# Below-The-Knee Prosthesis

David Collins

david.collins4@mydit.ie

C10736685

Supervisor: Dr. Ted Burke


Semester 2 2014

Electrical & Electronic Engineering (DT021)

# Declaration of Authenticity

I, the undersigned, declare that this report is entirely my own written work, except where otherwise accredited, and that it has not been submitted for a degree or other award to any other university or institution.

Signed: _____

Date: _____

**Abstract**

Traditional prostheses are passive devices which fall dramatically short of a natural limb's functionality. The goal of this project was to design and prototype an active transtibial prosthesis using a novel manufacturing technique, and to explore a means of neural control for the device also.

A single-axis prototype — of size and weight comparable to that of the natural human counterpart — was developed. Additive manufacturing (also known as '3D printing') was used to produce some of the essential mechanical components. 3D printing can be used to personalise medical devices to a significantly greater extent that is possible using traditional manufacturing techniques.

Mechanomyography was explored as a means of neural control of the prosthesis. Mechanomyography (MMG) is a viable alternative to electromyography (EMG) in prosthetic applications, and requires less pre-treatment of the skin over which the sensor is placed. Two lines of enquiry can be pursued in exploring the use of MMG signals in such applications. The first relies on the observation that the mean power frequency of the signal varies with the maximum voluntary contraction of the muscle. This relationship is hard to describe precisely, however. An alternative method is to search for 'signatures' in the MMG frequency spectrum that correspond to anatomical positions.

A simple potentiometer-based angle sensor was used to measure the degree of flexion in the ankle joint, and feedback from this allowed for a closed-loop control system.

Testing the device in a 'live' sense is difficult, given the ethical and logistic complications in finding a suitable test candidate. Nonetheless, a variety of meaningful tests were performed to assess the control system and the signal processing algorithm design — and to gauge how the power consumption requirements of the device compare to those of a natural lower limb.

The power consumption of the prosthesis in quiescent mode (no flexion of the ankle joint) is approximately 2 watts. This rises to approximately 6 watts during flexion without load-bearing. Further testing or research would permit the power requirements of the device to be determined in a wider range of operating conditions.

Experiments with mechanomyography were encouraging. The main frequency content of the signal lies between 5 and 45 Hz as expected, and distinct patterns were discernible in the frequency spectrum of the MMG for different degrees of plantar flexion.

Possible directions for future improvements or enhancements are discussed in closing.

**DT021 Year 4 — Final Project**

**May 2014**

**David Collins**

**Acknowledgements** Thanks first and foremost to Dr. Ted Burke. Dr. Burke proposed the use of mechanomyography (MMG) as an alternative to electromyography (EMG), and suggested the use of a potentiometer as a suitable angle sensor for joint positioning. He provided the needed impetus at various stages throughout the semester.

Thanks also to the technical staff in the electrical engineering department in Kevin Street, who helped with sourcing vital components on one or two occasions.

# Contents

# List of Tables



# List of Figures

# 1 Introduction

**Objective** The goal of this project was to design and prototype an active below-the-knee prosthesis. A below-the-knee prosthesis might also be referred to as a transtibial prosthesis. Transtibial prostheses have been available commercially for a long time. It is only in more recent years, however, that active prostheses have been developed.

**Motivation** Designing an active prosthesis presents an opportunity to apply principles from a variety of fields within electrical engineering — such as control theory, signal processing, as well as fundamental electrical theory. This opportunity in itself may be a sufficient reason to examine this problem. Reasons for performing research in prosthesis design should also be considered from the perspective of people who use them, however. One of the most significant challenges that designers and manufacturers of a prosthesis face is customising certain aspects of the device to an individual user's needs or preferences. A potentially viable means of overcoming this problem is explored here.

The incentives for this project fall in to two categories. Firstly, customisation of existing designs as well as rapid fabrication of prosthesis components during the prototyping stage is a challenge. Although the number of candidates who would benefit significantly from active prostheses is high [11], such devices are difficult to customise to individual needs. Certain aspects of a prosthesis design, such as the socket in to which the residual limb is fitted, may need to be tailored to each individual user. Thus, the use of 3D printing as a means to personalise prosthesis components was explored as part of this project, and was one of the primary incentives for the research. The project provided a unique opportunity to apply novel manufacturing techniques to an important biomedical challenge.

Secondly, optimal control of powered prostheses is a clinically-significant problem which merits close investigation [8]. The control engineering challenge here is not limited to just controller type or tuning — it involves figuring what the system is meant to be doing in the first place. In a typical control engineering problem, the set-point or command variable is well-defined and unambiguous. In the case of prosthesis control, the matter is not so straightforward.

Before attempting to even design a controller, the question needs to be posed: what exactly is the prosthesis meant to be doing? Should it be under volitional control from the individual using it? Should it respond to its environment in an autonomous manner? Should the control engineering strategy accommodate a combination of both tactics? It isn't purported that such complex issues are resolved here, but the approach to the problem should be framed against these sort of questions, and the opportunity to investigate these issues provided another strong incentive for investigations in this area.

**Implementation** Once these goals were established, a mechanical and electrical design was developed, and software for the control system was written. 3D models were designed using appropriate software, endeavouring throughout to produce a design that was aesthetically acceptable as well as mechanically sound. These models were then 'sliced' to prepare a CNC instruction set for a 3D printer. A 3D printer which uses the Fused-Deposition Modelling (FDM) technique was used to actually fabricate the parts.

It was resolved in the early stages of the project that — in order for the design to be highly customisable — a significant effort would be made to integrate 3D printed components in to all aspects of the design. In addressing the problem this way, it was deemed appropriate to start with a completely new design rather than adapt a prior prosthesis design, which might not be amenable to 3D printing techniques to begin with. Consequently, each decision regarding the mechanical or aesthetic aspects of the design was guided by the goal that, unless it threatened the mechanical integrity of the prosthesis, a design which could be fabricated using additive manufacturing techniques would be preferred to one which could not.

Simultaneously, an electrical design and control engineering strategy were developed. Three different classes of actuators were considered — pneumatic, DC electrical motors, and shape memory alloys. A DC motor was chosen on the basis that its electrical and mechanical characteristics are well-understood and suitable for integration in to robotic and prosthetic applications. An embedded Linux platform was chosen as the controller, and a program which performs proportional control of the actuator-joint complex was developed. The actual process variable to be controlled was the angle of the ankle joint in the sagittal plane. An angle sensor based on a potentiometer was used to provide position feedback and close the control loop. Then, using this feedback, both proportional and on-off control were explored as suitable candidates for accurate joint-positioning. The implementation is discussed fully in Section 3.

**Testing** The controller design was tested by examining the closed-loop system response to a series of step change inputs of different sizes. Two different controller configurations were used — proportional control and on-off control. The power consumption of the device was measured in two different modes — 1. quiescent (the actuator disabled), and 2. flexing under minimal or no load.

The viability of 3D printing as a means to fabricate necessary components for a lower-limb prosthesis can be evaluated according to two factors. Firstly, the cost — in terms of both time and money — can be considered. This factor is discussed in Section 6.1. Secondly, the mechanical integrity of the components can be considered. A typical means of assessing this factor is to perform some sort of stress tests. These tests are often destructive, however, which was highly problematic in this case. Damage to a mechanical component would have necessitated re-printing it — and the time was not available to do this. Alternative options for evaluating the mechanical system are suggested.

Tentative experiments with mechanomyography were performed also, to explore their viability as a means of myoelectric control and whether or not they could be easily integrated in to the existing design. A piezoelectric transducer was used to read the mechanomyogram and a short-time Fourier transform was used to examine the mean power frequency of the signal. A potential relationship between the MPF and reported muscular effort was explored. These tests were inconclusive. Thus, a second experiment was designed which aimed to identify 'signatures' in the Fourier transform which corresponded to different degrees of actuation in the joint of interest. The results, although preliminary and inconclusive, were encouraging. They are presented in Section 5.

Further details of the testing procedures are presented in Section 4.

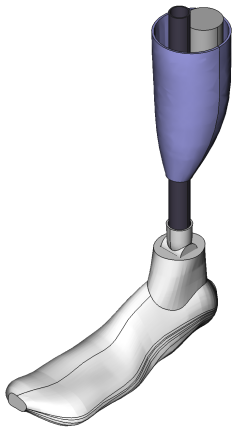

Figure 1: Design concept

**Results** Results of testing are presented in Section 5. Section 6 offers a balanced and critical evaluation of the project outcome, including suggestions for further development and testing.

# 2 Background

Before discussing the implementation in detail, a review of relevant research and literature is presented. A brief discussion of gait analysis is provided, and its relevance to the current design is explored. The impact of additive manufacturing on biomedical engineering is examined also. Recent research on the nature and underlying cause of the mechanomyogram, and the potential of mechanomyograph as a means of myoelectric control, is surveyed. Lastly, existing transtibial prosthesis designs of clinical and scientific interest will be discussed as a basis for comparison to the design developed here.

## 2.1 Human gait

A good topic with which to begin discussion on a transtibial prosthesis design is the functionality which the device aims to restore — a natural human gait. The mechanical and electrical system designs must take in to consideration the typical power consumption requirements during human locomotion. This, in turn, is greatly enriched by an analysis of human gait itself.

The human gait cycle can be divided in to two phases: 1. stance phase, and 2. swing phase. The stance phase for either the left or right side begins with heel contact of that foot with the ground and ends with toe-off of the same foot. The two phases of gait can be subdivided in turn. Stance phase, for example, can be regarded as consisting of four distinct components — 1. *heel contact*, 2. *mid-stance*, 3. *active propulsion*, and 4. *passive propulsion*.

A useful aid in gait analysis are gait graphs. Gait graphs are often used by clinicians — whether they are physiotherapists, orthopaedists, etc. — to identify pathologies. This is not the incentive for invoking them here, however. They are examined in this context, rather, as a potential design aid.

An example of a gait graph is presented in Figure 2. The graph portrays the degree of plantar or dorsi-flexion at various stages of a single gait cycle. The variable of interest in this case is the degree of rotation of the ankle joint in the sagittal plane. More specifically, it is the relative angle between the calcaneus and tibia (shown in Figure 2 in orange and blue respectively). Positive angles correspond to dorsi-flexion, negative angles to plantar-flexion. The data indicated was generated using the OpenSim biomechanics simulation software [5] and is that of a normal gait cycle.

Note that the parameter of interest (tibial rotation in this instance) returns almost to its starting point by the end of the graph. This highlights the fact that we are looking at a single cycle of a cyclic or periodic phenomenon. Graphs of adjacent cycles will be similar — obviating the need to show more than one.

The ankle joint actually exhibits rotation in three planes. However, the rotation in the two other planes is more limited than that in the sagittal plane.

Gait graphs — aside from their clinical use — provide a number of useful pieces of information.

**Range of tibial rotation** We see from the graph presented, for example, that the tibial angle typically varies between -16 and 12 degrees over a normal gait cycle (although this may vary significantly with age or other individual characteristics).

**Frequency information** From the perspective of a prosthesis design, the graph is a preliminary indication of the frequencies that will be present in the feedback from the angle sensor — which in turn allows

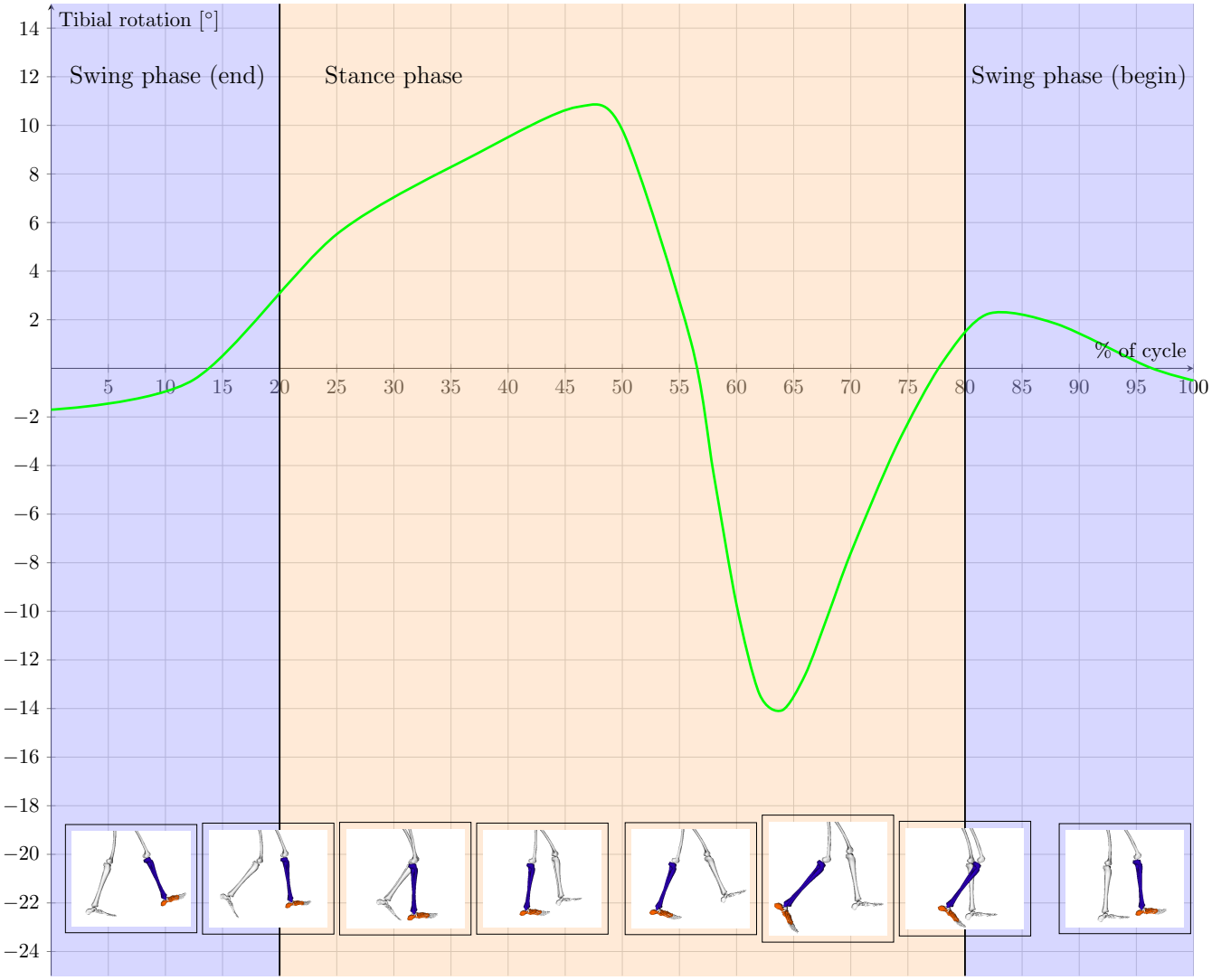


Figure 2: Tibial rotation in the sagitall plane (plantar/dorsi-flexion) throughout gait cycle.

