Newton-Euler First-Moment Gravity Compensation

Christopher Dellin

Friday, September 12, 2008

Contents

1 Introduction

This document briefly describes a method for gravity compensation applicable to any rigid-body robotic manipulator in an open kinematic chain (i.e. only one connection to ground). This method is designed to operate on the Barrett WAM™, an advanced torque-controlled robot.

The method consists of two steps. First, in the **calibration** step, the robot is made to hold a number of distinct poses, while torque measurements at each of the joints are taken. From these torque measurements, a vector μ is determined for each link j, using a linear regression. This vector μ_j, the *cumulative first moment of the mass*, has units of $\{\text{mass} \cdot \text{length}\}$, and represents the sum of link j's mass moment and the mass moments of all subsequent links. Expressed in joint coordinates, the vectors μ_j are pose-independent.

Once the μ vectors for the robot's mass configuration are determined, it is simple to derive the necessary torques for each joint, starting from the last link and recursively moving to

the first one. Thus, in the **compensation** scheme, torques can be provided to each joint to gravity-compensate the robot, relative to whatever mass configuration was calibrated.

For this step, it is assumed that the robot is capable of maintaining a fixed position; for this purpose, the WAM™uses a set of simple PID controllers in joint-space.

2 Newton-Euler Formulation

2.1 Conventions

We will use the following conventions in this document, which are based on the Denavit-Hartenberg conventions.

The robot is made up of n moving rigid links, numbered $j = 1, 2, ...n$. Rigidly attached to the end of each link is an origin frame, where the z axis of frame j is the axis of rotation for the joint between link j and link $j+1$. The world frame is labeled frame 0, and for now, it is assumed that frame 0 is an inertial frame.

m_j	=	the mass of link j
r_{mj}	=	the location of the joint's point of rotation
r_{cj}	=	the location of the center of mass
g^j	=	the gravity vector in frame j
f_j	=	the force from link $j-1$ to link j
f_{j+1}	=	the force from link j to link $j+1$
$\vec{\tau}_j$	=	the torque from link $j-1$ to link j
$\vec{\tau}_{j+1}$	=	the torque from link j to link $j+1$
a_{cj}	=	the COM acceleration in frame j
α_j	=	the COM ang acceleration in frame j
ω_j	=	the COM ang velocity in frame j
F^j	=	an arbitrary external force in frame j
T^j	=	an arbitrary external torque in frame j

2.2 Generic Equations

From Spong, pp 276-277, we have the following equations of motion for a rigid link:

$$f_j - \mathrm{R}^j_{j+1} f_{j+1} + m_j g^j + F^j = m_j a^j_{cj} \tag{1}$$

$$\vec{\tau}_j - \mathrm{R}^j_{j+1}\vec{\tau}_{j+1} + f_j \times (r_{cj} - r_{mj}) - \left(\mathrm{R}^j_{j+1} f_{j+1}\right) \times r_{cj} + T^j = I_j \alpha_j + \omega_j \times (I_j \omega_j) \tag{2}$$

Note that a number of terms have been adjusted from Spong to fit into the Denavit-Hartenberg conventions.

2.3 The Static Case

Next, we make the assumption that (a) there are no external forces or torques (F and T are zero) and (b) the system is static (a, α, and ω are zero). This leaves us with:

$$f_j = -m_j g^j + \mathrm{R}^j_{j+1} f_{j+1} \tag{3}$$

$$\vec{\tau}_j - \mathrm{R}^j_{j+1}\tau_{j+1} = -f_j \times (r_{cj} - r_{mj}) + \left(\mathrm{R}^j_{j+1} f_{j+1}\right) \times r_{cj} \tag{4}$$

First, we note that, for the last link $j = n$, both f_{j+1} and τ_{j+1} are zero. Thus:

$$f_n = -m_n g^n \tag{5}$$

Then, by substitution, we have:

$$\begin{aligned} f_{n-1} &= -m_{n-1} g^{n-1} + \mathrm{R}^{n-1}_n \left(-m_n g^n\right) && (6)\\ &= -m_{n-1} g^{n-1} - m_n g^{n-1} && (7)\\ &= -\left(m_{n-1} + m_n\right) g^{n-1} && (8) \end{aligned}$$

Thus, we find an explicit expression for the force f_j on each link:

$$f_j = -M_j\, g^j \;\text{ for }\; M_j = \sum_{k=j}^{n} m_k \tag{9}$$

(of course, this seems obvious in retrospect, but it's nice to derive it explicitly as well ...)

