# Gain scheduling for state space control of a dual-mode inverted pendulum

Laura Alvarez-Hidalgo
School of Engineering, Computing & Mathematics
University of Plymouth
Plymouth, UK
laura.alvarezhidalgo@students.plymouth.ac.uk

Ian S. Howard
School of Engineering, Computing & Mathematics
University of Plymouth
Plymouth, UK
ian.howard@plymouth.ac.uk

***Abstract*— The aim of the paper is to present an inverted pendulum system that operates in two different modes. Firstly, it operates in a static balancing mode, during which the controller tries to keep the pendulum balanced and maintain the current position of the pendulum carriage. Secondly, it also operates in a velocity control mode, so that the carriage can move at a selective velocity while simultaneously maintaining pendulum balance. In order to realize these two modes of control we implement a state feedback controller and schedule gain depending on the selected mode of operation of the system. We first describe the design and construction of the system. We then perform state space analysis, build state feedback controllers designed as linear quadratic regulators (LQR), and run tests to examine the operation of the system whilst subjecting the pendulum to impulsive disturbances. In particular, we investigate differences in control behavior in static position mode and in velocity-controlled mode. We present the experimental results and discuss their implications.**

**Keywords— State feedback controller, gain scheduling, inverted pendulum, Dual mode control, DIN Rail, LQR.**

## I. Introduction

Stabilizing an unstable system is a classical problem in control engineering. Indeed, in control theory and robotics, the inverted pendulum system has been one of the main benchmarks used almost for the last 100 years [1]. Due to its importance, many researchers have been developing different inverted pendulums and implemented different control systems methods to stabilize them. In this system, the balance of the pendulum has to be controlled by moving the cart back and forth within the limited travel of the cart.

The first researcher to demonstrate a solution to the inverted pendulum system was James Kerr Roberge in 1960 [2,3]. At the time, his system was the first to be thoroughly researched and successfully controlled, although there seemed to be few applications of the technology. Since then, it has proved to form an important basis for systems in several fields of engineering. For example, two systems that involves similar control of balancing are rockets at take-off and during flight, or stabilizing a quadcopter [4].

Many designs for inverted pendulums have been proposed [5 - 8]. The current system is an extended version of a previous inverted pendulum [9]. Here we modified the previous design to facilitate construction and improve safety. The new design is much lighter than the original, making it much more portable. It also makes use of more off-the-shelf components which will make it easier for other researchers to construct and replicate the system.

We also incorporate additional sensors to assist evaluation of behavior. In particular, we place an encoder on the stepper motor shaft so that we can accurately estimate cart position. In addition, we intercept the signal from the encoder located on the cart so we can also record the pendulum angle. This facilitates measurement of pendulum behavior during balancing.

As well as presenting a design for pendulum and controller, the contributions of this paper is also to investigate gain scheduling to change controller characteristics during different operating modes. We first describe a state space model for the inverted pendulum and its mechanical design. We then discuss controller design and simulation. Finally, we present the results from actual physical operation during balancing.

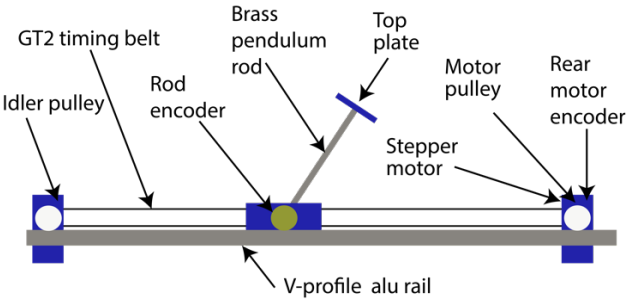


Fig 1. Schematic of pendulum showing all main components.

## II. Mathematical analysis

### A. Nonlinear analysis of pendulum dynamics

A schematic of the new pendulum design is shown in Fig. 1. The non-linear differential equation describing the inverted pendulum kinematics can be derived by consideration of forces. We extend the result of [10] by including a damping term, leading to the expression

$$(I + ml^2)\frac{d^2\theta}{dt^2} + \mu\frac{d\theta}{dt} = mgl\sin\theta + ml\frac{d^2x_P}{dt^2}\cos\theta \quad (1)$$

where the angle to the vertical is denoted by $\theta$, the coefficient of viscosity is denoted by $\mu$, the mass of the pendulum is denoted by $m$, the moment of inertia of the rod about its center of mass is $I$, the length to the center of mass is denoted by $l$ and the displacement of the pivot is given by $x_p$.

Here we focus on kinematic analysis. We note that a kinematic description is sufficient to derive control, and we do not need to control forces in the system if we use cart velocity as the control input (this is also the case if we use acceleration control). To control the cart using applied force (which is often done in many inverted pendulum implementations), we would also need to make use of additional equations to capture the cart forces and dynamics.

