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

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:

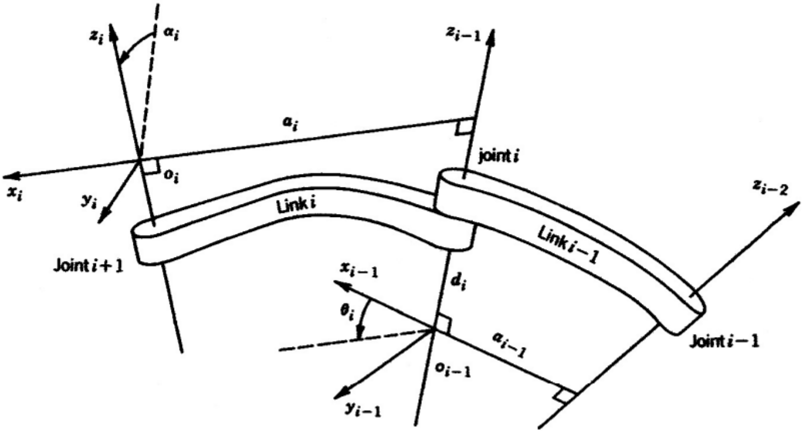
$$P = P_0 P_1 \ldots P_{n-1} P_n$$

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

$A_{n+1} =$

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

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,

$A = A_1 A_2 =$

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

Example

Then we have that the transformation carries the frame to some frame description $A = F$:

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

$$= \begin{pmatrix} n_x & o_x & a_x & p_x \\ n_y & o_y & a_y & p_y \\ n_z & o_z & a_z & p_z \\ 0 & 0 & 0 & 1 \end{pmatrix} = F$$

Example

Then the location of the end effector $(x, y, z) = (p_x, p_y, p_z)$:

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} a_2 \cos(\theta_1 + \theta_2) + a_1 \cos\theta_1 \\ a_2 \sin(\theta_1 + \theta_2) + a_1 \sin\theta_1 \\ 0 \end{pmatrix}$$

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?

Inverse Kinematics for the two link example

Recall that

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

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

Thus

$$A = A_1(\theta_1)A_2(\theta_2) = \begin{pmatrix} n_x & o_x & a_x & p_x \\ n_y & o_y & a_y & p_y \\ n_z & o_z & a_z & p_z \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Inverse Kinematics for the two link example

Right multiply to decouple: $A_1 = AA_2^{-1}$

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

$$= \begin{pmatrix} n_x & o_x & a_x & p_x \\ n_y & o_y & a_y & p_y \\ n_z & o_z & a_z & p_z \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \cos\theta_2 & -\sin\theta_2 & 0 & -a_2 \\ \sin\theta_2 & \cos\theta_2 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Note that $a_1\cos\theta_1 = p_x - a_2 n_x$ and $a_1\sin\theta_1 = p_y - a_2 n_y$

Inverse Kinematics for the two link example

This provides us with

$$\theta_1 = \text{atan2}\left(\frac{p_y - a_2 n_y}{a_1}, \frac{p_x - a_2 n_x}{a_1}\right)$$

From $\cos\theta_1 = \cos\theta_2 n_x - \sin\theta_2 o_x$ and $-\sin\theta_1 = \sin\theta_2 n_x + \cos\theta_2 o_x$ we can solve for θ_2.

$$\begin{pmatrix} \cos\theta_1 \\ -\sin\theta_1 \end{pmatrix} = \begin{pmatrix} n_x & -o_x \\ n_x & o_x \end{pmatrix} \begin{pmatrix} \cos\theta_2 \\ \sin\theta_2 \end{pmatrix}$$

$$\begin{pmatrix} \cos\theta_2 \\ \sin\theta_2 \end{pmatrix} = \frac{1}{2 n_x o_x} \begin{pmatrix} o_x & o_x \\ -n_x & n_x \end{pmatrix} \begin{pmatrix} \cos\theta_1 \\ -\sin\theta_1 \end{pmatrix}$$

So ... $\theta_2 = \text{atan2}\left(o_x(\cos\theta_1 - \sin\theta_1), -n_x(\cos\theta_1 + \sin\theta_1)\right)$

Two Link Example

There is a problem. The two link example has two degrees of freedom.

The assumption here is that you have four variables to input (four degrees of freedom): p_x, p_y, n_x, n_y.

You may not know n_x, n_y.[1] For general systems this approach will succeed if you have enough degrees of freedom in your robot.

[1] We will address the specific situation in a few slides.

Inverse Kinematics

The general approach is to form matrix A analytically and set to final pose matrix.

Then by applying inverses A_k^{-1}, examine intermediate results looking for terms which provide one of the angle variables: θ_j.

Producing actual robot motion means moving the end effector along some path $(x(t), y(t), z(t))$.

Inverse Kinematics

One really wants

$$\big(\theta_1(t), ..., \theta_n(t)\big) = f^{-1}(p(t), n(t), o(t), a(t))$$

There is no reason to expect that there exists a solution, that you can find the solution, or that the solution is unique.

