Howard IS (2019), A modular 3D-printed inverted pendulum. K. Althoefer et al. (Eds.): TAROS 2019, LNAI 11649, pp. 1–12, 2019. The final authenticated version is available online at: https://doi.org/10.1007/978-3-030-23807-0_34

A modular 3D-printed inverted pendulum

Ian S. Howard
Centre for Robotics and Neural Systems, University of Plymouth,
Plymouth, PL4 8AA UK.
ian.howard@plymouth.ac.uk

Abstract

Here we describe a simple modular 3D-printed design for an inverted pendulum system that is driven using a stepper motor operated by a microcontroller. The design consists of a stainless-steel pole that acts as the pendulum, which is pivoted at one end and attached to a cart. Although in its inverted configuration the pendulum is unstable without suitable control, if the cart travels backwards and forwards appropriately it is possible to balance the pole and keep it upright. The pendulum is intended for use as a research and teaching tool in the fields of control engineering and human sensori-motor control. We demonstrate operation of the design by implementing an observer-based state feedback controller, with augmented positional state of the cart and integral action, that can balance the pole in its unstable configuration and also maintains the cart at its starting position. When the controller is running, the pendulum can resist small disturbances to the pole, and it is possible to balance objects on its endpoint.

1 Introduction

Balancing an inverted pendulum is a classical problem in the field of control engineering. It is a task often used to demonstrate that it is possible to stabilize a system that is otherwise unstable without control and as such provides a test bed for control research, e.g. [1]. The inverted pendulum paradigm has also been adopted to investigate human balancing [2], [3], [4] as well as a model for human walking [5].

Inverted pendulums that are used to investigate control generally consists of a rod that acts as the pendulum, which is pivoted at one end and attached to a cart. In a linear inverted pendulum design, the cart can move along a linear track. Although many inverted pendulum designs and their variants exist (e.g. [6], [7]), very few are widely available. Thus, a major consideration of the current design was to ensure it would easy to construct using standard components and 3D printed parts, so other potential users can build their own pendulum units. To support this goal, the design has been made freely available for download (see Results and conclusions section).

2 Inverted pendulum components

To ensure the pendulum system easy to use and transport, an important design consideration was to make the unit a manageable size (e.g. only about 1m long), as well as being self-contained and be easy to program. The latter was the motivation to base the control on an Arduino microcontroller, since they are widely available. Here we adopt a modular approach to design, so that the system can be updated and expanded in the future. The system is comprised of several 3D printed parts, which are illustrated in Fig. 1. These are mounted on 20mm aluminum profile that forms the main structural support for the system.

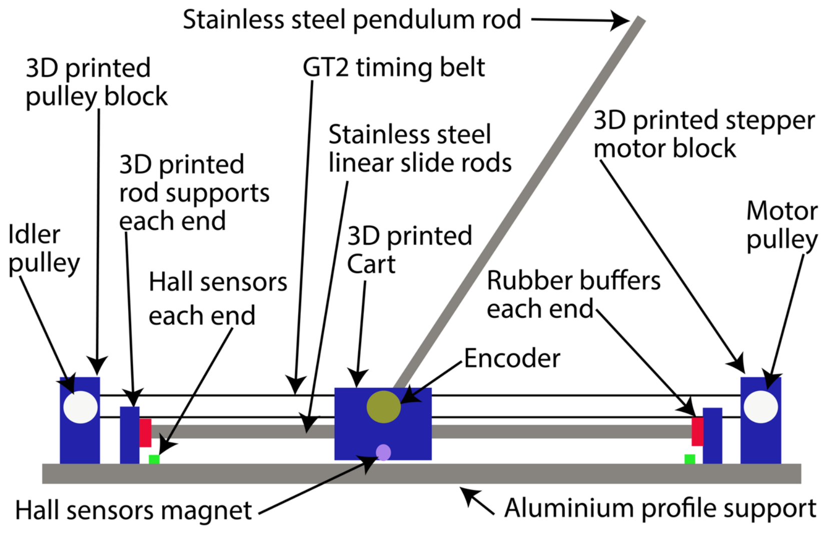
Stainless steel pendulum rod
3D printed pulley block
GT2 timing belt
Stainless steel linear slide rods
3D printed stepper motor block
3D printed rod supports each end
Idler pulley
3D printed Cart
Motor pulley
Hall sensors each end
Rubber buffers each end
Encoder
Hall sensors magnet
Aluminium profile support

Fig. 1. Schematic of the modular inverted pendulum design, illustrating its main components.