The non-linear expression in equation (1) can be linearized by calculating the system Jacobian and evaluating it around its equilibria positions. In the pendulum's inverted configuration,

this leads to a linearized differential equation which takes the form:

$$(I + ml^2)\frac{d^2\theta}{dt^2} + \mu\frac{d\theta}{dt} = mgl\theta + ml\frac{d^2x_P}{dt^2} \quad (2)$$

*B. Laplace analysis of pendulum dynamics*

It is enlightening to examine the linearized differential equation describing the inverted pendulum kinematics in the s-domain, since it yields important insights to the behavior of the system. From the differential equation (2) describing the linearized pendulum, we next apply Laplace transforms assuming zero initial conditions.

$$\big((I + ml^2)s^2 + \mu s - mgl\big)\Phi(s) = mls^2X_P(s) \quad (3)$$

We now write down the transfer function

$$\frac{\Phi(s)}{X_P(s)} = \frac{s^2ml}{\big((I + ml^2)s^2 + \mu s - mgl\big)} \quad (4)$$

If we wish to use velocity control, we can write that velocity as the differential of position. In the Laplace domain this leads to the relationship

$$V_P = sX_P \quad (5)$$

We next rearrange the transfer function to yield a unity term in front of the highest power of s.

$$\frac{\Phi(s)}{V_P(s)} = \frac{\frac{sml}{(I + ml^2)}}{\left(s^2 + s\frac{\mu}{(I + ml^2)} - \frac{mgl}{(I + ml^2)}\right)} \quad (6)$$

Comparing equation (6) with the canonical form of a 2nd order linear dynamical system, where $\omega_n$ is the natural frequency of the system and $\xi$ is its damping ratio:

$$\frac{sk}{(s^2 + 2\xi\omega_n s - \omega_n^2)} = \frac{\frac{sml}{(I + ml^2)}}{\left(s^2 + s\frac{\mu}{(I + ml^2)} - \frac{mgl}{(I + ml^2)}\right)} \quad (7)$$

We see that by inspection the natural frequency $\omega_n$ is given by the expression

$$\Longrightarrow \omega_n = \sqrt{\frac{mgl}{(I + ml^2)}} \quad (8)$$

We can relate angular frequency to frequency in cycles per second since

$$\omega_n = 2\pi f$$

Therefore

$$\Longrightarrow f_n = \frac{1}{2\pi}\sqrt{\frac{mgl}{(I + ml^2)}} \quad (9)$$

Similarly, by inspection is can be seen that the damping ratio $\xi$ is given by the expression

$$\Longrightarrow \xi = \frac{\frac{\mu}{(I + ml^2)}}{2\omega_n} = \frac{\frac{\mu}{(I + ml^2)}}{2\sqrt{\frac{mgl}{(I + ml^2)}}} = \frac{\mu}{2}\sqrt{\frac{mgl}{(I + ml^2)}} \quad (10)$$

These equations provide a useful means to evaluate the important system parameters from $\omega_n$ and $\xi$ by observing behavior of the pendulum, for example whilst it passively swings in the non-inverted configuration after an initial push.

*C. State space model of balancing*

To implement state feedback control of balancing in an inverted configuration, we use a mathematical expression of the system that captures its dynamics in state space format, following the derivation in [9] starting from the linearized system given in equation (2).

We implement the velocity control of the cart needed to balance the pole using a stepper motor, which can easily realize the desired cart velocity. That is, we use cart velocity as the control input (u = $V_c$). To stabilize the pendulum using velocity control, we first write the second derivative control term in equation (2) on the RHS as a first derivative of velocity

$$\frac{d^2x_P}{dt^2} = \frac{dv_c}{dt} \quad (11)$$

$$\Longrightarrow (I + ml^2)\frac{d^2\theta}{dt^2} + \mu\frac{d\theta}{dt} = mgl\theta + ml\frac{dv_c}{dt} \quad (12)$$

We choose the following state variables so no derivatives of the control velocity appear as control terms:

$$x_1 = \theta \quad (13)$$

$$x_2 = \frac{d\theta}{dt} - b_0Vc \quad (14)$$

As shown in [9], this leads to the state space matrix representation of the system

$$\Rightarrow \begin{bmatrix}\dot{x}_1\\ \dot{x}_2\end{bmatrix} = \begin{bmatrix}0 & 1\\ -a_2 & -a_1\end{bmatrix}\begin{bmatrix}x_1\\ x_2\end{bmatrix} + \begin{bmatrix}b_0\\ -a_1b_0\end{bmatrix}v_c \quad (15)$$

$$\mathrm{y} = \begin{bmatrix}1 & 0\end{bmatrix}\begin{bmatrix}x_1\\ x_2\end{bmatrix} \quad (16)$$

where we represented the constant coefficient terms as follows:

$$a_1 = \frac{\mu}{(I + ml^2)} \quad (17)$$

$$a_2 = \frac{-mgl}{(I + ml^2)} \quad (18)$$

$$b_0 = \frac{ml}{(I + ml^2)} \quad (19)$$

These values were numerically evaluated using measurements of the physical pendulum. This 2x2 system can be directly used as a basis for velocity control, in which the inverted pendulum balances and the cart can simultaneously move along the rail at a specified velocity.

In order to explicitly implement position control on the cart location, we add states to the system state vector that directly represent cart position. Thus, we first add a 3rd state to directly represent cart position. This can be achieved with by integrating the control velocity, providing one means of estimating the cart's location

$$\dot{x}_3 = Vc \tag{20}$$

Alternatively, the cart position can be directly measured from the stepper motor control signal, by appropriately counting drive pulses. The latter approach is more accurate since it represents what is actually happening to the cart, and was used in our controller implementation.

The additional positional state $x_3$ further expands the state space matrix representation of the system. In addition, we can include a 4th state to represent the integrated deviation of position state $x_3$ from the origin. Such incorporation of integral action has the effect of removing steady-state positional errors in cart position. This will ensure the system is able to effectively maintain the cart at its starting location. Incorporation of the additional two states $x_3$ and $x_4$ leads to the state space equations:

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \\ \dot{x}_4 \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ -a_2 & -a_1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} + \begin{bmatrix} b_0 \\ -a_1 b_0 \\ 1 \\ 0 \end{bmatrix} v_c \tag{21}$$

$$\mathrm{y} = \begin{bmatrix} 1 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} \tag{22}$$

To output pendulum angle from the state space model, which is later necessary to provide a correction for the observer used in the controller (discussed in the next section), we choose the C matrix to select state $x_1$, which corresponds to pendulum rod angle. Note that there is no D matrix term.

