Preprints of the 21st IFAC World Congress (Virtual)
Berlin, Germany, July 12-17, 2020

LQG controller for the LEGO MINDSTORMS EV3 Gyroboy Segway robot

Timothy H. Hughes * **Gareth H. Willetts** **
Jakub A. Kryczka ***

* *University of Exeter, Penryn Campus, Cornwall, U.K. (e-mail: T.H.Hughes@exeter.ac.uk).*
** *University of Exeter, Penryn Campus, Cornwall, U.K. (e-mail: ghw205@exeter.ac.uk)*
*** *University of Exeter, Streatham Campus, Devon, U.K. (e-mail: jak222@exeter.ac.uk)*

Abstract: This paper details the development of a Segway robot demonstrator for undergraduate and Masters level systems and control courses based on the LEGO MINDSTORMS EV3 robotics platform. The purpose of the demonstrator is to provide a physical and interactive device for explaining concepts that feature on many systems and control courses, notably: model-based control, linearisation, the Linear-Quadratic Regulator (LQR), pole placement, noise amplification due to differentiation, reference following, Kalman filtering, and the principle of separation of estimation and control. The demonstrator is designed using a standard LEGO MINDSTORMS model from the Education EV3 core set—the Gyroboy. This is interfaced with using the Simulink Support Package for LEGO MINDSTORMS EV3 Hardware. Reference inputs can be provided from either a keyboard or an Xbox One gamepad. To the best of our knowledge, this is the first example of the successful implementation of an observer-based reference-following feedback controller for a Segway robot built entirely using LEGO MINDSTORMS EV3 components, with previous designs being either not observer-based or based on the now outdated LEGO MINDSTORMS NXT platform.

Keywords: Linear-Quadratic-Gaussian control, Observer-based control, Control education, Model-based control, Linear-Quadratic Regulator, Pole placement, Kalman filter

1. INTRODUCTION

The value of LEGO MINDSTORMS for the teaching of control and robotics has widespread recognition (see, e.g., Canale and Brunet, 2013; Basso et al., 2013; Villacrés et al., 2016; Kim, 2011; Cruz-Martín et al., 2012; Markovsky, 2012). To date, most of the educational applications of LEGO MINDSTORMS detailed in the literature have focussed on the second generation NXT kit, which was superceded in 2013 by the third generation EV3 model. The use of LEGO MINDSTORMS in higher education is facilitated by the hardware interfacing capacities offered by the LEGO MINDSTORMS EV3 Support for Simulink package, which allows for controllers to be designed within MATLAB Simulink and then deployed directly to the LEGO MINDSTORMS EV3 intelligent brick. This also allows for data to be returned to the MATLAB workspace from LEGO MINDSTORMS sensors to inform controller design and enhance the teaching experience.

This paper details the design of a reference-following Linear-Quadratic-Gaussian (LQG) controller for the LEGO MINDSTORMS EV3 Gyroboy set. This is a two-wheeled Segway robot, comprised of the EV3 intelligent brick (the microprocessor for the robot), a gyroscopic sensor for measuring the Gyroboy's tilt velocity, and two electric motors, one for each wheel, each of which contains an encoder that measures the angle through which that motor has turned.

The developed demonstrator was integrated into a new third-year course on the Mathematical Sciences degree at the University of Exeter, UK, titled *Dynamical Systems and Control*. The intention was to provide an interactive physical demonstration of the key concepts introduced during the course, to improve students' engagement and their intuitive and conceptual understanding. The demonstrator also sought to develop students' awareness of the methods' limitations and the importance of considering the practical aspects of design alongside the theoretical. Moreover, the methods used to design the Gyroboy controller were mirrored in a simulation based coursework exercise in MATLAB Simulink, whereby students' understanding is reinforced through problem-based learning.

Several authors have noted the positive impact of LEGO MINDSTORMS on student motivation and outcomes, (see e.g., Panadero et al., 2010; Cruz-Martín et al., 2012). Since the demonstrator described in this paper was developed for a new course, it is not possible here to assess the effectiveness of the demonstrator in improving outcomes, but we do note that students assessed the module positively (with a mean score of 90% and a range of 77-100%).

In this paper, we describe the details of the demonstrator's design, and its use in demonstrating model-based control, linearisation, the Linear-Quadratic regulator (LQR), pole placement, noise amplification due to differentiation, reference following, Kalman filtering, and the principle of separation of estimation and control. MATLAB and Simulink files, and videos of the design implemented on the Gyroboy, have been made available by (Hughes et al., n.d.). The merits of the demonstrator that make it stand out relative to other teaching resources are its versatility (for demonstrating all of the aforementioned concepts), simplicity (the design only requires knowledge of control concepts that are commonly covered in undergraduate or Masters level courses, and the hardware interfacing can all be carried out within MATLAB Simulink), and affordability (the equipment cost is approximately $650).