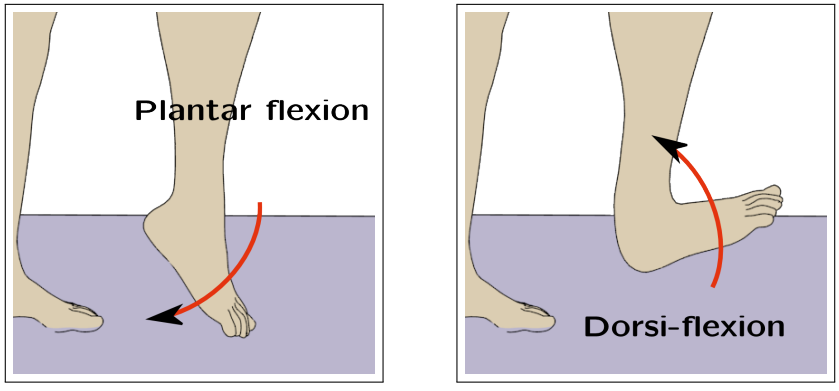


Figure 3: Plantar and dorsi flexion in the right ankle

us to select a suitable sampling frequency.

**Set point** In the case of an active prosthesis with a built-in controller, gait graphs reveal the time-varying nature of the set-point. Evidenly, typical graphs don't fall neatly in to simple categories such as step, ramp or sinusoidal over the entire cycle. However, they do approximate these classes of signal at various stages of the cycle.

**Power consumption** Lastly, a plot such as that shown in Figure 2 provides strong visual clues regarding the likely energy or power requirements at different phases of the gait cycle.

Other gait parameters — such as cadence and stride — may be used in conjunction with gait graphs to provide a fuller gait analysis.

Despite their utility, these graphs should not necessarily be used as standalone analysis tools, and may be misleading if they are. Analysing just a graph of the tibial rotation over a single cycle, for example, might suggest that tibial rotation is limited to the range exhibited during this cycle. A moment's reflection indicates, on the other hand, that this can not be the case. The ankle can plantar flex to a larger degree than is typically done during walking, for example. Thus, the range indicated in gait graphs should not be regarded as absolute limits.

## 2.2 Active prostheses

It was hoped from the outset that this could be an 'active' prosthesis design — in other words that it could be a powered device. A number of different actuator types were considered.

- Pneumatic cylinders,
- shape memory alloys, and
- DC motors (with or without a servo-mechanism)

The relative merits and drawbacks of each will be considered briefly.

Pneumatic cylinders could provide a very convenient form of linear motion. The ancillary pneumatic components (valves, etc.) and the cylinder itself could be quite expensive, however. Also, pneumatic cylinders don't necessarily allow fine-grained control over the extent of travel or position.

Research has been performed on the use of shape memory alloys in a number of upper-limb prosthesis designs [10]. Shape memory alloys have a high force-to-weight ratio and have received significant interest as potential robotic and prosthetic actuators for this reason. Research on the Rutgers robotic hand, for

example, suggests that forces of 40 N and greater are possible using Shape-Memory Alloy (SMA) wires in parallel [4] — configurations known as SMA bundles. Although these findings are very promising, there are significant challenges associated with the use of shape memory alloys. Perhaps most importantly, their behaviour is highly non-linear [12]. Secondly, they generate significant heat and thus ancillary cooling would be necessary.

DC motors, on the other hand, are well understood electro-mechanical systems which exhibit largely linear behaviour. Furthermore, there is a very strong precedent for their use in robotic and prosthetic applications. For these reasons, a DC motor was selected in this case. Although the presence of a motor in a replacement limb might seem 'unnatural', the are nonetheless being used in state-of-the-art prostheses [8, 14].

## 2.3 DC motor characteristics

### 2.3.1 Equivalent circuit

The DC motor is the 'machine' in the prosthesis — what transforms it from a passive object in to a dynamic prototype. As the active component in the device, it is worthwhile to consider its characteristics in detail.

A permanent magnet DC motor was used in this context. An equivalent circuit for such a motor is shown in Figure 4. There are really two circuits here — one electrical and one mechanical.

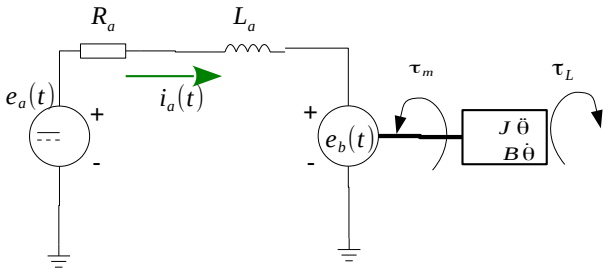


Figure 4: Equivalent circuit for a permanent magnet DC motor

This is a 'lumped component' model of the armature circuit. In reality, the resistance and inductance of the armature windings are not discrete devices, but are distributed throughout the entire length of the armature coil. As long as the wavelength of the voltage is much greater than the length of the coil, however,

this treatment of the electrical characteristics is entirely viable (otherwise it would be necessary to resort to transmission line theory).

The notation is as follows.

- $i_a(t)$ — the armature current
- $R_a$ — armature winding resistance
- $L_a$ — armature inductance
- $e_b(t) = k_b\omega$ — motor back-EMF
- $\tau_m$ — motor torque
- $\tau_L$ — load torque
- $J$ — moment of inertia of motor
- $B$ — motor damping

Applying Kirchoff's Voltage Law to the armature circuit yields

$$e_a(t) = Ri_a(t) + \frac{di_a(t)}{dt}L + e_b(t)$$

or

$$e_a(t) = Ri_a(t) + \frac{di_a(t)}{dt}L + k_b\omega. \tag{1}$$

In the steady state, the current will be unchanging and the above will simplify to Equation 2, which can be used to determine the back-EMF constant $k_b$ in instances where this is not provided.

$$e_a(t) = Ri_a(t) + k_b\omega \tag{2}$$

The motor torque, $\tau_m$, is proportional to the armature current according to Equation 3.

$$\tau_m = k_T i_a(t) \tag{3}$$

As long as the motor torque exceeds the torque exerted by the load, the angular position, $\theta$, of the motor will vary according to

$$B\frac{d\theta}{dt} + J\frac{d^2\theta}{dt^2} = \tau_m - \tau_L. \tag{4}$$

If we assume that all of the electromagnetic energy delivered to the motor shaft is converted to mechanical energy, we have

$$\begin{aligned} e_b(t)i_a(t) &= \tau_m\omega \\ \therefore k_b\omega i_a(t) &= k_T i_a(t)\omega \\ \therefore k_b &= k_T. \end{aligned}$$

Thus, a reasonable working assumption — in cases where electromagnetic losses are small at least — is that the back-EMF constant and torque constant are approximately equal.

The equations for the mechanical system simplify under certain conditions also. For example, if there is no load connected to the shaft and the shaft is rotating at a constant angular velocity, Equation 4 simplifies to Equation 5, since $\frac{d^2\theta}{dt^2} = 0$ and $\tau_L = 0$.

$$B\frac{d\theta}{dt} = \tau_m \tag{5}$$

This can be stated in terms of angular velocity ($\omega$) rather than angular displacement ($\theta$).

$$B\omega = \tau_m \tag{6}$$

### 2.3.2 Transfer function

Access to a Laplace-domain representation of a process or system greatly facilitates the task of designing a controller for that system. Thus, in order to implement a controller for our active prosthesis, an important step is transferring the system equations from the time domain in to the Laplace domain.[1]

Taking the Laplace transform of both sides of Equation 1, we have

$$\begin{aligned} E_a(s) &= R_a I_a(s) + sL_a I_a(s) + k_b\omega \\ \therefore I_a(s) &= \frac{E_a(s) - k_b\omega}{R_a + L_a s} \end{aligned}$$

Bringing Equation 4 in to the Laplace domain, on the other hand, yields

$$Bs\theta(s) + Js^2\theta(s) = \tau_m - \tau_L$$

or

$$B\omega(s) + Js\omega(s) = \tau_m - \tau_L.$$

Figure 5 is a block diagram view of these relationships. This block diagram, in turn, makes it easier for us

[1] The convention will be followed here that lower-case letters are used for time-domain variables, while upper-case letters are reserved for their frequency or Laplace-domain counterparts. (There are occasional exceptions to this.)

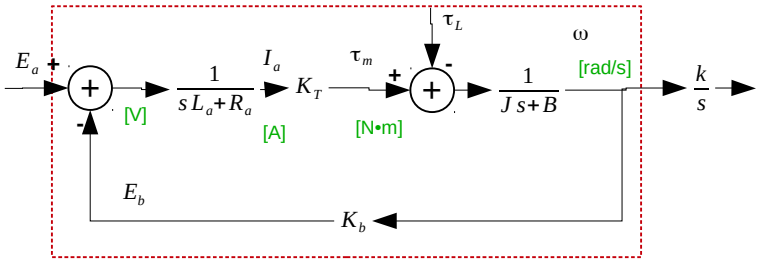


Figure 5: Block diagram representation of permanent magnet DC motor (dependency on $s$ is not indicated explicitly)

to determine an overall transfer function.

$$\begin{aligned}
\omega(s) &= \frac{\tau_m(s) - \tau_L(s)}{Js + B} = \frac{I_a(s)k_T - \tau_L(s)}{Js + B} \\
&= \frac{\frac{E_a(s) - E_b(s)}{L_a s + R_a} k_T - \tau_L(s)}{Js + B} \\
&= \frac{\frac{E_a(s) - k_b\omega(s)}{sL_a + R_a} k_T - \tau_L(s)}{Js + B} \\
\therefore \omega(s)(Js + B) &= \frac{E_a(s) - k_b\omega(s)}{sL_a + R_a} k_T - \tau_L(s) \\
\therefore \omega(s)(sL_a + R_a)(Js + B) &= (E_a(s) - k_b\omega(s))k_T \\
&\quad - \tau_L(s)(sL_a + R_a) \\
&= E_a(s)k_T - k_b k_T \omega(s) \\
&\quad - \tau_L(s)(sL_a + R_a)
\end{aligned}$$

Defining $k = k_b = k_T$, and rearranging, we have

$$\begin{aligned}
\therefore \omega(s) \left[(sL_a + R_a)(Js + B) + k^2\right] \\
= E_a(s)k - \tau_L(s)(sL_a + R_a).
\end{aligned}$$

$$\begin{aligned}
\therefore \omega(s) &= \frac{E_a(s)k - \tau_L(s)(sL_a + R_a)}{(sL_a + R_a)(Js + B) + k^2} \\
&= E_a(s)\frac{k}{(sL_a + R_a)(Js + B) + k^2} \\
&\quad - \tau_L(s)\frac{sL_a + R_a}{(sL_a + R_a)(Js + B) + k^2}
\end{aligned}$$

We thus have two transfer functions, indicative of different aspects of a motor's performance.

**Set-point tracking** This depends on the ratio between the motor shaft speed and the voltage across the armature windings

**Disturbance rejection** A measure of how load torque disturbances will affect the motor performance.

The motivation for modelling the motor in this fashion will become clearer when we look at the overall system diagram, with a view to selecting suitable parameters for a controller.

## 2.4 Additive Manufacturing

This project involved building a physical prototype. The prosthesis could have been built using standard, stock components. This would have limited the design possibilities significantly, however, and there was deemed to be a strong merit in custom mechanical components. Thus, some means of fabrication was required, and the most viable means for doing this in this context was Fused Deposition Modelling. Fused Deposition Modelling is a type of additive manufacturing — one of a large and increasing variety of techniques. Additive manufacturing in fact has significant relevance to the development of medical applications currently, and this link is worth examining briefly.

Additive manufacturing (or '3D printing' as it is commonly called) has been put to diverse uses for biomedical applications. The following list highlights some of the most important applications. 1. anatomical modelling (for demonstration or pedagogical purposes), 2. fabrication of surgical aids and tools, 3. dental implants (additive manufacturing has been used to produce dental implants made of zirconia [6]), 4. prostheses and orthoses, and 5. tissue engineering.

Beginning from 2012, interest in 3D printing for prostheses increased dramatically in the 'maker' and research communities. One example is the Open Hand project, which consists mainly of 3d-printed components [7]. This trend has been limited to upper-limb prostheses heretofore, however. For example, The E-NABLE project — a global movement of indiduals interested in 3D-printed prostheses — is almost exclusively focused on developing 3d-printed hands [15].

