SOUTH DAKOTA SCHOOL OF MINES & TECHNOLOGY

RIAS Robotics & Intelligent Autonomous Systems

Introduction to Mobile Robotics

Jeff McGough

Department of Computer Science
South Dakota School of Mines and Technology
Rapid City, SD 57701, USA

October 1, 2012

Examples and Applications

- GSL
  - Simple GSL code
  - Compiling
  - Random numbers
  - Matrices and Linear Systems
- Octave
  - Matrix notion
  - Linear solves
- Fixed frame
- Current frame

Combined Transforms

Begin with a point x in space. An application of a transformation, T_1, with respect to the global frame carries this point to a new point x':

$$x' = T_1 x$$

In essense, the fixed frame views the point as in motion. Now apply another transformation T_2 to the new point x':

$$x" = T_2 x' = T_2(x') = T_2(T_1 x) = T_2 T_1 x$$

Note that each transform was done with respect to the fixed frame.

Combined Transforms

Again, begin with a point x in space. If we view the transformation, T from the perspective of the point (which will be fixed), then it appears that the "fixed" frame is moving

AND

that the motion is in the *opposite* direction of the fixed frame transformation. Opposite here would be the inverse transformation: T^{-1}.

Thus combined transformations from the point's "point of view":

$$T^{-1} = T_2^{-1} T_1^{-1}, \quad \text{or} \quad T = \left(T_2^{-1} T_1^{-1}\right)^{-1}$$

$$T = T_1 T_2$$

Reverse order.

Combined Transforms

Successive transformations relative to the global frame are left multiplied:

$$T = T_n T_{n-1} \dots T_1 T_0$$

For example, take a rotation about z of 30 degrees, R_1, followed by a rotation about x by 60 degrees, R_2:

$$R = R_2 R_1 = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos 60 & -\sin 60 & 0 \\ 0 & \sin 60 & \cos 60 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \cos 30 & -\sin 30 & 0 & 0 \\ \sin 30 & \cos 30 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Combined Transforms

Successive transformations relative to the moving frame are right multiplied:

$$T = T_0 T_1 \dots T_{n-1} T_n$$

For example, take a rotation about x by 45 degrees, R, followed by a translation in z by 4 cm, T:

$$M = TR = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos 60 & -\sin 60 & 0 \\ 0 & \sin 60 & \cos 60 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 4 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Inverse Transforms

Inverting transformation matrices ...

$$T^{-1} = (T_n T_{n-1} \dots T_1 T_0)^{-1} = T_0^{-1} T_1^{-1} \dots T_{n-1}^{-1} T_n^{-1}$$

How does one invert the transformations?

Inverses

Rotation matrices are orthogonal and so

$$R^{-1} = R^T$$

For example, the inverse of the 60 degree rotation mentioned above:

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos 60 & -\sin 60 & 0 \\ 0 & \sin 60 & \cos 60 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}^{-1} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos 60 & \sin 60 & 0 \\ 0 & -\sin 60 & \cos 60 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Inverses

Translation matrices are simple as well. One just negates the translation components.

Thus:

$$\begin{pmatrix} 1 & 0 & 0 & a \\ 0 & 1 & 0 & b \\ 0 & 0 & 1 & c \\ 0 & 0 & 0 & 1 \end{pmatrix}^{-1} = \begin{pmatrix} 1 & 0 & 0 & -a \\ 0 & 1 & 0 & -b \\ 0 & 0 & 1 & -c \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Thus we can just undo the transformations individually.

RPY Angles

RPY angles provide the position and orientation of a craft by using a translation to body center and then three rotation matrices for craft pose.

- Rotation about a (z axis) - Roll
- Rotation about o (y axis) - Pitch
- Rotation about n (x axis) - Yaw

$$M = R_n R_o R_a T$$

Euler Angles

Euler angles provide the position and orientation of a craft by using a translation to body center and then three rotation matrices for craft pose. However - reference is with respect to the body, not the world coordinates.

- Rotation about a (z axis) - Roll
- Rotation about o - Pitch
- Rotation about a - Roll

$$M = R_a R_o R_a T$$

Forward and Inverse Kinematics

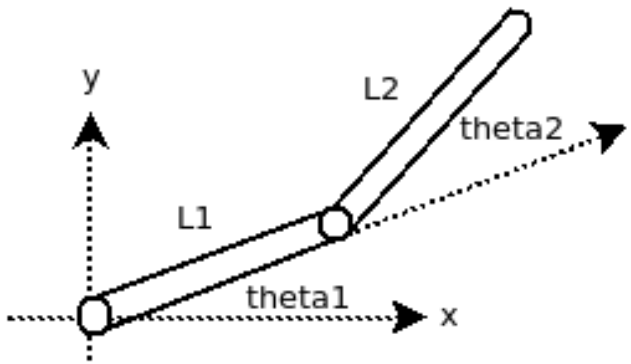
Given joint angles and actuator lengths it is straightforward to compute end effector position. Thus it is easy to find effector path as a function of rotations.

