MAE 546 Term Paper

Tarik Tosun

Due May 15, 2012

Contents

1 Introduction

In this project, we aim to develop a robust, high-performance feedback controller to work in conjunction with a previously developed general kinematic retargeting system. Given a source pose, the system is capable of generating an optimal imitation pose on an arbitrary target platform. Computation is performed in simulation using modeled kinematic chains with no dynamics. Motion is retargeted on a frame-by-frame basis, resulting in a series of poses that the target is to assume at specified times. This nominal retargeted trajectory is then "played back" on the robot to produce actual motion [8].

In order to control a robot using such a nominal trajectory, a good dynamic controller is required. During development of the retargeter, experimental trials with robots were plagued with controller issues. During tests with the PR2 [1], a large humanoid robot developed at Willow Garage, movement appeared to suffer from serious overshoot problems due to low damping. Motion performed on the CKbot platform [4] from the UPenn ModLab proved jerky and extremely slow, as motion was matched pose-by-pose using only servo position feedback and no interpolation. If a better tracking controller were available for either of these platforms, the retargeted solutions would be of a much higher quality overall. To illustrate how such a tracking can be developed this project aims to develop a gain-scheduled LQR state feedback controller for a simple robot model in Simulink, and then test its performance under a variety of conditions using a nominal trajectory developed with the retargeted system.

1.1 Robot Model

The robot model used is a simple two-link manipulator, shown in Figure 1 below, which is able to rotate its first link about its vertical axis and pitch its second link up and down. The model is taken from *Introduction to Robotics* by P. J. McKerrow. McKerrow derives equations for the manipulator's dynamics, which lead to the fourth-order, nonlinear state-space model shown in Equations 1-4.

$$\dot{x}_1 = x_2 \tag{1}$$

$$\dot{x}_2 = \frac{1}{\cos^2(x_3)} \left(x_2 x_4 \sin 2x_3 + \frac{u_1}{m l_2^2} \right) \tag{2}$$

$$\dot{x}_3 = x_4 \tag{3}$$

$$\dot{x}_4 = -\frac{g}{l_2}\cos(x_3) - \frac{x_2}{2}\sin 2x_3 + \frac{u_2}{ml_2^2} \tag{4}$$

Where:

$$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} Angle\,of\,1^{st}\,link,\,\theta_1,\,rad \\ Angle\,of\,1^{st}\,link,\,\dot{\theta}_1,\,rad/sec \\ Angle\,of\,2^{nd}\,link,\,\theta_2,\,rad \\ Angle\,of\,2^{nd}\,link,\,\dot{\theta}_2,\,rad/sec \end{bmatrix}$$

$$\begin{bmatrix} u_1 \\ u_2 \end{bmatrix} = \begin{bmatrix} \tau_1 \\ \tau_2 \end{bmatrix}$$

The angles, θ_1 and θ_2 (positive up), are zero when they are aligned with their respective x axes, and τ_1 and τ_2 are the torques at each joint.. Inertial properties are modeled simply by a point mass, m, at the distal end of the manipulator, and the length of the second link is l2. Thus the mass and inertia of the links are neglected. Link 2 is mounted at distance d1 from the base [6]. The model parameters are:

d_1 = 1.5 m

l_2 = 1 m

m =2 kg

g =9.807 m/s^2

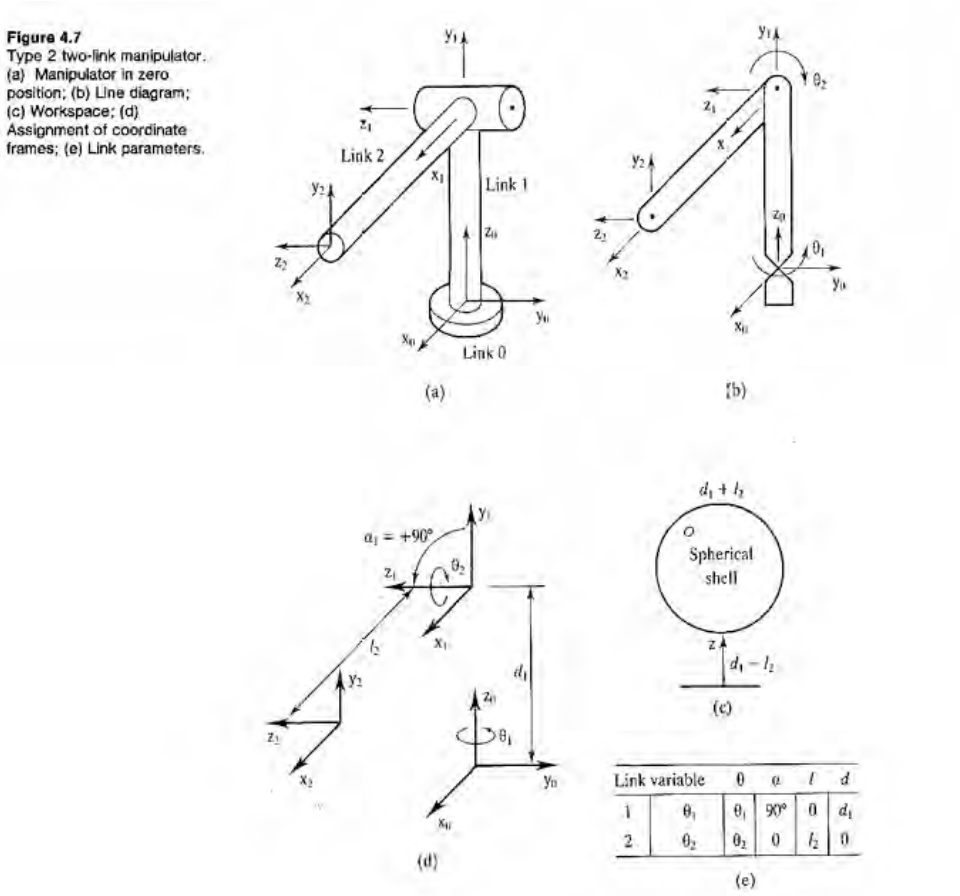


Figure 1: Robot Model, from McKerrow [3]

1.2 Controller Properties

A gain-scheduled Linear Quadratic Regulator was selected as the controller of choice for this system. LQR state-feedback controller exhibit a number of properties that make them advantageous for the purposes of this project. The LQR controller is designed to minimize the stochastic quadratic cost function:

$$J = \frac{1}{2}\left\{\int_0^t \left[\mathbf{x}^T(t)\mathbf{Q}\mathbf{x}(t) + \mathbf{u}^T(t)\mathbf{R}\mathbf{u}(t)\right] dt\right\}$$

subject to the system dynamics. In this equation, $\mathbf{Q}$ and $\mathbf{R}$ are positive-definite state and control cost weighting matrices which may be adjusted to prioritize tracking accuracy or economy of control use. The Linear Quadratic Regulator has the additional beneficial

quality of global asymptotic stability for any controllable system, regardless of the stability of the open-loop system [5].

2 Methods

A robust linear quadratic controller was designed for the system. Simulink was used to build both the robot model and controller.

2.1 LQR Design

LQR controllers require an LTI system, so an LQR controller could not be built directly from this system's nonlinear dynamics. Instead, the system was linearized at a series of operating points, and LQR gain matrices were computed at each operating point to form a gain scheduled controller.

Steady-State Operating Points [7]

Inspecting the dynamic equations of the robot (Equations 1-4), we see that the dynamics vary nonlinearly with $\dot{\theta}_1$,θ_2, and $\dot{\theta}_2$. Of these, the variation with θ_2 is the most significant, as it causes both the magnitude of the gravitational torque on the second joint and the rotational intertia properties about the first joint to vary.

To account for this, steady-state operating points were computed for different positions of the manipulator's horizontal arm. We desire the required torque as a function of joint angle to keep link 2 suspended in equilibrium at angles ranging from -90$^{\text{o}}$ to 90$^{\text{o}}$. Since we are only concerned with the second link under equilibrium conditions, x_1 is unimportant and x_2 must be zero. We derive a function for u_2 in terms of x_3 by setting the equation for x_4 equal to zero.

$$x_4 = \dot{x}_3 = -\frac{g}{l}\cos(x_3) - \frac{x_2}{2}\sin(2x_3) + \frac{u_2}{ml^2}$$

cancelling the second term and solving for u_2:

$$u_2(x_3) = mlg\cos(x_3)$$

A plot of this function is shown in Figure 2. The shape of this plot agrees with our natural intuitions. As we would expect, the greatest torque is required at $\theta_2 = 0^{\text{o}}$, when the arm is horizontal and gravity's moment arm is largest. At $\theta_2 = \pm 90^{\text{o}}$, the arm is vertical, and no torque is required. A sinusoidal shape is also consistent with a simple pendulum, which is essentially what this dynamic system represents.

