Kinematic Redundancy

Glen Henshaw
Craig Carignan

October 28, 2023

1 Formal Derivation of the Moore–Penrose Pseudo–Inverse

Last time we discussed how using the Moore-Penrose pseudo–inverse was a way to bypass the difficulty with non–square Jacobians, and that it had some nice properties that made it particularly suitable as the basis for an inverse kinematics scheme. But we threw it up on the board as a magic formula. Now we're going to derive it, because understanding how to derive it will let you derive your own customized inverse kinematics schemes.

The fundamental approach is to treat the velocity–space kinematics problem as an optimization problem (sound familiar?):

$$\min \|\Delta \underline{q}\|_2^2 \quad \text{given} \quad \Delta \underline{x} = J(\underline{q})\Delta \underline{q}$$

We'll use Lagrangian multipliers, so we'll formulate the problem as follows:

$$C = \frac{1}{2}\Delta \underline{q}^T \Delta \underline{q} + \underline{\lambda}^T(\Delta \underline{x} - J\Delta \underline{q})$$

to minimize this we must take the partial with respect to both $\Delta \underline{q}$ and $\Delta \underline{\lambda}$, set both equations to zero, and solve:

$$\frac{\partial C}{\partial \Delta \underline{q}} = \underline{0}^T = \Delta \underline{q}^T - \underline{\lambda}^T J \Rightarrow \Delta \underline{q} = J^T \underline{\lambda} \tag{1}$$

and

$$\frac{\partial C}{\partial \Delta \underline{\lambda}} = \underline{0}^T = \Delta \underline{x}^T - \Delta \underline{q}^T J^T \Rightarrow \Delta \underline{x} = J\Delta \underline{q} \tag{2}$$

Substituting 1 into 2 gives

$$\Delta \underline{x} = JJ^T \underline{\lambda} \Rightarrow \underline{\lambda} = (JJ^T)^{-1}\Delta \underline{x}$$

and substituting this into 1 gives

$$\Delta \underline{q} = J^T(JJ^T)^{-1}\Delta \underline{x} \triangleq J^\dagger \Delta \underline{x} \tag{3}$$

which gives us the result we want.

Note that you can change the initial optimization problem and derive different pseudo–inverses. For instance, you can add a term that penalizes distance from the arm to an object in the environment. Assume that $D(\underline{q} + \Delta\underline{q}\Delta t)$ is a positive, scalar function that has a "hat" shape with its maximum at a distance of zero and asymptotically approaches (or actually reaches) zero as the distance from the object increases. We can them write the following minimization problem:

$$\min \|\Delta\underline{q}\|_2^2 + \gamma D(\underline{q} + \Delta\underline{q}\Delta t) \quad \text{given} \quad \Delta\underline{x} = J(\underline{q})\Delta\underline{q}$$

We proceed as above:

$$C = \frac{1}{2}\Delta\underline{q}^T\Delta\underline{q} + \gamma D(\underline{q} + \Delta\underline{q}\Delta t) + \underline{\lambda}^T(\Delta\underline{x} - J\Delta\underline{q})$$

Take the partial with respect to both $\Delta\underline{q}$ and $\Delta\underline{\lambda}$, set both equations to zero, and solve:

$$\frac{\partial C}{\partial \Delta\underline{q}} = \underline{0}^T = \Delta\underline{q}^T + \gamma\frac{\partial D}{\partial \Delta\underline{q}}^T\Delta t - \underline{\lambda}^T J \Rightarrow \Delta\underline{q} = J^T\underline{\lambda} - \gamma\frac{\partial D}{\partial \Delta\underline{q}}^T\Delta t \qquad (4)$$

and

$$\frac{\partial C}{\partial \Delta\underline{\lambda}} = \underline{0}^T = \Delta\underline{x}^T - \Delta\underline{q}^T J^T \Rightarrow \Delta\underline{x} = J\Delta\underline{q} \qquad (5)$$

Substituting 4 into 5 gives

$$\begin{aligned}
\Delta\underline{x} &= J\left[J^T\underline{\lambda} - \gamma\frac{\partial D}{\partial \Delta\underline{q}}^T\Delta t\right] \\
\Rightarrow JJ^T\underline{\lambda} &= \Delta\underline{x} + \gamma J\frac{\partial D}{\partial \Delta\underline{q}}^T\Delta t \\
\Rightarrow \underline{\lambda} &= (JJ^T)^{-1}\left[\Delta\underline{x} + \gamma J\frac{\partial D}{\partial \Delta\underline{q}}^T\Delta t\right]
\end{aligned}$$

and substituting this into 4 gives

$$\begin{aligned}
\Delta\underline{q} &= J^T(JJ^T)^{-1}\left[\Delta\underline{x} + \gamma J\frac{\partial D}{\partial \Delta\underline{q}}^T\Delta t\right] - \gamma\frac{\partial D}{\partial \Delta\underline{q}}^T\Delta t \\
&= J^\dagger\Delta\underline{x} + \gamma J^\dagger J\frac{\partial D}{\partial \Delta\underline{q}}^T\Delta t - \gamma\frac{\partial D}{\partial \Delta\underline{q}}^T\Delta t \\
&= J^\dagger\Delta\underline{x} + \left[J^\dagger J - I\right]\frac{\partial D}{\partial \Delta\underline{q}}^T\Delta t \qquad (6)
\end{aligned}$$