## 2.5 Mechanomyography

An important focus of modern prosthesis design is some form of volitional control. MMG and Electromyography (EMG) are the two main candidates for achieving this. Mechanomyography (MMG) was chosen rather than EMG, as research indicated three advantages in this context: 1. less complex circuitry needed;

2. higher signal-to-noise ratio compared to EMG; and 3. minimal pre-preparation of the skin surface required. The electromyogram is subject to electrical noise from devices such as fluorescent lights, etc. [9], for example.

The muscle which is perhaps the best candidate for mechanomyography and volitional control of a transtibial prosthesis is the gastrocnemius. The gastrocnemius is one of two muscles which comprise the 'calf muscle' — the other being the soleus. At the distal end, the aponeuroses of the gastrocnemius and soleus combine to form the Achilles tendon which attaches these muscles to the calcaneus (heel). The gastrocnemius lies superficially to the soleus and inserts proximally just above the knee. While the muscles responsible for finer control of foot movement might not be preserved in a below-the-knee amputation, the gastrocnemius — or at least its proximal portion — generally would be.

### 2.5.1 Factors affecting the MMG signal

In trying to understand the precise cause of the mechanomyogram, one relationship which has been investigated is that between the mean power frequency of the signal and the firing rate of motor units in the underlying muscle [2]. In addition to the motor unit firing rate, however, other factors which potentially affect the mechanomyogram include 1. the depth of the motor unit within the muscle or below the skin [2]; 2. the type of motor unit; 3. temperature; and 4. the degree of connective, adipose, or other tissue surrounding the muscle. All of these factors, in turn, can be affected by individual characteristics such as age [1] and gender.

## 2.6 Microprocessors

One of the earliest and most challenging decisions to be made in this project was the selection of a suitable microprocessor. Given the investment of time required to familiarise oneself with a new microprocessor or microcontroller platform, it was not really viable to experiment with a variety of different microprocessors in any great depth, or to switch to a different development platform part-way through the project. Each microprocessor or microcontroller requires different ways of dealing with hardware registers for timers, analog-to-digital converters, and so on.

Several factors were considered in selecting a suitable microprocessor.

**Processor speed** The primary factor of concern when considering the processor speed in this context was not the processor speed *per se*; it was, rather, the ability of the microprocessor or microcontroller to

support an operating system. As long as the processor was powerful enough to run Linux, it should be more than capable of running whatever software controller is devised.

**ADC channels** At least two Analog-to-digital converter (ADC) channels were desired — one for retrieving feedback from the potentiometer regarding angle position, and one for reading a mechanomyogram from a piezoelectric sensor should it be feasible to implement or test this feature.

**Analog outputs** Speed control of the actuator requires a varying output from the processor. Pulse-width modulation (PWM) is sometimes used as an alternative to true digital-to-analog conversion in this context, and this option was used here. The microprocessor used should have a hardware Pulse-width modulation (PWM) module ideally.

A variety of microprocessors with characteristics more-or-less suitable for this application were considered. These include the BeagleBone Black, the Raspberry Pi, and Olimex's oLinuXino platform. All of these platforms are capable of running a fully-fledged operating system, and this capability was one of the primary incentives for considering them. All of these have adequate processing speed. The oLinuXino i.MX233 (the Nano variant in particular) was chosen because of its reduced form factor.

## 2.7 State-of-the-Art

The last few years have witnessed a burgeoning of research and development in the area of prosthesis design. Furthermore, the undertaking of this project coincided with an increased interest in the use of additive manufacturing for prosthesis development. Inspired by examples where 3D printing and myoelectric control were applied to upper-limb prostheses, the goal of this project was to apply the same concepts and techniques to the development of a transtibial (below-the-knee) prosthesis.

A design which is particularly relevant to the current project is SPARKy (Spring Ankle with Regenerative Kinetics) — a powered transtibial prosthesis[8]. The device uses a novel means of preserving energy as well as an optical sensor located in the foot to detect the point of heel-strike in the gait cycle. Perhaps the most advanced of all transtibial prostheses currently available, however, is BiOM. Developed initially in the Massachusetts Institute of Technology biomechatronics's laboratory, BiOM is reported to give people the ability to walk with a near-normal metabolic demand [14]. The BiOM prosthesis is also of interest here on the basis that the socket in the design is fabricated using 3D printing techniques.

Advances in materials science and engineering complement these developments in prosthesis design. Some of these prostheses use custom carbon fibre components, for example.

# 3 Implementation

The following characteristics are desirable in a prosthesis.

**Neural or myoelectric control** This may be the factor that most clearly distinguish a prosthetic limb from that of a humanoid robot.

**Biomimetic** A mechanical design which is comparable to that of its natural human counterpart, in terms of size and weight, and which replicates the most important degrees of freedom.

**Efficiency** A level of power consumption not significantly above that of a natural limb, and no dramatic increase in metabolic demand due to presence of the prosthesis.

It is not claimed that all of these characteristics can be found in the design presented here — such a goal would require an experienced interdisplinary team with more time and resources than could be afforded to this project. Nonetheless, meaningful research has been performed on these areas, and a unique design was obtained which might be a useful to other investigators in this area. How precisely this was accomplished is discussed in the subsections that follow.

Figure 6: Degrees of freedom in the current prosthesis design vs. a simplified biomechanical model

## 3.1 High-level view of design

Figure 7 offers a block-diagram view of the design, from a control systems perspective. It is simply a closed-loop system with negative feedback, familiar to most engineers. We have a single loop here as the prosthesis has one degree of freedom. The design could be expanded to include more degrees of freedom and many aspects of the design would still be valid.

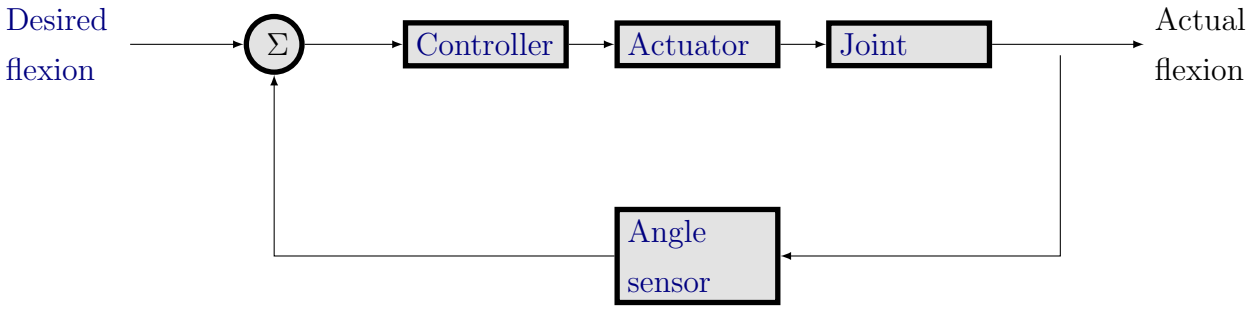


Figure 7: System overview

## 3.2 Actuatation

As there is a clear precedent for the use of motors in active lower-limb prostheses, and to minimise the number of uncertainties, it was decided to use DC motors rather than shape-memory alloys or pneumatic components. An obvious problem presents itself in this context, however. The actuator in a prosthesis such as this is intended to mimic the function of a muscle and/or tendon — this is a linear motion. The motion generated by a DC motor, in contrast, is rotary. An important challenge, thus, was to integrate a linear actuator of the appropriate size and gearing ratio in to the mechanical design.

### 3.2.1 Linear actuator design

The task of finding a linear actuator with the desired mechanical and electrical characteristics was non-trivial, so a custom linear actuator was designed for the prosthesis. In order for this to be successful, some consideration needed to be given to the power requirements of the prosthesis and the torque-speed characteristic of the motor. The values used in the following example may vary to that used in a prosthesis, but not significantly.

Consider, as an example, a case where the power requirement of the ankle during walking or exercise is

$$P_m \approx 20\text{W}.$$

$$\therefore \tau\omega = 20\text{W} = 20\text{N}\,\text{m}\,\text{s}^{-1}$$

If the tendon attaches to the ankle joint approximately 2cm from its pivot point, a rotation of 30 degrees (as might be performed during plantar flexion) corresponds to a linear travel of $\frac{30}{180}\pi(2\text{cm}) \approx 2.1\text{cm} \approx .02\text{m}$.

With a leadscrew pitch of 1mm = .001m, this will require 20 revolutions in one second.

$$\begin{aligned} \therefore f &= 20\text{s}^{-1} \\ \omega &= 2\pi f \approx 125\text{s}^{-1} \\ \therefore \tau &= \frac{P_m}{\omega} = \frac{20\text{N m}\cancel{\text{s}^{-1}}}{125\cancel{\text{s}^{-1}}} \\ &= .15\text{N m} \end{aligned}$$

Thus, with the ankle and leadscrew parameters given, the motor — during a typical instance of plantar flexion — would be operating at approximately 1200 RPM, delivering a torque of approximately .15 N m. We want the motor to operate somewhere in the middle of its torque-speed characteristic, to ensure that the maximum power is attained. Thus, a motor with a no-load speed of approximately 3000 RPM, and stall torque in the region of .3 N m would be ideal.

The motor used in this context was a Mabuchi motor whose parameters (Table 1) varied slightly from this, but not to the extent that it precluded the possibility of testing and development.

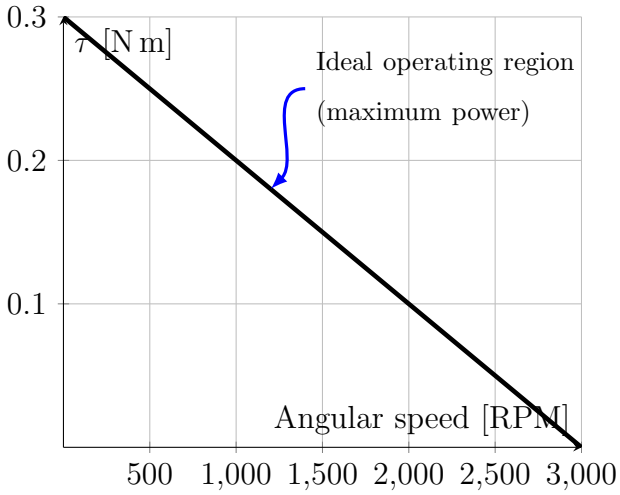


Figure 8: Torque-speed characteristic for a DC motor

### 3.2.2 Motor parameters

The motor used in this context was a Mabuchi RS-550PC. It's characteristics are summarised in Table 1.

From the torque and current values indicated for maximum efficiency in Table **??**, combined with Equation 3, we can determine the torque constant.

$$k_T = \frac{\tau_{max}}{i_{max}} = \frac{41.7\text{mN}}{10.8\text{A}} = 3.9 \cdot 10^{-3}\text{N A}^{-1}$$

| Parameter | Symbol | Value | Unit |
|---|---|---|---|
| Manufacturer-provided | | | |
| Armature resistance | $R_a$ | 12 | $\Omega$ |
| Armature inductance | $L_a$ | 1.8 | mH |
| No-load speed | $RPM_{noload}$ | 15000 | RPM |
| No-load current | $i_{noload}$ | 1.4 | A |
| Current at maximum efficiency | $i_{max}$ | 10.8 | A |
| Torque at maximum efficiency | $\tau_{max}$ | 41.7 | $\mathrm{mN \cdot m}$ |
| Derived | | | |
| Back-EMF constant | $k_b$ | $3.9 \cdot 10^{-3}$ | $\mathrm{V\,rad^{-1}\,s^{-1}}$ |
| Torque constant | $k_T$ | $3.9 \cdot 10^{-3}$ | $\mathrm{N\,m\,A^{-1}}$ |
| Damping coefficient | $B$ | $3.5 \cdot 10^{-6}$ | $\mathrm{N\,m\,s\,rad^{-1}}$ |
| Estimated | | | |
| Moment of inertia | $J$ | $1 \cdot 10^{-6}$ | $\mathrm{N\,m\,rad^{-1}\,s^{-2}}$ |

Table 1: Mabuchi RS-550PC DC motor — Important parameters

If there are minimal or no losses, the back-EMF constant is approximately equal to the torque constant.

$$k_b \approx k_T = 3.9 \cdot 10^{-3}\mathrm{N\,A^{-1}}$$

From Equation 6 and the no-load parameters indicated in Table 1, we can also determine the motor damping coefficient.

$$\begin{aligned} B = \frac{\tau_m}{\omega} &= \frac{k_T i_{noload}}{2\pi \frac{RPM_{noload}}{60}} \\ &= \frac{(3.9 \cdot 10^{-3}\mathrm{N\,A^{-1}})(1.4\mathrm{A})}{500\pi \mathrm{s^{-1}}} \\ &\approx 3.5 \cdot 10^{-6}\mathrm{N\,m\,s\,rad^{-1}} \end{aligned}$$

The moment of inertia is less straightforward to derive; nor was it available in the manufacturer's datasheet. A typical value for other DC motors of this torque capacity is approximately $1 \cdot 10^{-6}\mathrm{N\,m\,rad^{-1}\,s^{-2}}$. These values will be used shortly in simulations of a closed loop control system.

## 3.3 Motor driver

The output pins of the i.MX233 microprocessor can not provide enough current to drive the motors directly. They are used, instead, to switch a power transistor which has the required current capacity. A common

designs employs these transistors in a H-bridge configuration, which prevents transient currents due to the motor inductance from damaging the transistors or power supply. Such circuits can be designed anew for a given application, or acquired ready-made in a standard chip package.