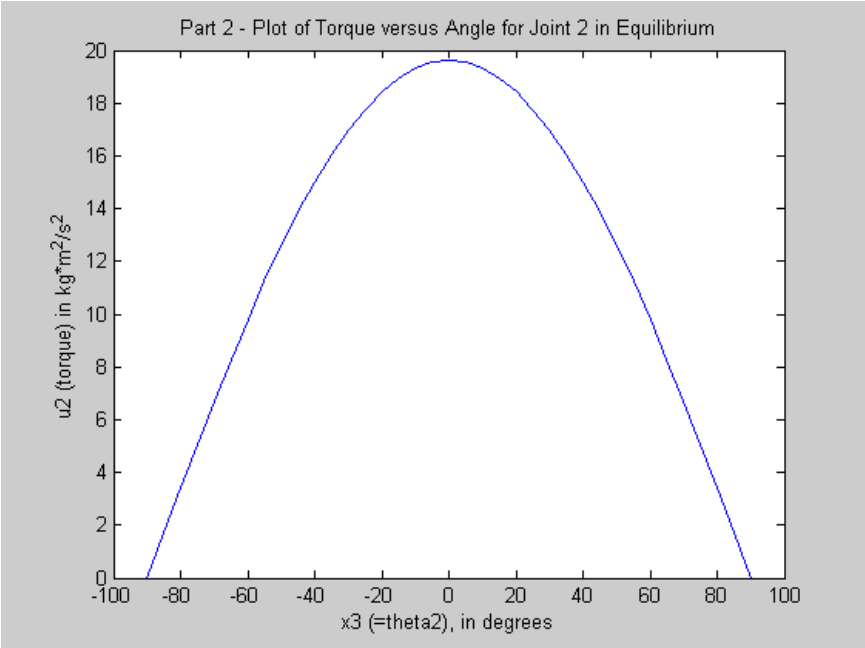


Figure 2: Plot of $u_2(x_3)$

Linearization

We wish to express our system in the linearized form given in Equation 5 at each operating point. As the operating points are steady-state, the nominal solutions are simply $\dot{\mathbf{x}}_0(t) = 0$. We wish to express the perturbation solution $\Delta\dot{\mathbf{x}}(t)$ in the linear time-varying form shown in Equation 6, where $\mathbf{F}$, $\mathbf{G}$, and $\mathbf{H}$ are Jacobian matrices of $\mathbf{f}\left(\dot{\mathbf{x}}_0(t),\, \mathbf{u}_0(t),\, \mathbf{w}_0(t),\, t\right)$ with respect to $\mathbf{x}$, $\mathbf{u}$, and $\mathbf{w}$, respectively. These matrices are given in Equations 7, 8, and 9. Disturbances are modeled as torques applied to the joints, so disturbance dynamics are identical to control dynamics and $\mathbf{L} = \mathbf{G}$.

$$\dot{\mathbf{x}}_0(t) + \Delta\dot{\mathbf{x}}(t) = \mathbf{f}\left(\dot{\mathbf{x}}_0(t),\, \mathbf{u}_0(t),\, \mathbf{w}_0(t),\, t\right) + \Delta\mathbf{f}\left(\dot{\mathbf{x}}_0(t),\, \mathbf{u}_0(t),\, \mathbf{w}_0(t),\, t\right) \tag{5}$$

$$\Delta\dot{\mathbf{x}}(t) = \mathbf{F}(t)\Delta\mathbf{x}(t) + \mathbf{G}(t)\Delta\mathbf{u}(t) + \mathbf{L}(t)\Delta\mathbf{w}(t) \tag{6}$$

$$\mathbf{F} = \frac{\partial \mathbf{f}}{\partial \mathbf{x}} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & \frac{\mathrm{x}_4 \sin(2\mathrm{x}_3)}{\cos^2(\mathrm{x}_3)} & \frac{2\sin(\mathrm{x}_3)\left(\frac{\mathrm{u}1}{l^2 m} + \mathrm{x}_2 x_4 \sin(2\mathrm{x}_3)\right)}{\cos^3(\mathrm{x}_3)} + \frac{2\mathrm{x}_2\mathrm{x}_4\cos(2\mathrm{x}_3)}{\cos(x_3)^2} & \frac{\mathrm{x}_2\sin(2\mathrm{x}_3)}{\cos(\mathrm{x}_3)^2} \\ 0 & 0 & 0 & 1 \\ 0 & \sin(2\mathrm{x}_3) & \frac{g\sin(\mathrm{x}_3)}{l} - \mathrm{x}_2\cos(2\mathrm{x}_3) & 0 \end{bmatrix} \tag{7}$$

$$\mathbf{G} = \frac{\partial \mathbf{f}}{\partial \mathbf{u}} = \begin{bmatrix} 0 & 0 \\ \frac{1}{(\cos(x3))^2 ml^2} & 0 \\ 0 & 0 \\ 0 & \frac{1}{ml^2} \end{bmatrix} \tag{8}$$

$$\mathbf{L} = \mathbf{G} \tag{9}$$

LQR Gain Matrix Computation

LQR state feedback gain matrices were computed for the linearized system at each operating point. θ_2 was varied between -90 and 90 degrees in intervals of five degrees; $\dot{\theta}_2$, θ_1, and $\dot{\theta}_1$ were set to zero. Cost matrices $\mathbf{Q} = \mathbf{I}_4$ and $\mathbf{R} = \mathbf{I}_2$ were used. Figure 3 shows the variation of these gains with θ_2. There is a great deal of variation over the domain for all elements of the gain matrix, indicating that linearization of the system at a single operating point would likely produce an unsuccessful controller.

2.2 Simulink Model

The system dynamic models was translated into the SIMULINK block model subsystem shown in Figure 4. The controller was also implemented in SIMULINK, and is show in the block model in Figure 5. The gain scheduler is implemented using a number of linear interpolation table lookup blocks as shown in Figure 6.

