Activity Log

Rodolfo Jordao

September 10, 2014

Contents

List of Algorithms

List of Figures

List of Tables

Acronyms

LSR least squares regression. 13, 14

LTI Linear time-invariant. 12

ODE ordinary differential equation. 11

SISO Single-input-single-output. 12

Chapter 1

Modelling

1.1 Kinematics

1.1.1 Angles and Positions dependence:

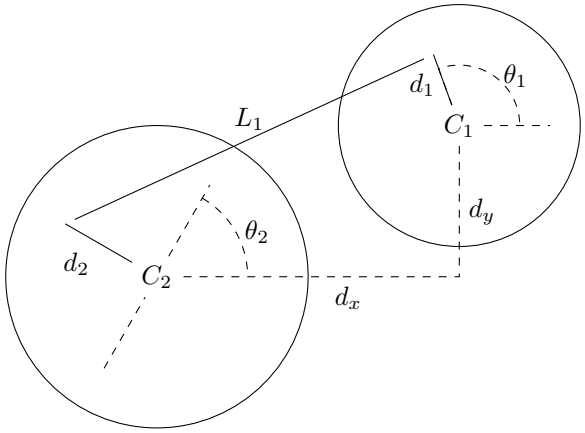

Figure 1.1: First sketch.

From Fig. 1.1,

$$L_1 c_3 - d_2 s_2 - d_1 c_1 = d_x$$
$$L_1 s_3 + d_2 c_2 - d_1 s_1 = d_y$$

rearrange to isolate the pseudo angle,

$$L_1 c_3 = d_x + d_2 s_2 + d_1 c_1$$
$$L_1 s_3 = d_y - d_2 c_2 + d_1 s_1$$

square both equations and sum them,

$$L_1^2 = d_x^2 + d_y^2 + d_1^2 + d_2^2 + 2\left(d_x d_2 s_2 + d_x d_1 c_1 + d_1 d_2 c_1 s_2 - d_y d_2 c_2 + d_y d_1 s_1 - d_1 d_2 s_1 c_2\right)$$

for simplicity, let,

$$K = \frac{L_1^2 - \left(d_x^2 + d_y^2 + d_1^2 + d_2^2 + 2 d_x d_1 c_1 + 2 d_y d_1 s_1\right)}{2}$$
$$a = d_x d_1 + d_1 d_2 c_1$$
$$b = -(d_y d_2 + d_1 d_2 s_1)$$

giving,

$$K = a s_2 + b c_2$$

then,

$$c_2 = \frac{bK + \sqrt{b^2K^2 - (a^2+b^2)(K^2-a^2)}}{a^2+b^2}$$
$$= \frac{bK + a\sqrt{a^2+b^2-K^2}}{a^2+b^2}$$

finally,

$$\theta_2 = Atan2\left(\sqrt{1-c_2^2}, c_2\right)$$

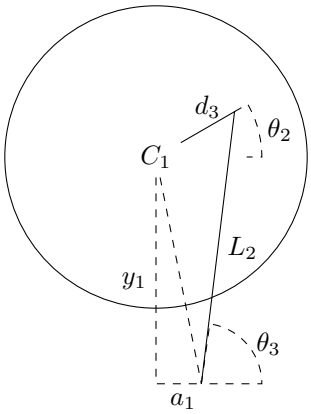

Figure 1.2: Second Sketch.

$$a_1 + L_2\cos\theta_3 = d_3\cos\theta_2$$
$$d_3\sin\theta_2 + y_1 = L_2\sin\theta_3$$

rearranging these equations:

$$L_2\cos\theta_3 = d_3\cos\theta_2 - a_1$$
$$L_2\sin\theta_3 = y_1 + d_3\sin\theta_2$$

these can be squared and summed to give:

$$y_1^2 + 2y_1d_3s_2 + s_2^2d_3^2 - L_2^2 + (d_3c_2 - a_1)^2 = 0$$

getting the roots:

$$y_1 = -d_3s_2 + \sqrt{L_2^2 - (a_1 - d_3c_2)^2}$$

the same reasoning can be applied to get y_2.

1.1.2 Best linearization candidate

Since most of the operational routine the system will oscillate about $\theta_2 = 0$, a good candidate for linearization of the system should be the values of θ_1 which results in this condition. For this, it is possible to use all equations developed in direct Kinematics. So the starting equation to solve for θ_1 is:

$$L_1^2 = d_x^2 + d_y^2 + d_1^2 + d_2^2 + 2\left(d_xd_1c_1 - d_yd_2 + d_yd_1s_1 - d_1d_2s_1\right)$$

$$\therefore \frac{L_1^2 - \left(d_x^2 + d_y^2 + d_1^2 + d_2^2\right)}{2} + d_yd_2 = d_xd_1c_1 + d_1\left(d_y - d_2\right)s_1$$

making the following simplifications,

$$W = \frac{L_1^2 - \left(d_x^2 + d_y^2 + d_1^2 + d_2^2\right)}{2} + d_yd_2$$
$$h = d_xd_1$$
$$g = d_1\left(d_y - d_2\right)$$

the solution for θ_1 is acquired,

$$c_1 = \frac{hW + g\sqrt{g^2 + h^2 - W^2}}{g^2 + h^2}$$

$$s_1 = \frac{gW - h\sqrt{g^2 + h^2 - W^2}}{g^2 + h^2}$$

$$\theta_0 = Atan2\left(\sqrt{1 - c_1^2}, c_1\right)$$

where θ_0 is the angle chosen to lienarize the whole system about.