Notice that this result is identical to the one above except for the addition of the strange partial derivative of the distance function. It turns out that $\left[J^\dagger J - I\right]$

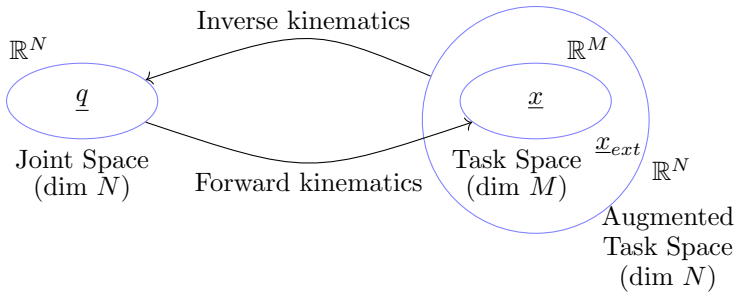
is a projection into the nullspace of J, i.e. we can choose any vector $\underline{r}$ and multiply it by the projection and it will not change the result:

$$\Delta x = J(\underline{q})\Delta\underline{q} = J\left(\Delta\underline{q} + \left[J^{\dagger}J - I\right]r\right)$$

In practical terms, what we end up doing here is simultaneously fulfilling the tracking constraint $\Delta\underline{x} = J(\underline{q})\Delta\underline{q}$ and using whatever redundant degrees of freedom our arm has to move along the gradient away from the obstacle *at the same time*. Recall our discussion last time about **self motion**: this technique uses the self motion of the robot arm to move away from a potential collision while maintaining end effector tracking. This may be an advantage or a disadvantage: the advantage is that you can guarantee that the end effector is exactly where you want it. The disadvantage is that if you command the end effector itself to collide with the environment, no amount of self motion can prevent a collision.

2 Augmented Jacobian Approach

Sometimes, we don't want our inverse kinematics routine to implicitly control the self motion of the arm; instead, we may want to control it directly. We can do this through augmenting the Jacobian. It can be augmented until it contains $\mathbb{N} - \mathbb{M}$ extra rows, making it square.



The basic idea is that we are going to increase the dimensionality of "task space" to include not only end effector trajectories but trajectories in terms of other parts of our arm. If our original Jacobian relationship is

$$\Delta\underline{x} = J_{\mathbb{M}\times\mathbb{N}}(\underline{q})\Delta\underline{q}$$

where $\dim(\underline{x}) = \mathbb{M}$ and $\dim(\underline{q}) = \mathbb{N}$ then we're going to "extend" the Jacobian as follows:

$$\underbrace{\begin{bmatrix}\Delta\underline{x}\\ \Delta\underline{x}_A\end{bmatrix}}_{\Delta\underline{x}_{ext}} = \underbrace{\begin{bmatrix}J\\ J_A\end{bmatrix}}_{J_{ext}}\Delta\underline{q}$$

and our new Jacobian equation will be written as

$$\begin{aligned}\Delta\underline{x}_{ext} &= J_{ext}\Delta\underline{q}\\ \Delta\underline{q} &= J_{ext}^{-1}\Delta\underline{x}_{ext}\end{aligned}$$

Note that we no longer need a pseudo–inverse because our extended Jacobian is square.

2.1 7 DOF Anthropomorphic Arm

Here's an example of how this works. This is a 7 DOF "anthropomorphic" arm, which means it has a three–axis shoulder, a pitch elbow, and a three–axis wrist. We want to control the motion of the end effector independently of the motion of the elbow's plane.

Here's a diagram:

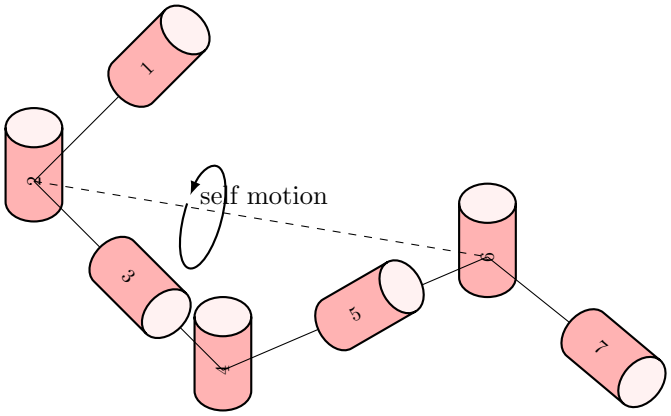
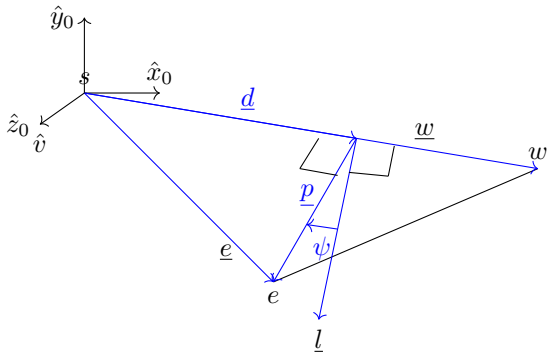