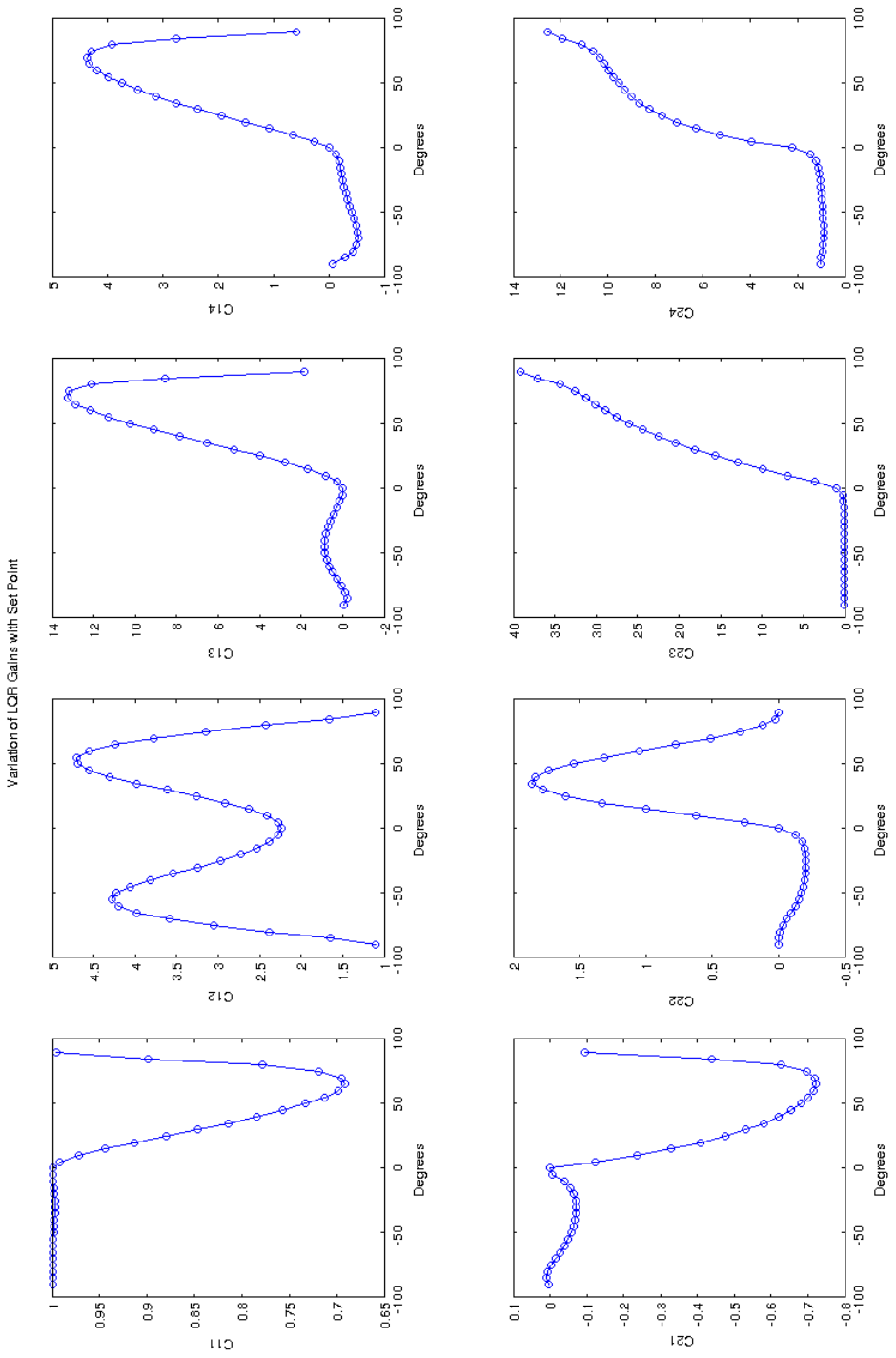


Figure 3: Plot of LQR gains by operating point

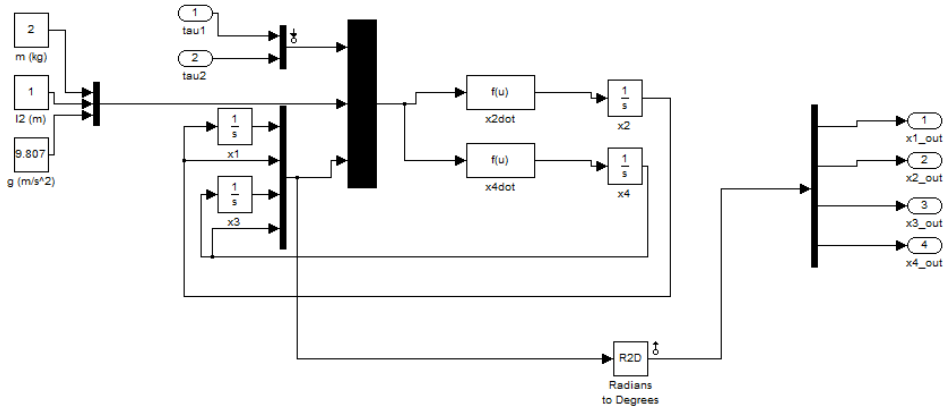


Figure 4: SIMULINK Robot Model

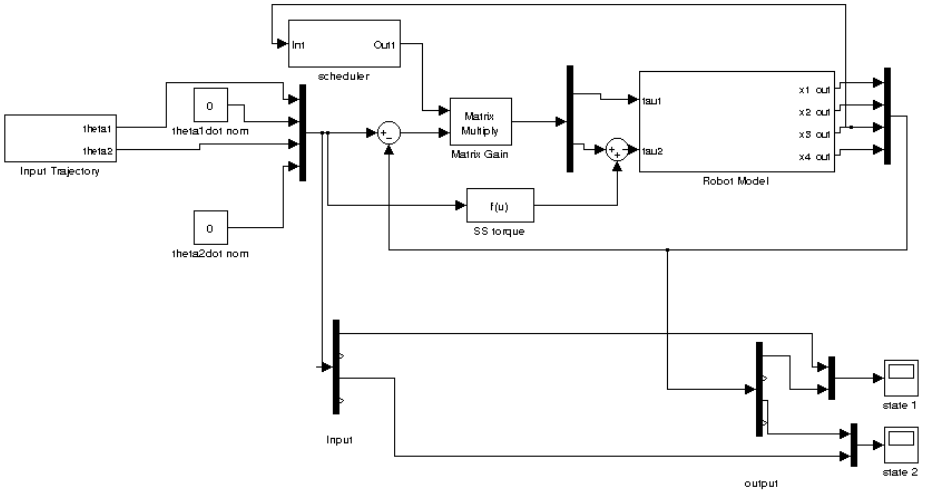


Figure 5: SIMULINK Controller Model

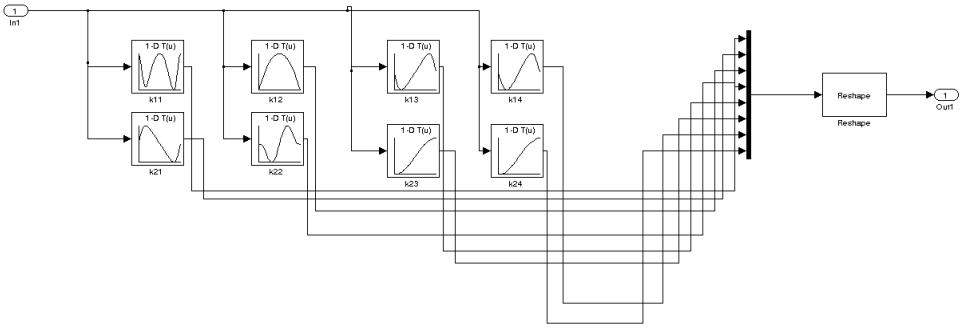


Figure 6: SIMULINK Gain scheduling model

3 Results

Controller performance was tested using a nominal trajectory from the retargeter. Parameters were varied in order to test their effect on performance.

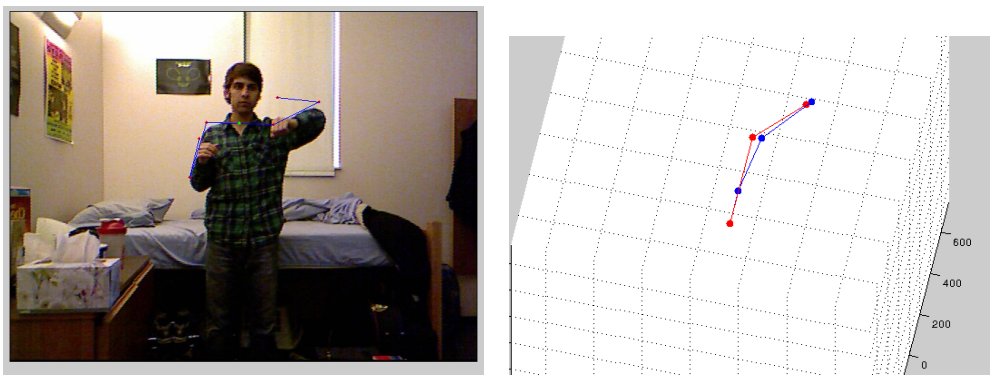
3.1 Generation of Retargeted Trajectory

Before retargeting solutions could be computed, a kinematic model of the robot had to be formed in Matlab just Denavit-Hartenberg parameters. These DH parameters are listed in Table below.

	d	θ	l	α
link 1	0	0	d_1	θ_1
link 2	0	θ_2	l_2	0

Table 1: Denavit-Hartenberg Parameters

Motion was then retargeted from my own arm to the robot, using the Microsoft Kinect for motion capture. To make the control movements easier to perform, and ninety-degree rotation was applied to the input data before retargeting it to the simulated robot, so that the vector pointing horizontally outward from my right shoulder corresponded to vertical in the robot's frame. This was done purely to make the source motions more comfortable to perform; holding one's arm vertically from the shoulder for any length of time will quickly become taxing. Figure 7 shows snapshots of the retargeting process. Due to the disparity in workspace size between a human arm and the two-link manipulator retargeted solutions were not perfect, but pose imitations were of reasonable quality over the duration of the motion.