Next, we turn to the torque equation.

$$\begin{aligned} \vec{\tau}_j - \mathrm{R}^j_{j+1}\vec{\tau}_{j+1} &= -\left(-M_j\, g^j\right) \times (r_{cj} - r_{mj}) + \left(\mathrm{R}^j_{j+1}\left(-M_{j+1}\, g^{j+1}\right)\right) \times r_{cj} && (10)\\ &= g^j \times \left(M_j\,[r_{cj} - r_{mj}]\right) - g^j \times \left(M_{j+1}\, r_{cj}\right) && (11)\\ &= g^j \times \left(M_j\, r_{cj} - M_j\, r_{mj} - M_{j+1}\, r_{cj}\right) && (12)\\ &= g^j \times \left(m_j\, r_{cj} - M_j\, r_{mj}\right) && (13) \end{aligned}$$

Thus, we arrive at the following static governing equation:

$$\vec{\tau}_j - \mathrm{R}^j_{j+1}\vec{\tau}_{j+1} = g^j \times \mu_j \;\text{ for }\; \mu_j = m_j\, r_{cj} - M_j\, r_{mj} \tag{14}$$

3 Calibration - Coupled Matrix

3.1 Denavit-Hartenberg Representation

Next, we take advantage of our Denavit-Hartenberg frame formulation to write each torque:

$$\vec{\tau}_j = \mathrm{R}_{j-1}^{j} \begin{bmatrix} a_j \\ b_j \\ \tau_j \end{bmatrix} \text{ and } [\,\vec{a}\,]_\times = \begin{bmatrix} 0 & -a_3 & a_2 \\ a_3 & 0 & -a_1 \\ -a_2 & a_1 & 0 \end{bmatrix} \tag{15}$$

$$\mathrm{R}_{j-1}^{j} \begin{bmatrix} a_j \\ b_j \\ \tau_j \end{bmatrix} - \begin{bmatrix} a_{j+1} \\ b_{j+1} \\ \tau_{j+1} \end{bmatrix} = \left[g^j\right]_\times \mu_j \tag{16}$$

3.2 Matrix Formulation - Single Link j

Next, we get things in matrix language so we can design a matrix solution. We make a couple of useful definitions to separate the rotation matrix for multiplication with known and unknown quantities:

$$L = \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 0 \end{bmatrix} \text{ and } Z = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \text{ and } c_j = \begin{bmatrix} a_j \\ b_j \end{bmatrix} \tag{17}$$

$$\mathrm{R}_{j-1}^{j} L c_j + \mathrm{R}_{j-1}^{j} Z \tau_j - L c_{j+1} - Z \tau_{j+1} = \left[g^j\right]_\times \begin{bmatrix} \mu_j \end{bmatrix} \tag{18}$$

$$\mathrm{R}_{j-1}^{j} Z \tau_j - Z \tau_{j+1} = \left[g^j\right]_\times \begin{bmatrix} \mu_j \end{bmatrix} - \mathrm{R}_{j-1}^{j} L c_j + L c_{j+1} \tag{19}$$

$$\begin{bmatrix} \mathrm{R}_{j-1}^{j} Z \tau_j - Z \tau_{j+1} \end{bmatrix} = \begin{bmatrix} [g^j]_\times & \begin{bmatrix} -\mathrm{R}_{j-1}^{j} L & L \end{bmatrix} \end{bmatrix} \begin{bmatrix} \mu_j \\ c_j \\ c_{j+1} \end{bmatrix} \tag{20}$$

We see that this is now in the form $\vec{y} = \mathrm{GT}\ \vec{x}$, with $\vec{x}$ the parameter matrix to be determined by regression. Note that the vector μ is pose-independent, while the torques a and b are different for each pose.

3.3 Matrix Formulation - Coupled Links, Single Pose

We now determine a sample GT matrix for a manipulator with $n = 4$ links, going through 5 poses A, B, C, D, and E.

For pose A, we form the following G matrix:

$$ {}^{A}G = \begin{bmatrix} \left[{}^{A}g^{1}\right]_{\times} & 0 & 0 & 0 \\ 0 & \left[{}^{A}g^{2}\right]_{\times} & 0 & 0 \\ 0 & 0 & \left[{}^{A}g^{3}\right]_{\times} & 0 \\ 0 & 0 & 0 & \left[{}^{A}g^{4}\right]_{\times} \end{bmatrix} \tag{21} $$

For pose A, we also form the following T matrix:

$$ {}^{A}T = \begin{bmatrix} -{}^{A}\mathrm{R}_{0}^{1}L & L & 0 & 0 \\ 0 & -{}^{A}\mathrm{R}_{1}^{2}L & L & 0 \\ 0 & 0 & -{}^{A}\mathrm{R}_{2}^{3}L & L \\ 0 & 0 & 0 & -{}^{A}\mathrm{R}_{3}^{4}L \end{bmatrix} \tag{22} $$

