AE353: Design Problem 04

Two-Wheel Robotics

April 11, 2019

1 Goal

This simulation will follow the life of a two-wheeled robot – known as Bobby – as it utilizes its individually torque-powered wheels to follow a generated path in Matlab. The robot is equipped with two actuators which control τ_R and τ_L, the torque of the individual wheels and a smart sensors which tracks lateral error, heading error, the radius of curvature of the road, and the angular velocity of each wheel.

Figure 1: Modern robot friend by Segway (Photo from Mac Sources).

1.1 Requirements

The robot is required to reach and maintain a forward-oriented velocity of $3\frac{\text{m}}{\text{s}}$. The robot must not crash, diverge from path, or exceed a pitch angle of $\phi = 30^\circ$ in order to maintain optimal speed and stability.

1.2 Verification

The simulation will contain a verification test to ensure the requirements are satisfied. This will be done by checking if $v \geq 3\frac{\text{m}}{\text{s}}$. The `processdata.v` vs. `processdata.t` will be plotted to verify whether the robot met the desired velocity within the time requirement.

2 Model

The system of non-linearized ODE's defining the equations of motion are provided as `DesignProblem04_EOMs.mat` and will be accessed in conjunction with `fsolve()` and `jacobian()` functions to determine equilibrium points and state space model. Three separate controller tests will be created: slow, speed, optimized speed. The slow test will verify the robot's controller can successfully allow navigation at slow speeds; the speed test will attempt to make the robot reach the required speed but will likely fail offering valuable information for future tests; the optimized speed test will satisfy the requirements.

2.1 Robot Dynamics

In general, the non-linearized differential system may be defined as shown in eqn 1:

$$\begin{bmatrix} \ddot{\phi} \\ \dot{v} \\ \dot{w} \end{bmatrix} = f(\phi, \dot{\phi}, v, w, \tau_R, \tau_L), \quad \text{State} = \begin{bmatrix} \dot{\phi} - \dot{\phi}_E \\ v - v_E \\ w - w_E \\ \phi - \phi_E \\ e_{lateral} - e_{lateral,E} \\ e_{heading} - e_{heading,E} \end{bmatrix}$$
$$\text{Input} = \begin{bmatrix} \tau_R - \tau_{R,E} \\ \tau_L - \tau_{L,E} \end{bmatrix}, \qquad \text{Output} = \begin{bmatrix} e_{lateral} - e_{lateral,E} \\ e_{heading} - e_{heading,E} \\ \omega_L - \omega_{L,E} \\ \omega_R - \omega_{R,E} \end{bmatrix} \tag{1}$$

Each of the variables correspond to the following real-life values as defined in table 2

$\phi \rightarrow$	Chassis Angle	$\dot{\phi} \rightarrow$	Change in Chassis Angle
$v \rightarrow$	Forward Velocity	$w \rightarrow$	Turning rate
$\tau_L \rightarrow$	Left Motor Torque	$\tau_R \rightarrow$	Right Motor Torque

(2)

The simulator is designed to accept three equations of motion in reference to the position of the robot, (x, y) and θ, defined by (3):

$$\begin{aligned} \dot{x} &= v\cos\theta \\ \dot{y} &= v\sin\theta \\ \dot{\theta} &= w \end{aligned} \tag{3}$$

In order to let the robot keep track of the centerline of the road, equation (4) is implemented to provide a method of letting the robot mitigate the error between robot position and the centerline. Additionally, 5 outlines the effects of the two ω values [1]:

$$\begin{aligned} w_{road} &= \frac{v_{road}}{r_{road}} \\ \dot{e}_{lateral} &= -v\sin(e_{heading}) \\ \dot{e}_{heading} &= w - \left(\frac{v\cos(e_{heading})}{v_{road} + w_{road}e_{lateral}}\right)w_{road} \end{aligned} \tag{4}$$

$$\omega_R = \frac{(v + \frac{\omega b}{2})}{r}, \quad \omega_L = \frac{(v - \frac{\omega b}{2})}{r} \tag{5}$$

2.2 Equilibrium Values

In order to find equilibrium values which best fit the requirements, the following values in table 6 were selected for the first and second trials:

$\dot{\phi}$	v	w	ϕ	$e_{lateral}$	$e_{heading}$	τ_R	τ_L	r_{road}	v_{road}
0	2	0	0	0	0	0	0	∞	2
0	3	0	0	0	0	0	0	∞	2

(6)