(a) Source data, captured with the Kinect (b) Source (blue) and retargeted (red)

Figure 7: Snapshots of the retargeting process

3.2 Trajectory Following

Figure 8 shows the joint-angle trajectory for the retargeted motion. The objective is for the controller to track this trajectory as closely as possible. We test the controller's performance by varying a number of parameters.

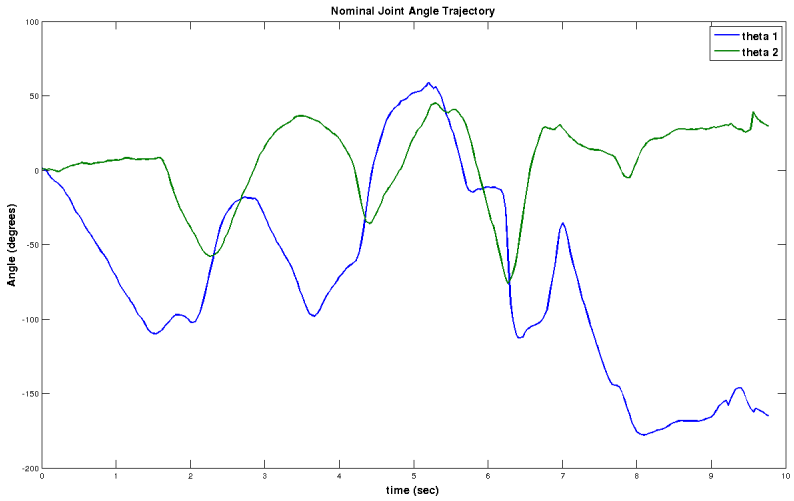
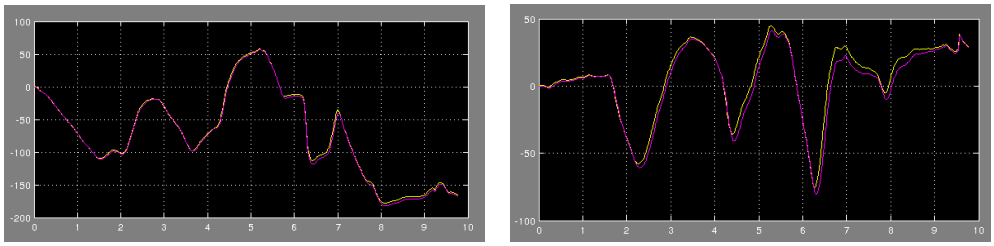


Figure 8: Nominal retargeted trajectory

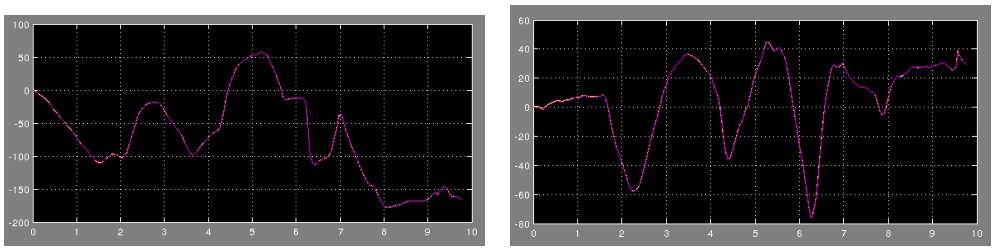
Full-State Feedback

As our first test case, we compute the response of a full-state feedback controller designed with$\mathbf{Q} = \mathbf{I}_4$ and $\mathbf{R} = \mathbf{I}_2$. To establish nominal values for $\dot{\theta}_1$ and $\dot{\theta}_2$, the input signal was passed through a derivative filter. Tracking results are shown in Figure 9. Tracking is fairly accurate, but deviates slightly from the nominal state at times. Since these are deviations of only a few degrees, these results would likely be sufficient for an application such as gesture imitation. If the task at hand requires great accuracy, the LQR cost matrices may be tuned accordingly. Figure 10 shows the results of increasing $\mathbf{Q}$ factor of 100 - tracking is effectively perfect.