*D. Designing the state feedback controllers*

To stabilize the pendulum in its inverted configuration, we use a full state feedback controller. This can maintain balance even when noise and disturbances are present. To implement such a controller, we need to calculate the state feedback gain vector K. This gain can be found in many ways, including using pole placement or by optimal control methods, which we adopted here. To find gain K to implement a linear quadratic regulator, we made use of the MATLAB lqr command [11].

For velocity control, we base the design on the 2x2 system shown in equations (15,16). Diagonal terms in the 2x2 Q matrix and 1x1 R matrix were specified to appropriately penalized the system states and control respectively.

We note that penalizing state ensures it approaches its desired target value – in our case it keeps the pendulum angle near zero, resulting in effective balancing. Penalizing R suppresses excessively high velocity movement of the cart. The optimal control cost terms Q and R were found by experimentation.

$$\mathrm{Q} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \tag{23}$$

$$\mathrm{R} = 0.1 \tag{24}$$

For position control using integral action, we base the design on the 4x4 system shown in equations (21, 22). Diagonal terms in the 4x4 Q matrix and 1x1 R matrix were specified to appropriately penalized the system states and control respectively. The optimal control cost terms Q and R were again found by experimentation.

$$\mathrm{Q} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 5 \end{bmatrix} \tag{25}$$

$$\mathrm{R} = 0.5 \tag{26}$$

Notice to achieve cart position control, we more heavily weight the integral action state $x_4$ and ignore state $x_3$ in the optimization. Changing operation between velocity and position control can be achieved by switching between the corresponding gains. Note it is necessary to reset the integral state $x_4$ to zero each time the mode changes.

In position mode, using integral action, the reference input is set to zero to maintain cart position at its current location. In velocity control mode, to achieve good tracking of the desired reference velocity input, it is necessary to compute the feedforward pre-emphasis term $\bar{N}$:

$$\bar{N} = -[\mathrm{C}(\mathrm{A} - \mathrm{BK})^{-1}\,\mathrm{B}]^{-1} \tag{27}$$

The pre-scaling of the reference velocity input by $\bar{N}$ ensured the robot position moves at the desired velocity. Here we note that $\bar{N}$ had a value of 1, since the reference input directly controlled the output cart velocity of the system.

*E. Designing the Luenberger observer*

We used an observer to estimate full system state. The Luenberger update for estimated system state is given by

$$\dot{\hat{X}} = A\hat{X} + BU + L(Y - C\hat{X}) \tag{28}$$

Where the hat symbol denotes state estimate. Here we once again use the MATLAB lqr command, to design the Luenberger gain L by designing another linear quadratic regulator. Again, diagonal terms in the 2x2 Q matrix and 1x1 R matrix were specified to penalize the system states and control respectively. Once again, the parameter values were found by experimentation.

$$\mathrm{Q} = \begin{bmatrix} 1000 & 0 \\ 0 & 1000 \end{bmatrix} \tag{29}$$

$$\mathrm{R} = 0.1 \tag{30}$$

Notice that we used strong penalization on both observer states. Also, we note that we do not predict the velocity or position of the cart using the observer. Indeed, we can easily

estimate the velocity of the cart, since we use velocity control of the stepper motor.

## III. Mechanical Design of the Inverted Pendulum

### A. Pensulum rod

The pendulum rod was constructed from a relatively short 340mm length of brass. Brass was chosen since it is an easy to machine and relatively high-density material. It was tapped at one end so it could be screwed firmly into a mounting clamp attached to the main shaft to ensure very firm attachment (Fig. 2). A short pendulum length makes balancing more challenging since the cart needs to move and react faster due to the system's correspondingly higher natural frequency $\omega_n$. However, a short rod length is easier to accommodate and is less dangerous to the operator, since the chances of hitting anyone is much reduced. It is also a better model for other smaller systems, such as balancing robots.

A round flat 3D printed disc was attached to the end of the pendulum to improve safety and to provide a platform on which to balance objects. A PLA 3D printed cart plate supported a shaft that attached to the pendulum rod so it could swing freely. It made use of a bearing on one end and an encoder the other end so the angle of the rod to vertical could be measured (Fig. 2). In addition, a low-cost MPU-6050 6 DOF gyro/accelerometer IMU was attached to the pendulum rod close to its axis of rotation, providing an additional means of angle measurement.

### B. V-groove profile

The mechanical design made use of V-groove profile section for the linear axis to facilitate construction using an off-the-shelf gantry plate, which ran along the V-grooves in its sides. The pendulum structure was supported on a base constructed from profile. (Fig. 3). The cart was driven back and forwards along the aluminum V-rail using a GT2 belt operated by a NEMA 23 stepper motor located at the end of the rail. An encoder was mounted on its shaft to enable measurement of cart position to be made (Fig. 4).