Implementing a custom H-bridge driver might be warranted in a production application. It was deemed appropriate in this case, however, to use a ready-made solution and concentrate instead on aspects of the design which harboured more uncertainty. Thus, an L293 dual-channel MOSFET driver was used for the DC motor. The L293 can sustain a peak output current of 1.2 amperes, which — although perhaps somewhat low for a live context — is useful for testing purposes. It uses voltage levels compatible with standard microprocessors.

**Pulse-width modulation** The direction of the motor was controlled by two digital outputs on the i.MX233. A third, Pulse-width modulation (PWM), signal was connected to the enable channel on the L293 driver. Pulse-width modulation (PWM) is only used in the case of the P and PI controllers — it is not used for the on-off controller. Software Pulse-width modulation (PWM) — sometimes referred to as 'bit-banging' — was used in this context. The duty cycle was varied according to the current state of the controller.

## 3.4 Control

Expanding on the high-level block-diagram representation of the system in Figure 7, we arrive the Laplace-domain view presented in Figure 9.

There are three options for determining the controller parameters:

- **Manual tuning** This involves a certain amount of trial and error, essentially, to arrive at acceptable parameters. If the process to be controlled is unlikely to change, manual tuning might in fact be the most viable option.
- **Ultimate cycle tuning method** The controller is placed in proportional-only mode (i.e. the integral and derivative gains are set to zero), and the proportional gain is carefully increased until the process variable oscillates at a steady amplitude. The period of oscillation, referred to in this context as the 'ultimate period', is measured and, in conjunction with the ultimate gain, is used to determine suitable parameters for P, PI, or PID controllers.
- **Process Reaction Curve** This method lends itself most readily to third-order systems, which can in turn be modelled as first-order systems with a transport delay or 'dead-time'. The step response

— or 'process reaction curve' — of the system is obtained, from which a time constant, steady-state gain and time delay are determined. These values, in turn, can be used to select sensible values for a particular controller configuration.

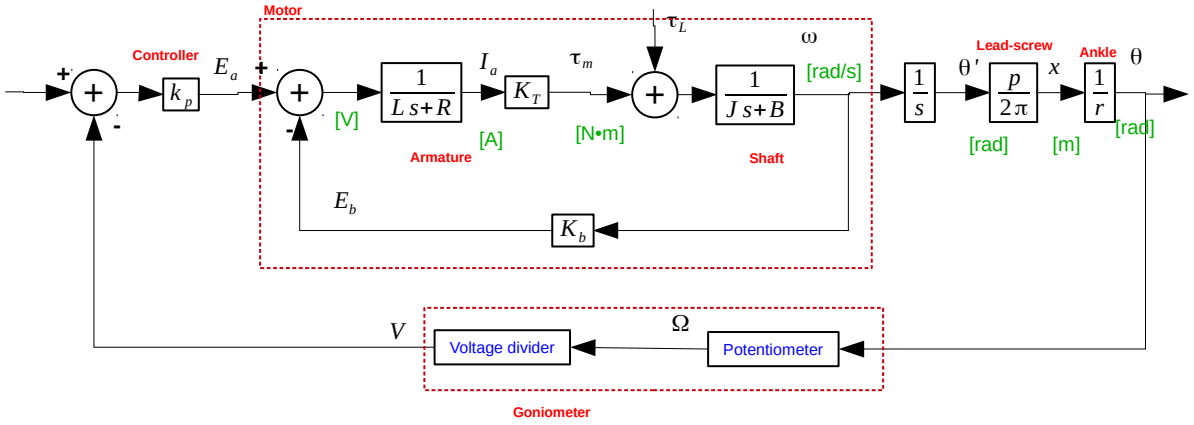


Figure 9: Laplace-domain view of system

In the case of no load disturbance (i.e. $\tau_L = 0\text{N m}$), the ratio of the motor shaft angle to the armature voltage is

$$\begin{aligned}\frac{\theta'(s)}{E_a(s)} &= \frac{1}{s}\frac{k}{(R_a + sL_a)(Js + B) + k^2} \\ &= \frac{k}{s^3 JL_a + s^2[R_a J + BL_a] + s[R_a B + k^2]}.\end{aligned}$$

Considering the gain of the lead-screw and pulley, we can determine how the actual tibial angle varies with the armature voltage.

$$\frac{\theta(s)}{E_a(s)} = \frac{p}{2\pi r}\left(\frac{k}{s^3 JL_a + s^2[R_a J + BL_a] + s[R_a B + k^2]}\right)$$

This is the 'plant' of our system. It is shown in a simplified s-domain view of our prosthesis in Figure 10.

Using the values shown in Table 1, we can simulate the response of the system to a step input. The results is shown in Figure 11, and can be compared with the test results presented in Section 5. The long settling time is due, in part, to the relatively large resistance of the armature windings. The simulation was performed in Scilab, and the necessary code can be found in Appendix D.

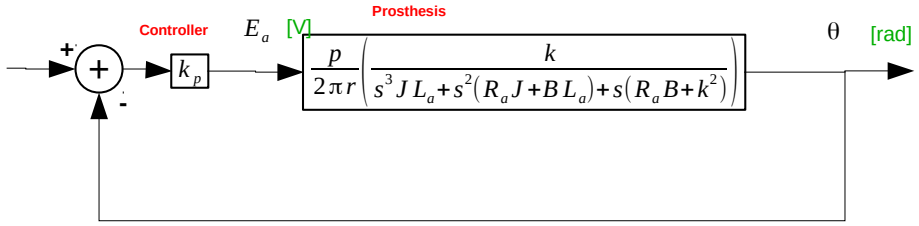


Figure 10: Simplified Laplace-domain view of system (assuming no load torque disturbance)

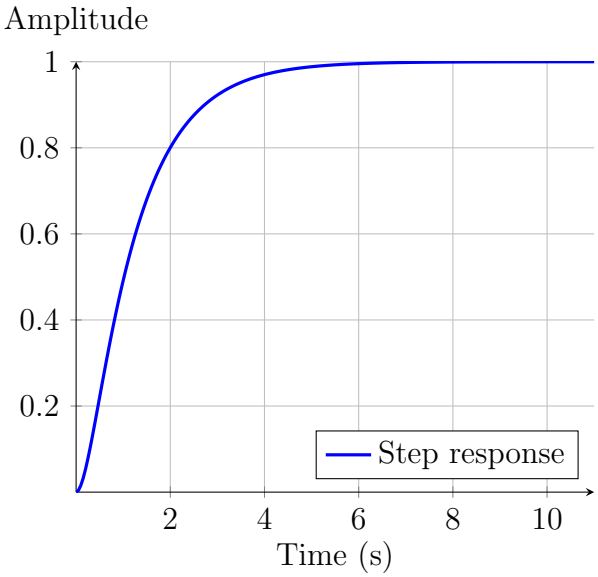


Figure 11: Simulated response of closed-loop system to a step input. Proportional control. $k_p = .35$

## 3.5 Controller feedback

In order to implement a meaningful control system, some way of 'closing the loop' needs to be found. Once this is done, i.e. once the position of the end-effector is 'fed back' to our controller, we might have a limited form of 'proprioception'.

### 3.5.1 Servo vs. direct measurement

Rather than using a DC motor with a servomechanism — where we rely on the control system within the servo motor to minimise the error — it was decided instead to measure the controlled variable and

determine the error directly.

In this case, the process variable — an angle — was measured using a potentiometer. By using the voltage divider configuration shown (Figure 12), the variable resistance generated a voltage which was connected to the microprocessor ADC. It was necessary to use a voltage divider in this context because the i.MX23 analog to digital converter hax a maximum input voltage of 1.85 V — it clips voltages above this value. The potentiometer used in this case had a resistance of 1 kΩ. The analog input channels on the i.MX23 have an input impedance of approximately 150 kΩ, which was sufficiently large to prevent loading of the potentiometer output.

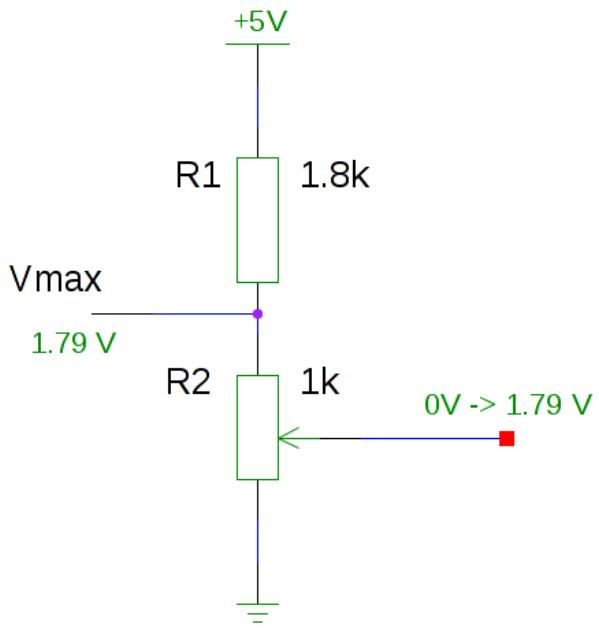


Figure 12: Goniometer

## 3.6 Microprocessor Platform

### 3.6.1 i.MX23 System-on-a-Chip

The oLinuXino platform used for this project features an i.MX23 System-on-a-Chip, which has the ability to run a Linux operating system. A system with a fully-fledged operating system has a number of advantages.

- **Extensibility** Integrating in to a more sophisticated prosthetic platform is possible.

- **Data-logging** AVR and PIC microcontrollers have reasonably sophisticated debugging facilities, but these do not quite match the logging capabilities available on a true operating system.
- **Network access** Network access with, e.g., an Arduino or other microcontroller platform, although possible, is time-consuming. Using an operating system such as Linux, in contrast, makes available a full TCP/IP stack, which makes certain networking operations significantly eaiser. Although networking facilities might be exploited only rarely (if at all) in the day-to-day functioning of an active prosthesis, it does have the potential to be very useful in the device development stages and perhaps in the early stages of rehabilitation also. The developers of the BiOM transtibial prosthesis, for example, use bluetooth capabilities to transmit real-time information regarding kinematics while an indivdual is testing a prosthesis [14].

In the explanation which follows, the register and field names used are those in the Freescale Semiconductor manual [13].

### 3.6.2 Analog-to-digital conversion

The i.MX233 features a 16-channel analog-to-digital conversion block with 12-bit resolution. These 16 physical channels are mapped to 8 virtual channels using a configurable multiplexer. The ADC hardware includes 8 result registers (one for each channel), 5 control registers, a status register and several ancillary registers. An ADC conversion on any virtual channel is invoked by setting the relevant schedule bit in the first control register. Once a scheduled conversion is completed, the hardware resets the schedule bit.

The invoking thread or process has two ways of ensuring that the conversion is finished before fetching the result.

**Software polling** It can poll the schedule bit (which is reset by the hardware upon completion).

**Interrupts** Alternatively, it can use interrupts. This requires enabling LRADC$n$_IRQ_EN in the relevant control register, where $n$ is the virtual channel for which the conversion is desired. If this bit is enabled, an interrupt will be generated once the conversion is complete. It is the responsibility of the calling process to reset or clear this bit.

Although interrupts obviate the need for software polling of registers, the onus is on the programmer to prepare an approprite interrupt handling routine in this instance. This is more challening in a Linux operating system environment than on a standalone microcontroller.

The source code used to configure and use the ADC in this project is presented, with comments, in

Appendix A.

### 3.6.3 Runtime configuration

GNU/Linux, like any other operating system, has multiple processes running simultaneously all the time. Most of these are background processes with which a user will have no interaction. To minimise start-up delays and computational overhead, the number of these background processes should be kept to a bare minimum for this application. Thus, all unnecessary services and daemons were disabled, resulting in a shorter start-up time on power-on.

## 3.7 Modelling

A number of mechanical components were custom designed specifically for this project, prior to fabrication. Figure 13, for example, shows the prosthetic foot used for this purpose.

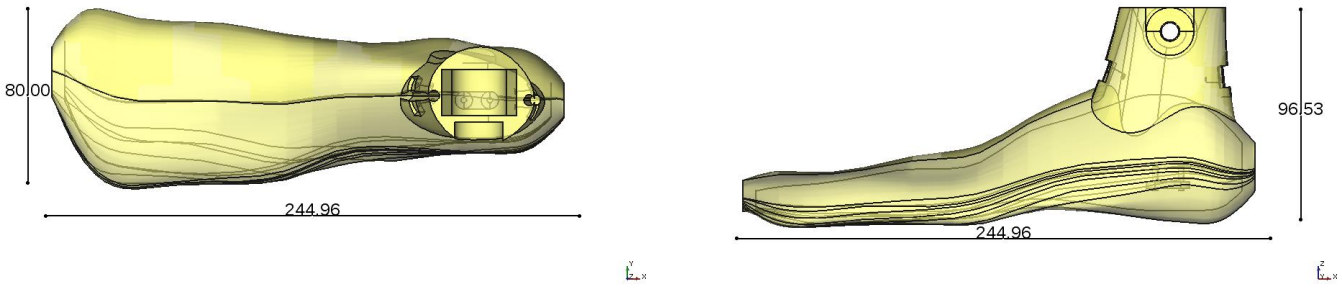


Figure 13: CAD model of foot (Dimensions in mm)

To obtain a suitable CAD model for 3D printing, there are two options. Firstly, one can use anatomical data derived from either medical imaging or mould generation. A plaster mould might be generated by pressing against the residual limb of the individual for whom the prosthesis is being provided, for example. The mould could then be scanned to generate a 3D model of the desired socket. This obviates the need for geometric modelling from scratch, but the data is difficult to acquire.

The alternative approach is to perform geometric modelling from scratch. This is certainly time-consuming, but was the most viable option in this case.