(a) θ_1 tracking (b) θ_2 tracking

Figure 9: Tracking results for full-state feedback, $\mathbf{Q} = \mathbf{I}_4$ and $\mathbf{R} = \mathbf{I}_2$ (input yellow, output purple)

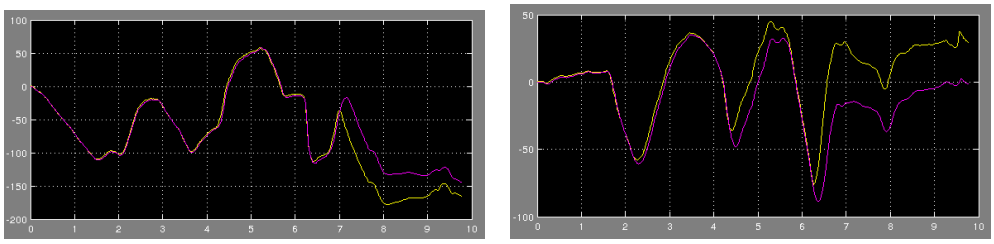


(a) θ_1 tracking (b) θ_2 tracking

Figure 10: Tracking results for full-state feedback, $\mathbf{Q} = 100 * \mathbf{I}_4$ and $\mathbf{R} = \mathbf{I}_2$ (input yellow, output purple)

Torque Limiting

Up to this point, the manipulator has been modeled as capable of exerting any torque commanded by the controller. In reality, and the actuators of the manipulator would have a torque maximum, which we model with a flat cutoff. Figure 11 shows system performance after introducing an torque limit of 60 N-m, which is a reasonable actuator limit for a manipulator of this size [2]. This limitation seriously degrades tracking performance toward the end of the trajectory, and we can see in Figure 12 that the applied torques clip at the 60 N-m frequently during this part of the trajectory. However, outfitting the manipulator with more powerful motors (with a 100 N-m torque limit) yields great performance improvements (Figure 13).



(a) θ_1 tracking (b) θ_2 tracking

Figure 11: Tracking results with a torque limit of 60 N*m (input yellow, output purple)

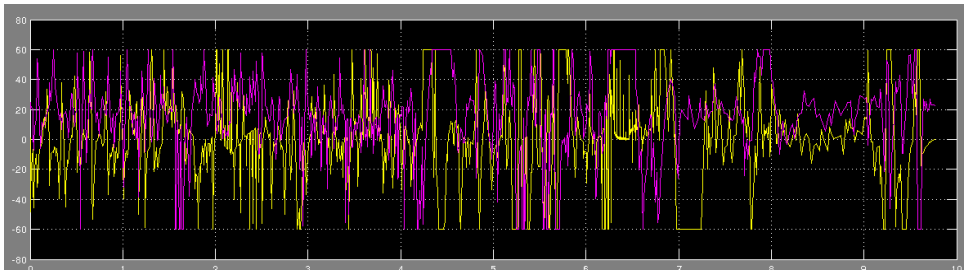
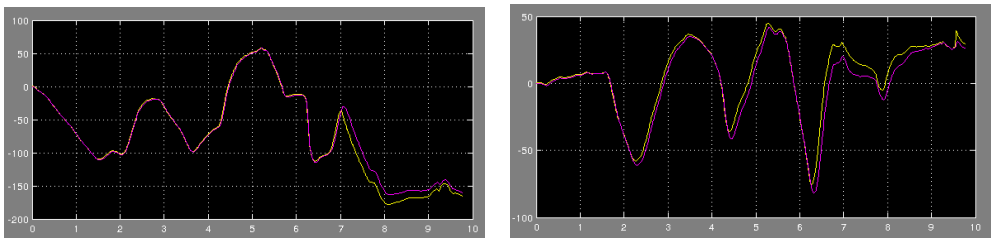


Figure 12: Plot of torques over time when limited to 60 N*m (τ_1 yellow, τ_2 purple)

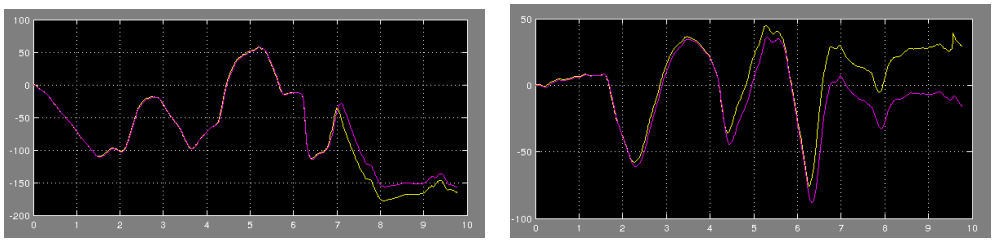


(a) θ_1 tracking (b) θ_2 tracking

Figure 13: Tracking results with a torque limit of 100 N*m (input yellow, output purple)

Robustness to Disturbance

Robustness to disturbance is one of the most important measures of any controller's performance. To test this, we instantaneously applying a constant load of -10 N*m to the arm joint beginning at time t=3 seconds - perhaps this is caused by the weight of an object which the arm has picked up and must carry for the remainder of the motion. To maintain realism, we also include a torque limitation of 100 N-m. We see in Figure 14 θ_2 tracking performance is degraded by the applied load, but that the general trajectory shape is still maintained.