Applications of LQR, model predictive control, sliding mode control and fuzzy logic control to the stabilisation of LEGO Segway robots have previously appeared in the literature (see Canale and Brunet, 2013; Villacrés et al., 2016; Yamamoto, n.d.; Akmal et al., 2017; Behera and Mija, 2016). Our approach in this paper takes the LQR designs of Yamamoto (n.d.) and Roslovets (n.d.) as a starting point, and extends these designs in a number of directions. It is shown that the design of forward and turning motion controllers of the Segway robot can be decoupled. Explicit expressions are provided for the controller gains as a function of the closed loop characteristic polynomial, and pole placement is then used in order to improve the forward motion by altering the fast dynamics of the initial LQR design. Also, a Kalman filter based observer is implemented, based on an estimate of input and output noise due to quantisation and other sensor errors, and the resulting LQG controller demonstrates a substantially smoother behaviour. Finally, the robot is designed to take reference inputs from either a keyboard or an Xbox One gamepad.

2. OPEN LOOP DYNAMICS

Following Yamamoto (n.d.), the open loop dynamics for the Gyroboy can be described as a system with state vector

$$x = \left[\theta \;\; \psi \;\; \frac{d\theta}{dt} \;\; \frac{d\psi}{dt} \;\; \phi \;\; \frac{d\phi}{dt}\right]^T,$$

where ψ is the tilt angle (in radians); θ is the average of the two wheel angles (in radians), which depends on the angles θ_l and θ_r of the left and right motor through the relationship $\theta - \psi = \frac{1}{2}(\theta_l + \theta_r)$; and ϕ is the rotation angle about the vertical (in radians), in accordance with Fig. 1.

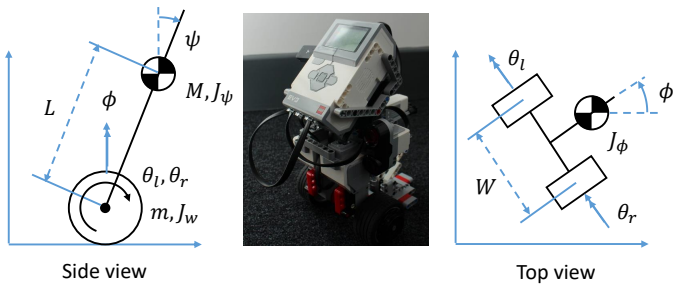

Fig. 1. LEGO MINDSTORMS EV3 Gyroboy schematic.

Estimates of the Gyroboy's properties are as follows:

$m = 0.031$kg (wheel's mass),
$M = 0.641$kg (body's mass),
$g = 9.81\text{ms}^{-2}$ (acceleration due to gravity),
$L = 0.107$m (distance from wheel axis to centre of mass)
$R = 0.027$m (wheel radius),
$J_w = 1.13\times10^{-5}\text{kgm}^2$ (wheel's moment of inertia),
$J_\psi = 2.77\times10^{-3}\text{kgm}^2$ (body's pitch moment of inertia),
$W = 0.105$m (body's width),
$J_\phi = 1.12\times10^{-3}\text{kgm}^2$ (body's yaw moment of inertia)
$\alpha = 0.505\times10^{-3}$ (motor calibration constant),
$\beta = 3.58\times10^{-3}$ (motor calibration constant).

The estimation of these constants required only the LEGO MINDSTORMS EV3 sensors themselves, in addition to a ruler and a pair of calipers (for measuring L, R and W), and a set of gram-accurate weighing scales (for measuring m and M). The remaining constants are then estimated as follows. By measuring the time period of pendulum oscillations (T) when the Gyroboy robot swings freely about the wheel axis, the moment of inertia J_ψ can be estimated using the formula $J_\psi + ML^2 = MgLT^2/((2\pi)^2)$.

To obtain the moments of inertia J_ϕ and J_w, the depth of the Gyroboy robot is estimated as $D = 0.1$m, and the mass of the body and wheel is assumed to be evenly distributed, whereupon $J_\phi = M(W^2 + D^2)/12$ and $J_w = (mR^2)/2$.

The motor calibration constants α and β correspond to coefficients in a first order model of LEGO EV3 large servo motor's dynamics. This motor takes integer inputs from -100 to +100, and α and β have units of $\text{kgm}^2\text{s}^{-2}$ per integer unit and $\text{kgm}^2\text{s}^{-1}$ per integer unit, respectively. These parameters have been estimated by adding a third wheel to balance the Gyroboy, and then applying a step input to both motors of the Gyroboy. The dynamics of this system is then described by the differential equation

$$\left(M + 2m + \frac{2J_w}{R^2}\right)\frac{d^2z}{dt^2} + \frac{2\beta}{R^2}\frac{dz}{dt} = \frac{2\alpha u}{R}. \qquad (1)$$

Here, u refers to the integer input to the motors, and z refers to the horizontal distance of travel. The parameters α and β can then be estimated by applying a step input of magnitude $U = 100$ at time $T_0 = 1$. By fitting a straight line to the steady-state behaviour of the cart, and determining the slope V of this line and the intercept T_1 with the time axis, then estimates for α and β that are consistent with equation (1) are obtained as follows:

$$\beta = \frac{R^2(M+2m)+2J_w}{2(T_1 - T_0)} \text{ and } \alpha = \frac{\beta V}{RU}.$$

The correspondence between the experimental data and the predictions of the fitted model are shown in Fig. 2.