It was not attempted to produce a prosthesis which is anatomically accurate at all levels of detail. Such efforts would have been redundant, and anatomically accuracy in this case may have actually impeded the

mechanical performance. Having said that, the mechanical components were designed in such a way that their *envelope* matches that of the human foot (at least approximately). Thus, it should be feasible for an individual wearing the prosthesis to don footwear of the appropriate size, for example.

There was a limit on the size of the components which could be printed with the 3D printer available. Thus, larger models — such as the foot — had to be divided in to smaller parts for printing.

More details about the modelling techniques employed, as well as the CAD models themselves, are discussed in Appendix E

## 3.8 Fabrication

3D printers use Gcode instructions, generated by a 'slicing' algorithm which examines cross-sections of a CAD model to generate cartesian motion and tooling paths. These cross-sections — in the case of the FDM 3D printing technology used in this project at least — are in the X-Y plane and performed at successive heights in the Z-direction. The distance between successive cross-sections is a tunable 'layer height' parameter in most slicing software packages.

Slicing packages often use the Stereolithography file format for importing geometry data. The only geometric structure understood by the Stereolithography format are polygons and triangles; it has no means for storing geometry information about solids such as spheres or cylinders directly. Solid object models, instead, must be converted to a set of triangles — a process known as tesselation — where the precise number of triangles stored depends on the desired resolution. In the case of complex geometries, this can lead to very large file sizes.

It is generally a good idea to double-check the STL file for potential artifacts before slicing the model. Figure 14 shows this check being performed in Netfabb Studio. The program also shows useful information such as the dimensions and volume of the model. This is also a good opportunity to reorientate the model such that it is most suitable for 3D printing.

### 3.8.1 Printing parameters

Parameters such as layer height and infill density can be varied according to the user's needs and wishes. Reducing the layer height results in a higher resolution and smoother appearance in the printed component, but also increases the printing time. Increasing the infill density, on the other hand, can lead to increased strength in the printed parts, although this relationship is not necessarily linear. The outer boundary at

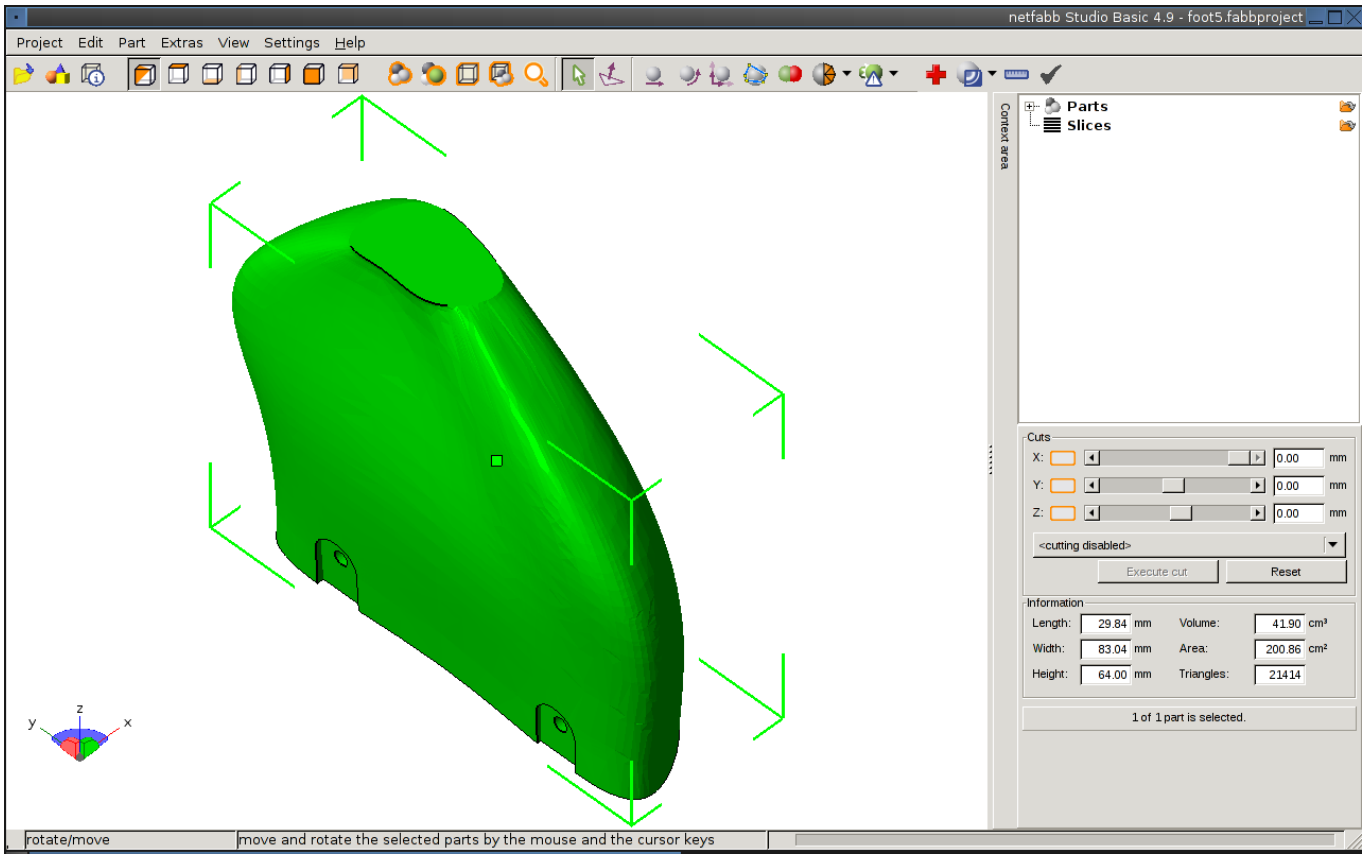


Figure 14: Checking the CAD model in Netfabb for integrity

each layer is always fused completely, at 100% density, and the thickness of this wall is a configurable parameter like any other.

An infill density of approximately 40% offers a reasonable compromise between printing time and strength. In this case, however, significant strength was desired, and an infill density of 60% to 80% was used for those components subject to significant stresses.

A layer height between 200 and 300 µm was used — 225 µm for most of the foot components.

Tall and thin components are liable to literally fall over during the print unless precautions are taken. There are a number of ways to prevent this from happening.

**Increase bed temperature** Increasing the temperature of the print bed surface leads to better adhesion between the printed part and the bed;

**Brim** adding a 'brim' around the base of the part — analogous to the stabilisers on a bicycle;

**Adhesive** Adding a few drops of PVA (Poly-Vinyl Alcohol) — a variety of wood glue — at the base of

| **Parameter** | **Typical value** | **Typical range** | **Unit** | Comment |
|---|---|---|---|---|
| Layer height | 300 | 20 – 600 | µm | |
| Infill density | 40 | 0 – 100 | % | |
| Infill pattern | rectilinear | | | A variety of different infill patterns may be available — depending on the slicing software used. |
| Brim | 5 | 0 – 20 | mm | |
| Extruder temperature (PLA) | 185 | 170 – 205 | ° | |
| Extruder temperature (ABS) | 210 | 200 – 230 | ° | |
| Heated bed temperature (PLA) | 70 | 60 – 85 | ° | |
| Heated bed temperature (ABS) | 115 | 110 – 130 | ° | |

Table 2: Typical parameters used for 'slicing' a CAD model.

the printed part is a really simple way of increasing the chances of completion for awkward-shaped pieces. PVA has very suitable characteristics for this purpose — while it bonds the printed part securely to the bed as long as heat is applied, it is readily removed from both once their temperature returns to normal. Care just needs to be paid to ensure that the printing process is not disturbed and that the piece is not inadvertently shifted while applying the glue.

Figure 15 shows a prototype.

Table 3 is a summary of some of the mechanical characteristics of the prosthesis. It should be noted that the weight value indicated is for the prosthesis without any socket attached. Attaching a socket would also increase the overall length of the prosthesis.

| **Parameter** | **Value** |
|---|---|
| Weight [g] | 600 |
| Dimensions ($L \times W \times H$) [cm] | $.4 \times .28 \times .09$ |
| Tibial rotation range [°] | -30 . . . 30 (approximately) |

Table 3: Approximate values for important mechanical characteristics

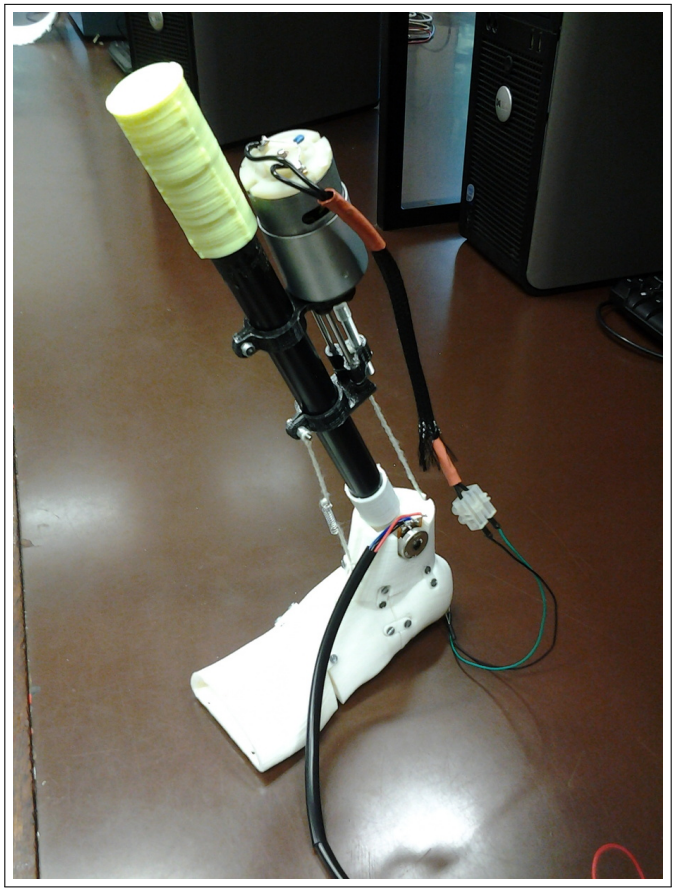

Figure 15: Prototype

# 4 Testing

## 4.1 Mechanomyography experiments

For all of the mechanomyograph experiments, a piezoelectric transducer of 12 mm in diameter was used. Initially, experiments were performed on the raw, unamplified signal with a handheld single-channel oscilloscope. This provided useful visual cues on the characteristics of the mechanomyogram — approximate amplitude range, frequency range and level of noise.

Following this, two controlled experiments were performed. In both cases, mechanomyographic signals were read from the lateral gastrocnemius in to the line-in port on a laptop soundcard. The signals were recorded at a sampling frequency of 8 kHz and stored for subsequent analysis. The testing setup was limited in the sense that the sample size was limited to a single individual. It nonetheless provided useful insights on the nature of the mechanomyogram.

### 4.1.1 Experiment 1

The analysis was performed in GNU Octave, and consisted of the following steps.

1. A mechanomyogram was recorded over a period of approximately 10 seconds and stored.
2. The signal was downsampled from 8 kHz to 500 Hz.
3. The data was passed through to a band-pass filter centered at 50 Hz to remove electrical noise at mains frequency.
4. The DC offset was removed.
5. A Short-Time Fourier Transform was performed. This involves time-dividing the signal in to periods of length appropriate for gait analysis, and performing a Fast Fourier Transform on each section multiplied by a suitable window function (e.g. a Hanning window).
6. From the Fast Fourier Transform of each section, the MPF of the corresponding section of the mechanomyogram was determined.
7. Once the MPF for each section was determined and an array of values had been gathered, these values were compared with the reported muscular effort at the corresponding instant.

### 4.1.2 Experiment 2

The goal of the second experiment was not to examine the relationship between the mean power frequency and muscular effort, but to identify potential patterns in the mechanomyogram corresponding to different degrees of flexion or muscular contraction. Thus, the analysis procedure was slightly different.

1. A mechanomyogram of duration approximately 10 seconds was recorded from the lateral gastrocnemius while the muscle was in one of three states: (a) at rest; (b) plantar flexing with moderate force; (c) plantar flexing with high force.
2. The signal was downsampled from 8 kHz to 500 Hz.
3. 50 Hz electrical interference was removed using a bandstop filter
4. The Fast Fourier transform was performed on the signal.
5. The DC offset was removed.

6. The Fourier Transform of the signals were normalised such that the sum of the values over the length of the signal is equal to one.
7. The frequency range 0 to 75 Hz was examined in attempt to identify potential similarities.

The results are presented in Section 5.

## 4.2 Controller performance

The test method for each controller configuration consisted of the following steps.

- A flexion angle between -30 and 30 degrees (positive for plantar-flexion and negative for dorsi-flexion) was arbitrarily selected as the set-point, and this value was passed was passed as a command-line argument when invoking the controller process. There was no load disturbance in these instances — i.e. no torque was applied to the foot itself.
- On each test-run, four parameters — 1. time offset, 2. set point, 3. process variable, and 4. error — were logged to a file as space-delimited values.
- These log files were saved on the embedded microprocessor itself initially, and subsequently transmitted over serial to a host computer for plotting and analysis.

## 4.3 Power consumption

The human leg is a highly efficient mechanism. It may not be possible to achieve the same level of efficiency in this prosthesis, but care should nonetheless be taken to measure the power consumption of the device and to address any issues that cause it to be excessive. Thus, voltage and current measurements were taken at various stages of testing and in different operating modes. Indicative figures are presented in Section 5.