To provide a low friction track, 16mm diameter stainless steel rods are attached to the profile frame on supports by means of standard aluminum clamps and 3D printed mounting blocks. The large diameter rods were needed to ensure minimal bending due to load of the cart whilst the pendulum rod was swung around. At the end of each track rod section, a protective rubber grommet was used to damp any collisions that may occur between the cart and the end support blocks. Hall effect limit-switch sensors were also located on the frame, to deactivate the motor drive system when the cart reached the end of its travel. A neodymium magnet mounted at the base of the cart provides the necessary switching signal to achieve this.

The cart runs along the two track rods on linear ball bearings to keep friction to a low level. The pendulum element itself is a 600mm long x 8mm diameter stainless steel rod section, with an optional 3D printed end piece, so item can be balanced on it. It is attached to a rotary shaft mounted in the cart. An important design requirement here was to ensure the pendulum rod could swings freely through 360° and clear the profile frame structure. The shaft is supported by two sets of ball bearing and its far end is coupled to an incremental encoder. A unit with 2000 pulses per revolution was found suitable.

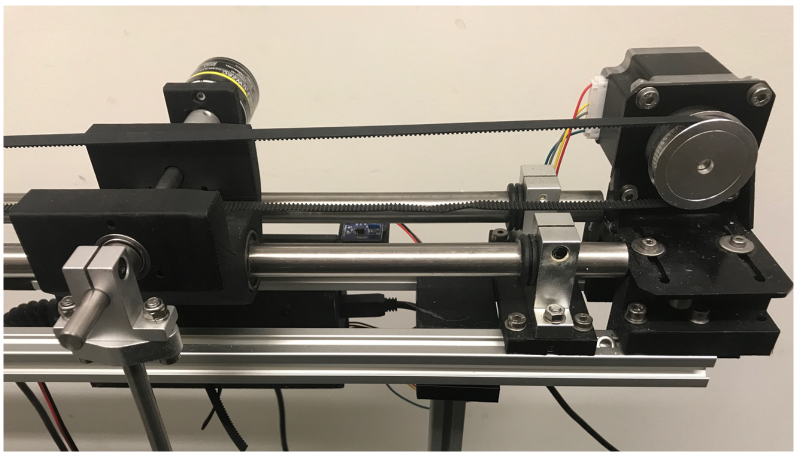

Fig. 2. Pendulum drive motor and cart mechanism. The drive pulley attached to the stepper motor can be seen on the right and the attachment clamp for the pendulum rod can be seen on the left.

The cart is pulled along its rails by means of a GT2 timing belt, which is attached to the cart using a simple toothed clamp. A tooth GT2 pulley is mounted on the motor to engage the belt. At the opposing end of the frame, a mounting block holds another GT2 pulley so that the belt loops over the length of the track and back and can therefore be used to pull the cart in both directions. Ball bearing were used in the passive pulley to ensure all moving parts could rotate with high precision and with little frictional resistance despite high belt tension.

The motor drive component consists of a 3D printed base located in-between the profile rails, which supports a 2A 1.8° per step standard NEMA23 industrial standard stepper motor attached by means of a standard NEMA23 angle bracket. The motor was driven using a Pololu A4988- type stepper driver. This implements intelligent current control up to 2A. The stepping mode is set via control pins – and in our application we made use of 4x micro stepping. The A4988 operates using a simple step and direction control interface. To operate a stepper motor using this controller, appropriately control pulses (1 pulse per micro step) and direction signals (so motor turns in desired direction) need to be sent to the driver. Overall the motor achieved a maximum cart speed of 0.6ms^{-1}. We note a slow speed is desirable in many educational settings, since, it reduces the chance of operator (student) injury from the mechanism's moving parts.

The pendulum controller was implemented using an Arduino Mega 2560 R3 microcontroller because it meets the design requirements. Two digital input that support interrupts were needed to support the Hall sensors and another two for the incremental encoder, giving four in total. We note that the Mega has 6 Digital I/O Pins which support interrupts, whereas the cheaper Arduino Uno only supports 2. The Arduino Mega also has considerably more memory than the Uno and therefore this platform will also support future development of more memory-demanding software. It also offers the possibility of extending the current design to a dual pendulum system, since an additional encoder can be accommodated. The main pendulum application was programmed in Arduino-style C-code. Library functions to realize stepper motor velocity operation and state feedback control were written in C++. The system was run from a 20v 2A power supply. A close up of the cart and motor drive assemble is shown in in Fig. 2.