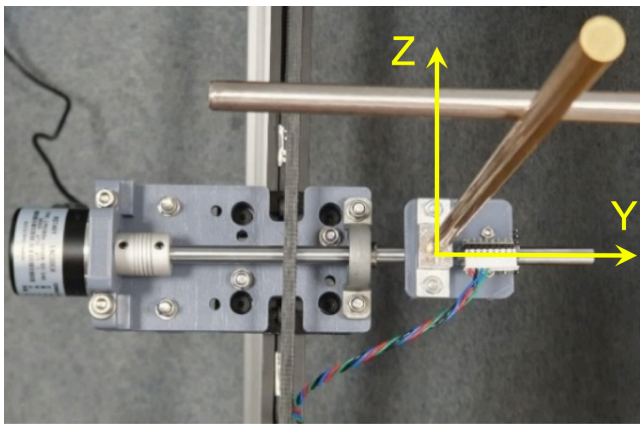


Fig 2. 3D printed Pendulum Cart. The pendulum shaft is supported by a bearing on the RHS and an encoder on the LHS. The encoder measures angular position of the pendulum rod. The IMU is located on the pendulum rod assembly on a support such that when the pendulum is in its inverted configuration, rotation occurs around the y-axis of the IMU sensor, with its x-axis pointing downwards in the opposirte direction to the rod, with the z-axis in the horizontal plane. The directions of IMU y- and z-axes are shown in yellow.

## IV. DIN Rail Controller Panel

### A. DIN rail construction

As part of this project, a DIN rail controller panel was constructed that would operate the current inverted pendulum system, and also support future projects.

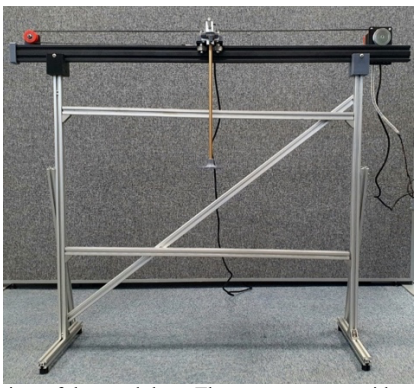

Fig 3. Side view of the pendulum. The stepper motor with encoder fitted is mounted on the RHS and drives a GT2 timing belt loop. The latter runs the length of the rail and it kept tight using a pulley system on the LHS. The belt attaches to the cart shown here in the middle of the rail. The rail assembly is firmly attached to an aluminum profile base to minimize unwanted movement during operation.

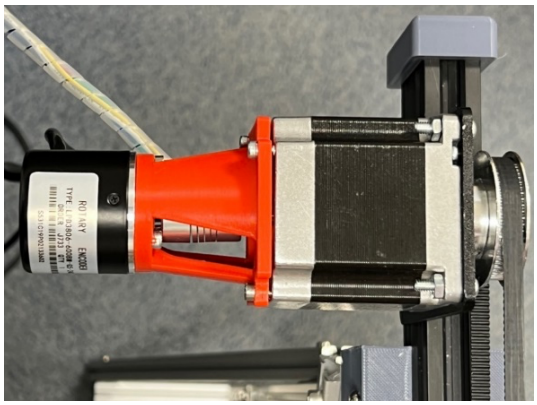

Fig 4. The custom-made mechanical components were 3D printed with PLA using a Creality 6SE printer, such as the encoder mount for the stepper motor.

### B. Top connection panel

The controller panel used a female USB-B connection to connect the Arduino Mega 2560 within the controller assembly with the PC that was used to develop software and also send control commands.

### C. D connectors

The panel also made use of 7 female D-connectors to interface to one SPI device, one encoder, one I2C device and 4 stepper motors. This panel is easy to connect-up to the inverted pendulum apparatus due to the use of the D-connectors.

The plugs on each of the different attached component connected into this panel have a unique connection pattern, to prevent accidental inappropriate connections causing damage. The power pins assignments were the main issue here, and care was taken to keep them away from other signal inputs and outputs to ensure connection errors would not lead to

component damage. The panel connector interface and wring diagrams are shown in Figs. 5-9.

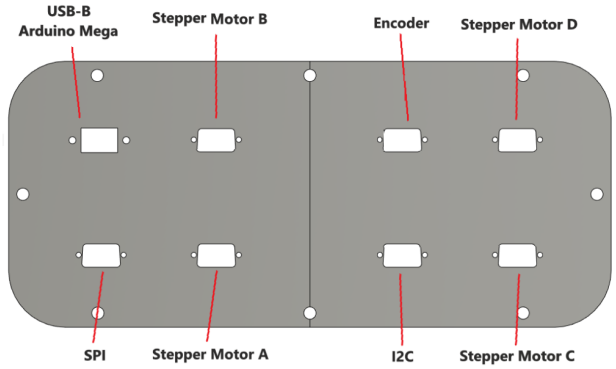


Fig 5. 3D design in Autodesk Fusion of the controller panel cover

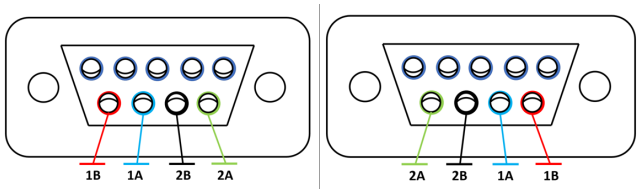


Fig 6.Rear view of solder D-Connections between two stepper motors and control panel