1.1.3 Localization of all points of interest

Looking at Fig. 1.3 and assuming that everything has homogeneous distribution, which can safely be switched later on (even more assuming that most or all links have axial homogeneity):

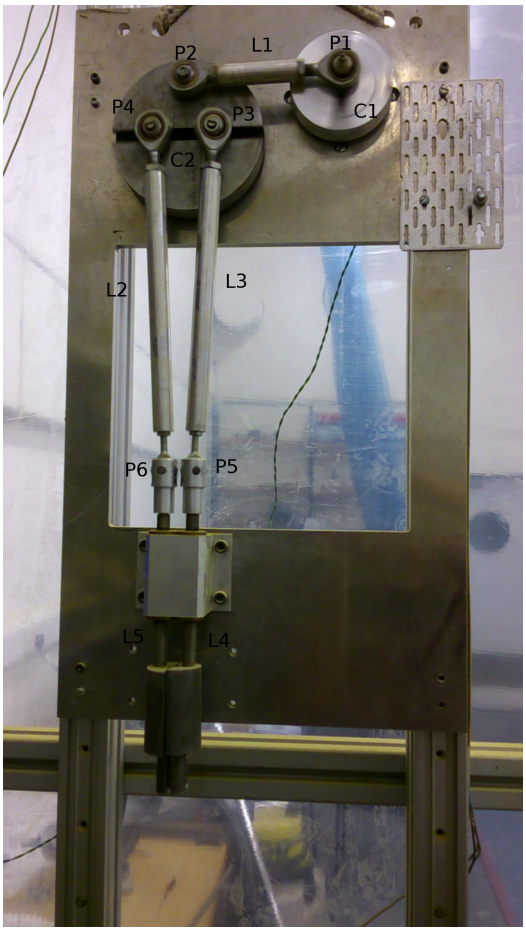

Figure 1.3: Photo of test rig indicating points.

1. $\mathbf{r}_{C_1} = (0, 0)$
2. $\mathbf{r}_{C_2} = (c_x, c_y)$
3. $\mathbf{r}_{P_1} = \mathbf{r}_{C_2} + (d_1 \cos\theta_1, d_1 \sin\theta_1)$
4. $\mathbf{r}_{P_2} = (d_2 \sin\theta_2, d_2 \cos\theta_2)$
5. $\mathbf{r}_{P_3} = \frac{1}{2}(d_3 \cos\theta_2, d_3 \sin\theta_2)$
6. $\mathbf{r}_{P_4} = -\frac{1}{2}(d_4 \cos\theta_2, d_4 \sin\theta_2)$
7. $\mathbf{r}_{P_5} = (a_1, -y_1)$
8. $\mathbf{r}_{P_6} = (a_2, -y_2)$
9. $\mathbf{r}_{L_1} = \frac{1}{2}(\mathbf{r}_{P_1} + \mathbf{r}_{p_2})$
10. $\mathbf{r}_{L_2} = \frac{1}{2}(\mathbf{r}_{P_4} + \mathbf{r}_{P_6})$
11. $\mathbf{r}_{L_3} = \frac{1}{2}(\mathbf{r}_{P_3} + \mathbf{r}_{P_5})$
12. $\mathbf{r}_{L_4} = \mathbf{r}_{P_5} + \frac{1}{2}(0, -L_4)$
13. $\mathbf{r}_{L_5} = \mathbf{r}_{P_6} + \frac{1}{2}(0, -L_5)$

1.1.4 Velocities of all points of interest

This is necessary for dynamical formulation. Actually it is enough to differentiate every point localization with respect to time, the only another requirement is the angular velocity of every body, which are:

1. $\omega_{C_1} = \omega_{P_1} = \dot{\theta}_1$
2. $\omega_{C_2} = \omega_{P_2} = \omega_{P_3} = \omega_{P_4} = \dot{\theta}_2$
3. $\omega_{L_1} = -\dfrac{\frac{d_1 s_2 \dot{\theta}_1 \dot{\theta}_2 - d_2 c_1 \dot{\theta}_1}{d_1 c_1 + d_1 s_2 + c_x} + \frac{\left(d_1 c_2 \dot{\theta}_1 \dot{\theta}_2 - d_1 s_1 \dot{\theta}_1\right)(d_1 c_2 + d_2 s_1 + c_y)}{(d_1 c_1 + d_1 s_2 + c_x)^2}}{\frac{(d_1 c_2 + d_2 s_1 + c_y)^2}{(d_1 c_1 + d_1 s_2 + c_x)^2} + 1}$
4. $\omega_{L_2} = \dfrac{\frac{(d_4 s_2 - y_2(\theta_1(t)))d_4 s_2 \dot{\theta}_1 \dot{\theta}_2}{(d_4 c_2 + a_2)^2} + \frac{d_4 c_2 \dot{\theta}_1 \dot{\theta}_2 - \dot{\theta}_1 \dot{y_2}}{d_4 c_2 + a_2}}{\frac{(d_4 s_2 - y_2(\theta_1(t)))^2}{(d_4 c_2 + a_2)^2} + 1}$
5. $\omega_{L_3} = \dfrac{\frac{(d_3 s_2 - y_1(\theta_1(t)))d_3 s_2 \dot{\theta}_1 \dot{\theta}_2}{(d_3 c_2 + a_1)^2} + \frac{d_3 c_2 \dot{\theta}_1 \dot{\theta}_2 - \dot{\theta}_1 \dot{y_1}}{d_3 c_2 + a_1}}{\frac{(d_3 s_2 - y_1(\theta_1(t)))^2}{(d_3 c_2 + a_1)^2} + 1}$

Also, the rotation of the screws (Every P) can possibly be ignored as it *may* cuase little contribution to the overall system dynamics if neglected as rigid bodies.

1.2 Dynamics

The symbolic calculations are just too gigantic to fill in one paper. I made many assumptions that simplified the model but that still did not make it small enough, I'll try other things. The sagemath code in Sec. **??** basically does that.