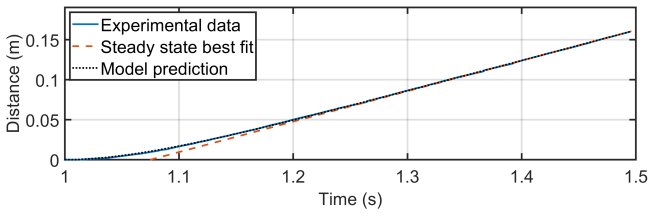

Fig. 2. Motor calibration experiment.

To model the Gyroboy's dynamics, we let
$u_1{=}\frac{1}{2}(v_l{+}v_r)$, $u_2{=}\frac{1}{2}(v_r{-}v_l)$, $q_1{=}u_1{+}w_{1a}$ and $q_2{=}u_1{+}w_{2a}$. Here, v_l and v_r are the desired inputs to the left and right motors, respectively. Also, w_{1a} and w_{2a} correspond to quantisation and saturation errors, which arise as the motors only accept integer inputs between -100 and $+100$. Then, with the notation

$$\gamma_1 = MgL, \quad \gamma_2 = MLR, \quad \gamma_3 = 2J_w + (2m+M)R^2,$$
$$\gamma_4 = ML^2 + J_\psi, \text{ and } \gamma_5 = ML^2,$$

the dynamics of the Gyroboy robot satisfy the equation $\frac{dx}{dt} = f(x, z_1, z_2)$, where $f(x, z_1, z_2)$ takes the form

$$[f_1(x)\ f_2(x)\ f_3(x,z_1)\ f_4(x,z_1)\ f_5(x)\ f_6(x,z_2)]^T,$$

with $f_1(x) = \frac{d\theta}{dt}$, $f_2(x) = \frac{d\psi}{dt}$, $f_5(x) = \frac{d\phi}{dt}$,

$$f_3(x,z_1) = \frac{\gamma_2 \sin(\psi)\left(\gamma_4\left(\frac{d\psi}{dt}\right)^2 - \gamma_1\cos(\psi) - \gamma_5\left(\frac{d\phi}{dt}\right)^2\cos^2(\psi)\right)}{\gamma_3\gamma_4 - \gamma_2^2\cos^2(\psi)} + \frac{2\left(\alpha z_1 + \beta\left(\frac{d\psi}{dt} - \frac{d\theta}{dt}\right)\right)(\gamma_4 + \gamma_2\cos(\psi))}{\gamma_3\gamma_4 - \gamma_2^2\cos^2(\psi)},$$

$$f_4(x,z_1) = \frac{\gamma_3 \sin(\psi)\left(\gamma_1 + \gamma_5\left(\frac{d\phi}{dt}\right)^2\cos(\psi)\right) - \gamma_2^2\sin(\psi)\cos(\psi)\left(\frac{d\psi}{dt}\right)^2}{\gamma_3\gamma_4 - \gamma_2^2\cos^2(\psi)} - \frac{2\left(\alpha z_1 + \beta\left(\frac{d\psi}{dt} - \frac{d\theta}{dt}\right)\right)(\gamma_3 + \gamma_2\cos(\psi))}{\gamma_3\gamma_4 - \gamma_2^2\cos^2(\psi)}, \text{ and}$$

$$f_6(x,z_2) = \frac{\frac{W}{R}\alpha z_2 - \frac{W^2}{2R^2}\beta\frac{d\phi}{dt} - 2ML^2\frac{d\psi}{dt}\frac{d\phi}{dt}\sin(\psi)\cos(\psi)}{\frac{mW^2}{2} + J_\phi + \frac{W^2}{2R^2}J_w + ML^2\sin^2(\psi)}.$$

The derivation of these equations follows Yamamoto (n.d.). The motor's moment of inertia has been omitted as it is negligible in comparison with the body pitch and body yaw moments of inertia, so the error due to its omission is deemed insignificant relative to other model errors.

By partitioning the system's states into the *forward states* x_1 and *turning states* x_2 thus,

$$x_1 = \left[\theta\ \psi\ \frac{d\theta}{dt}\ \frac{d\psi}{dt}\right]^T \text{ and } x_2 = \left[\phi\ \frac{d\phi}{dt}\right]^T, \tag{2}$$

and by linearising this system about the equilibrium point at the origin, we obtain an approximation for the dynamics of the LEGO Gyroboy robot of the form