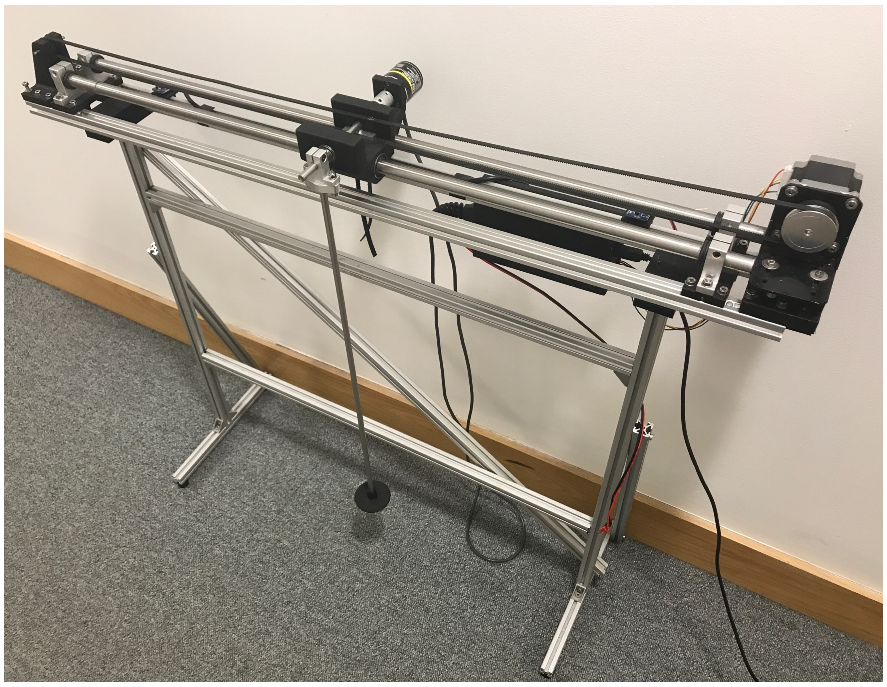

Fig. 3. Pendulum system mounted on its aluminum profile stand. This provides a convenient and elegant means of support and allows the pendulum to rotate freely and avoid collisions.

The main track holders, motor drive and end pulley support are all separately attached to the aluminum profile sections with T-nuts, so they can be easily removed and adjusted. This attachment methods also facilitates tensioning the drive belt, since the T-nuts can be loosened, and the motor block slid along the profile until desired tension is achieved. The modular construction ensures that the parts of the inverted pendulum system are easy to change and also upgraded with future designs, if so desired. This is particularly useful in teaching scenarios because different tasks can then be given to different groups of students with only minor modifications to the apparatus. For example, the pole rod can easily be changed with one of a different length. Similarly, the stepper motor unit could be exchanged for a drive unit employing a faster torque-controlled brushless DC motor, enabling the use of force control instead of velocity control and supporting one-shot swing-up operation in reinforcement learning experiments.

The mounting blocks and pendulum cart were designed using AutoCAD Fusion 360. This was subsequently used to generate STL format files and the mechanical parts were manufactured in PLA using a Flash forge Creator Pro 3D printer. We note that higher impact materials such as nylon would greatly improve the robustness of the design, but PLA has also proved to be an adequate choice.

During operation, the inverted pendulum unit requires mounting at a suitable height off the ground, so that so that the rod can swing freely. A custom-made support stand made out of 20mm aluminum profile was therefore constructed to mount the pendulum system. This provided a strong but light weight construction that can be easily transported. It consists of two aluminum profile pillars. These are filled at their base with cross member section attached with feet. At the top they are capped with 3D printed support sections that fit in between the aluminum frame of the pendulum structure. Diagonal aluminum profile sections are also used to brace the structure to increase its stiffness. An assembled pendulum on its stand is shown in Fig. 3.