1.2.1 plant model

the Lagrangian has the form:

$$L = \sum \frac{m_i}{2} \left(\dot{r}_i(\theta_1)\right)^2 + \frac{I_i}{2}(\omega_i(\theta_1, \dot{\theta}_1))^2 - m_i r_i(\theta_1) g$$

seeing that $\forall i, \omega_i(\theta_1, \dot{\theta}_1) = \Omega(\theta_1)\dot{\theta_1}$, then:

$$\begin{aligned} L &= \sum \frac{m_i}{2} \left(\frac{r_i(\theta_1)}{d\theta_1}\right)^2 \dot{\theta}_1^2 + \frac{I_i}{2}(\Omega_i(\theta_1))^2 {\dot{\theta_1}}^2 - m_i r_i(\theta_1) g \\ &= {\dot{\theta_1}}^2 \left[\sum \frac{m_i}{2} \left(\frac{r_i(\theta_1)}{d\theta_1}\right)^2 + \frac{I_i}{2}(\Omega_i(\theta_1))^2\right] - \sum m_i r_i(\theta_1) g \end{aligned}$$

if the element enclosed in brackets is denominated V while the sum U, then:

$$L = {\dot{\theta_1}}^2 V(\theta_1) - U(\theta_1)$$

where the equation of motion, disregarding for now outside influence and friction, can be derived:

$$\begin{aligned} \tau &= \frac{d}{dt}\left(\frac{\partial L}{\partial \dot{\theta}_1}\right) - \frac{\partial L}{\partial \theta_1} \\ &= \ddot{\theta_1}(2V(\theta_1)) + \dot{\theta}_1^2 \frac{dV(\theta_1)}{d\theta_1} + \frac{dU(\theta_1)}{d\theta_1} \end{aligned}$$

or written more compactly:

$$\tau = 2\ddot{\theta}_1 V(\theta_1) + \dot{\theta}_1^2 V'(\theta_1) + U'(\theta_1) \tag{1.1}$$

finally giving it into more familiar form:

$$\begin{aligned} \dot{\theta}_1 &= \omega_1 \\ \dot{\omega}_1 &= -\omega_1^2 \frac{V'(\theta_1)}{2V(\theta_1)} - \frac{U'(\theta_1)}{2V(\theta_1)} + \frac{\tau}{2V(\theta_1)} \end{aligned} \tag{1.2}$$

1.2.2 linearised plant model

the linear model basically consists into the jacobian of Sys. (1.2) evaluated about equilibrium point discussed in Sec. 1.1.2. First define:

$$\begin{aligned} g(\omega_1, \theta_1) &= -\omega_1^2 \frac{V'(\theta_1)}{2V(\theta_1)} - \frac{U'(\theta_1)}{2V(\theta_1)} \\ f(\omega_1, \theta_1, \tau) &= \frac{\tau}{2V(\theta_1)} \end{aligned}$$

then clearly, the linearisation will be given by:

$$\dot{\theta}_1 = \frac{\partial \omega_1}{\partial \theta_1}\theta_1 + \frac{\partial \omega_1}{\partial \omega_1}\omega_1$$
$$\dot{\omega}_1 = \left[\frac{\partial g}{\partial(\theta_1,\omega_1)}(\theta_0)\right]\begin{bmatrix}\theta_1-\theta_0\\ \omega_1\end{bmatrix} + \left[\frac{\partial f}{\partial(\theta_1,\omega_1)}(\theta_0)\right]\begin{bmatrix}\tau\\ \tau\end{bmatrix}$$

where the partial is in fact a gradient. These in turn can be written as,

$$\begin{aligned}\dot{\theta}_1 &= \omega_1\\ \dot{\omega}_1 &= a_1\theta_1 + b_1\omega_1 + c_1 + d_1\tau\end{aligned} \tag{1.3}$$

where the coefficients are, symbolically:

$$\begin{aligned}a_1 &= \left[-\omega_1^2\left(\frac{V(\theta_1)V''(\theta_1)-(V'(\theta_1))^2}{2V(\theta_1)}\right) - \left(\frac{V(\theta_1)U''(\theta_1)-V'(\theta_1)U'(\theta_1)}{2V(\theta_1)}\right)\right]_{\theta_1=\theta_0,\omega_1=0}\\ b_1 &= \left[-\omega_1\frac{V'(\theta_1)}{V(\theta_1)}\right]_{\theta_1=\theta_0,\omega_1=0}\\ c_1 &= -a_1\theta_0\\ d_1 &= a_1+b_1\end{aligned}$$

by the chosen equilibrium point, some simplifications can be made:

$$\begin{aligned}a_1 &= -\left(\frac{V(\theta_0)U''(\theta_0)-V'(\theta_0)U'(\theta_0)}{2V(\theta_0)}\right)\\ b_1 &= 0\\ c_1 &= -a_1\theta_0\\ d_1 &= a_1+b_1\end{aligned}$$

An even better way to linearise this sytem is to look into Eq. (1.1), and make this the starting equation, since this can easily then give the linearised torque the motor should generate. First, see that the torque is function of three independent variables: $\theta, \omega, \dot{\omega}$, thus:

$$\nabla\tau = \left(\frac{\partial\tau}{\partial\theta}, \frac{\partial\tau}{\partial\omega}, \frac{\partial\tau}{\partial\dot{\omega}}\right)$$

seeing that $x_0 = (\theta_0, 0, 0)$ is the equilibrium point, the sought equation for along the torque will be:

$$\tau = \nabla\tau|_{x=x_0}\begin{bmatrix}\theta-\theta_o\\ \omega\\ \dot{\omega}\end{bmatrix}$$

or expanded in explicit form,

$$\begin{aligned}\tau &= \left(2V'(\theta_0)\right)\dot{\omega} + \left(2\omega_0 V'(\theta_0)\right)\omega + \left(2\dot{\omega}_0 V'(\theta_0) + \omega_0^2 V''(\theta_0) + U''(\theta_0)\right)(\theta-\theta_0)\\ \therefore \tau &= 2V'(\theta_0)\dot{\omega} + U''(\theta_0)(\theta-\theta_0)\end{aligned} \tag{1.4}$$

