The Effect of Ill-conditioned Inertia Matrix on Controlling Manipultor Robot

Yueshi Shen
Department of Systems Engineering
Australian National University
Canberra ACT 0200, Australia
yueshi@syseng.anu.edu.au

Roy Featherstone
Department of Systems Engineering
Australian National University
Canberra ACT 0200, Australia
roy@syseng.anu.edu.au

Abstract

Though proved as symmetric-positive-definite, the inertia matrix of a robot manipulator can sometimes be ill-conditioned, and in such case, the ill-conditioning will make the robot more difficult to simulate and control. In this paper, we first describe a problem that has been encountered in real experimentation of controlling 4-dof robot WAM (Whole Arm Manipulator). Then some analysis is made to show that the ill-conditioned inertia matrix causes such problem, and some ways to cope with it will be suggested at last.

1 Background

The development of control strategies for robot manipulator has attracted considerable interest, and the mathematical model of robot manipulator greatly influences the design of the controller. For robots with high gear ratios, the dynamics of the robot's mechanism is neglectable (for example, industrial robot). A very simple way to control the robot manipulator is to model it as n independent electrical motors, and control the robot in joint space with a separate controller for each joint.

However, for more advanced research-used robot manipulator, usually the dynamics of actuator is small comparing to that of rigid multi-link robot, whose mathematical model can be given as a non-linear second-order differential equation:

$$\mathbf{H}\ddot{\mathbf{q}} + \mathbf{C} + \mathbf{f} = \tau \quad (1)$$

where τ is the driving torque, $\mathbf{C}$ is the sum of Coriolis and gravitation force, $\mathbf{f}$ is the joint friction term, $\mathbf{q}$ is the joint position, and $\mathbf{H}$ is the inertia matrix that we will talk about in later sections. The common way to control the above system is to apply inverse dynamics as feedback linearization, and the linearized system then has n independent joint accelerations as its new output

Figure 1: 4-Dof WAM (Whole Arm Manipulator)

(Eq. 2) [Spong and Vidyasagar, 1989].

$$\ddot{\mathbf{q}} = \mathbf{v} \quad (2)$$

where $\mathbf{v}$ is the control signal of the new system.

2 A Practical Problem

2.1 Experimental Robot: WAM

WAM (Whole Arm Manipulator) is a 4-DOF (originally designed as 7-DOF configuration) robot manipulator (Fig. 1) with human-like kinematics (Fig. 2). The dumb bell probe mounted on the end-tip of robot is designed for later force control experiment.

With an advanced cable-drive system, WAM has no backlash, and extremely low friction [Barrett, 1998], so the joint friction in Eq. 1 could be considered as neglectable in the rigid robot model. The WAM's four joints are driven by four brushless servo-motors, which are further controlled by four current amplifiers in torque mode. In practice, we could ignore the dynamics of the actuators, because of the low gear ratio and control mechanism inside the amplifiers. However, the motor torque ripple must be modeled and integrated into the

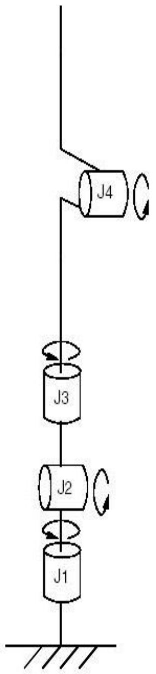

Figure 2: The Kinematic Model of WAM (The joint angles are zero in this position)