3 State space analysis of the pendulum

To demonstrate operation of the inverted pendulum and provide a useful basis for experiments and demonstrations of control, here we implement observed-based state feedback control with augmented positional state (of cart) and integral action [8]. The non-linear differential equation describing the inverted pendulum kinematics is given by

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

This expression can be linearized around the unstable equilibria of the pendulum to give the linearized differential equation describing an inverted pendulum kinematics, which is given by

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

Where: The angle to the vertical is denoted by θ, the coefficient of viscosity is denoted by μ, mass of the pendulum is denoted by m, moment of inertial of the rod about its center of mass is I, length to the center of mass is denoted by l and the displacement of the cart is given by x_P. We note that this kinematic description is sufficient to derive control, provided we use cart acceleration or velocity as the control input. Velocity control of the cart was chosen here over acceleration control because of the relative ease of implementing velocity control using a stepper motor. This can be achieved with a simple function that uses a timer to generates pulses at a frequency corresponding to the desired rotational speed of the motor. Writing the differential equation describing the inverted pendulum with the highest state related term on the LHS be have

$$\Rightarrow \frac{d^2\theta}{dt^2} = -\frac{\mu}{(I + ml^2)}\frac{d\theta}{dt} + \frac{mgl}{(I + ml^2)}\theta + \frac{ml}{(I + ml^2)}\frac{d^2x_P}{dt^2} \tag{3}$$

Since we wish to stabilize the pendulum using velocity control, we first write the acceleration control term on the RHS as

$$\frac{d^2 x_P}{dt^2} = \frac{dv_c}{dt} \tag{4}$$

We now let the constant terms be represented by coefficients as follows:

$$a_1 = \frac{\mu}{(I + ml^2)} \tag{5}$$

$$a_2 = \frac{-mgl}{(I + ml^2)} \tag{6}$$

$$b_0 = \frac{ml}{(I + ml^2)} \tag{7}$$

We let the constant terms be represented by coefficients terms, leading to the equation for dynamics

$$\frac{d^2\theta}{dt^2} = -a_1 \frac{d\theta}{dt} - a_2\theta + b_0 \frac{dv_c}{dt} \tag{8}$$

We now choose the state variables

$$x_1 = \theta \tag{9}$$

$$x_2 = \frac{d\theta}{dt} - b_0 v_c \tag{10}$$

$$\Rightarrow \frac{d\theta}{dt} = x_2 + b_0 v_c \tag{11}$$

The choice of x_1 is clear since it is simply pendulum angle θ. However, we note that the choice of x_2 is not simply angular velocity. It also includes a term made to cancel-out the time differential of control velocity term, as will shortly become apparent. Differentiating the state x_1 with respect to time we get:

$$\Rightarrow \dot{x}_1 = \frac{d\theta}{dt} \tag{12}$$

Representing this in terms of the control and the state x_2:

$$\Rightarrow \dot{x}_1 = x_2 + b_0 v_c \tag{13}$$

$$\Rightarrow \dot{x}_2 = \frac{d^2\theta}{dt^2} - b_0 \frac{dv_c}{dt} \tag{14}$$

$$\Rightarrow \frac{d^2\theta}{dt^2} = \dot{x}_2 + b_0 \frac{dv_c}{dt} \tag{15}$$

Substituting the terms into equation (8)

$$\Rightarrow \dot{x}_2 + b_0 \frac{dv_c}{dt} = -a_1(x_2 + b_0 v_c) - a_2 x_1 + b_0 \frac{dv_c}{dt} \tag{16}$$

Cancelling terms and re-arranging this leads to

$$\Rightarrow \dot{x}_2 = -a_1 x_2 - a_2 x_1 - a_1 b_0 v_c \tag{17}$$

Rewriting equations (13) and (17) in state space matrix notation we see that

$$\dot{x}_1 = 0x_1 + 1x_2 + b_0 v_c \tag{18}$$

$$\dot{x}_2 = -a_2 x_1 - a_1 x_2 - a_1 b_0 v_c \tag{19}$$

Writing equations in matrix format we therefore have

$$\Rightarrow \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -a_2 & -a_1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} b_0 \\ -a_1 b_0 \end{bmatrix} v_c \tag{20}$$

$$\Rightarrow \mathrm{y} = [1 \quad 0] \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \tag{21}$$

4 Observer to estimate state

We use a Luenberger observer to estimate the full inverted pendulum system state by using the state space a model of the plant. This is captured by the matrices A and B given in equation (20), and a correction term arising from the measured output. We note that this is necessary since the state x_2 is not available for measurement in our implementation. The observer generates the state estimate according to the dynamical equation:

$$\dot{\hat{X}} = A\hat{X} + BU + L(Y - C\hat{X}) \tag{22}$$

The observer gain vector L needs to be found such that the eigenvalue solutions λ to the characteristic equation for its error dynamics all have suitable negative real values:

$$|(A - LC - \lambda I)| = 0 \tag{23}$$

Here we choose observer correction gain L using Matlab (The MathWorks Inc., Natick, MA, USA) by means of pole placement using the place command, with poles set to [-20 -24]. The values were found by experimentation.