# 5 Results

## 5.1 Power consumption

Some preliminary measurements are shown in Table 4. The current and power consumption indicated in Table 4 are those of the complete device. The 5V supply required by the i.MX233 SoC was in some cases provided by a L7085 regulator, and in other cases directly from a bench-top power supply.

| Main task(s) / Processes | Controller mode | Approximate duty cycle | Supply voltage (V) | Approx. Current consumption (A) | Approx. Power consumption (W) |
|---|---|---|---|---|---|
| Flexing | | | 12 | 0.4 | 4.8 |
| Quiescent | | | 9 | 0.25 | 2.25 |
| Plantar flexion | Proportional | 0.5 | 6 | 0.75 | 4.5 |
| Plantar flexion | Proportional | 0.8 | 5 | 1.1 | 5.5 |
| Flexion | On-off | | 9 | | 0 |
| Quiescent | | | 5 | 0.175 | 0.875 |
| Flexion | On-off | | 11 | 1 | 11 |
| Quiescent | | | 5.1 | 0.25 | 1.275 |

Table 4: Device power consumption

## 5.2 Control system

The results of step response tests with both controller configurations are presented in Figures 16 to 17. The test programs, in this instance, terminated once the set-point was reached. The high frequency variations in the graphs are believed to be artifacts of the analog-to-digital conversion rather than fluctuations in the process variable itself.

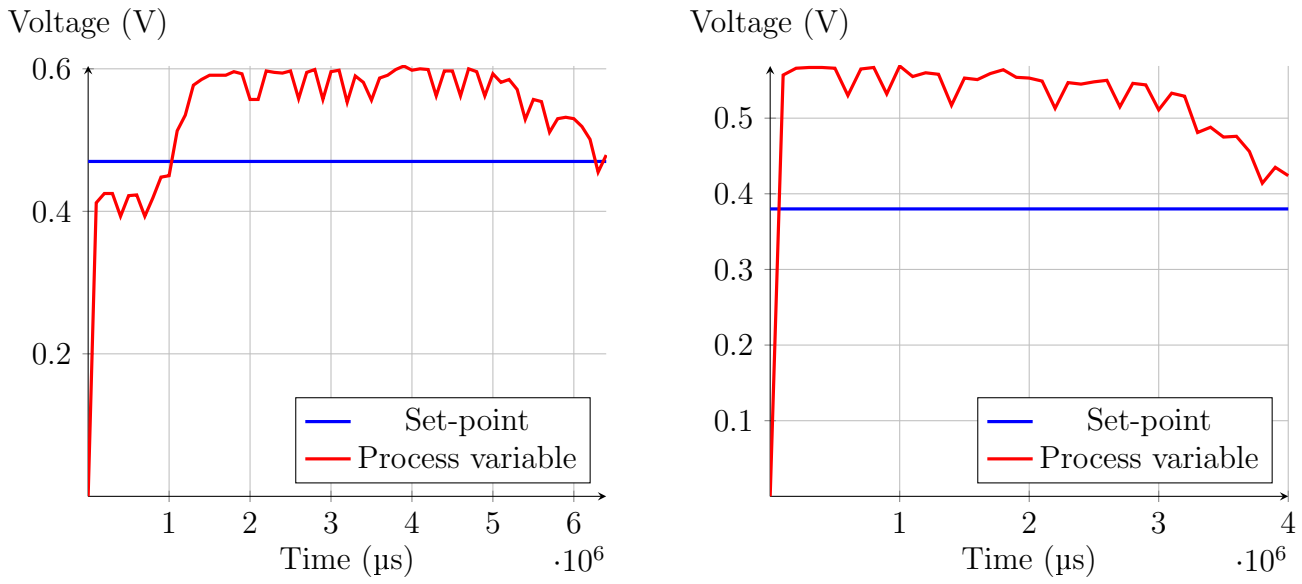


Figure 16: Step response of system using Proportional control. Proportional gain is $k = .35$.

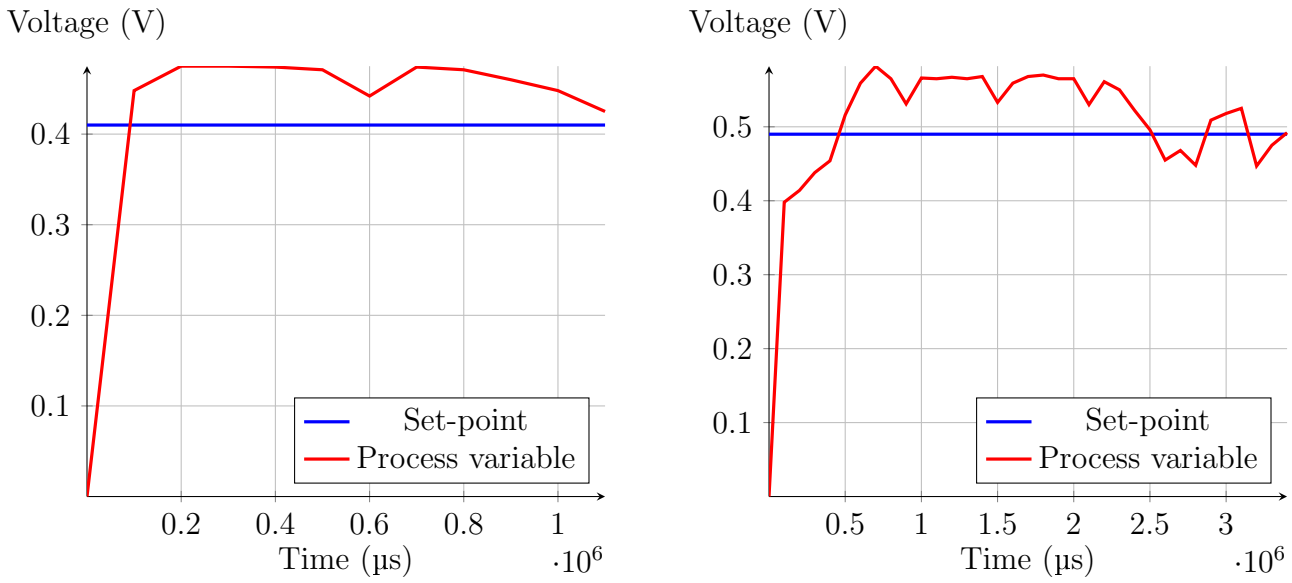


Figure 17: Step response of system using On-off control

## 5.3 Mechanomyography patterns

Figures 18 to 20 show the results of the test procedures outlined in Section 4.1. These are the lower frequency components of a calf muscle MMG while 1. at rest (Figure 18, 2. plantar flexing with moderate force (Figure 19), and 3. flexing with maximum voluntary force (Figure 20). The patterns are normalised — as the relative amplitudes are potentially significant. The effect of the 50 Hz notch filter is discernible. Although this might interfere slightly with pattern identification, the effects should be negligible.

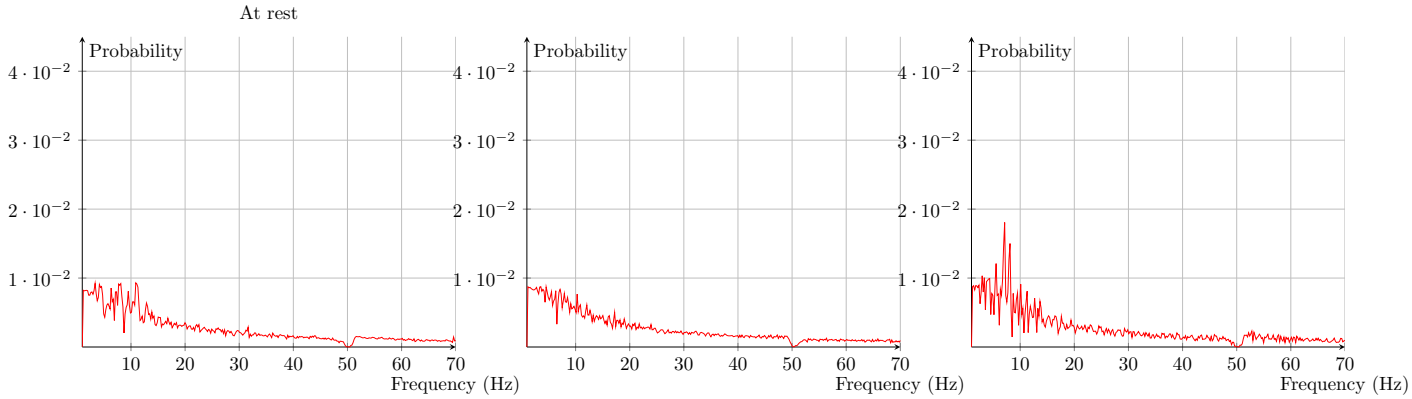


Figure 18: Calf muscle MMG — at rest

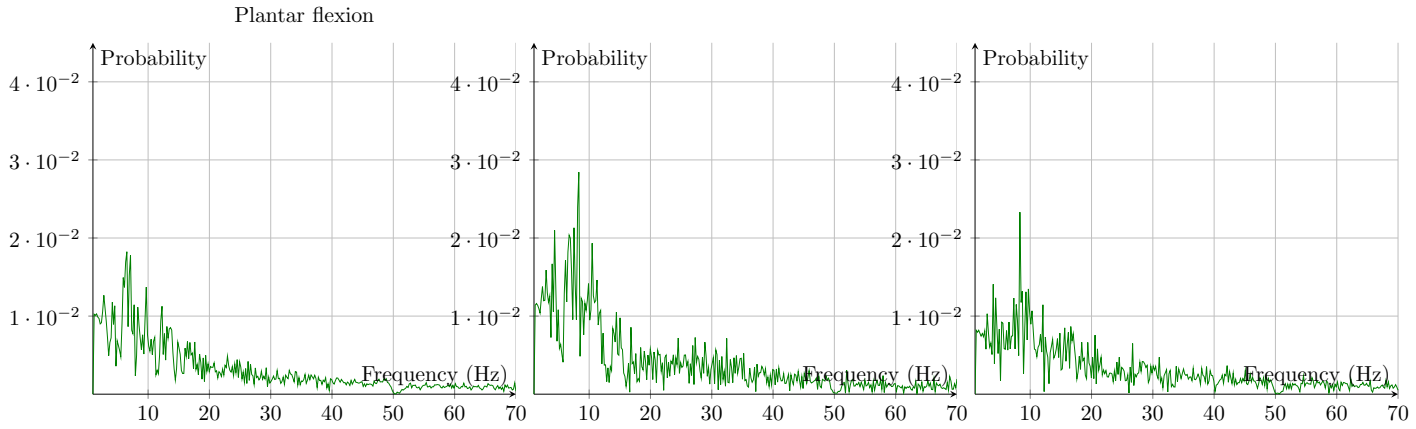


Figure 19: Calf muscle MMG — plantar flexion with moderate force

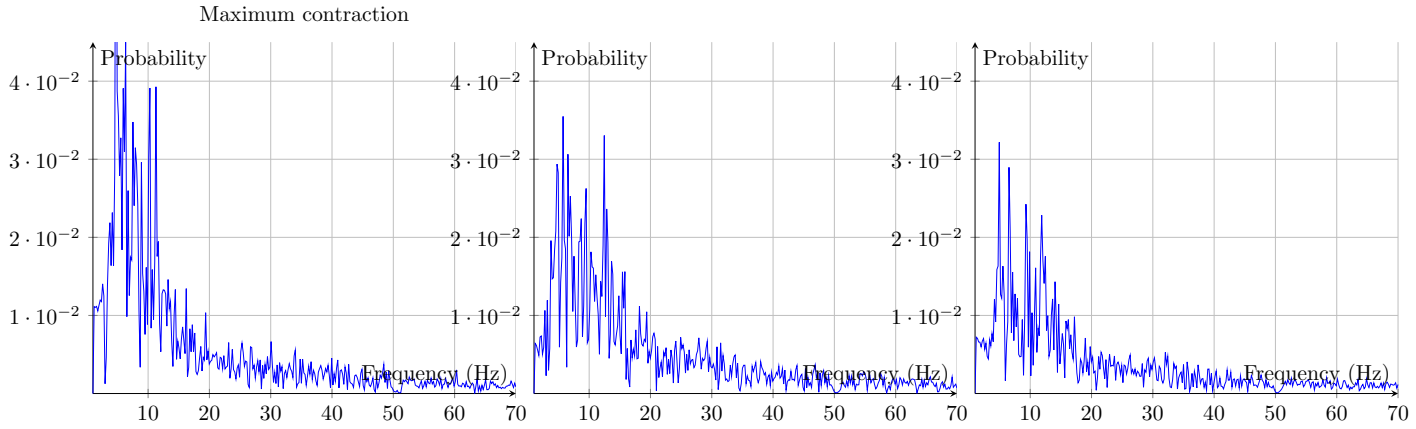


Figure 20: Calf muscle MMG — maximum voluntary contraction

# 6 Discussion

## 6.1 3D printing

One of the most challenging aspects of the project was the fabrication of necessary mechanical components, and 3D printing was used for this purpose as already explained. The opportunity to use 3D printing in this project was exciting, and it was rewarding to transforms designs from just an idea on paper to tangible objects using this technique. The task was not without frustrations however.

The main drawback of fabricating parts this way is that it is slow. For example, the prosthetic foot design used for this project was divided in to five parts of suitable size for 3D printing. Each of these parts, in turn, took approximately four hours to print. Factoring in the time required for slicing models (i.e. converting STL files in to Gcode instructions), setting up the printer, and post-processing (e.g. drilling) of

the resulting parts, the overall 3d-printing process can be very time-consuming.

This would not be such a concern if the printing process could be left unattended. As it stands, however, most 3D printer setups require close, or at least intermittent, supervision. Furthermore, most entry-level 3D printers have no vision inspection system to provide feedback on integrity of a piece while the print is ongoing. Thus, if something goes wrong during the print (e.g. a stepper motor misses a few steps), precious hours can be lost.