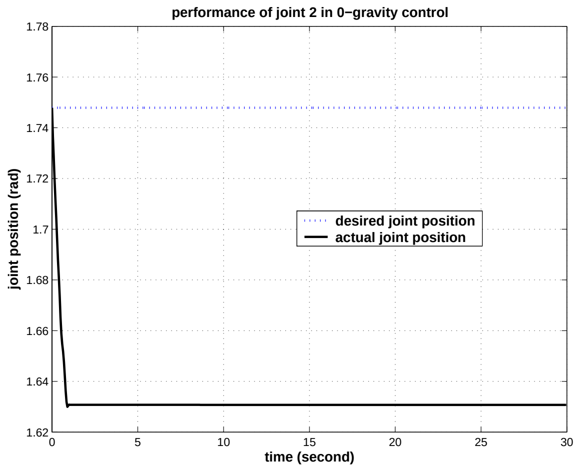

Figure 3: Performance of Joint 2 in Zero-gravity Control

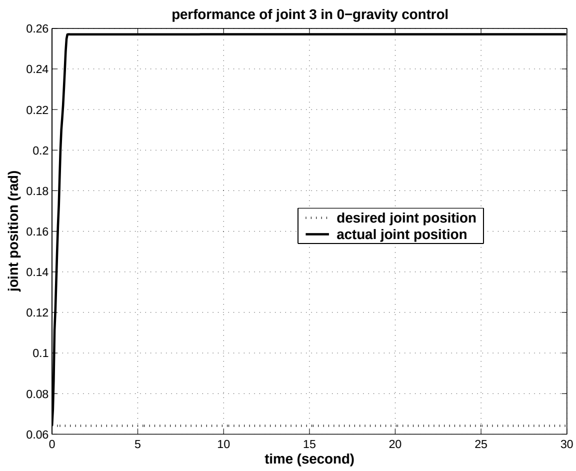

Figure 4: Performance of Joint 3 in Zero-gravity Control

system, because it can be transmitted to the joints and have a noticeable effect on the motion of the WAM. A cursory way to model the torque ripple is to consider it as a function of only shaft position [Ansar *et al.*, 2001], thus the compensation can be achieved by a feed-forward term. Actually, the complete model is a function of both motor position and command torque, because the torque ripple is caused by electromagnetic torque fluctuation (a function of load) and cogging torque (a function of rotor position) [Hanselman, 1994]. Including torque ripple, the whole dynamics model of WAM can be expressed as:

$$\mathbf{H}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}}) + \mathbf{T}(\mathbf{q}, \tau) = \tau \tag{3}$$

where $\mathbf{C}$ is the sum of Coriolis and gravitation force, and $\mathbf{T}$ is 4 by 1 motor torque ripple vector.

2.2 Zero-Gravity Control

Consider the following control law:

$$\tau = \mathbf{C} + \mathbf{T} \tag{4}$$

By applying this drive torque on WAM, we can realize the zero-gravity control, which means the robot now becomes weightless so that it can stay still at any pose and be moved around with only little external force.

The usual way of doing gravity compensation on WAM is the Lagrange approach [Ansar *et al.*, 2001], which approximates the upper and lower arm as two mass points and uses least square estimation to acquire relevant mass and center of mass data. But we implement the calculation of Coriolis-gravitation term in a different way, which is based on the rigid robot model recorded in WAM's mass property documentation [Barrett, 1998] and the Inverse Dynamics Algorithm expressed in Spatial Algebra [Featherstone, 1987]. The result of our zero-gravity control is satisfactory, and robot is able to stand still except for a little bit of drift in shoulder joints 2 and 3 in some particular poses. Fig. 3 and Fig. 4 show examples of the movement of those two joints, which are supposed to stay still from the beginning.

The reason for imperfect zero-gravity control is complicated. It could be the inaccuracy of WAM's dynamic model, or inadequate torque ripple compensation etc. Besides those factors, the most important thing we be-

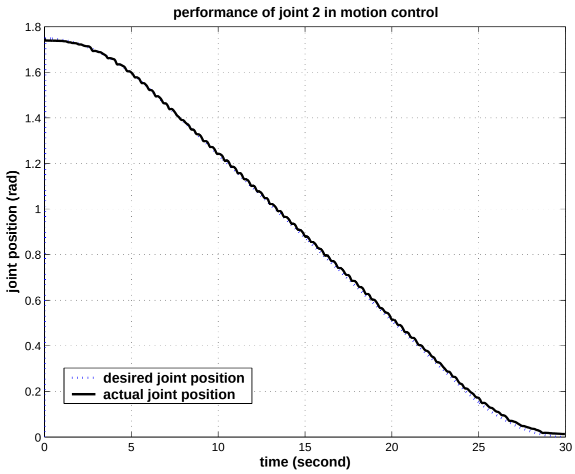

Figure 5: Performance of Joint 2 in Motion Control

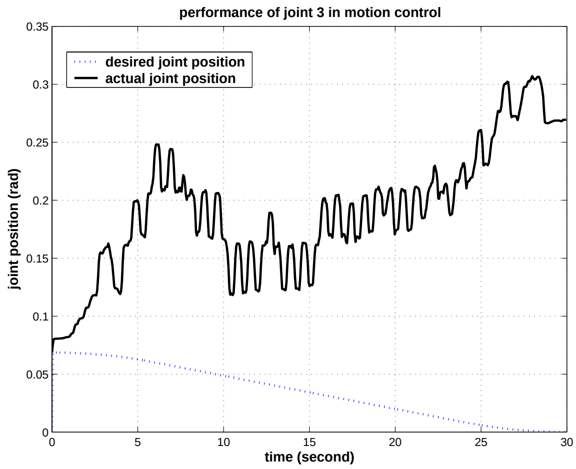

Figure 6: Performance of Joint 3 in Motion Control

lieve is that because two motors drive a cabled differential which connects joint 2 and 3, there exists coupling between those two joints. Under this circumstance, the error from any single motor will affect the actual torque applied on both joints. Furthermore, the desired drive torque for joint 2 is usually much bigger than that of joint 3 (see Fig. 2) in normal operation, so the motor error has bigger proportional effect on joint 3 than joint 2.