$$\big(\theta_1(t), ..., \theta_n(t)\big) \rightarrow p(t)$$

It is MUCH harder to find the angle functions if you are given the end effector path:

$$p(t) \rightarrow \big(\theta_1(t), ..., \theta_n(t)\big)$$

Two link manipulator



Let the base of the first link be the origin of the fixed frame shown in the figure. The location of the second joint is $(\tilde{x}, \tilde{y})$ given by

$$\tilde{x} = L_1 \cos\theta_1, \quad \tilde{y} = L_1 \sin\theta_1$$

The location of the end effector is given by (x, y)

$$x = L_2 \cos(\theta_1 + \theta_2) + \tilde{x}, \quad y = L_2 \sin(\theta_1 + \theta_2) + \tilde{y}$$

Thus one obtains

$$x = L_2 \cos(\theta_1 + \theta_2) + L_1 \cos\theta_1, \quad y = L_2 \sin(\theta_1 + \theta_2) + L_1 \sin\theta_1$$

Review

Represent points by $\begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix}$ and vectors by $\begin{pmatrix} v_x \\ v_y \\ v_z \\ 0 \end{pmatrix}$

Recall a rotation about x by 30 degrees

$$\begin{pmatrix} x' \\ y' \\ z' \\ 1 \end{pmatrix} = \begin{pmatrix} \cos 30 & -\sin 30 & 0 & 0 \\ \sin 30 & \cos 30 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix}$$

Denavit-Hartenberg Parameters

Provides a standard way to build kinematic models for a robot.

Simple concept.

Follow out the links of the manipulator, and see them as rotations and translations of the coordinate system:

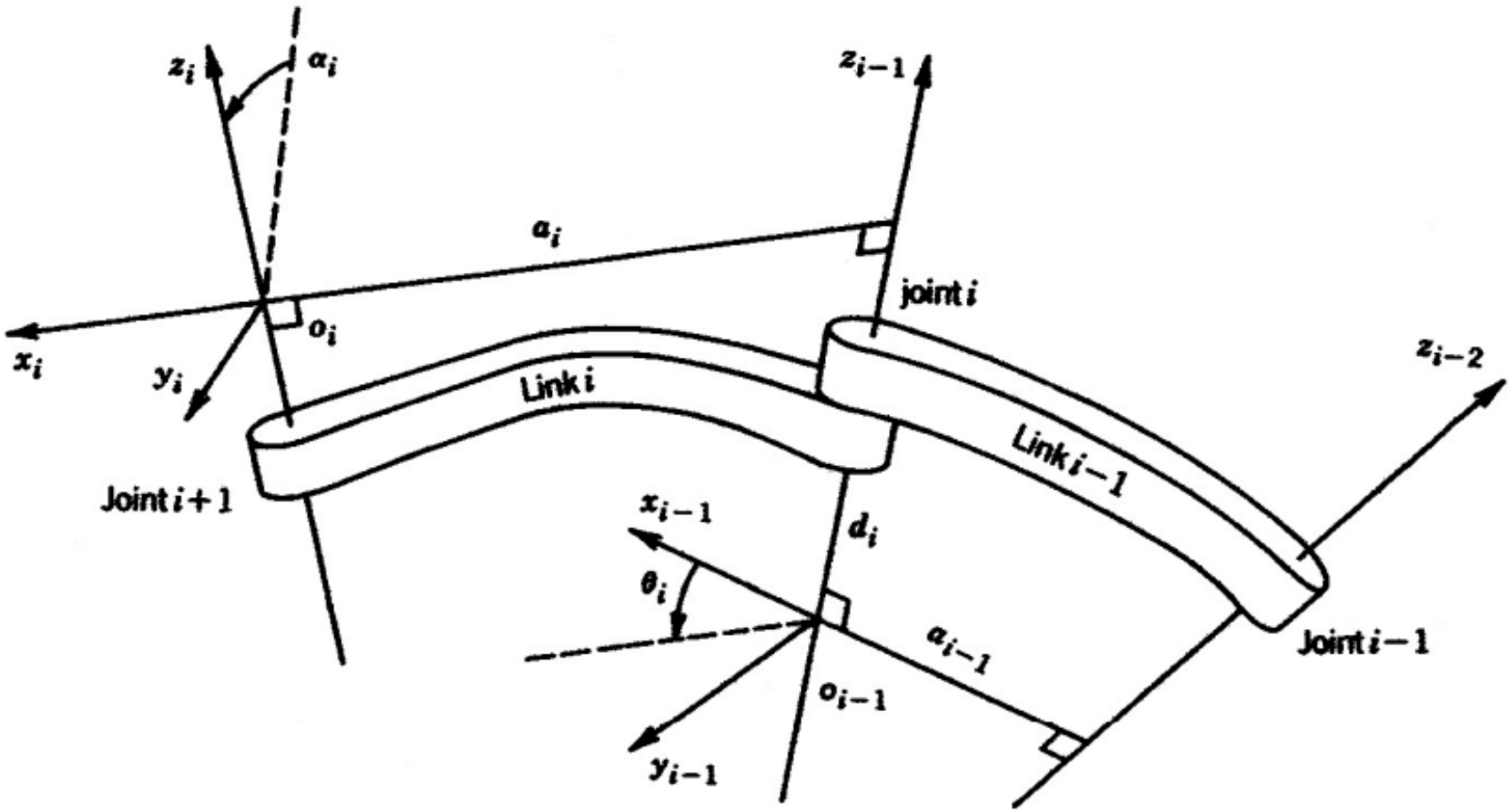
$$P = P_0 P_1 ... P_{n-1} P_n$$

where $P_k = R_z T_z T_x R_x$

Denavit-Hartenberg Parameters

- Each link is assigned a number. Normally start with the base and work towards the effector.
- All joints are represented by the z axis, z_i where the z axis is the axis of revolution (right hand rule for orientation).
- θ_i will represent the rotation about the joint.
- The x axis, x_i is in the direction that connects the links. [Well, connects the z axes of each joint.]
- a_i is link length.
- α_i will be the angles between z axes (if they are not parallel).
- d_i will represent the offset along the z axis.

Denavit-Hartenberg Parameters



Denavit-Hartenberg Parameters

Thus, the translation from one joint to the next involves a rotation, translation, translation and a rotation:

- Rotate about the local z axis angle θ.
- Translate along the z axis amount d.
- Translate along x amount a.
- Rotate about the new x axis (the joint twist) amount α.

This set of transformations will then change the coordinate system to the next link in the serial chain.

Denavit-Hartenberg Parameters

$$A_{n+1} =$$

$$\begin{pmatrix} \cos\theta_{n+1} & -\sin\theta_{n+1} & 0 & 0 \\ \sin\theta_{n+1} & \cos\theta_{n+1} & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & d_{n+1} \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 & a_{n+1} \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$$\times \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\alpha_{n+1} & -\sin\alpha_{n+1} & 0 \\ 0 & \sin\alpha_{n+1} & \cos\alpha_{n+1} & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Denavit-Hartenberg Parameters

$$A_{n+1} = \begin{pmatrix} \cos\theta_{n+1} & -\sin\theta_{n+1}\cos\alpha_{n+1} & \sin\theta_{n+1}\sin\alpha_{n+1} & a_{n+1}\cos\theta_{n+1} \\ \sin\theta_{n+1} & \cos\theta_{n+1}\cos\alpha_{n+1} & -\cos\theta_{n+1}\sin\theta_{n+1} & a_{n+1}\sin\theta_{n+1} \\ 0 & \sin\alpha_{n+1} & \cos\alpha_{n+1} & d_{n+1} \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

A parameter table keeps track for each link, the values of θ, d, a and α.

Starting from the base of the robot, we can built the transformation that defines the kinematics:

$$A = A_1 A_2 \ldots A_n$$

D-H Two Link Example

Link	θ	d	a	α
1	θ_1	0	a_1	0
2	θ_2	0	a_2	0

$$A_1 = \begin{pmatrix} \cos\theta_1 & -\sin\theta_1 & 0 & a_1\cos\theta_1 \\ \sin\theta_1 & \cos\theta_1 & 0 & a_1\sin\theta_1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$$A_2 = \begin{pmatrix} \cos\theta_2 & -\sin\theta_2 & 0 & a_2\cos\theta_2 \\ \sin\theta_2 & \cos\theta_2 & 0 & a_2\sin\theta_2 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Example

So,

$$A = A_1 A_2 = \begin{pmatrix} \cos(\theta_1 + \theta_2) & -\sin(\theta_1 + \theta_2) & 0 & a_2 \cos(\theta_1 + \theta_2) + a_1 \cos\theta_1 \\ \sin(\theta_1 + \theta_2) & \cos(\theta_1 + \theta_2) & 0 & a_2 \sin(\theta_1 + \theta_2) + a_1 \sin\theta_1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Inverse Kinematics

How can we use this technology to solve the inverse kinematics problem?

$$T^{-1} = T_0^{-1} T_1^{-1} \dots T_{n-1}^{-1} T_n^{-1}$$

In each matrix one can solve algebraically for θ_i in terms of the orientation and displacement vectors.

What does this look like for the two link manipulator?