... along with a known torque vector y and unknown torque parameter vector p:

$$ {}^{A}y = \begin{bmatrix} {}^{A}\mathrm{R}_{0}^{1}Z^{A}\tau_{1} - Z^{A}\tau_{2} \\ {}^{A}\mathrm{R}_{1}^{2}Z^{A}\tau_{2} - Z^{A}\tau_{3} \\ {}^{A}\mathrm{R}_{2}^{3}Z^{A}\tau_{3} - Z^{A}\tau_{4} \\ {}^{A}\mathrm{R}_{3}^{4}Z^{A}\tau_{4} - 0 \end{bmatrix} \text{ for } {}^{A}p = \begin{bmatrix} {}^{A}c_{1} \\ {}^{A}c_{2} \\ {}^{A}c_{3} \\ {}^{A}c_{4} \end{bmatrix} \tag{23} $$

For pose A, then, we have:

$$ {}^{A}y = \begin{bmatrix} {}^{A}G & {}^{A}T \end{bmatrix} \begin{bmatrix} U \\ {}^{A}p \end{bmatrix} \tag{24} $$

3.4 Matrix Formulation - Multiple Poses

However, this parameterization is ambiguous, since the known torque vector y is of length $3n$, while the parameter vector $[U, p]^T$ is of length $3n + 2n = 5n$. This reflects the fact that one pose is insufficient to uniquely determine the parameterization. However, since the parameters μ_j are independent of pose, the solution can be found with a sufficient number of poses k.

For multiple poses, we form the GT matrix as follows:

$$ \begin{bmatrix} {}^{A}y \\ {}^{B}y \\ {}^{C}y \\ {}^{D}y \\ {}^{E}y \end{bmatrix} = \begin{bmatrix} {}^{A}G & {}^{A}T & 0 & 0 & 0 & 0 \\ {}^{B}G & 0 & {}^{B}T & 0 & 0 & 0 \\ {}^{C}G & 0 & 0 & {}^{C}T & 0 & 0 \\ {}^{D}G & 0 & 0 & 0 & {}^{D}T & 0 \\ {}^{E}G & 0 & 0 & 0 & 0 & {}^{E}T \end{bmatrix} \begin{bmatrix} U \\ {}^{A}p \\ {}^{B}p \\ {}^{C}p \\ {}^{D}p \\ {}^{E}p \end{bmatrix} \tag{25} $$

Thus, for k poses, the known torque vector y is of length $\mathrm{len}\{y\} = 3nk$, while the parameter vector p is of length $\mathrm{len}\{p\} = 3n + 2nk$. Thus, you need at least $k = 3$ poses for a complete parameterization, and you can use more poses and a regression analysis to determine the best-fit parameters.

4 Calibration - Iterative Algorithm

While the coupled matrix method described above works in theory, it does not yield very precise solutions under linear regression, especially for the end links. There may be a way to weight the regression in some way, but attempts so far have been unsuccessful. Therefore, we take advantage of the directional coupling of the pose matrices G and T and offer a different solution method.

4.1 Matrix Forumulation

We start with the single link, single pose equation (Eqn. 19), and rewrite:

$$\mathrm{R}_{j-1}^{j} Z\tau_j - Z\tau_{j+1} - Lc_{j+1} = \left[g^j\right]_\times \begin{bmatrix} \mu_j \end{bmatrix} - \mathrm{R}_{j-1}^{j} Lc_j \tag{26}$$

We then extend this equation across multiple poses, for a single link j:

$$\begin{bmatrix} {}^A\mathrm{R}_{j-1}^{j} Z^A\tau_j - Z^A\tau_{j+1} \\ {}^B\mathrm{R}_{j-1}^{j} Z^B\tau_j - Z^B\tau_{j+1} \\ {}^C\mathrm{R}_{j-1}^{j} Z^C\tau_j - Z^C\tau_{j+1} \\ \vdots \end{bmatrix} - \begin{bmatrix} 0 & L & 0 & 0 & \dots \\ 0 & 0 & L & 0 & \dots \\ 0 & 0 & 0 & L & \dots \\ \vdots & \vdots & \vdots & \vdots & \ddots \end{bmatrix} \begin{bmatrix} \mu_{j+1} \\ {}^A c_{j+1} \\ {}^B c_{j+1} \\ {}^C c_{j+1} \\ \vdots \end{bmatrix} \tag{27}$$

$$= \begin{bmatrix} \left[{}^A g^j\right]_\times & -{}^A\mathrm{R}_{j-1}^{j} L & 0 & 0 & \dots \\ \left[{}^B g^j\right]_\times & 0 & -{}^B\mathrm{R}_{j-1}^{j} L & 0 & \dots \\ \left[{}^C g^j\right]_\times & 0 & 0 & -{}^C\mathrm{R}_{j-1}^{j} L & \dots \\ \vdots & \vdots & \vdots & \vdots & \ddots \end{bmatrix} \begin{bmatrix} \mu_j \\ {}^A c_j \\ {}^B c_j \\ {}^C c_j \\ \vdots \end{bmatrix} \tag{28}$$

By solving this system for each link (starting at the robot end-point and working down to the base), we arrive at the solution vectors μ, one for each link.