5 Augmenting positional state and adding integral action

It is possible to build a controller just using feedback of the 2-dimesional state derived from the observer in equation (22). However, in practice we want to control cart position as well as pendulum angle, since otherwise the cart has no reason to stop moving or remain at a given location. To control cart position too, we add a third state x3 to represent cart position. In the current pendulum design, there are two options to obtain cart position. We can either integrate the velocity control signal or we can count the pulses sent to the stepper motor and scale the count appropriately. In our analysis, here we demonstrate the former approach.

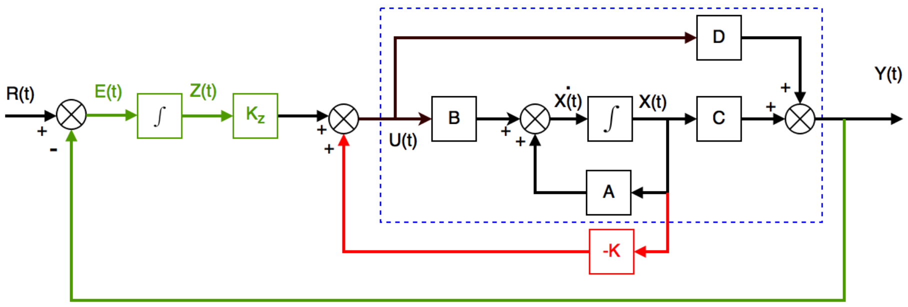

Fig. 4. Adding integral feedback using an integrator to reduce zero steady state error. The output from the plant is compared against a reference input (in our case zero). The resulting error is then integrated and weighted by the integral gain and added to the state feedback.

To use to the control signal in this way to estimate cart position, we note that differential of x3 is simply given by the input velocity control signal. Therefore, we can write:

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 \\ -a_2 & -a_1 & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} + \begin{bmatrix} b_0 \\ -a_1 b_0 \\ 1 \end{bmatrix} v_c \tag{24}$$

$$\mathrm{y} = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} \tag{25}$$

Integral action for a general system is illustrated in Fig. 4 and it provides an effective means to reduce steady state error for a state space controller. To reduce steady state positional error of the pendulum cart location, we also add an additional state within our controller that computes the integral of the positional error. To implement integral action, we further augment the pendulum system matrices given in equation (24) by adding a fourth state x_4 to represent integrated cart position error. This formulation assumes that our reference input is zero.

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \\ \dot{x}_4 \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ -a_2 & -a_1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} + \begin{bmatrix} b_0 \\ -a_1 b_0 \\ 1 \\ 0 \end{bmatrix} v_c \tag{26}$$

$$\mathrm{y} = \begin{bmatrix} 1 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} \tag{27}$$

6 State feedback control

Finally, we design the feedback gain vector K needed to implement full feedback state control in the full state vector as given in equation (26), although we use estimates of states x_1 and x_2 obtained from the Luenberger observer, i.e. $xhat_1$ and $xhat_2$. The eigenvalues λ of the state feedback control system are found by solving its characteristic equation

$$|(A - BK - \lambda I)| = 0 \tag{28}$$

We can thus influence the location of eigenvalues of the system by changing gain matrix K. We determine K using pole placements using the Matlab place command, with poles set to [-8.8 -9.6 -10.4 -1.6]. The feedback gains were also found using a process of experimentation. We note that the observer poles were deliberately selected to be more aggressive than those used for the feedback controller, to ensure faster settling of its state estimate. The structure of the observer-based controller is illustrated in Fig. 5.

7 Pseudocode and Arduino implementation

After the system matrices for the system, and the observer and controller gains L and K were calculated, they were tested using a Matlab simulation that made use of a non-linear simulation of the inverted pendulum, based on equation (1). Simulations (Fig 6) indicated that with appropriate pole placement, the design was able to resist small velocity disturbances to the balanced pole (modeled as small non-zero initial rod velocity) and maintain the cart at its origin. Importantly this was achieved when cart velocity was limited to the maximum cart speed that could be achieved by the stepper motor. We then implemented the observer-based real-time controller using an Arduino Mega. Fig 7. Shows the controller unit with its lid removed. This controller case was 3D printed and attaches to the pendulum frame with T-nuts. We note that the fan was essential in this design to prevent the stepper driver from overheating. The control processing was carried out in a poll loop. Its operation involves reading the encoder to determine pendulum angle, calculating the velocity control signal and using it to generate an output pulse train to drive the stepper motor in. State updates were performed using Euler integration [9].