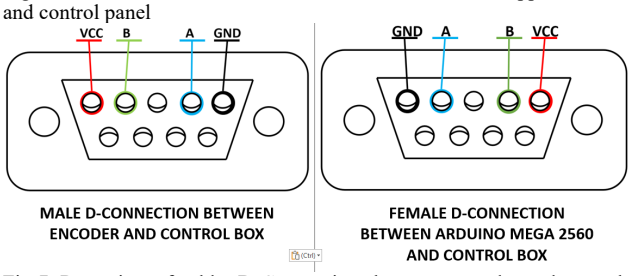


Fig 7. Rear view of solder D-Connections between encoder and control panel

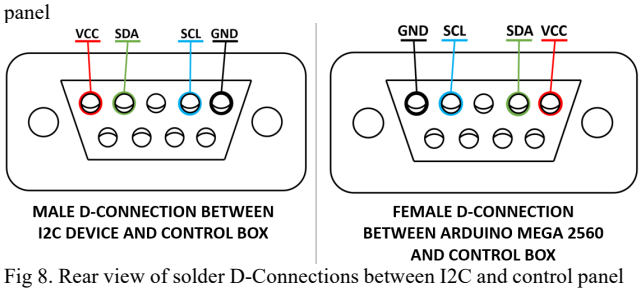


Fig 8. Rear view of solder D-Connections between I2C and control panel

The internal space of the controller panel was divided up by three rails; a power supply rail, a motor driver and power converter rail and a microcontroller rail (Fig. 10). In addition, slotted conduit was placed between the rails to route and organize the cables, and avoid any chance of the cables touching and interfering with other components. To construct a robust and neat DIN rail system, where possible we made use of components which had rear DIN rail mounts. In the cases where the components were not already setup for use in the DIN rail system, supports were designed and fitted with DIN rail clamps.

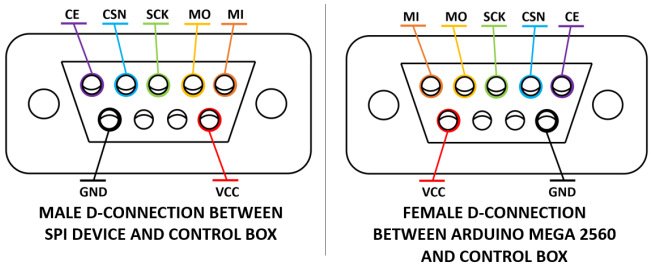


Fig 9. Rear view of solder D-Connections between SPI and the control panel

The power rail was divided in three main parts, 1) an AC circuit breaker section. 2) a PULS Dimension DIN rail power supply section, which supplied 24v at 5A, and 3) a section consisting of three DC Circuit breakers to enable power to the main control components be disconnected individually and to protect against possible short circuit damage.

*D. Motor drivers*

The motor driver and power converter rail operated up to four stepper motors that could be interfaced the control panel. This rail used two 3D printed stepper motor carriers and each of them supported two A4988 stepper motor drivers. They were supplied with 24v from the mains AC/DC power supply via the motor driver power breaker.

The A4988 chip has 8 different types of micro-stepping resolution. This project used ¼ step resolution for all four stepper motors, which meant the full step was sub-divided into 4 smaller sub-steps and smoothed-out the rotation of the motor. To select ¼ stepping mode, DIL switches on the expansion boards were set appropriately. The A4988 can run from a supply voltage between 8v and 35v, and deliver a current between 1.5A and 2.2A. The breakout board has a potentiometer that allows manual setting of its current limit. To do so, it is necessary to measure the voltage on the potentiometer and use the following formula to calculate the required output voltage:

$$Current\ Limit = VRef * 2 \qquad (31)$$

It was important to adjust this value to ensure the stepper motors would run properly, and in this case the current limited needed for our NEMA23 stepper motors was 1.5A. Higher current could damage the motors and the drivers. Also, if the current was set less than 1.5A, the motors would not operate correctly and could possibly stall.

To avoid the motor drivers overheating, it was crucial to place a small cooling fan just in top of them, to cool them. Thus, to avoid thermal of damage of these chips, two DC axial fans were used, which need a supply voltage of 12v and was supplied using a Buck converter operating on the 24v power supply output (Fig. 11).

The micro-controller rail was at the heart of the controller panel and also communicated with the host PC. The Arduino Mega 2560 [12] was chosen as the microcontroller in the panel because it has good I/O capabilities; 54 digital input/output pins, with 15 that can be used for PWM, a flash memory of 256KB and a clock speed of 16MHz (Fig. 13).

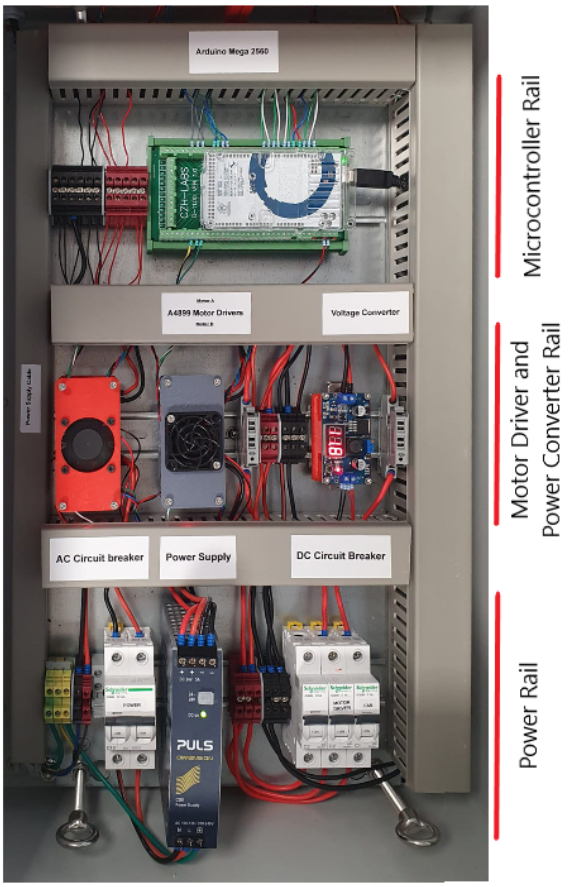


Fig 10. Controller panel internal layout, showing the partition into power, motor control and microcontroller sections.