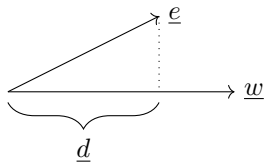
Figure 1: Diagram of the 7 DOF anthropomorphic arm

The self motion is an "orbit" of the elbow about the line $\overrightarrow{sw}$. We need to characterize the elbow's orbit dynamics. Let's assign a base frame and define some helpful vectors:



Define

$$\begin{aligned}
\underline{w} &= \overrightarrow{sw} = \underline{p}_w - \underline{p}_r \\
\underline{e} &= \overrightarrow{se} = \underline{p}_e - \underline{p}_s \\
\hat{x} & \quad \text{is reference vector in base frame} \\
\underline{w} \times \hat{v} & \quad \text{defines the reference plane} \\
\underline{l} & \quad \text{is perpendicular to } \underline{w} \text{ in reference plane} \\
\underline{d} &= \hat{w}(\hat{w}^T \underline{e}) \\
& \quad \underline{d} \text{ is the projection of } \underline{e} \text{ onto } \underline{w} :
\end{aligned}$$



Then

$$\begin{aligned}
\underline{p} &= \underline{e} - \underline{d} = (I - \hat{w}\hat{w}^T)\underline{e} \\
\underline{l} &= (\underline{w} \times \underline{\hat{v}}) \times \underline{w}
\end{aligned}$$

ϕ is the angle between $\underline{l}$ and $\underline{p}$:

$$\tan(\phi) = \frac{\hat{w}^T(\underline{l} \times \underline{p})}{\underline{l} \cdot \underline{p}} = \frac{\hat{w}^T \left(|\underline{l}||\underline{p}| \sin(\phi)\hat{w} \right)}{|\underline{l}||\underline{p}| \cos(\phi)} = \frac{\sin(\phi)}{\cos(\phi)} \hat{w}^T \hat{w}$$

J_ϕ is found by taking the derivative of ϕ with respect to $\underline{q}$:

$$\dot{\phi} = J_\phi \dot{\underline{q}}$$

where

$$J_\phi = \begin{bmatrix} x & x & x & x & 0 & 0 & 0 \end{bmatrix}$$

and note that the last three entries do not depend on n_5, n_6, n_7. So the extended Jacobian is:

$$\begin{bmatrix} \dot{\underline{x}} \\ \dot{\phi} \end{bmatrix} = \begin{bmatrix} J \\ J_\phi \end{bmatrix} \dot{\underline{q}} \tag{7}$$

3 Example Problem

One very common way to use redundant degrees of freedom is to try to keep the manipulator's joints in the middle of their range of motion, so that it's less likely to hit a joint limit. Suppose we wanted to derive a pseudoinverse to make a manipulator with $f > 6$ degrees of freedom do this.

We'll define an auxiliary loss term that penalizes angular distance from the center of the robot's range of motion:

$$D = \sum_{i=1}^{n} \left(\frac{q_i - q_i^{\text{mid}}}{q_i^{\min} - q_i^{\max}} \right)^2 \quad (8)$$

$$q^{\text{mid}} \triangleq \frac{1}{2} \left(q_i^{\min} + q_i^{\max} \right) \quad (9)$$

where $q_i^{\min}$ and $q_i^{\max}$ are the minimum and maximum joint angles for joint i.

We want to use Resolved Motion Rate Control (RMRC) with a "custom" pseudoinverse that uses the extra degrees of freedom to minimize this loss term. What is the RMRC Jacobian relationship for this term — in other words, what is $\Delta \underline{q}$ with respect to $\Delta \underline{x}$?

3.1 Answer

Recall that we derived a general formula to do this in Lecture 12 (specifically, equation 6):

$$\Delta \underline{q} = J^{\dagger} \Delta \underline{x} + \left[J^{\dagger} J - I \right] \frac{\partial D}{\partial \Delta \underline{q}}^{T} \Delta t$$

So all we have to do is take the partial of D with respect to each of the joint angles q_i:

$$\frac{\partial D}{\partial \Delta \underline{q}} = 2 \begin{bmatrix} \frac{q_0 - n_0^{\text{mid}}}{q_0^{\min} - q_0^{\max}} & 0 & \cdots & 0 \\ 0 & \frac{q_1 - q_1^{\text{mid}}}{q_1^{\min} - q_1^{\max}} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \frac{q_f - n_f^{\text{mid}}}{q_f^{\min} - q_f^{\max}} \end{bmatrix} \triangleq G$$

Noting that is diagonal, $D = D^T$, and therefore the full RMRC algorithm is:

$$\Delta \underline{q} = J^{\dagger} \Delta \underline{x} + \left[J^{\dagger} J - I \right] G \ \Delta t$$

Citation: Hanai, Aaron, Doctoral dissertation, "A Unified Autonomus Underwater Vehicle–Manipulator System", University of Hawai'i–Manoa, 2010, pg. 12