Pseudo-code for the main controller loop operation is as follows:

```
loop
        Calculate time since last update
        Read the pendulum angle
        Compute control u using full state $[xhat_1\ xhat_2\ x_3\ x_4]^T$
        Calculate observer correction term using encoder position
        Update observer state estimates $[xhat_1\ xhat_2]^T$
        Use control velocity from input to update cart position $x_3$
        Update integral action positional error state $x_4$
        Generate stepper velocity drive pulses from u
End
```

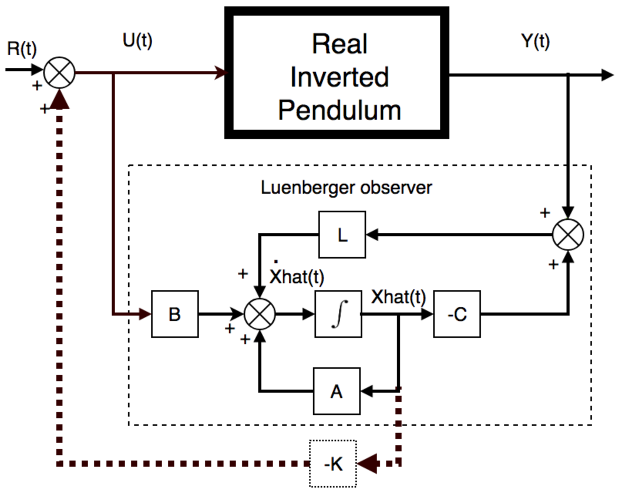

Fig. 5. State feedback control using a Luenberger observer to estimate pendulum state. We only have access to the output angle of the real pendulum Y(t), which is shown in the upper part of the diagram, and not all of its states. The observer, shown in the lower part of the diagram, is simply a model of the real inverted pendulum dynamics and generates a state estimate Xhat, which is used to provide state feedback. The output angle from the pendulum encoder is represented by Y(t), and this signal is only used to correct the state estimate.

8 Results and conclusions

The pendulum operated effectively and is able to balance the pendulum in its inverted state. When the controller was running, the pendulum could resist small tapping disturbances and it was possible to balance objects on its endpoint. Indeed, provide the encoder is precisely adjusted so that the balance state precisely aligns with a reading corresponding to 0°, balancing is maintained for many 10s of minutes (if not longer).

The unit is relatively cheap to build, with the parts for a unit certainly costing (in 2019) no more than around £400, and potentially less, provided good value-for-money components are used, which can be obtained from internet suppliers such as Amazon and eBay. More information on the inverted pendulum including videos of its operation are available at Howardlab.com/pendulum. The design is freely available for download, as well as the list of parts. In addition, an Arduino state feedback control program is supplied, as well as a set of utility programs to test the encoder, stepper motor, Hall sensors and menu control system. All these resources will be updated as improvements to the pendulum hardware and software are made.