1.2.3 overall model

It can be found easily in the internet or elsewhere that the coupled ordinary differential equations (ODEs) modelling the motor's behaviour are given by:

$$\begin{aligned}\dot{i} &= -\frac{R}{L}i - \frac{k_e}{L}\omega_m + \frac{1}{L}V_s\\ \dot{\omega}_m &= \frac{k_t}{J}i - \frac{k_e}{J} + \frac{1}{J}\tau_m\end{aligned}$$

these can be combined with the previous equation of motion developed for the drill rig:

$$\begin{aligned}\dot{i} &= -\frac{R}{L}i - \frac{k_e}{L}\omega_m + \frac{1}{L}V_s\\ \dot{\omega}_m &= \frac{k_t}{J}i - \frac{k_e}{J}\omega_m + \frac{1}{J}\frac{1}{N}\left(a\dot{\omega}_o + b\omega_o + c\theta_o + d\right)\\ \dot{\theta}_m &= \omega_m\end{aligned}$$

finally noting that $N\theta_0 = \theta_m$:

$$\dot{\omega}_m = \frac{k_t}{J} i - \frac{k_e}{J}\omega_m + \frac{1}{J}\frac{1}{N^2}\left(a\dot{\omega}_m + b\omega_m + c\theta_m + d\right)$$

$$\left(1 - \frac{a}{N^2}\right)\dot{\omega}_m = \frac{k_t}{J} i + \left(\frac{b}{JN^2} - \frac{k_e}{J}\right)\omega_m + \frac{c}{JN^2}\theta_m + \frac{d}{JN^2}$$

finally, the overall linearized system coupled ODEs can be written:

$$\begin{aligned}
\dot{i} &= -\frac{R}{L} i - \frac{k_e}{L}\omega_m + \frac{1}{L} V_s \\
\dot{\omega}_m &= \frac{N^2 k_t}{J(N^2 - a)} i + \frac{1}{J}\left(\frac{b - N^2 k_e}{N^2 - a}\right)\omega + \frac{c}{J(N^2 - a)}\theta + \frac{d}{J(N^2 - a)} \\
\dot{\theta}_m &= \omega_m
\end{aligned} \tag{1.5}$$

using $x = (i, \omega, \theta)$, the standard form for dynamical systems:

$$x = Ax + BV_s + Ex_o$$

where:

$$A = \begin{bmatrix} -\frac{R}{L} & \frac{k_e}{L} & 0 \\ \frac{N^2 k_t}{J(N^2 - a)} & \frac{1}{J}\left(\frac{b - N^2 k_e}{N^2 - a}\right) & \frac{c}{J(N^2 - a)} \\ 0 & 1 & 0 \end{bmatrix}$$

$$B = \begin{bmatrix} \frac{1}{L} \\ 0 \\ 0 \end{bmatrix}$$

$$E = \begin{bmatrix} 0 \\ \frac{d}{J(N^2 - a)} \\ 0 \end{bmatrix}$$

1.3 Control Measures

1.3.1 Inicial considerations

The main goal is to control the reciprocating frequency, which can make use of the efficient internal loop of the controller into the system. the IxR methodology to account for voltage drop is given in Fig. 1.4.

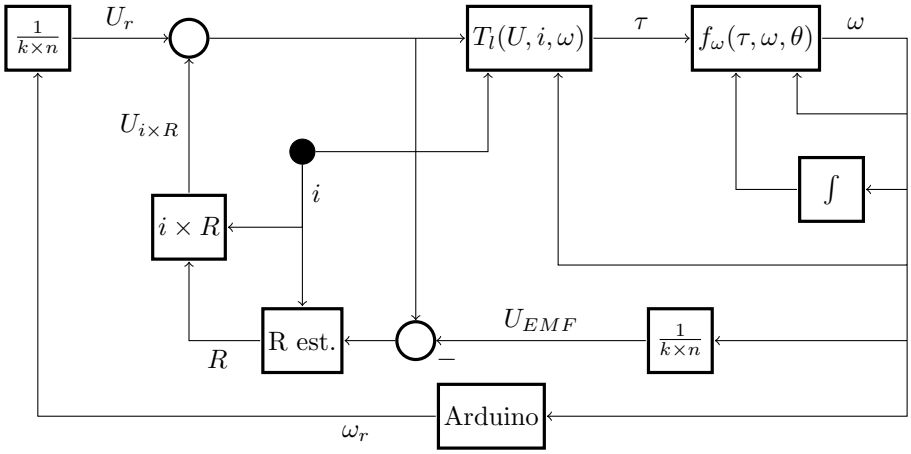

Figure 1.4: Overall diagram showing internal voltage drop control "IxR methodology". Every summation point is positive unless noted otherwise.

The reciprocating motion is dictated by the equation:

$$y(t) = f(\theta_2(\theta_1(t)))$$

the period of $y(t)$ is then given by the time elapsed between two equal values of y for different time. Since this is true for the period of θ_1, it comes that the period p of reciprocation is the same as the period of rotation:

$$y(t+p) = f(\theta_2(\theta_1(t+p))) = f(\theta_2(\theta_1(t))) = y(t)$$

Thus the reciprocating frequency is given by:

$$\frac{f_y}{2\pi} = \omega_1$$

If the values for the maximum and minimum heights in each stroke is required, their deduction follows:

$$\dot{y} = 0 = \frac{df}{d\theta_2}\frac{d\theta_2}{d\theta_1}\omega_1$$

since the maxima of f and θ_2 occur at the same points for their respective variables, it is enough to solve for:

$$\frac{df(\theta_2)}{d\theta_2} = 0$$

this is given by:

$$-d_3\cos(\theta_2) + \frac{(d_3\cos(\theta_2) - a_1)\, d_3\sin(\theta_2)}{\sqrt{{L_2}^2 - (d_3\cos(\theta_2) - a_1)^2}} = 0$$

which has solutions (along two more):

$$c_2 = \pm\frac{a_1}{L_2 + d_3}$$

which can be substituted back onto $y(\theta_2)$ to give its maxima and minima:

$$y_m = \frac{L_2 - d_3}{L_2 + d_3}\sqrt{(L_2^2 + d_3^2) - a_1^2}$$

$$y_m^* = -d_3\sqrt{1 - \frac{{a_1}^2}{(L_2 + d_3)^2}} + \sqrt{{L_2}^2 - \left(-\frac{d_3 a_1}{L_2 + d_3} - a_1\right)^2}$$