These cautionary remarks notwithstanding, the decision to employ 3D printing techniques for the project was certainly not regretted. The project probably wouldn't have possible without it.

In cases where personalisation of devices is not required, it may be economically more efficient to use conventional manufacturing and prototyping technologies. Using injection moulding, for example, it might be possible to produce dozens of components in minutes or even seconds. Additive manufacturing, in contrast, is slow — even with advanced and well-calibrated equipment. A compromise might be achieved in certain circumstances, wherein the mold itself is generated using additive techniques. In any case, additive manufacturing techniques and 3D-printing are certainly viable for small volume and limited-run manufacturing instances.

## 6.2 Choice of software platform

In hindsight, an alternative software platform might have been more suitable. The use of a fully-fledged operating system (Linux) is in one sense justified. It provided filesystem and network access, which was useful for testing and debugging. On the other hand, a microcontroller with signal processing capabilities — which might have been easier to program — could have been a viable candidate also.

A noteworthy platform which has only recently become available is Intel's Edison microcomputer. Featuring the Intel Quark CPU, this has a form factor similar to that of an SD card, and is targeted partly at wearable devices — which might include prostheses. As well as the ability to run Linux, it features Wi-Fi and bluetooth connectivity.

## 6.3 Controller

Experiments which showed a longer step response would be needed to fully evaluate the controller designed. Tentative remarks can be made on the results obtained nonetheless. Firstly, set-point tracking works

reasonably well in the case of on-off control. Although the proportional controller is somewhat slower to respond, this is due to the controller gain chosen.

Naturally, the gain of the proportional controller is an important determining factor of the overall system performance. With a controller gain of .35, for example, the system takes on average of 5 seconds to reach the set point (Figure 16). The length of a complete gait cycle at typical walking speed is approximately 1.5 s, so the settling time in this instance would be unacceptable.

The results presented in Section 5 suggest that on-off control might be viable in certain scenarios also. This is worth nothing in case a controller needed to be implemented on a platform without pulse-width-modulation features or capability. More comprehensive testing of the controller design is necessary however — including an analysis of its disturbance rejection characteristics.

The disturbance in this context is the load torque, marked $T_L$ in Figure 9. More specifically, it is the torque exerted on the motor shaft and leadscrew by the foot itself. This will vary constantly during the gait cycle. Thus, reliable disturbance rejection is desirable in this context, and should be an area of focus in further testing.

## 6.4 Electrical design

The electrical design could be refined in certain ways. The motor driver used in the design presented, for example, has a peak current of of approximately 1.2 A. While adequate for most testing purposes and instances of limited payload, this was dramatically undersized for the motor used (Table 1) and is unlikely to be adequate in a 'live' scenario.

In the analysis of the electromechanical characteristics, it was assumed that the mechanical power delivered to the shaft was equal to the electrical power dissipated by the armature circuit. A more thorough analysis might account for these losses.

## 6.5 Mechanomyography

Although the mechanomyography experiments provided potentially useful insights, the experimental design could be improved. Most importantly, the physiological parameters — such as the degree of plantar flexion or muscular effort — were estimated based on visual inspection and the subject's own proprioceptive feedback. This provided a starting point for comparisons, but was still partly subjective in nature. A more rigorous setup might use instrumentation to measure the actual torque exerted by the user.

Piezoelectric transducers are only one of several options for retrieving a mechanomyogram. Alternatives include accelerometers and condenser microphones, and these may yield valuable alternative insights on the nature and cause of the surface vibrations. Accelerometers, for example, can yield vector information about displacement in all three axes, while the piezoelectric device used for this project has only one channel of information.

## 6.6 Actuators

The linear actuator used in this implementation was adequate for a proof of concept. In a fully-functioning prosthesis, however, the custom solution presented would probably need to be revised. If it was decided to maintain a motor-leadscrew combination, more attention would need to be devoted to certain aspects of the solution offered here. The PVC tubing used to couple the motor shaft and leadscrew, for example, has already shown signs of wear during testing. Although it was very useful for testing purposes, it would probably not be satisfactory for permanent use, as it has a tendency to gradually 'slip' off the screw.

If further development was to be carried out, the choice of actuator might be reconsidered also. Although the use of DC motors was sensible in this context, in order to provide an initial proof-of-concept, shape memory alloys seem to offer exciting potential. Despite their added complexity, the potential for their use in lower-limb prostheses undoubtedly merits further investigation. They are lightweight, compact and their geometric form partly mimics that of natural muscle fibers (at least more so than conventional actuators).

## 6.7 Mechanical integrity

That aspect of the design over which there is the greatest question mark is probably the mechanical integrity of the components. It was not really viable to subject the fabricated parts to rigorous mechanical testing in this context. 3D-printing, although a versatile fabrication technique, can be very time-consuming, and re-printing the parts in the case of mechanical failure would have hindered development in all other areas of the project. Finite Element Analysis might be a viable alternative means of stress analysis.

# 7 Conclusion

A fledgling design for an active transtibial prosthesis has been presented. The mechanical, electrical and software designs developed specifically for the project have been examined. A Laplace-domain model of the

system was developed, and the relevance of this model to controller design was explored briefly. Lastly, the potential of mechanomyography as a means of volitional control was discussed. Although progress towards a useful design was made, the research and design could be built upon or improved in a number of ways.

First and foremost, the prosthesis is capable of only a reduced payload. A larger payload could be accommodated by replacing the motor driver circuit and one or two other minor modifications of the electrical configuration. Secondly, only preliminary research and experiments were performed on mechanomyography. If further research was to be performed, more carefully-controlled experiments with a larger sample size might be desirable. Lastly, careful refactoring of the software written would need to be performed if it was to be used in a functioning design.

In closing, suggestions for improvements and potential directions for further research — some practical, some abstract — are presented.

## 7.1 Possible improvements

**Degrees of freedom** One of the most obvious enhancements to the device would be additional joints, or degrees of freedom, It must be borne in mind, when considering this, that — for every degree of freedom added in a system — a new state variable is introduced. This, in turn, begs the question of how precisely this state variable is to be controlled. This question is all the more relevant in the case of a residual limb, where there is a limited amount of muscular tissue from which to read myoelectric signals. Nonetheless, additional degrees of freedom — if justified and feasible — could open the door to exciting control engineering, such as the use of state-space models, etc..

**3D printing materials** Two composite materials which have recently become available to the 3D printing community might be harnessed to advance this project. Firstly, the use of carbon-fiber reinforced PLA might permit creative alternatives to standardised carbon-fiber products.[2] Rather than carbon-fibre tubing of pre-defined diameters, for example, it might be possible to design a prosthetic 'bone' which more closely resembles a real tibia. Secondly, for those components which require more flexibility, Filaflex could be used [3]. The latter material was considered for use in this project, but requires the extruder to be modified in certain instances.

**Mechanomyography research** Pattern-based MMG interpretation, or the identification of 'signatures' in the MMG signal, seems to be the most viable method for using MMG in a myoelectric control

[2] http://www.proto-pasta.com/datasheets/Carbon-Fiber-PLA.pdf
[3] http://recreus.com/

scenario (more viable than, for example, a method based on the relationship between motor unit recruitment and the mean power frequency of the signal).

**Shape memory alloys** A rationale for using DC motors rather than shape memory alloys was offered in Section 3.2. Nonetheless, research in to shape memory alloys as actuator solutions for lower-limb prostheses certainly seems merited. Although their thermal and mechanical characteristics might make them more challenging to integrate in to a prosthesis design in some respects, they bear a closer resemblance to natural muscles than conventional solutions such as motors or pneumatic components. This 'biomimetic' quality suggests that they might find increased use in prosthetic applications in the future.

**Gait analysis** The prosthesis design presented here wasn't developed to a stage where a proper gait analysis was possible. Nonetheless, such an analysis would certainly be a very important step in a final evaluation of a functional prosthesis. Ultimately, it might not be viable to appraise this aspect of the design until someone actually wears the prosthesis and attempts to walk with it. On the other hand, sophisticated techniques for biomechanical simulation are available also, which may obviate the need for *in vivo* testing. Thus, there are potentially a number of methods for gait analysis.

1. **Dynamic Markers** This generally requires sophisticated gait laboratary facilities which is expensive, and relatively cumbersome.
2. **Image Processing** More recently, a method for evaluating gait performance using efficient image processing techniques have been explored [3].
3. **Simulation** Using modern software such as OpenSim [5], it may be feasible to evaluate certain aspects of prosthesis-assisted gait through simulation. Such options could be very useful for evaluating prosthetic and orthopaedic devices in the future.

**Position sensing** A potentiometer provided an elegant and efficient means of measuring the degree of flexion in the ankle joint in this case. By selecting a potentiometer of the right resistance and rotation range, it might be possible to use this with any hinge joint. However, in the case of more complex joints — such as ball joints or saddle joints — or in cases where the travel to be measured is not purely rotary, a potentiometer will probably not be suitable. For this reason, some tentative research was made in to an alternative flexion sensor. This is discussed more fully in Appendix F.

# Glossary

**ABS** Acrylonitrile Butadiene Styrene — a lightweight thermoplastic with a wide variety of engineering applications. Like PLA, it is commonly used in Fused Deposition Modelling 3d-printing.. 27

**Analog-to-digital converter (ADC)** As the name implies, ADCs transform an analog voltage in to a digital representation. The details of the conversion process may vary, but it always involves at least two steps — sampling and quantisation.. 16

**Fast Fourier Transform** An efficient means of computing the Discrete Fourier Transform. 30

**fused deposition modelling** An additive manufacturing technique. Fused-deposition modelling (FDM) involves the layer-by-layer deposition of molten thermoplastic to form a solid object. In the open-source 3D-printing community, the technique is sometimes referred to as Fused-Filament Fabrication (FFF), in order to avoid potential patent issues.. 6, 13, 27, 41

**MMG** Mechanomyography is the measurement of skin surface displacement due to vibrations in underlying muscle. 14

**MPF** Mean Power Frequency. 8, 30

**PLA** Poly-lactic acid — a biodegradable thermoplastic commonly used in Fused Deposition Modelling 3d-printing.. 27

**Pulse-width modulation (PWM)** Pulse-width modulation is a technique used to provide a varying voltage by varying the duty cycle of a digital output. It can be used a substitute for true digital-to-analog conversion in many cases. The two important parameters in configuring pulse-width modulation are the period and the duty cycle. The modulation can be performed in software or in hardware. 16, 17, 23, 24

**shape-memory alloy (SMA)** A shape-memory alloy is an alloy which can 'remember' its original shape when deformed, and which returns to this shape when heated. This property can be exploited to provide a torque, and shape memory alloys are used as actuators in a variety of engineering applications for this reason.. 12, 37

**STFT** Short-Time Fourier Transform. 15

**System-on-a-Chip (SoC)** A System-on-a-Chip consists of a microprocessor with associated memory and hardware. As the name implies, these are intended to be used as self-contained systems, if needed.. 16

**Universal Asynchronous Receiver-Transmitter (UART)** UARTs provide the parallel-to-serial conversion necessary for serial communications.. 16

# References


[1] Kumi Akataki et al. "Age-related change in motor unit activation strategy in force production: A mechanomyographic investigation". In: *Muscle & Nerve* 25.4 (2002), pp. 505–512.

[2] Travis W. Beck et al. "Does the frequency content of the surface mechanomyographic signal reflect motor unit firing rates? A brief review". In: *Journal of Electromyography and Kinesiology* 17.1 (2007), pp. 1 –13. ISSN: 1050-6411. URL: http://www.sciencedirect.com/science/article/pii/S1050641106000058.

[3] Jane Courtney and Annraoi M De Paor. "A monocular marker-free gait measurement system". In: *Neural Systems and Rehabilitation Engineering, IEEE Transactions on* 18.4 (2010), pp. 453–460.

[4] Kathryn J De Laurentis et al. "Optimal design of shape memory alloy wire bundle actuators". In: *Robotics and Automation, 2002. Proceedings. ICRA'02. IEEE International Conference on*. Vol. 3. IEEE. 2002, pp. 2363–2368.

[5] Scott L Delp et al. "OpenSim: open-source software to create and analyze dynamic simulations of movement". In: *Biomedical Engineering, IEEE Transactions on* 54.11 (2007), pp. 1940–1950.

[6] J Ebert et al. "Direct inkjet printing of dental prostheses made of zirconia". In: *Journal of dental research* 88.7 (2009), pp. 673–676.

[7] Joel Gibbard. *The Open Hand Project*. Accessed on May 26 2014. 2013. URL: http://www.openhandproject.org/.

[8] Joseph Hitt et al. "Robotic transtibial prosthesis with biomechanical energy regeneration". In: *Industrial Robot: An International Journal* 36.5 (2009), pp. 441–447.

[9] Ellen Kreighbaum and Katharine M. Barthels. *Biomechanics: a qualitative approach for studying human movement*. 4th ed. Boston: Allyn and Bacon, 1996. ISBN: 0205186513.

[10] Kevin T OToole and Mark M McGrath. "Mechanical design and theoretical analysis of a four fingered prosthetic hand incorporating embedded SMA bundle actuators". In: *Proceedings of World Academy of Science Engineering And Technology*. Vol. 25. 2007.

[11] P. Parker, K. Englehart, and B. Hudgins. "Myoelectric signal processing for control of powered limb prostheses". In: *Journal of Electromyography and Kinesiology* 16.6 (2006), pp. 541 –548. ISSN: 1050-6411. DOI: http://dx.doi.org/10.1016/j.jelekin.2006.08.006. URL: http://www.sciencedirect.com/science/article/pii/S1050641106001027.