2.3 State Space Model

By use of the `fsolve()` to find equilibrium values and `jacobian()` to find the coefficients for the state-space matrix, the generalized State Space model may be defined as shown in 7:

$$\begin{aligned} \dot{m} &= Am + Bu \\ n &= Cm + Du \end{aligned} \tag{7}$$

While there were three different controllers, the difference among trials arose from modifying the gain matrix for LQR. The coefficients, A, B, and C may be found in 8:

$$A = \begin{bmatrix} 0 & 0 & 0 & 27.0 & 0 & 0 \\ 0 & 0 & 0 & -3.5 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 1.0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -3.0 \\ 0 & 0 & 1.0 & 0 & 0 & 0 \end{bmatrix}, \quad B = \begin{bmatrix} -2.2 & -2.2 \\ 0.59 & 0.59 \\ 2.5 & -2.5 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}, \quad C = \begin{bmatrix} 0 & 0 & 0 & 0 & 1.0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1.0 \\ 0 & 5.0 & 1.0 & 0 & 0 & 0 \\ 0 & 5.0 & -1.0 & 0 & 0 & 0 \end{bmatrix} \tag{8}$$

3 Controller Design

3.1 Controller

To begin implementing the controller, the input must first be defined as in 9:

$$u = -Km \tag{9}$$

In order to find an optimized K value, the data found in Section 2.3 will be used with a gain matrix to evaluate `K = lqr`(A, B, Q_c, R_c). It is very important to note the Controller gains ratio is extremely sensitive and can result in critical failure depending on initial values. The open loop controllability may be tested by determining the full-rank and condition of the controllability matrix, `cond(ctrb(A,B)); rank(ctrb(A,B))`. Closed-loop asymptotic stability was verified by use of finding all `eig(A−BK)` values negative. The controller was successfully verified as controllable and asymptotically stable.

3.2 Observer

Since the only sensors equipped on the robot are $e_{lateral}, e_{heading}, \omega_R, \omega_L,$ and r_{road}, it is crucial to create an observer. This may be done by first finding an optimized value for L by `L = lqr`$(A', C', R_o^{-1}, Q_o^{-1})$. The ultimate observer is designed as follows in 10:

$$\hat{\dot{m}} = A\hat{m} + Bu - L(C\hat{m} - n) \tag{10}$$

In order to determine the open-loop observability, the condition number and rank may be used by finding `cond(obsv(A, C)); rank(obsv(A,C))`. The closed-loop asymptotic stability may be verified by determining the signs of `eig(A−LC)`. Both were verified and therefore the observer is both observable and asymptotically stable.

4 Simulation Tests

With the observer and controller completed, the two actuators may be assigned as follows: `actuators.tauR = u(1)+ tauRE; actuators.tauL = u(2)+ tauLE` for τ_{RE} and τ_{LE} equilibrium values for τ. Since both conditions numbers were on the order of magnitude of $e4$ and $e3$, a relatively high ratio of $\frac{Q}{R}$ will need to be implemented.

4.1 Trial 1

In the first, slower trial, a low equilibrium velocity of $v = 2$ meters per second was designated to analyze whether the controller and observer operate nominally. The results were positive with an operational robot tracking the road.

4.2 Trial 2

As the equilibrium value was bumped up to the required velocity of $v = 3$ meters per second, the robot successfully tracked the majority of the road but failed whenever the robot reached a double-bend or 180° turn, indicating the controllability matrix was not powerful enough to provide quicker reactions to the actuators. Hypothesis: increasing K value by changing the $\frac{Q_c}{R_c}$ ratio will let the robot properly adjust the actuators.

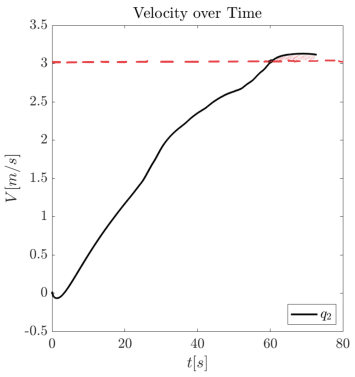

Figure 2: Trial 3 Successful Velocity

4.3 Trial 3

Inspiration for a modified I matrix was shown in [2] and provided a useful matrix designation for Q_c, resulting in a stable and efficient K value. The Q matrix is defined by 11:

$$Q_c = \begin{bmatrix} 6 \cdot 10^4 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 4 \cdot 10^2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}, \qquad R_c = 200 \cdot \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \tag{11}$$

This resulted in a significantly more stable controller, capable of reaching the required speed and passing the verification.

References

[1] M. West, "Rolling motion dynamics reference." [Online]. Available: http://dynref.engr.illinois.edu/rko.html

[2] P. N. Brian Bonafilia, Nicklas Gustafsson and S. Nilsson, "Self-balancing two-wheeled robot." [Online]. Available: https://pdfs.semanticscholar.org/ad7a/3b833695e8e01c1d41043c75c770276c923c.pdf