1.3.2 Optimal gain calculation

The system, although being defined in three dimensional state space is Single-input-single-output (SISO). Moreover the applied linearization rendered the lumped parameter model (1.5) Linear time-invariant (LTI), thus the classical approach via frequency analysis is readily avaiable for use.

In Eq. (1.5), its laplace transform is given by:

$$\begin{aligned} si &= a_{11}i + a_{12}\omega + bV_s \\ s\omega &= a_{21} + a_{22}\omega + a_{23}\theta + D \\ s\theta &= \omega \end{aligned}$$

where $a_{ij} = [A]_{ij}$, for simplicty of notation. Manipulating the first and third of these and substituting into the second:

$$s\omega = \frac{a_{12}\omega + bV_s}{s - a_{11}} + a_{22}\omega + \frac{a_{23}}{s}\omega + D$$

which gives:

$$\left(s - \frac{a_{12}}{s - a_{11}} - a_{22} - \frac{a_{23}}{s}\right)\omega = bV_s + D$$

or, into a rational form:

$$\left(\frac{s^3 - (a_{11} + a_{22})s^2 + (a_{11}a_{22} - a_{12} - a_{23})s + a_{11}a_{23}}{s(s - a_{11})}\right)\omega = bV_s + D$$

As D depends upon the choice of initial conditions, and since any of them make the physical system stable, it can be assumed that $D = 0$. Then the transfer function of the overall system:

$$G(s) = \frac{\omega}{V_s} = \left(\frac{bs(s - a_{11})}{s^3 - (a_{11} + a_{22})s^2 + (a_{11}a_{22} - a_{12} - a_{23})s + a_{11}a_{23}}\right) \tag{1.6}$$

The controller in question is a PI-like controller, hence its equation can be assumed to be:

$$C(s) = K_p\left(1 + \frac{1}{T_i s}\right) \tag{1.7}$$

where K and T_i are constants well defined in literature. The overall closed-loop transfer function changes to:

$$H(s) = \frac{C(s)G(s)}{1 + C(s)G(s)}$$

If a Integral Square error (ISE) is defined as:

$$I_c = \int_0^\infty (x_r(t) - y(t))^2 + \sigma^2 u^2 dt \tag{1.8}$$

where u is the control variable, then Parseval's theorem can be used to calculate this integral by knowing the transfer functions if each variable. Therefore, it is necessary to minimize:

$$I_c = \frac{1}{2\pi}\int_{-\infty}^{\infty} \left(V_s(i\xi) - \omega(i\xi)\right)\left(V_s(-i\xi) - \omega(-i\xi)\right) + \sigma^2 U(i\xi)U(-i\xi)d\xi$$

where i is the imaginary unit. It is desirable to manipulate the integrand before hand of computing the integral, as to minimize the effort required:

$$\left[V_s(s) - H(s)V_s(s)\right]\left[V_s(-s) - H(-s)V_s(-s)\right] + \sigma^2 V_s(s)C(s)C(-s)V_s(-s)$$

assuming that the voltage is an unity step input $1/s$ to prioritize reducing errors in the system,

$$E_2(s) = -\frac{1}{s^2}\left[1 - H(s)\right]\left[1 - H(-s)\right] - \sigma^2\frac{1}{s^2}C(s)C(-s)$$

as it is expected from the linearizations performed and computed transfer functions, $H(s) = N_h(s)/D_h(s)$ while $C(s) = N_c(s)/D_c(s)$, therefore:

$$\begin{aligned} E_2(s) &= \frac{1}{s}\left[1 - \frac{N_h(s)}{D_h(s)}\right]\left[1 - \frac{N_h(-s)}{D_h(-s)}\right]\frac{1}{-s} + \sigma^2\frac{1}{s}\frac{N_c(s)}{D_c(s)}\frac{N_c(-s)}{D_c(-s)}\frac{1}{-s} \\ &= \frac{1}{s}\left[\frac{D_h(s) - N_h(s)}{D_h(s)}\right]\left[\frac{D_h(-s) - N_h(-s)}{D_h(-s)}\right]\frac{1}{-s} + \sigma^2\frac{1}{s}\frac{N_c(s)}{D_c(s)}\frac{N_c(-s)}{D_c(-s)}\frac{1}{-s} \end{aligned}$$

these can be evaluated via the residue theorem of compelx analysis; and if so desired, evaluated numerically via an iterative procedure about the control gains.

1.4 Algorithms for Tuning

1.4.1 Theory and Deductions

After trying the implemented a stall detection algorithm which tried to exhaust some sequence properties and test them in the Arduino, they showed no promise. It has become clear that something a little more sophisticated, however possible of on-line implementation; emulating maybe a short-term memory was necessary. Certainly the current flowing through the motor can give good hints upon stalling, but no before prior knowledge of the system the motor will effectively drive; thus focus at the feedback speed is the most general option. Using the same fundamental definitions as before:

Definition 1 (Stall)**.** The positive definite error function $e(t)$ is said to be *stalling about* L if there is $\epsilon, t_o \in \mathbb{R}$ such that $\forall t > t_0, |e(t) - L| < \epsilon$. In words, the error $e(t)$ is *stalling* if it oscillates (ou stabilizates) about some value L. The discrete version is similar.