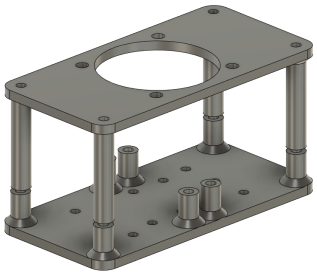

Fig 11. A4899 Driver Module DIN Rail holder designed in Autodesk Fusion. This assembly firmly attached two stepper controller bearout boards onto a DIN rail. It also housed two cooling fans needed to prevenbt the controller from overheating.

## V. Software Implementation on the Arduino

After the inverted pendulum controller was designed, it was tuned and tested using a MATLAB simulation. This involved examining the inverted pendulum whilst setting the initial conditions of the second state to non-zero values to simulate the effect of small knocks, as well as varying the reference input to move the cart. Example results for the integral action position mode are shown in Fig. 12.

The control processing was finally implemented on the Arduino Mega, with the state feedback controller operating within the Arduino main loop. The control procedure first involved reading the encoder. To support extended operational capabilities, the MPU-6050 IMU was also read, and an associated complementary filter was run. The selected feedback controller then calculated the velocity control signal and used it to generate an output pulse train to drive the stepper motor to move the cart at the appropriate velocity to balance the pendulum. Euler integration was used to compute the state updates [13].

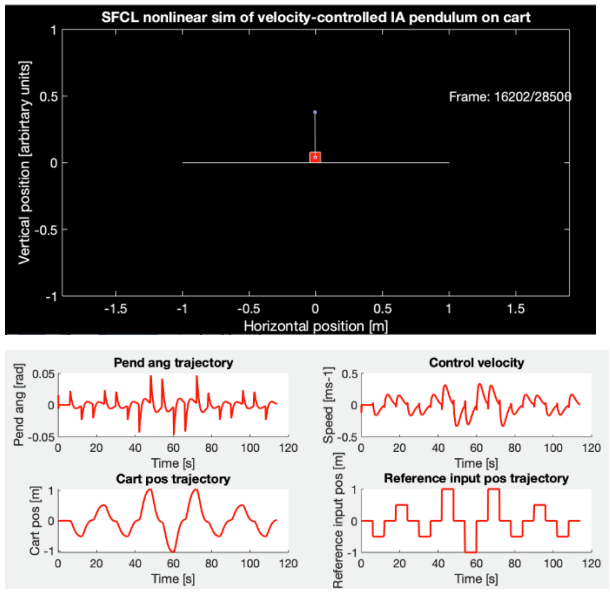


Fig 12. MATLAB inverted pendulum simulation in integral action position control mode. Upper panel shows screen shot of animation used to examine behaviour. Lower panel shows signals arisng from the sigmulation, indicating the pendulum angular trajectory, correspoiding control velocty signal and cart position whilst tracking a rerefence carty postion.

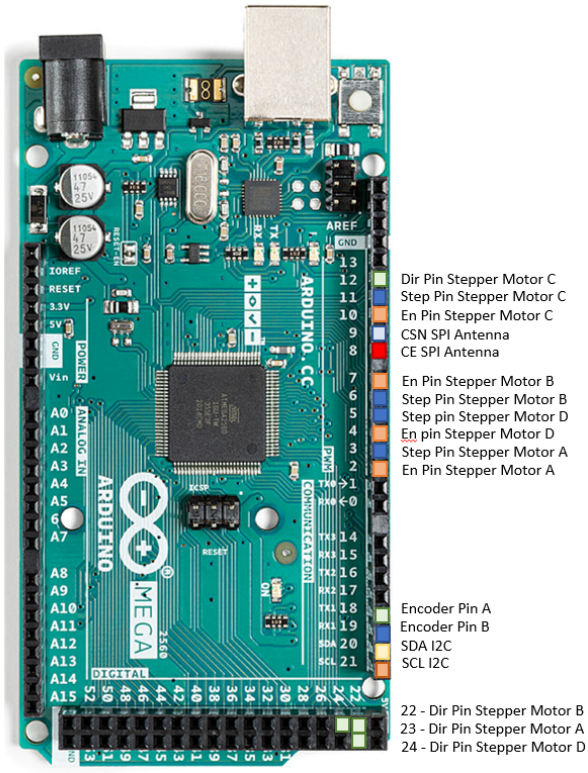


Fig 13. Arduino Mega 2560 pinout connections in the controller panel.