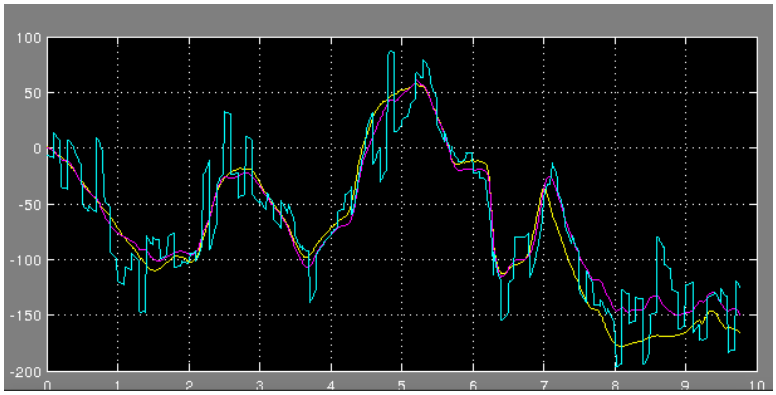


(a) θ_1 tracking (b) θ_2 tracking

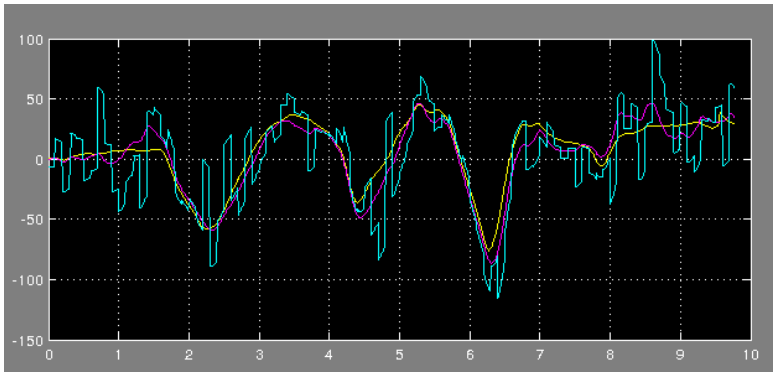
Figure 14: Tracking results with -10 N*m load applied at time t=3 (input yellow, output purple)

Noise Rejection

Finally, we test the controller's ability to reject sensor noise, another important performance metric for any dynamic controller. We model sensor noise by adding Gaussian white noise independently to each state (variance of 1 and scaled by a gain of 25 in each case). Once again, we include a torque limitation of 100 N-m for realism. Figure 15 shows the results. The controller exhibits some noise rejection, but the overall trajectory is still perturbed quite a bit by the sensor noise. To improve noise rejection, sensor inputs could be low-pass filtered. Or, a better alternative might be to implement a well-designed Kalman filter for state estimation that uses a model of the dynamic system and statistical characteristics of the process noise to de-noise the sensor measurements.



(a) θ_1 tracking



(b) θ_2 tracking

Figure 15: Tracking results with Gaussian sensor noise applied (input yellow, output purple, measurements green)

4 Conclusions and Future Work

4.1 Conclusions

The designed LQR controller proved effective in directing a simulated robotic manipulator to follow a trajectory output by the motion retargeter. Controller response was fast enough to capture the details of the retargeted trajectory, especially when the LQR cost matrices were adjusted for closer tracking. Controller disturbance rejection and noise rejection were found to be adequate but imperfect. Noise rejection might be greatly improved through the addition of a Kalman filter. Torque limiting was found to have an adverse effect on tracking, showing that this manipulator would require very powerful actuators to achieve human-like motion.

4.2 Future Work

In the future, it would be very interesting to implement a controller such as this on a real robot. The CKbot from the UPenn Modlab would be of particular interest for the purposes of retargeting. Previous experience in retargeting to CKbots resulted in very slow, jerky performance due to poor controller design (the controller used simply implemented servo feedback on position, progressing in discreet steps on a point-to-point basis). Using a more advanced controller such as this one would likely improve motion retargeting performance significantly.

For more complicated robot systems, full state feedback is not feasible; feedback information is provided by a limited number of sensors. It also be advantageous to expand this controller by adding a Kalman filter-based observation module, as this would allow for both state estimation and sensor noise rejection.

References

[1] S. Cousins. Ros on the pr2 [ros topics]. *Robotics Automation Magazine, IEEE*, 17(3):23 –25, sept. 2010.

[2] http://www.directindustry.com/prod/motor-power-company/brushless-electric-servo-motors 19978-356319.html. Brushless electric servo-motor, 2012.

[3] P. J. McKerrow. *Introduction To Robotics*. Addison-Wesley, 1991.

[4] University of Pennsylvania ModLab. Ckbot research. http://modlab.seas.upenn.edu/research.html.

[5] Robert F. Stengel. *Optimal Control and Estimation*. Dover Books, 1994.

[6] Robert F. Stengel. Mae 345: Robotics and intelligent systems assignment 3. Problem set, Princeton University, October 2011.

[7] Tarik Tosun. Mae 345 homework 3. Problem set, Princeton University, 2011.

[8] Tarik Tosun. Development of a system for general kinematic retargeting. Master's thesis, Princeton University, May 2012.

I pledge my honor this paper represents my own work in accordance with University regulations.