Once again, the same fundamental fact that connect a mathematical test to the physical problem is stated:

Fact 1. *if the error function $e(t)$ is not strictly monotonic decreasing, it is either stalling on some value or increasing.*

This can be proved without too much difficulty. Now, the test that can be used is that of linear least squares regression (LSR) of a certain amount of values. If n error values are sampled giving the vector $\mathbf{e} = \{e_1, \ldots, e_n\}$ for the following time vector $\mathbf{t} = \{T, \ldots, nT\}$. But since the check is for a non-strictly decreasing function, the vector $\mathbf{t}$ can be simplified to $\mathbf{t}_c = \{1, \ldots, n\}$. The reason for so is given by the following:

Definition 2. Let $a, b \in \mathbb{R}$ be such that,

$$e(t) = at + b$$

is the best approximation of the sampled data $\mathbf{e} = \{e_1, \ldots, e_n\}$ upon time vector $\mathbf{t} = \{t_1, \ldots, t_n\}$. Then a is denominated *trend* of the data and b *start* of the data. Moreover, these are given by:

$$a = \frac{n\left(\sum_{i=1}^{n} e_i t_i\right) - \left(\sum_{i=1}^{n} e_i\right)\left(\sum_{i=1}^{n} t_n\right)}{n\left(\sum_{i=1}^{n} t_i^2\right) - \left(\sum_{i=1}^{n} t_i\right)^2}$$
$$b = \frac{\left(\sum_{i=1}^{n} t_i^2\right)\left(\sum_{i=1}^{n} e_i\right) - \left(\sum_{i=1}^{n} e_i t_i\right)\left(\sum_{i=1}^{n} t_n\right)}{n\left(\sum_{i=1}^{n} t_i^2\right) - \left(\sum_{i=1}^{n} t_i\right)^2}$$

Since the micro controller samples at approximately a constant interval, the time vector can be considered $\mathbf{t} = \{T, \ldots, nT\}$, and the test for non strictly decreasing function follow:

Proposition 1 (LSR test)**.** *Let the error function $e(t)$ be sampled with time vector $\mathbf{t} = \{T, \ldots, nT\}$. If the trend obtained by using as a time vector $\mathbf{t}_c = \{1, \ldots, n\}$ is non-negative, then $e(t)$ cannot be monotonic decreasing.*

Proof. First, a connection between both time vectors is established; let a_1 be the trend with $\mathbf{t}$ and a_2 be the trend with $\mathbf{t}_c$. Then,

$$a_1 = \frac{n\left(\sum_{i=1}^{n} e_i iT\right) - \left(\sum_{i=1}^{n} e_i\right)\left(\sum_{i=1}^{n} iT\right)}{n\left(\sum_{i=1}^{n} iT^2\right) - \left(\sum_{i=1}^{n} iT\right)^2} = \frac{1}{T}\left[\frac{n\left(\sum_{i=1}^{n} e_i i\right) - \left(\sum_{i=1}^{n} e_i\right)\left(\sum_{i=1}^{n} i\right)}{n\left(\sum_{i=1}^{n} i^2\right) - \left(\sum_{i=1}^{n} i\right)^2}\right] = \frac{1}{T} a_2$$

that is, both trends are related by a constant, nominally $1/T$. To prove the incompatibility between strictly monotonic decreasing and the trend non-negativity, let $e(t)$ be a strictly monotonic decreasing function; then its trend with time $\mathbf{t}_c$,

$$\begin{aligned} a &= \frac{n\left(\sum_{i=1}^{n} e_i i\right) - \left(\sum_{i=1}^{n} e_i\right)\left(\sum_{i=1}^{n} i\right)}{n\left(\sum_{i=1}^{n} i^2\right) - \left(\sum_{i=1}^{n} i\right)^2} \\ &= \frac{n\left(\sum_{i=1}^{n} e_i i\right) - \left(\sum_{i=1}^{n} e_i\right)\frac{n}{2}(n+1)}{n\left(\sum_{i=1}^{n} i^2\right) - \left(\sum_{i=1}^{n} i\right)^2} \\ &= \frac{\sum_{i=1}^{n} e_i \left(ni - n\frac{(n+1)}{2}\right)}{n\left(\sum_{i=1}^{n} i^2\right) - \left(\sum_{i=1}^{n} i\right)^2} \end{aligned}$$

since the denominator is a constant, focus is given to the numerator. By assumption,

$$\begin{aligned} \sum_{i=1}^{n} e_i \left(ni - n\frac{(n+1)}{2}\right) &< e_1 \sum_{i=1}^{n} \left(ni - n\frac{(n+1)}{2}\right) \\ &= e_1 \left(\sum_{i=1}^{n} ni - n^2\frac{(n+1)}{2}\right) \\ &= e_1 \left(n\left(n\frac{(n+1)}{2}\right) - n^2\frac{(n+1)}{2}\right) \\ &= 0 \end{aligned}$$

while the denominator is positive by using the cauchy-schwarz inequality with sequences $\{i\}_i$ and $\{1\}_i$. Since trends with both time vectors are multiple, it is true that sampling with $\mathbf{t}$ must also give a negative trend. This completes the proof. □

All formalities aside, a regression test for data is useful as to determine the possibility of stalling. The implementation is given in Secs. **??** and **??**.