For the controller to operate, it needed an estimation of the full state of the system. This was achieved by implementing a Luenberger observer, and directly measuring cart position and calculating the integral error. Then the full state estimate was scaled by the feedback gain and used generate the control command. Pseudocode for the Arduino implementation is shown here:

% initialize the state estimate
Initialize xHat to zero value

% run controller
for (each time point)
{
Read pendulum angle, which is the real pendulum output

Read the cart position and use to directly set state variable $x_3$

Compute state feedback control variable u on basis of SFC gain K and the estimated observer state xHat:
U = -K * xHat

Calculate the observer correction term using the real pendulum output:
ycorr = L * (y - C * xHat)

Update the observer state Xhat using Euler integration with correction term for the states $x_1$, $x_2$ and $x_4$ (NB: $x_3$ is explicitly set above):
Xhat = Xhat + h * (A * Xhat + B * U + ycorr)

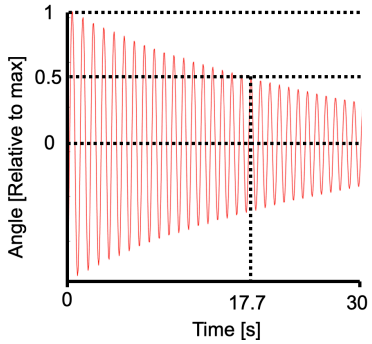


Fig 14. Angle decay of pendulum as a functon of time, measured with an encoder. This trace provided a means to estimate the canonical parameters off the 2nd order pendulum system.

## VI. Experimental Results

### A. *Testing task-dependent disturbance*

To examine the dynamical characteristics of the uncontrolled pendulum, it was pushed and allowed to swing whilst hanging-down in its stable configuration (Fig. 14). It can be seen that 31 cycles were made in the period of 30 seconds. In addition, the envelope of the oscillations decayed to 50% in 17.7s. This yielded damping and natural frequency values of 0.006 and 1.03Hz respectively. Using equations (9,10) and the physical parameters of the pendulum, we calculate the viscous damping factor of the pendulum system as 0.00049. This value was used to set the numeric state space model coefficients given in equations (17-19).

An enlightening way to investigating how the control laws behave involved simultaneously recording cart position and the pendulum angle (measured by the encoder located on the stepper motor and pendulum axis respectively) during balancing and then applying a disturbance. Results from a simple test during position control mode shown in Fig. 15A. After first being brought into the balancing condition, the pendulum was repeatedly tapped. The effect of this disturbance can be seen in both traces pendulum angle and cart position traces. The pendulum encoder plot (which indicates pendulum angle) shows small changes in angle of the pendulum and at the same time, it can be seen in the motor encoder values, that the cart repositioned rapidly to find a stable balance of the system, and then slowly moved back to the starting location.

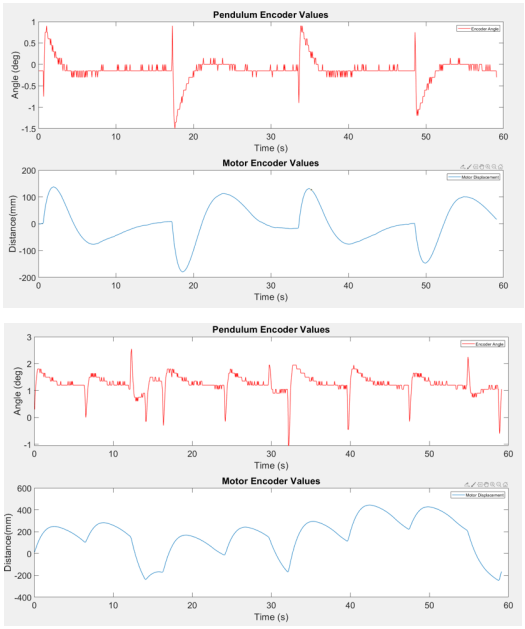


Fig 15. Relative balancing pendulum reponses to discurbances. A shows the reaction of the pendulum to being tapped whilst under position control. B shows the reaction of the pendulum to being tapped whilst under velocity control.

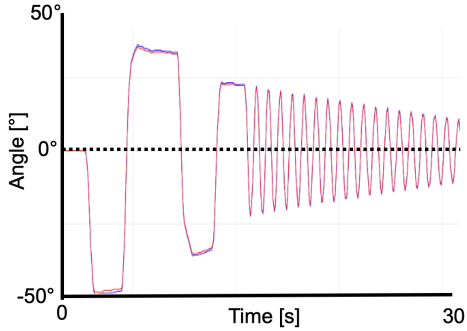


Fig 16. Comparing encoder (red) and IMU reading (blue) during pendulum movement. Initially the traces show moving the rod a few times times by hand between around ±40 degrees, and then free damped oscillation of the pendulum. It can be seen the two estimate of rod angle are in close agreement.

The pendulum was also examined in velocity control mode, as shown in Fig.15B. Again, the pendulum was brought to the upright position where it immediately activated the position controller and started to balance effectively. Velocity control was then switched on using a keyboard command and a velocity of 0.05 meters per second selected, which resulted in a corresponding movement of the cart. The cart was able to move both left and right. In order to test resistance to knocks (which constitute impulsive disturbances) whilst balancing in position mode, the rod was again lightly tapped. It was shown to be quite stable and able to compensate the disturbance. We then ran a test in both position control and velocity control modes, which we balanced a roll of wire on the top of the pendulum to demonstrate that it could also maintain balance when additional loads are placed on the pendulum, even though they change its dynamical characteristics (see YouTube video link in video demonstrations section).

### B. *Testing IMU*

To test the validity of the IMU as a function of time and verify that there is little offset at any point arising from drift, the pendulum was moved from about ±40° and held relatively stationary at these angles for a couple of seconds. After repeating this twice, the pendulum was released and allowed to oscillate freely. Fig. 16 shows angular measurements from both IMU and encoder sensors, indicating there was good agreement between the two sources of measurement. Finally, we briefly mention that we ran the pendulum balancing tasks as before but made use of the IMU instead of the encoder as a source of pendulum angle. The IMU proved slightly less robust that using an encoder, but balance could be maintained and resistance to disturbances was still achieved.