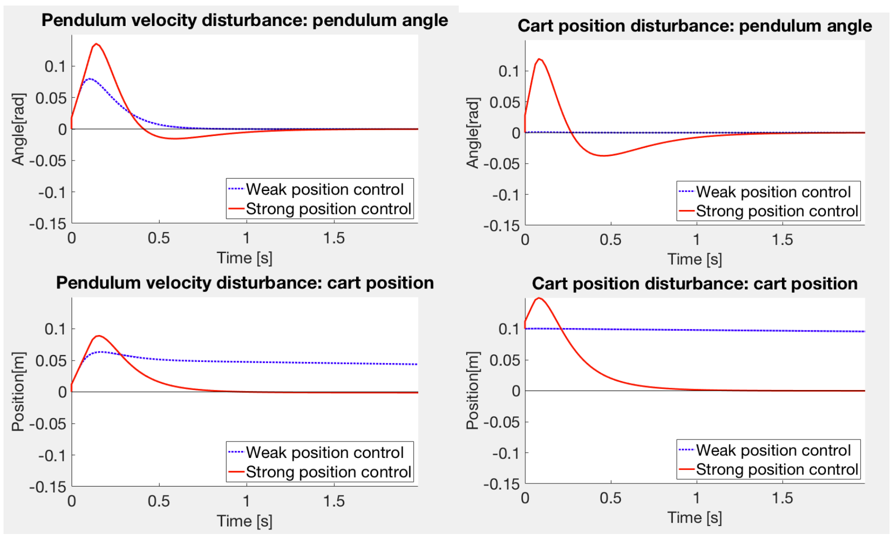

Fig. 6. Simulation of the pendulum system in Matlab showing effect of weak (poles at [-8.8 -9.6 -0.016 -0.008]; blue line) and strong control of cart position (poles at [-8.8 -9.6 -0.016 -4.0]; red line). Left panels shown the response to an initial perturbation of pendulum rod angular velocity (a simulated tap of the rod). Right panels shown the response to an initial displacement of pendulum cart position from its origin. In both cases the pendulum angle stabilizes to zero. However, it can be seen that when poles relating to cart position and its integral error are weak (close to zero), cart position is barely compensated.

9 Acknowledgments

Two 3rd year University of Plymouth student projects, undertaken by Jacob Threadgould and Daniel Hunt, influenced and contributed to the inverted pendulum design presented here. We thank Phil Culverhouse and two anonymous reviewers for commenting on an earlier version of the manuscript.

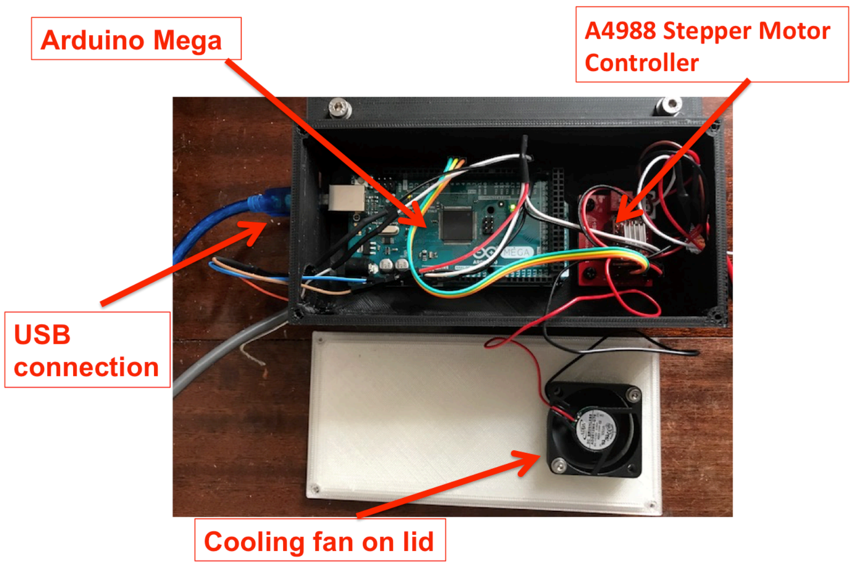

Fig. 7 Arduino Mega-based stepper motor controller unit using am A4988 stepper driver.

10 References.

[1] C. A. I. C. S. Magazine1989, "Learning to control an inverted pendulum using neural networks," cs.colostate.edu
[2] J. L. Cabrera, J. M. N. Studies, Buss, "Stick balancing: on-off intermittency and survival times," researchgate.net.
[3] 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.
[4] S. Franklin, J. Cesonis, and D. W. Franklin, "Influence of Visual Feedback on the Sensorimotor Control of an Inverted Pendulum," presented at the 2018 40th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC), pp. 5170–5173.
[5] S. Kajita, F. Kanehiro, K. Kaneko, K. Y. R. A. Systems, 2001, "The 3D Linear Inverted Pendulum Mode: A simple modeling for a biped walking pattern generation," ieeexplore.ieee.org

[6] S. Awtar, N. King, T. Allen, I. Bang, M. Hagan, D. Skidmore, and K. Craig, “Inverted pendulum systems: rotary and arm-driven - a mechatronic system design case study,” Mechatronics, vol. 12, no. 2, pp. 357–370, Mar. 2002.

[7] F. Grasser, A. D'arrigo, S. C. I. T. on, 2002, “JOE: a mobile, inverted pendulum,” robonz.com.

[8] R. C. Dorf and R. H. Bishop, Modern control systems. 2011.

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