. and Siciliano, B., "Control of robotic systems through singularities," Advanced Robot Control, Springer–Verlag, pp. 285–295, 1990.