$$\frac{dx_1}{dt} = A_1x_1 + B_1(u_1 + w_{1a}) \text{ and} \tag{3}$$
$$\frac{dx_2}{dt} = A_2x_2 + B_2(u_2 + w_{2a}), \text{ where} \tag{4}$$
$$A_1 = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & -\frac{\gamma_1\gamma_2}{\gamma_3\gamma_4-\gamma_2^2} & -\frac{2\beta(\gamma_2+\gamma_4)}{\gamma_3\gamma_4-\gamma_2^2} & \frac{2\beta(\gamma_2+\gamma_4)}{\gamma_3\gamma_4-\gamma_2^2} \\ 0 & \frac{\gamma_1\gamma_3}{\gamma_3\gamma_4-\gamma_2^2} & \frac{2\beta(\gamma_2+\gamma_3)}{\gamma_3\gamma_4-\gamma_2^2} & -\frac{2\beta(\gamma_2+\gamma_3)}{\gamma_3\gamma_4-\gamma_2^2} \end{bmatrix}, \tag{5}$$
$$B_1 = \begin{bmatrix} 0 & 0 & \frac{2\alpha(\gamma_2+\gamma_4)}{\gamma_3\gamma_4-\gamma_2^2} & -\frac{2\alpha(\gamma_2+\gamma_3)}{\gamma_3\gamma_4-\gamma_2^2} \end{bmatrix}^T, \tag{6}$$
$$A_2 = \begin{bmatrix} 0 & 1 \\ 0 & \frac{-\beta W^2}{2J_\phi R^2 + (mR^2+J_w)W^2} \end{bmatrix}, \text{ and} \tag{7}$$
$$B_2 = \begin{bmatrix} 0 & \frac{2R\alpha W}{2J_\phi R^2 + (mR^2+J_w)W^2} \end{bmatrix}^T. \tag{8}$$

There are three sensors on the Gyroboy: the gyroscopic sensor measuring tilt angular velocity, and an encoder on each of the two wheels measuring the two motor angles θ_l and θ_r. The gyroscopic sensor exhibits a steady-state offset and can drift (see e.g., (Canale and Brunet, 2013, Sec. IID) and (Basso et al., 2013, Sec. III)). Following some experimentation, the following procedure was found to effectively compensate for these effects. First, the offset is estimated in an initial calibration phase, in which the Gyroboy is supported upright. This offset is subsequently subtracted from the gyroscopic sensor measurement. Moreover, to correct for calibration errors in this offset, the resulting signal is passed through a (discrete-time) high-pass filter with a time constant of 10 seconds. Finally, to compensate for drift, the resulting signal is integrated and passed through a further (discrete-time) high-pass filter with the same time constant, thereby providing an estimate for the tilt angle (in radians). [1] We then take as forward output y_1 a vector whose first entry is the average wheel angle estimate $(\theta_l + \theta_r)/2$ (in radians), and whose second entry is the tilt angle estimate from the filtered gyroscopic sensor measurement. Also, as turning output y_2 we take the estimate for the turning angle $\frac{R}{W}(\theta_r - \theta_l)$ (again in radians). These outputs then satisfy the relationships

$$y_1 = C_1x_1 + w_{1b} \text{ and } y_2 = C_2x_2 + w_{2b} \tag{9}$$

where w_{1b} and w_{2b} are disturbances due to sensor noise,

$$C_1 = \begin{bmatrix} 1 & -1 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}, \text{ and } C_2 = [1\ 0]. \tag{10}$$

3. CONTROLLER DESIGN

In this section, we present the stages in the design of an observer-based reference-following feedback controller in an incremental manner that can be followed in an undergraduate control systems course. We first present a reference-following proportional controller for the turning motion. Subsequently, we present the development of the forward motion controller, which illustrates the concepts of LQR, pole placement, noise amplification due to differentiation, Kalman filtering, and the principle of separation of estimation and control. The final design comprises a reference-following LQG controller that accepts reference inputs from either a keyboard or an Xbox One gamepad.

3.1 TURNING MOTION CONTROLLER

The turning motion of the Gyroboy is an example of a second order system that can be stabilised using proportional feedback. The error w_{2b} in the estimate of the turn angle y_2 in equation (9) is dominated by quantisation errors in the motor encoders, which measure motor angles to the nearest degree, and is sufficiently small to be neglected. Similarly, the motor quantisation error w_{2a} in equation (4) will also be neglected. The open-loop turning dynamics are then approximated in the vicinity of the upright equilibrium point by the two-state system $\frac{dx_2}{dt} = A_2x_2 + B_2u_2$, where $x_2 = \left[\phi\ \frac{d\phi}{dt}\right]^T$, and A_2 and B_2 are as given in equations (7) and (8). It is easily verified that this model of the open loop turning dynamics is marginally stable due to the presence of a simple eigenvalue of A_2 at the origin. The

[1] It should be noted that this filtering method is only effective in this application as an adequate controller will maintain a small tilt angle. An alternative controller was also successfully implemented that used the calibrated but unfiltered gyroscopic sensor measurement. For this controller, the angle estimate from the gyroscopic sensor drifted over time, and feedback was required on the integral of the motor angle to prevent this from causing the Gyroboy's location to drift.

system can be stabilised using the proportional control $u = -k(\phi - \phi_{\text{ref}})$, leading to the closed loop dynamics:

$$\frac{dx_2}{dt} = \begin{bmatrix} 0 & 1 \\ \frac{-2R\alpha Wk}{\Delta} & \frac{-\beta W^2}{\Delta} \end{bmatrix} x_2 + \begin{bmatrix} 0 \\ \frac{2R\alpha Wk}{\Delta} \end{bmatrix} \phi_{\text{ref}},$$

where $\Delta = 2J_\phi R^2 + (mR^2 + J_w)W^2$.