### C. *Video demonstrations*

Operations of the inverted pendulum on these tests, as well as others, are shown on the YouTube channel: Robotics, Control and Machine Learning in the playlist: ICSSE2022 & NSSSE2022 Inverted Pendulum:

https://www.youtube.com/playlist?list=PLjKvJX8cBCKWv0DIV5zhKSC-Z-qn4UiKP

## VII. Discussion

Here we built a new pendulum to investigate state feedback control of balancing. We first simulated the pendulum system in MATLAB using a position controller and a velocity controller, both based on the state-space model of the linearized dynamical system of the pendulum. We constructed a general-purpose control panel to implement our control tasks that could also easily be used for other tasks in the future too. Specifically, in order to implement a real-time controller, a DIN rail based micro control panel was designed and built. The panel had sufficient functionality to enable it to record critical signals from the pendulum system, including position of the cart and angle of the pendulum rod to vertical.

We investigated balancing during static operation under position control and also during constant velocity movement. We ran tests on the system and investigated behavior when we perturbed the balanced pendulum rod using impulses generated by tapping lightly. We show difference in behavior between the velocity controller and the position controller. We found the control was more responsive in static balance mode and more resistant to perturbations than it was in velocity control mode. This is perhaps unsurprising because when it is controlling both the velocity and also balancing, the controllers resources are somewhat more divided. The problem of controlling velocity while balancing is clearly more challenging than just maintaining a static posture and balancing at a fixed position. Of course, the gains found here using LQR were done so on the basis of experimentation, so it is always possible better values also exists for both modes. In the future it will also be interesting to further investigate human behavior on such tasks and compare it to control models [14,15], as well as the factors that affect control in both human and machines, such as the latency of the sensory feedback [16].

In this work we made use of state feedback control. We note that is certainly possible to use PID control for this task, and may previous designs have adopted this approach. However, here we specifically chose to use full state feedback control instead since it is less frequently used to control a pendulum and because LQR controller design lets the engineer formulate the criterion of controller optimality, and corresponding feedback law, in a principled way. We also note that the slight increase in computational complexity of a state feedback control approach over PID is relatively small and the computational load needed the for-motor controller implementation certainly did not lead to any problems on the Arduino Mega microcontroller used.

## Acknowledgments

We thank the EPSRC for supporting LAH. We thank the University of Plymouth, and in particular David Mozley, for supporting ISH as part of the proof-of-concept project entitled “Design of robotic grippers with build-in sensors for picking operations that are evaluated using direct human control.” Thanks also to Hooman Samani for commenting on the manuscript.

## References

[1] K.H. Lundberg, & T.W, Barton. (2010). History of inverted-pendulum systems. IFAC Proceedings Volumes, 42(24), 131-135.

[2] J. Roberge, 1960. The mechanical Seal. Bachelor of Science. Massachusetts Institute of Technology.

[3] O. Boubaker, (2013). The Inverted Pendulum Benchmark in Nonlinear Control Theory: A Survey. International Journal of Advanced Robotic Systems, [online] 10(5), p.233. Available at: https://journals.sagepub.com/doi/pdf/10.5772/55058.

[4] P. Wang, Z. Man, Z. Cao, J. Zheng, & Y. Zhao, (2016). Dynamics modelling and linear control of quadcopter. In 2016 International Conference on Advanced Mechatronic Systems (ICAMechS) (pp. 498-503). IEEE

[5] S. Awtar, N. King, T. Allen, I. Bang, M. Hagan, D. Skidmore & K. Craig, (2002). Inverted pendulum systems: rotary and arm-driven-a mechatronic system design case study. Mechatronics, 12(2), 357-370

[6] F. Grasser, A. D'arrigo, S. Colombi & A.C. Rufer, (2002). JOE: a mobile, inverted pendulum. IEEE Transactions on industrial electronics, 49(1), 107-114.

[7] C.A. I. C. S. Magazine (1989), “Learning to control an inverted pendulum using neural networks,” cs.colostate.edu

[8] K. Yoshida, (1999). Swing-up control of an inverted pendulum by energy-based methods. American Control Conference, [online] 6(6), pp.4045-4047. https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=786297

[9] I.S. Howard, (2019). A modular 3D-printed inverted pendulum. In Annual Conference Towards Autonomous Robotic Systems (pp. 413-424). Springer, Cham.

[10] Control Tutorials for MATLAB and Simulink (CTMS): Inverted Pendulum: System Modeling https://ctms.engin.umich.edu/CTMS/index.php?example=InvertedPendulum&section=SystemModeling

[11] https://uk.mathworks.com/help/control/ref/lqr.html

[12] Arduino, Arduino Mega 2560 Rev 3.Available at: https://store.arduino.cc/products/arduino-mega-2

[13] K. J. Aström and R. M. Murray, Feedback Systems. Princeton University Press, 2010

[14] I. D. Loram, P. J. Gawthrop, and M. Lakie, “The frequency of human, manual adjustments in balancing an inverted pendulum is constrained by intrinsic physiological factors,” The Journal of Physiology, vol. 577, no. 1, pp. 417–432, Nov. 2006.

[15] R. Leib, J. Česonis, S. Franklin & D.W. Franklin, (2019). LQG framework explains performance of balancing inverted pendulum with incongruent visual feedback. In 2019 41st Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC) (pp. 1940-1943). IEEE.

[16] S. Franklin, J. Česonis, R. Leib & D.W. Franklin, (2019). Feedback delay changes the control of an inverted pendulum. In 2019 41st Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC) (pp. 1517-1520).