1.4.2 Online update improvement

After some more familiarization with these topics, I've improved all formulas to the point in making its implementation real-time; that is, tweaking these estimations to make it recursive. For the sake of understandability, Define a placeholder for multiplicative sums as follows:

$$s_{x_1 x_2 \ldots x_m}(n) = \sum_{i=1}^{n} \left(\prod_{j=1}^{m} x_{ij}\right)$$

then, it is possible to generate an updating factor for the linear regression parameters as follows. let $D(n) = ns_{xx}(n) - (s_x(n))^2$, and:

$$\begin{aligned} D(n+1)a_{n+1} &= (n+1)s_{xy}(n+1) - s_x(n+1)s_y(n+1) \\ &= ns_{xy}(n) + s_{xy}(n) + (n+1)x_{n+1}y_{n+1} - (s_x(n) + x_{n+1})(s_y(n) + y_{n+1}) \\ &= D(n)a_n + s_{xy}(n) + (n+1)x_{n+1}y_{n+1} - s_x(n)y_{n+1} - s_y(n)x_{n+1} - x_{n+1}y_{n+1} \\ &= D(n)a_n + s_{xy}(n) + nx_{n+1}y_{n+1} - s_x(n)y_{n+1} - s_y(n)x_{n+1} \end{aligned}$$

The updating law for the denominator is also easy to derive and is much related to what has just been done:

$$\begin{aligned} D(n+1) &= (n+1)s_{xx}(n+1) - s_x(n+1)s_x(n+1) \\ &= ns_{xx}(n) + s_{xx}(n) + (n+1)x_{n+1}x_{n+1} - (s_x(n) + x_{n+1})(s_x(n) + x_{n+1}) \\ &= D(n) + s_{xx}(n) + (n+1)x_{n+1}x_{n+1} - s_x(n)x_{n+1} - s_x(n)x_{n+1} - x_{n+1}x_{n+1} \\ &= D(n) + s_{xx}(n) + nx_{n+1}^2 - 2s_x(n)x_{n+1} \end{aligned}$$

this completes the characterization of all the updating laws for the trend. To achieve same results for the linear constant, a similiar deduction can be employed:

$$\begin{aligned} D(n+1)b_{n+1} &= s_y(n+1)s_{xx}(n+1) - s_x(n+1)s_{xy}(n+1) \\ &= s_y(n)s_{xx}(n) + s_y(n)x_{n+1}^2 + y_{n+1}(s_{xx}(n) + x_{n+1}^2) - s_x(n)s_{xy}(n) - s_x(n)y_{n+1} - x_{n+1}(s_{xy}(n) + x_{n+1}y_{n+1}) \\ &= D(n)b_n + s_y(n)x_{n+1}^2 + y_{n+1}(s_{xx}(n) + x_{n+1}^2) - s_x(n)y_{n+1} - x_{n+1}(s_{xy}(n) + x_{n+1}y_{n+1}) \\ &= D(n)b_n + s_y(n)x_{n+1}^2 + y_{n+1}s_{xx}(n) - s_x(n)y_{n+1} - x_{n+1}s_{xy}(n) \end{aligned}$$

these can all be brought together to form a online estimation of the linear trend and constant.

The downside is that at least to values are required in order to start the algorithm correctly and attain the same estimation as the non-recursive version of LSR. Another downside would be the eventual digital saturation of the microcontroller. This algorithm is equivalent to re-running a full regression after every smapling is done; thus, the possiblity exists. Although it would require that a operation would remain stable and sane for a extended period of time.

Chapter 2

Implementation

2.1 Motor attachments

Since the motor to be tested do not fit right away in in the test rig, two parts have been designed to fulfill the purpose of fixing it in place. These are an axis attachment to expand the final gear ratio as seen in Fig. 2.1 and an outer ring to secure it in place as seen in Fig. 2.2.

The CAD program used in the design was the solid edge proprietary software, by siemens.

2.2 Arduino programming

The choice was a modular approach so that the next person in the project can work with what was done without having to look the entire code again and understand each line. To accomplish modularity, the semi-complete C/C++ capabilities of the arduino platform were used; abstraction of the hardware components such as engine, sensor etc were encapsulated into new data structures and respective allocating fucntions; in the same manner as a class design would be done.

2.3 Computer Interface

The Graphical Interface is build upon the Qt framework to strike a balance between deployment and speed. A quick picture if it can be seen in Fig. 2.3. In the underlying code, the ch

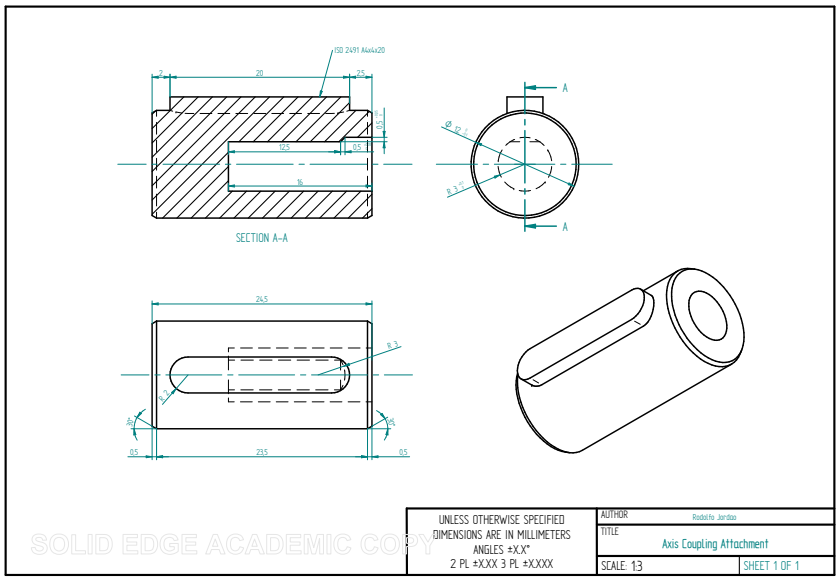

Figure 2.1: Motor axis coupling drawing.