It can be verified that this model of the turning dynamics is stabilised for any given $k > 0$. The sum of the eigenvalues of the closed loop A matrix is given by its trace $(-\beta W^2/\Delta)$, and is independent of k, hence the fastest possible response of the linearised closed loop system will occur when the real part of these eigenvalues is equal to $-\beta W^2/(2\Delta)$. The value of k for critical damping can then be obtained as $k = \beta^2 W^3/(8R\alpha\Delta)$.

This is then implemented on the Gyroboy (together with a stabilising controller for the forward motion, as will be considered in Subsection 3.2). Fig. 3 shows the response of the Gyroboy when the reference signal ϕ_{ref} is ramped up for 5 seconds, held constant for 5 seconds, then ramped back down to zero. Note that the Gyroboy's response to the reference signal is slower than expected from the model, owing in part to the coupling between the turning and forward dynamics in the real system.

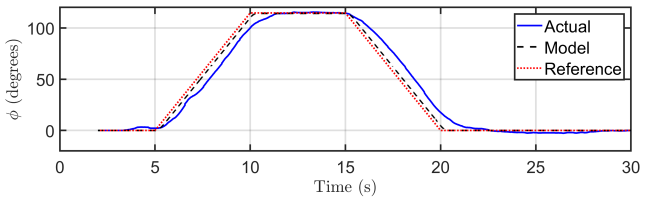

Fig. 3. Experimental and simulation data for the turning motion of the Gyroboy.

3.2 FORWARD MOTION CONTROLLER

The forward motion of the Gyroboy is modelled by the four state system in equation (3), whose state vector is as in equation (2), and where A_1 and B_1 are as given in equations (5) and (6). The measured forward outputs y_1 then take the form of equation (9). To begin with, we neglect the noise terms w_{1a} and w_{1b}, whereupon $\theta = [1\ 1]\, y_1$ and $\psi = [0\ 1]\, y_1$. We subsequently use discrete differentiation to estimate the remaining forward states $\frac{d\theta}{dt}$ and $\frac{d\psi}{dt}$. A discrete low pass filter is applied to the differentiated signals with a time constant of 8ms, which is sufficiently fast as to not interfere with the robot's dynamics. Despite this filtering, this still results in particularly noisy state estimates (especially for $\frac{d\psi}{dt}$ and $\frac{d\theta}{dt}$), and noisy motor inputs, which are improved subsequently through the use of an observer.

LQR As a starting point, an LQR controller is designed. Here, we let $Q \in \mathbb{R}^{4\times4}$ denote the state penalty and $R \in \mathbb{R}$ denote the input penalty, so in the absence of any reference signal the LQR controller will minimise the integral $\int_0^\infty (x_1(t)^T Q x_1(t) + u_1(t)^T R u_1(t))\text{dt}$ for the linearised model of the Gyroboy forward motion (see, e.g., Åström and Murray, 2008, Sec. 6.3). The controller is designed using the MATLAB `lqrd` command as it is to be implemented on a sampled data system, and the state and input penalties are specified as $Q = C_1^T C_1$ and $R = 0.01$. Thus, this LQR controller penalises deviations in the mean angle of the two motors, the Gyroboy's tilt angle, and the average input applied to the two motors. Moreover, the penalty corresponding to a motor input of ten LEGO units is equivalent to the penalty corresponding to a deviation in mean motor angle, or in tilt angle, of one radian. This results in the controller gain matrix $K = [-9.83\ -913\ -17.2\ -100]$, which provides the forward motion control law $u_1 = -K(x_1 - x_{\text{ref}})$. The experimental results from implementing this controller on the Gyroboy robot are shown in the top half of Fig. 4. It can be seen that the mean wheel angle successfully tracks the reference. However, the tilt angle measurement and the motor input u_1 are very noisy, resulting in occasional saturation of the motor. Note that this LQR controller can be improved through further fine tuning, to provide students with an appreciation of heuristic methods for assigning the state and input penalty matrices Q and R.

Pole placement The LQR controller has given a good starting point for the controller design, but as the system has only a single input it is possible to obtain an explicit expression for the closed-loop characteristic equation to examine the relationship between the controller gains and the closed-loop pole locations as follows.