Kinematic equations are derived by the developer of the robot. Inverse kinematic formulas are derived in an "ad hoc" manner.

Inverse Kinematics for Curves

How?

$$p(t) \rightarrow \left(\theta_1(t), ..., \theta_n(t)\right)$$

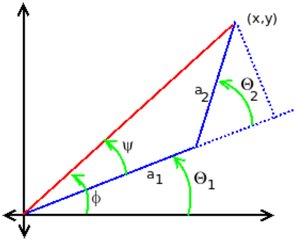
Assume that you have $(\theta_1, ..., \theta_n) = f(p, n, o, a)$.

For each t, solve

$$\begin{bmatrix} \theta_{1k} \\ \theta_{2k} \\ \vdots \\ \theta_{nk} \end{bmatrix} = \begin{bmatrix} \theta_1(t_k) \\ \theta_2(t_k) \\ \vdots \\ \theta_{nk} \end{bmatrix} = \begin{bmatrix} f_1(p(t_k), n(t_k), o(t_k), a(t_k)) \\ f_2(p(t_k), n(t_k), o(t_k), a(t_k)) \\ \vdots \\ f_n(p(t_k), n(t_k), o(t_k), a(t_k)) \end{bmatrix}$$

Two Link Example

Recall the links and angles:



The forward kinematics are:

$$\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} a_2 \cos(\theta_1 + \theta_2) + a_1 \cos\theta_1 \\ a_2 \sin(\theta_1 + \theta_2) + a_1 \sin\theta_1 \end{pmatrix}$$

Two Link Example

Using a little trig (law of cosines):

$$x^2 + y^2 = a_1^2 + a_2^2 - 2a_1 a_2 \cos(\pi - \theta_2)$$

Solve for cos

$$\cos(\theta_2) = \frac{x^2 + y^2 - a_1^2 - a_2^2}{2a_1 a_2} \equiv D$$

Using a trig formula:

$$\sin(\theta_2) = \pm\sqrt{1 - D^2}$$

Dividing the sin and cos expressions to get tan and then inverting:

$$\theta_2 = \tan^{-1} \frac{\pm\sqrt{1 - D^2}}{D}$$

The tangent form has the +/- and gives the elbow up and elbow down solutions.

Two Link Example

From the diagram, we have

$$\theta_1 = \phi - \psi = \tan^{-1}\frac{y}{x} - \psi$$

and

$$\psi = \tan^{-1}\frac{a_2 \sin\theta_2}{a_1 + a_2\cos\theta_2}.$$

And so we now have the solution

$$\theta_1 = \tan^{-1}\frac{y}{x} - \tan^{-1}\frac{a_2 \sin\theta_2}{a_1 + a_2\cos\theta_2}$$

Summary

Given

$$x = a_2 \cos(\theta_1 + \theta_2) + a_1 \cos \theta_1$$

$$y = a_2 \sin(\theta_1 + \theta_2) + a_1 \sin \theta_1$$

The solutions are

$$D = \frac{x^2 + y^2 - a_1^2 - a_2^2}{2a_1 a_2}$$

$$\theta_2 = \tan^{-1} \frac{\pm\sqrt{1 - D^2}}{D}$$

$$\theta_1 = \tan^{-1} \frac{y}{x} - \tan^{-1} \frac{a_2 \sin \theta_2}{a_1 + a_2 \cos \theta_2}$$

Summary - example

Let $a_1 = 15,\ a_2 = 10,\ x = 10,\ y = 8$.

Find θ_1 and θ_2:

1. $D = (10^2 + 8^2 - 15^2 - 10^2)/(2 * 15 * 10) = -0.53667$
2. $\theta_2 = \tan^{-1}(\sqrt{1 - (-0.53667)^2}/(-0.53667)) \approx -2.137278$
3. $\theta_1 = \tan^{-1}(8/10) - \tan^{-1}[(10\sin(-2.137278))/(15 + 10\cos(-2.137278))] \approx 1.394087$

Check

$x = 10 * \cos(1.394087 - 2.137278) + 15 * \cos(1.394087) = 10.000$

$y = 10 * \sin(1.394087 - 2.137278) + 15 * \sin(1.394087) = 8.000$

Summary - example

```
int main()
{
  double x,y, t1, t2, d, a1, a2, x1, y1;
  a1 = 15, a2 =10;
  x = 10, y = 8;

  d = (x*x+y*y-a1*a1-a2*a2)/(2*a1*a2);
  t2 = atan2(-sqrt(1.0-d*d),d);
  t1 = atan2(y,x) - atan2(a2*sin(t2),a1+a2*cos(t2));

  printf("D = %f, t1 = %f, t2 = %f\n", d, t1, t2);

  x1 = a2*cos(t1+t2) + a1*cos(t1);
  y1 = a2*sin(t1+t2) + a1*sin(t1);

  printf("x1 = %f, y1 = %f\n", x1, y1);

}
```