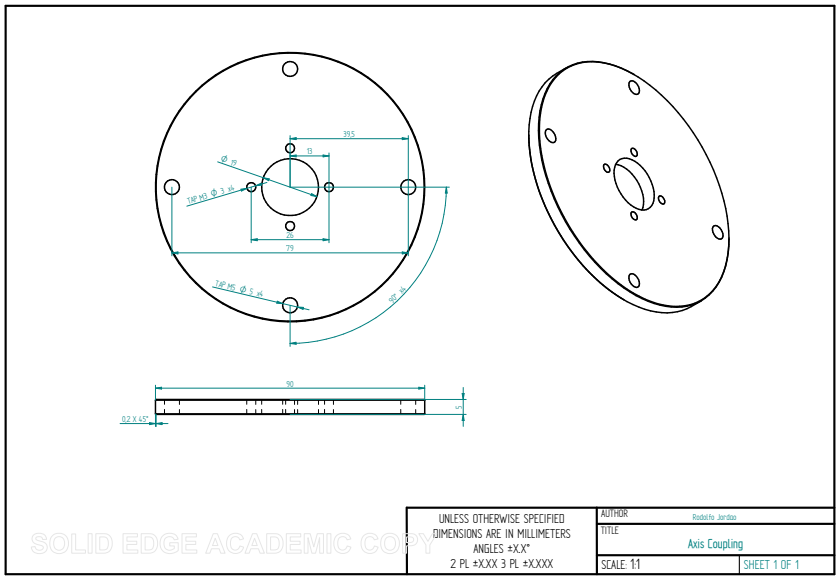

Figure 2.2: Motor ring coupling drawing.

Figure 2.3: Controller Prototype interface.

Bibliography

Appendix A

Deprecated Stuff

A.1 Stall Detection (Deprecated)

It really depends upon the behavior of the system, but if focus is mode on the speed error, then some things can be asserted about the mathematical model:

Definition 3 (Stall)**.** The positive definite error function $e(t)$ is said to be *stalling about* L if there is $t_o \in \mathbb{R}$ such that $\forall t > t_0, |e(t) - L| < \epsilon$ where $\epsilon \in \mathbb{R}$. In words, the error $e(t)$ is *stalling* if it oscillates (ou stabilizates) about some value L. The discrete version is similar.

Stalling formally defined, it is easier to analyze and develop any algorithm. In a correct working mechanism, the error will just drop continuously to zero; hence, in an ideal condition the error function should be *strictly monotonic deceasing.* The data available consists of a sequence of real number $\{e(T), \ldots, e(nT)\}$ sampled approximately at times kT, where $k \in \mathbb{Z}$ and $T \in \mathbb{R}$.

now, the behavior of $e(t)$ in respect to time by sampling can be tracked by many different ways, three based in direct measure, sampled derivative and sampled integral, respectively, follows:

Proposition 2 (Straight check)**.** *if $e(nT) \geq e(T)$, then $e(t)$ is not strictly monotonic decreasing in the interval.*

Proof. Follows straight from definition. □

Proposition 3 (Ratio integral test)**.** *if*

$$\sum_{i=1}^{N} \frac{e_{i+1}}{e_i} \geq N$$

where $e_i = e(iT)$, then $e(t)$ is not strictly monotonic decreasing.

Proof. Suppose that $e(t)$ is monotonic decreasing and let

$$\sum_{i=1}^{N} \frac{e_{i+1}}{e_i} \geq N$$

where N is the size of the sampled set. Now, since $e(t)$ is monotonic decreasing,

$$\frac{e_{k+1}}{e_k} \leq 1 \quad \forall k \in \mathbb{N}$$

it follows that,Suppose

$$\sum_{i=1}^{N} \frac{e_{i+1}}{e_i} < vectorN$$

which is a contradiction. Thus $e(t)$ cannot be strictly monotonic decreasing. □

Proposition 4 (Meta integral test)**.** *let*

$$I(a, b) = \sum_{i=a+1}^{b} e_k$$

where $a, b \in \mathbb{N}$. If,

$$I(0, N) \leq I(N, 2N)$$

then $e(t)$ cannot be strictly monotonic decreasing.

Proof. Suppose that $e(t)$ is strictly monotonic decreasing. From definition it follows that,

$$e_k > e_{k+a} \quad \forall k \in \mathbb{N}$$

where $a \in \mathbb{N}$. Now, summing this inequality with $a = N$ from 1 to N,

$$I(0,N) = \sum_{i=1}^{N} e_k > \sum_{i=1}^{N} e_{k+N} = \sum_{i=N+1}^{2N} e_k = I(N,2N)$$

henceforth, if

$$I(0,N)l \leq I(N,2N)$$

a contradiction follows. This completes the proof. □

These propositions can be used to test for stalling of the motor, since there is a intimate relationship between stalling and a monotonic decreasing function with error checking. For instance, the error function by definition should stall at the final error value, as being the objective; but it should not stall in values greater than the tolerance for error.

Fact 2. *if the error function $e(t)$ is not strictly monotonic decreasing, it is either stalling on some value or increasing.*

From this fact, it becomes clear that the checks for a strictly monotonic decreasing function followed by a check of the total error of the system returns a safe estimate of its stalling. The actual implementation can use the three tests as to increase security. This is done at Sec. A.2.

A.2 Stall detection (deprecated)

The implementation is rather simple and it's based on simultaneous tests pass. If more two out of three tests asserts that the system is possibly stalling, the system is shutdown if the error is above some tolerance. The code is show in Alg. 1 in pseudo code.

```
Program Stall_detection
Var error_decision: integer
error_decision := 0
If straight_error_check is true Then
error_decision := error_decision + 1
If ratio_integral_check is true Then
error_decision := error_decision + 1
If meta_integral_check is true Then
error_decision := error_decision + 1
If error_decision > 2 Then
stop motor
Else
error_decision := 0
Rerun Stall_detection after sampling
End
```

Algorithm 1: Error Algorithm.