The closed-loop poles for the linearised model of the Gyroboy's forward motion are the eigenvalues of $A_1 - B_1 K$ Letting $K = [k_1\ k_2\ k_3\ k_4]$, then the following expression for the closed-loop characteristic polynomial is obtained:

$$s^4 + \frac{2((\gamma_2+\gamma_4)(\alpha k_3+\beta)-(\gamma_2+\gamma_3)(\alpha k_4-\beta))}{\gamma_3\gamma_4-\gamma_2^2} s^3 + \frac{2(\gamma_2+\gamma_4)\alpha k_1-2(\gamma_2+\gamma_3)\alpha k_2-\gamma_1\gamma_3}{\gamma_3\gamma_4-\gamma_2^2} s^2 - \frac{2\gamma_1(\alpha k_3+\beta)}{\gamma_3\gamma_4-\gamma_2^2} s - \frac{2\alpha\gamma_1 k_1}{\gamma_3\gamma_4-\gamma_2^2}.$$

By equating this to the general quartic function $s^4 + a_3 s^3 + a_2 s^2 + a_1 s + a_0$, we obtain four linear equations relating k_1–k_4 to a_0–a_3, which can be solved to give

$$k_1 = \frac{-a_0(\gamma_3\gamma_4-\gamma_2^2)}{2\alpha\gamma_1},\ k_2 = \frac{-((\gamma_3\gamma_4-\gamma_2^2)(\gamma_1 a_2+(\gamma_2+\gamma_4)a_0)+\gamma_1^2\gamma_3)}{2\alpha\gamma_1(\gamma_2+\gamma_3)},$$
$$k_3 = -\frac{a_1(\gamma_3\gamma_4-\gamma_2^2)}{2\alpha\gamma_1} - \frac{\beta}{\alpha}\ \&\ k_4 = \frac{\beta}{\alpha} - \frac{(\gamma_3\gamma_4-\gamma_2^2)(\gamma_1 a_3+(\gamma_2+\gamma_4)a_1)}{2\alpha\gamma_1(\gamma_2+\gamma_3)}.$$

These relationships can be used for a detailed study of the sensitivity of the pole locations to variations in the parameters, and to compute theoretical limits on the controller gains for stability. In particular, one of the issues with the LQR controller is the large amount of noise amplification owing to the relatively high controller gains. One effective method that improves this is to slow down the fastest pole of the LQR controller. Specifically, the locations of the closed-loop poles of the LQR controller are -54, -7.9, -5.5 and -1.4. By moving these to -30, -7.9, -5.5 and -1.4, it is found that the controller gains reduce to $K = [-3.65\ -591\ -12.7\ -58.0]$. This results in a smoother behaviour. The experimental results from implementation on the Gyroboy are shown on the right of Fig. 4. Relative to the LQR controller (on the left of that figure), the motor input is less noisy and rarely saturates. This is at the expense of slightly more overshoot in response to a reference input in mean wheel angle θ, and larger amplitude oscillations in wheel angle when the robot self balances while stationary. Nevertheless, the motor input remains very noisy, due primarily to noise in

ψ and the estimates of $\frac{d\psi}{dt}$ and $\frac{d\theta}{dt}$. We look to improve this next by implementing an observer.

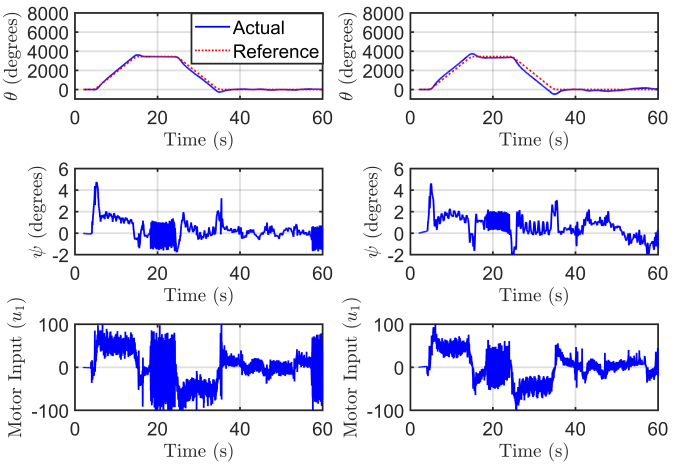

Fig. 4. Experimental results for the LQR controller (left) and pole placement controller (right).

LQR with observer The previous controller iterations, which use discrete differentiation to estimate the systems' states, provide a clear illustration of the downsides of noise amplification due to differentiation. An order of magnitude estimate of the resulting noise can be obtained by considering the errors due to quantisation, which are then multiplied by the inverse of the sample time upon differentiation, and subsequently by the controller gains, to show that this process causes jumps in the motor input equal to a substantial proportion of the motor saturation input. This is improved in this section through the implementation of a Kalman filter based observer.

We recall that the forward motion of the Gyroboy is modelled by the four state system in equation (3), where the measured forward outputs y_1 are as in equation (9). Here, w_{1a} represents errors in the motor input which are predominantly due to quantisation and saturation; the first entry in w_{2a} represents error in the estimate of the average wheel angle, which is predominantly due to quantisation; and the second entry in w_{2a} represents error in the estimate of the tilt angle, which is due to errors in the filtered gyroscopic sensor measurement. A Kalman filter is then designed to estimate the forward state x_1 for this system. This assumes that the signals w_{1a} and w_{2a} correspond to zero mean white noise with associated power spectral densities W and V, respectively.