2.3 Motion Control with PD-Controller

Having done gravity compensation, now we are ready to do motion control for WAM by applying the control law as below (Eq. 5)

$$\tau = \mathbf{H}\ddot{\mathbf{q}} + \mathbf{C} + \mathbf{T} \tag{5}$$

where

$$\ddot{\mathbf{q}} = K_p(\mathbf{q}_d - \mathbf{q}) + K_v(\dot{\mathbf{q}}_d - \dot{\mathbf{q}}) \tag{6}$$

The PD controller is supposed to overcome the error and draw the robot to its desired position, and because we've implemented the inverse dynamics of rigid robot, it's not necessary to apply different PD-coefficients for different joints.

The experiment starts with the robot lying on a cushion, fully stretching out (as in Fig. 1), then the drive torque is applied to make the robot rise up to a vertical position. Even in such a simple task, the performance of the robot is not quite as we expect and the result is a little bit confusing: every joint can follow the desired trajectory except joint 3. Fig. 5 and Fig. 6 show the movement of joint 2 and 3 during the experiment.

Both joint 2 and 3 are WAM's shoulder joints, whose rotation axes are perpendicular and intersecting. Though driven by a cabled differential, which might cause some error, why does the closed-loop performance of these two joints differ so such? The following section will give us the explanation.

3 Analysis

3.1 Hypothesis

A well-known property of the inertia matrix is its symmetric positive definiteness [Tourassis and Neuman, 1985]. Though the inertia matrix could never be singular, it will be ill-conditioned sometimes and that will cause problem when we control the robot by using the control law described above (Eq. 5).

First let's study the property of $\mathbf{H}(\mathbf{q})$ at the time when the above experiment starts. Using the Composite-Rigid-Body method [Featherstone, 1987], we can get the value of $\mathbf{H}$ when the robot lies on the cushion as:

$$\mathbf{H} = \begin{bmatrix} 1.6008 & 0.0001 & -0.0146 & -0.0000 \\ 0.0001 & 1.4880 & 0.0024 & 0.3643 \\ -0.0146 & 0.0024 & 0.0071 & 0.0001 \\ -0.0000 & 0.3643 & 0.0001 & 0.1413 \end{bmatrix} \tag{7}$$

Applying the eigenvalue decomposition on $\mathbf{H}$ we can get the eigenvalues and eigenvectors as follows:

$$\begin{aligned} E_1 &= 1.6010 \\ E_2 &= 1.5802 \\ E_3 &= 0.0490 \\ E_4 &= 0.0069 \\ \mathbf{U}_1 &= \begin{bmatrix} -1.0000 & -0.0018 & 0.0092 & -0.0004 \end{bmatrix}^T \\ \mathbf{U}_2 &= \begin{bmatrix} 0.0019 & -0.9494 & -0.0015 & -0.2454 \end{bmatrix}^T \\ \mathbf{U}_3 &= \begin{bmatrix} -0.0001 & -0.2454 & -0.0129 & 0.9693 \end{bmatrix}^T \\ \mathbf{U}_4 &= \begin{bmatrix} 0.0092 & -0.0046 & 0.9999 & -0.0121 \end{bmatrix}^T \end{aligned} \tag{8}$$

As can be seen from the eigenvectors, $\mathbf{U}_1$ aligns approximately with joint 1, $\mathbf{U}_2$ with joint 2, $\mathbf{U}_3$ with joint 4, and $\mathbf{U}_4$ with joint 3; and their corresponding eigenvalues are S_1, S_2, S_3, S_4 respectively. So, for the same desired joint acceleration $\ddot{\mathbf{q}}$, the actual gravitation-free drive torques calculated from the inverse dynamics control law (Eq. 9) differ enormously for different joints.

$$\tau_{gravity-free} = \mathbf{H}\ddot{\mathbf{q}} \tag{9}$$

In this particular case, the gravity-free torque for joint 3 is 232.03 times smaller than gravity-free torque for joint 1.