We observe that the motors do not saturate, so the error in w_{1a} is due to quantisation only, and W can be estimated by calculating the variance of the quantisation error. As the desired forward motor input is the average of v_l and v_r, each of which is quantised to the nearest integer, then we obtain the power spectral density estimate $W = 1/24$.

Similarly, the first entry in w_{2a} is predominantly due to quantisation. This is the error in the average of the wheel angles, each of which is quantised to the nearest degree, whereupon we estimate its power spectral density as $\pi^2/(24 \times 180^2)$. The power spectral density for the second entry in w_{2a} is estimated by measuring the filtered gyroscopic sensor output when the Gyroboy is stationary. The resulting signal does not closely resemble zero mean white noise and is dominated by low frequency noise (see the left-hand side of Fig. 5). Such low frequency noise is usual due to gyroscopic sensor offset and drift, and does not have significant adverse effects on the controller's performance. We therefore adjust the high-pass filters in this experiment to have a time constant of 0.1s to reduce this low frequency noise, resulting in the signal on the right-hand side of Fig. 5. The power spectral density of this filtered signal is then computed, whereupon the following power spectral density estimate is obtained: [2]

$$V = \left(\frac{\pi}{180}\right)^2 \begin{bmatrix} \frac{1}{24} & 0 \\ 0 & 5{\times}10^{-6} \end{bmatrix}.$$

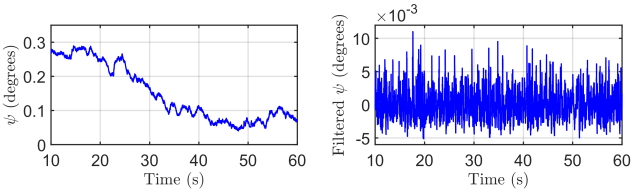

Fig. 5. Noise on tilt angle measurement. The left-hand figure uses the regular high-pass filters with time constants of 10s (as implemented in the controllers themselves). In the right-hand figure, the time constants have been adjusted to 0.1s.

The Kalman filter provides an observer for the system whose state estimate $\tilde{x}_1$ minimises the power spectral density of the state estimate error $x_1 - \tilde{x}_1$ (see, e.g., Åström and Murray, 2008, Sec. 7.4). For implementation on the LEGO MINDSTORMS intelligent brick, it is necessary to instead use a discrete time analogue to the continuous time Kalman filter, which can be obtained using the MATLAB command `kalmd`.

The forward motion input is then specified as $u = -K(\tilde{x}_1 - x_{\text{ref}})$, where K is the gain matrix corresponding to the LQR controller obtained in subsection 3.2.1, to obtain a reference following LQG controller due to the principle of separation of estimation and control (see, e.g., Åström and Murray, 2008, Sec. 7.3). The experimental results are shown in Fig. 6. The resulting behaviour of the Gyroboy is considerably smoother. However, there is notably more variation in the error of the mean motor angle (θ) particularly when the Gyroboy is rotating (at around 50s), owing to the coupling between the nonlinear forward and turning dynamics. Note that the state estimates follow the actual measurements very closely for both the mean wheel angle θ and tilt angle ψ. Also, the noise on the motor input is substantially reduced relative to the LQR and pole placement controllers in Fig. 4, owing to the estimates for the states $\frac{d\theta}{dt}$ and $\frac{d\psi}{dt}$ being considerably smoother.

3.3 REFERENCE INPUT

A reference input is provided to the Gyroboy observer as a state reference $x_{\text{ref}} = \begin{bmatrix}\theta_{\text{ref}} & 0 & \frac{d\theta_{\text{ref}}}{dt} & 0\end{bmatrix}^T$ for the for-

[2] The estimation of the bottom right entry in V is somewhat heuristic since the corresponding signal is poorly approximated as white noise. However, a good performance was observed for a range of values.

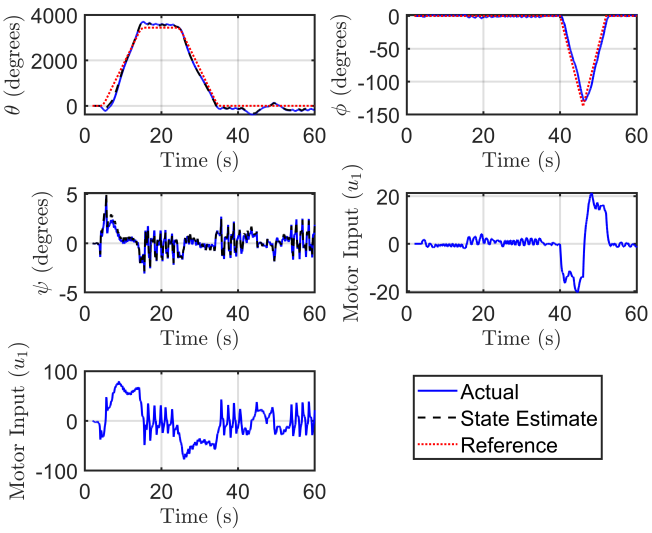

Fig. 6. Experimental results for the LQG controller.

ward motion and a turning angle reference ϕ_{ref} for the turning motion. The reference inputs used in the experiments presented in this paper were pre-programmed, but user control has also been implemented using a keyboard and an Xbox One gamepad. In each case, the resulting signal is sent to the Gyroboy using UDP send and UDP receive blocks. A MATLAB function block then contains the logic to convert these into velocity references from which the state references are subsequently computed. In the case of keyboard control, the keyboard presses are logged in Simulink using the S-function `sfun_keyboard_input_v1_01` of Compere (n.d.). For Xbox One control, the gamepad interfaces with Simulink through a Gamepad Simulator block from the Simulink Coder Support Package for ARM Cortex-based VEX Microcontroller, and is compatible with a standard Windows PC using either a USB or Bluetooth connection.

4. CONCLUSIONS AND EXTENSIONS

We presented an LQG controller for a Segway robot based on the standard LEGO MINDSTORMS EV3 Gyroboy model. This provides physical and interactive demonstrations of concepts typically taught on undergraduate and Masters level systems and control courses: model-based control, linearisation, LQR, pole placement, noise amplification due to differentiation, reference following, Kalman filtering, and the principle of separation of estimation and control. Hardware interfacing is carried out through MATLAB Simulink, and the Gyroboy model also provides a platform for exploring further control and robotics applications in research or teaching projects. For example, collision avoidance can be implemented using the LEGO MINDSTORMS ultrasonic sensors, or the Gyroboy can be used as a testbed for novel control algorithms.

REFERENCES

Akmal, M., Jamin, N., and Ghani, N.A. (2017). Fuzzy logic controller for two wheeled EV3 LEGO robot. *in 2017 IEEE Conference on Systems, Process and Control (ICSPC 2017), Melaka, Malaysia*, 134–139.

Basso, M., Innocenti, G., and Rosa, A. (2013). Simulink meets LEGO: Rapid controller prototyping of a stabilized bicycle model. *in 52nd IEEE Conference on Decision and Control, Florence, Italy*, 330–335.

Behera, P.T. and Mija, S. (2016). Balancing of two wheeled inverted pendulum using SOSMC and validation on LEGO EV3. *in First IEEE International Conference on Power Electronics, Intelligent Control and Energy Systems (ICPEICES-2016)*.

Canale, M. and Brunet, S. (2013). A LEGO MINDSTORMS NXT experiment for model predictive control education. *In 2013 European Control Conference (ECC), Zürich, Switzerland*, 2549–2554.

Compere, M. (n.d.). Simulink keyboard input. uk.mathworks.com/matlabcentral/fileexchange/3630-simulink-keyboard-input. Accessed: 25/10/2019.

Cruz-Martín, A., Fernández-Madriga, J., Galindo, C., González-Jiménez, J., Stockmans-Daou, C., and Blanco-Claraco, J. (2012). A LEGO MINDSTORMS NXT approach for teaching at Data Acquisition, Control Systems Engineering and Real-Time Systems undergraduate courses. *Computers & Education*, 974–988.

Hughes, T., Willetts, G., and Kryczka, J. (n.d.). Lego Mindstorms EV3 Gyroboy. uk.mathworks.com/matlabcentral/fileexchange/71872-lego-mindstorms-ev3-gyroboy. Accessed: 25/10/2019.

Kim, Y. (2011). Control systems lab using a LEGO MINDSTORMS NXT motor system. *IEEE Transactions on Education*, 452–461.

Markovsky, I. (2012). Dynamical systems and control mindstorms. In *Proceedings of the 2012 20th Mediterranean Conference on Control & Automation (MED)*, 54–59.

Panadero, C.F., Román, J.V., and Kloos, C.D. (2010). Impact of learning experiences using LEGO Mindstorms® in engineering courses. In *Proceedings of the IEEE EDUCON Education Engineering 2010 – The Future of Global Learning Engineering Education*, 503–512.

Roslovets, P. (n.d.). Gyroboy - self-balancing two-wheel robot (segway) based on Lego EV3. uk.mathworks.com/matlabcentral/fileexchange/60322-gyroboy-self-balancing-two-wheel-robot-segway-based-on-lego-ev3. Accessed: 25/10/2019.

Åström, K. and Murray, R. (2008). *Feedback Systems: An Introduction for Scientists and Engineers*. Princeton.

Villacrés, J., Viscaino, M., Herrera, M., and Camacho, O. (2016). Controllers comparison to stabilize a Two-wheeled Inverted Pendulum: PID, LQR and Sliding Mode Control. *International Journal of Control Systems and Robotics*, 1, 29–36.

Yamamoto, Y. (n.d.). Nxtway-GS model based design: Control of self-balancing two-wheeled robot built with LEGO MINDSTORMS NXT. uk.mathworks.com/matlabcentral/fileexchange/19147-nxtway-gs-self-balancing-two-wheeled-robot-controller-design. Accessed: 25/10/2019.