A good index for a matrix to see if it is ill-conditioned or well-conditioned is the condition number, which can be defined as:

$$k(\mathbf{H}) = \frac{s_{max}(\mathbf{H})}{s_{min}(\mathbf{H})} \tag{10}$$

where s_{max}, s_{min} are respectively the largest and smallest singular-value of H. For a symmetric positive definite matrix, they are also eigenvalues. The bigger the condition number is, the more ill-conditioned a matrix is. It is known that the ill-conditioning will cause the solution of a system to be extremely sensitive to a small disturbance [Meyer, 2000]. In Robotics application that will lead to the loss of precision in simulation etc. In control point of view, the ill-conditioning causes different sensitivities in different directions, if those directions align with robot joints, the performance of the control law will be significantly worse on joints with high sensitivity.

The condition number k is a function of joint position $[q_1 q_2 q_3 q_4]^T$ (because the inertia matrix $\mathbf{H}$ itself is a function of robot pose). However, for WAM robot, the position of joint 4 is much more important on deciding the value of $\mathbf{H}$ comparing with the other three joints. As shown in Fig. 7, when $q_4 = 0$ (robot fully stretching out), the condition number is close to its peak value no matter what q_2 or q_3 is. In other words, the control task that we described before is actually the most difficult case for inverse dynamics control law (Eq. 5).

A more intuitive way to understand why joint 3 cannot be well controlled is because when $q_4 = 0$ the inertia along joint 3's axis is very small, so no matter how big the position/velocity error or PD-coefficient is, the actually error-correction torque applied on this joint is still small compared to the dominant torque offset, and that causes the drift as we see in Fig. 6.

3.2 Verification

In order to verify our hypothesis presented above, we did a simple experiment. The control task is almost the same as described in the previous section, but WAM is now folding its arm by 90 degrees (Fig. 8). In such a pose, the robot has a less ill-conditioned inertia matrix $\mathbf{H}$, and

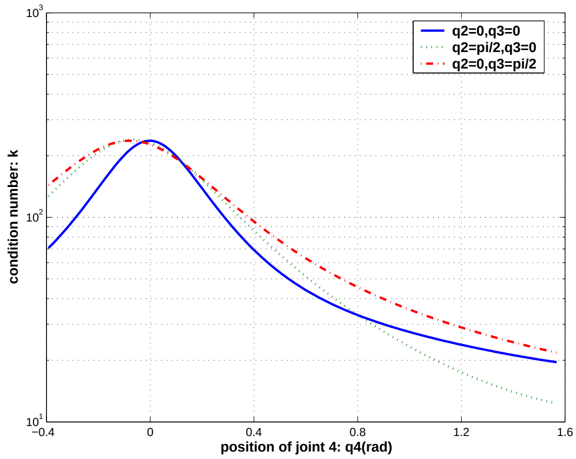

Figure 7: Change of Condition Number k Due to Position of Joint 2,3,4

Figure 8: WAM folding its arm by 90 degrees

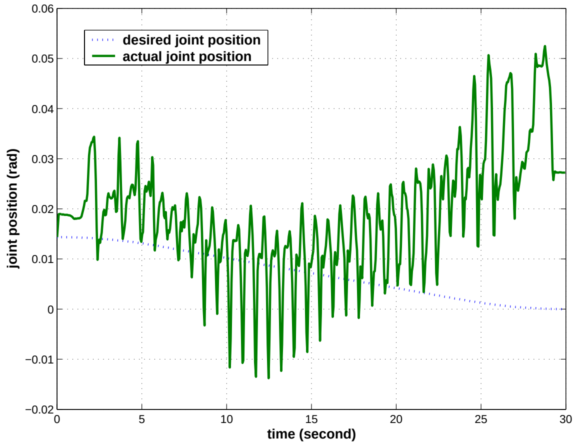

Figure 9: Performance of Joint 3 in Motion Control Using Eqs. 5 and 6

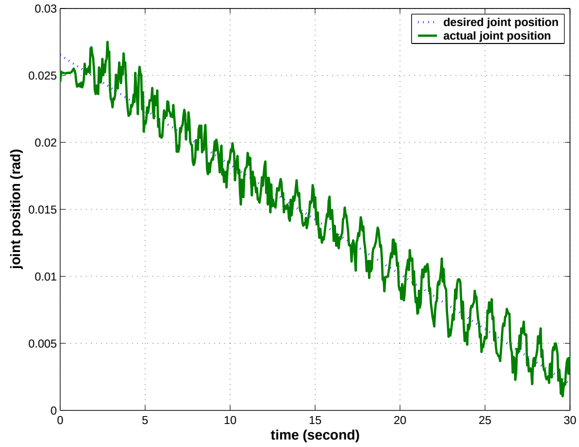

Figure 10: Performance of Joint 3 in Motion Control Using Eq. 12

the value of **H**, its eigenvalues and condition number are shown as follows:

$$\mathbf{H} = \begin{bmatrix} 1.0427 & 0.0015 & -0.2922 & 0.0020 \\ 0.0015 & 1.0321 & 0.0061 & 0.1364 \\ -0.2922 & 0.0061 & 0.1859 & 0.0000 \\ 0.0020 & 0.1364 & 0.0000 & 0.1413 \end{bmatrix}$$
$$\mathbf{E} = \begin{bmatrix} 0.0956 \\ 0.1209 \\ 1.0526 \\ 1.1329 \end{bmatrix} \qquad (11)$$
$$k = 11.8464$$

We can see that when the inertia matrix is well-conditioned (condition number 11.8464 vs. 232.0290), for same control law, the tracking performance of joint 3 becomes much better (see Fig. 6 and Fig. 9).

3.3 Suggested Solution

As we proposed above, the problem described in Section 1.3 comes from the error caused by the mechanical property of differential joint 2/3, also the ill-conditioning of robot inertia matrix. For the robotic system like that, the solution for our problem is to use a different controller design. Let's consider the PD controller with gravity compensation as follows (Eq. 12)

$$\tau = \mathbf{K}_p(\mathbf{q}_d - \mathbf{q}) + \mathbf{K}_v(\dot{\mathbf{q}}_d - \dot{\mathbf{q}}) + \mathbf{C} + \mathbf{T} \qquad (12)$$

where $\mathbf{K}_p$ and $\mathbf{K}_v$ are diagonal matrices of position and differential gains for different joints. The PD controller here directly converts joint position/velocity error to drive torque, and because the real gains for all joints are same for any robot pose, this control law won't be affected by the inertia matrix's ill-conditioning.

Fig. 10 shows the experimental result for same control task as in Section 1.1 (robot fully stretching out goes to vertical). The tracking performance for joint 3 is much better than the inverse dynamics control law.

For control law in Eq. 12 although it ignores the coupling between joints, as long as the PD-gains are chosen properly, the globally asymptotical stability will be guaranteed [Kelly, 1997], [Sage *et al.*, 1999]. However, the inverse dynamics controller should achieve better accuracy because of having complete knowledge about the robot dynamics, and the feedback linearization is useful in many applications such as motion/force hybrid control. High quality feed-forward compensation of torque ripple and Coriolis-gravitation force might be essential. Also adding an integral term might help the controller overcome the joint torque offset. We should leave this part as the future work of this paper.

4 Conclusion

This paper shows a practical problem in a motion control experiment. We show that the reason for the problem is the ill-conditioning of inertia matrix, and this hypothesis has been verified by experimental result. Also, a suggested solution for such a problem is proposed at last. For further work, it will be interesting to make the inverse dynamics work and see if it brings the improvement of performance comparing with the direct PD controller.

References

[Ansar *et al.*, 2001] Ansar, A., Rodrigues, D., Desai, J.P., Danilidis, K., Kumar, V., and Capos, M.F.M. Visual and Haptic Collaborative Tele-presence. *Computer & Graphics*, 25: 789-798, 2001.

[Barrett, 1998] Barrett Technology Inc. *User Manual, Version 1.* 1998.

[Featherstone, 1987] Featherstone, R. *Robot Dynamics Algorithms.* Boston: Kluwer Academic Publishers, 1987.

[Hanselman, 1994] Hanselman, D.C. Minimum Torque Ripple, Maximum Efficiency Excitation of Brushless Permanent-Magnet Motors. *IEEE Transactions on Industrial Electronics*, 41(3):292-300, 1994.

[Kelly, 1997] Kelly, R. PD Control with Desired Gravity Compensation of Robotic Manipulators: A Review. *The International Journal of Robotics Research*, 16(5):660-672, 1997.

[Meyer, 2000] Meyer, C.D. *Matrix Analysis and Applied Linear Algebra.* The Society for Industrial and Applied Mathematics, 2000.

[Sage *et al.*, 1999] Sage, H.G., De Mathelin, M.F., and Ostertag, E. Robust Control of Robot manipulators: A Survey. *Int. J. Control*, 72(16): 1498-1522, 1999.

[Spong and Vidyasagar, 1989] Spong, M.W. and Vidyasagar, M. *Robot Dynamics and Control.* New York: Wiley, 1989.

[Tourassis and Neuman, 1985] Tourassis V.D. and Neuman C.P. Properties and Structure of Dynamic Robot Model for Control Engineering Applications. *Mechanism & Machine Theory*, 20(1):27-40, 1985.