[12] Charles Pfeiffer, Kathryn DeLaurentis, and Constantinos Mavroidis. "Shape memory alloy actuated robot prostheses: initial experiments". In: *Robotics and Automation, 1999. Proceedings. 1999 IEEE International Conference on*. Vol. 3. IEEE. 1999, pp. 2385–2391.

[13] Freescale semiconductor. *i.MX23 Applications Processor Reference Manual*. Accessed on May 24 2014. Nov. 2009. URL: http://cache.freescale.com/files/dsp/doc/ref_manual/IMX23RM.pdf.

[14] Eliza Strickland. *A Robot Ankle for Amputees*. Accessed on May 24 2014. Aug. 2012. URL: http://spectrum.ieee.org/video/biomedical/bionics/a-robot-ankle-for-amputees.

[15] E-NABLE team. *E-NABLING THE FUTURE: A network of passionate volunteers using 3D printing to give the World a "Helping Hand."*. Accessed on May 26 2014. 2014. URL: http://enablingthefuture.org/.

# A Software

The software for this project was written in the C programming language, and compiled using the GNU C Compiler. Listings 1 and 2 show the necessary configuration of the analog-to-digital hardware registers, and a software-based proportional controller. A number of other software tests were written over the course of the project also.

The source code for all of these — as well as CAD models and other relevant documents — are accessible in the design repository which has been maintained for this project: http://indigo.uk.to/repo/prosthesis/.

Listing 1: Set up of the ADC hardware

```
#include "imx233.h"
#include "adc.h"

void adc_setup(void)
{
  /*
    ANALOG CHANNEL MAPPING
    Virtual channels are mapped to actual hardware inputs (physical channels).
    We need to configure this mapping before anything else.
  */
  imx233_wr(HW_LRADC_CTRL4_CLR, 0xff); // Clear all mappings presently stored
  imx233_wr(HW_LRADC_CTRL4_SET, 0x01); // Virtual channel 0 <-> phys. chan. 1
  imx233_wr(HW_LRADC_CTRL1_SET, 0x00010000); // Enable interrupts for channel 0
  imx233_wr(HW_LRADC_CTRL1_CLR, 0x00000001); // Clear interrupt bit for channel 0
  if (DIVIDE_BY_TWO)
    imx233_wr(HW_LRADC_CTRL2_SET, 0x01000000); // Set divide-by-two bit for CH0
  else
    imx233_wr(HW_LRADC_CTRL2_CLR, 0x01000000); // Clear divide-by-two bit for CH0

  // Do we need to set the accumulate bit?
  imx233_wr(HW_LRADC_CH0_SET, 0x20000000);
  /*
    We only want one conversion per interrupt, so we need to set the NUM_SAMPLES
    field (bits 28:24) in HW_LRADC_CH0 register to zero.
  */
  imx233_wr(HW_LRADC_CH0_CLR, 0x1f000000);
}

double adc_read(void)
{
  int adc_value;
  double voltage;
  // Clear the accumulator
  imx233_wr(HW_LRADC_CH0_CLR, 0x0003ffff);
  /*
    CONVERSION
```

```
    Schedule a conversion on channel 0
  */
  imx233_wr(HW_LRADC_CTRL0_SET, 0x01);
  /*
    WAIT FOR INTERRUPT (COMPLETION OF CONVERSION)
    As soon as the ADC conversion is complete, the schedule bit
    for this channel will be unset in HW_LRADC_CTRL0. In the case of
    channel 0 this is bit 0.
    So we wait for this.
  */
  while (imx233_rd(HW_LRADC_CTRL0) & 0x00000001);
  /* There might be a slight delay between the schedule conversion bit
    being reset and the result actually being placed in the result register.
  */
  usleep(CONVERSION_DELAY);
  /*
    RESULT
    Read result
    We are taking the lower 12-bits of the result register, and
    placing them in a variable of type 'int'. */
  adc_value = imx233_rd(HW_LRADC_CH0) & 0x3fff;
  voltage = ((double) adc_value/(double) ADCMAX) * VMAX;
  return voltage;
}
```

Listing 2: Proportional control test

```
/*
  Experiment with feedback and Proportional control

  Software-based PWM is used to control the motor speed.
*/

#include "io.h"
#include "adc.h"
#include "imx233.h"

int dir, timed_out;
int pwm_period;
double duty, duty_min;
double e; // error
double pv; // process variable
double sp; // set-point
double u; // controller output
double k;
double tol; // tolerance

void *timer(void *args)
{
  while (1)
  {
//    printf("%.3f\n", pv);
```

```
  }
  pthread_exit(NULL);
}

void *pwm(void *args)
{
  while (!timed_out)
  {
    GPIO_WRITE_PIN(PLANTARFLEXEN,1);
    usleep((double)(pwm_period * duty));
    GPIO_WRITE_PIN(PLANTARFLEXEN,0);
    usleep((double)(pwm_period * (1-duty)));
  }
  pthread_exit(NULL);
}

int main(int argc, char **argv) {
  pthread_t thread_pwm, thread_timer;

  sp = atof(argv[1]);

  timed_out = 0;
  pwm_period = 20000;
  k = .35;
  e = .3;
  duty_min = .29;
  tol = .005;

  gpio_map();
  set_pin_as_output(PLANTARFLEX0);
  set_pin_as_output(PLANTARFLEX1);
  set_pin_as_output(PLANTARFLEXEN);

  if (argc < 2) {
    fprintf(stderr, "Usage: %s <set_point>\n", argv[0]);
    exit(EXIT_FAILURE);
  }

  GPIO_WRITE_PIN(PLANTARFLEX0,1);
  GPIO_WRITE_PIN(PLANTARFLEX1,0);

  pthread_create(&thread_pwm, NULL, &pwm, NULL);
//  pthread_create(&thread_timer, NULL, &timer, NULL);

  // Enable motor
  GPIO_WRITE_PIN(PLANTARFLEXEN,1);

  adc_setup();

  duty = duty_min;
  // The controller output, u, determines the motor's speed.
  while (e>tol || e<-tol) {
```

```
    pv = adc_read();
    e = sp - pv;
    u = k*e;
    /*
      If the controller output is negative, we need to change the
      motor direction
    */
    if (u < 0) {
      duty = -(u+duty_min);
      GPIO_WRITE_PIN(PLANTARFLEX0,1);
      GPIO_WRITE_PIN(PLANTARFLEX1,0);
    }
    else {
      duty = u+duty_min;
      GPIO_WRITE_PIN(PLANTARFLEX0,0);
      GPIO_WRITE_PIN(PLANTARFLEX1,1);
    }
  }
  timed_out = 1;

  // Disable motor!
  GPIO_WRITE_PIN(PLANTARFLEXEN,0);

  pthread_join(thread_pwm,NULL);
//  pthread_join(thread_timer,NULL);
  pthread_exit(NULL);

  exit(EXIT_SUCCESS);
}
```

# B Circuits

Figure 21 presents the main circuit diagram for the prosthesis.

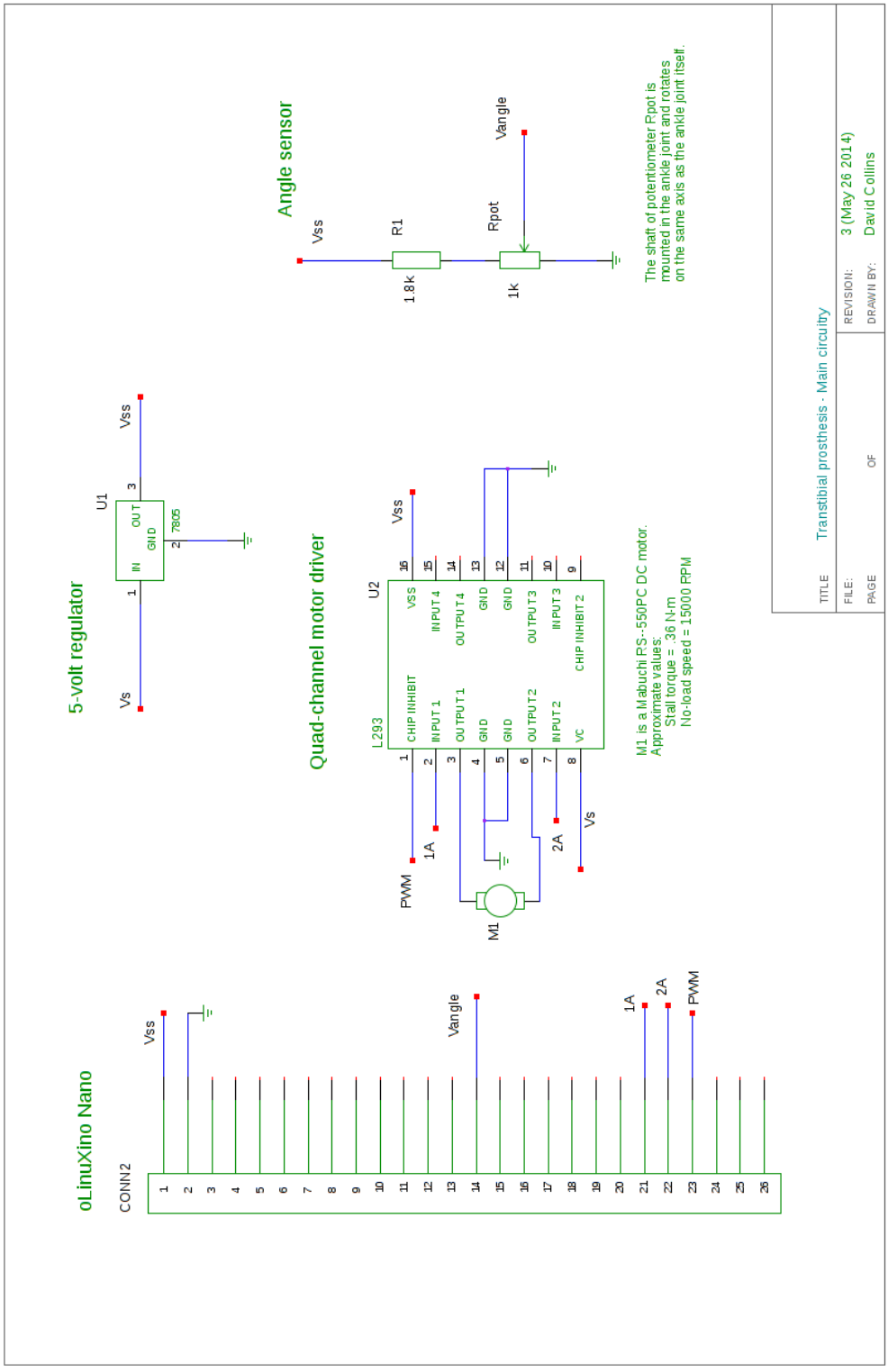


Figure 21: Main circuit diagram

# C Mechanomyography Analysis

The MMG analysis was performed in GNU Octave. The code used is presented in Listing 3. The signal was sampled at a frequency of 8 kHz, due to the nature of the PC-based sampling setup. All of the relevant frequency information of the mechanomyogram, on the other hand, is concentrated between approximately 5 and 75 Hz. Thus, downsampling was performed as a preliminary step.

Listing 3 is the analysis used for the first mechanomyography experiment. For the second experiment, a slightly different analysis approach was adopted, based on pattern identification rather than a one-to-one relationship between the mean power frequency and any parameter of physiological state. The code used for analysis in this case is presented in Listing 4.

Listing 3: MATLAB / Octave analysis for mechanomyography experiment 1

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
%                       MECHANOMYOGRAPHY ANALYSIS
%
%                                David Collins
%                                  May 2014
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% Clear workspace
clear;
close all;
clf();

pkg load signal;

% Frequencies - original and new sampling frequency, and Nyquist frequency
fsorig = 8e+3; % sampling frequency
fs = 500;
fNy = fs/2;

% Load MMG data
fName = "plantar-flexion-isometric-30-degrees-2.wav";
mmg = wavread(fName);

% Resample
mmg = resample(mmg, fs, fsorig);

% Bandstop filter (remove 50 Hz noise)
fl = 49;
fh = 51;
% Convert frequencies from Hz to radians
%wc = [fl/fNy fh/fNy];
wc = [fl/fNy fh/fNy];
```

```
[b,a] = butter(3, wc, 'stop');

[H,w] = freqz(b,a);
freqrange = linspace(1,fNy,length(H));
plot(freqrange, 20*log10(abs(H)));
pause();

mmgClean = filter(b, a, mmg);
%subplot(2,1,1);
%plot(mmg);
%subplot(2,1,2);
%plot(mmgClean);
%pause();

tstep = .1; % 10 milliseconds
inc = fix(fs*tstep);
num_coeff = inc*4;
win_size = num_coeff*5/4;
mmgSTFT = stft(mmgClean,  win_size, inc, num_coeff);
nSections = size(mmgSTFT)(2)

sectionMPFs = zeros(1,nSections);
sectionPOWER = zeros(1,nSections);

% The DC offset is ignored by zeroing the first component in each Fourier transform.
% This is needed as otherwise an incorrect value could be returned for the MPF.
for i = 1:nSections
  mmgSTFT(1,i) = 0;
  extract = mmgSTFT(:,i);
  % Estimate the power _before_ normalising
  sectionPOWER(i) = sum(extract);
  % Normalise
  extract = extract/sum(extract);
  MPF = sum((1:num_coeff).*extract')/sum(extract);
  sectionMPFs(i) = MPF;
end

clf();
subplot(2,1,1);
plot(sectionMPFs);
subplot(2,1,2);
plot(sectionPOWER);
```

Listing 4: MATLAB / Octave analysis for mechanomyography experiment 2

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
%                        MECHANOMYOGRAPHY ANALYSIS
%
%                                David Collins
%                                  May 2014
%
```

```
%
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
%
% This script extracts frequencies in the range of 0 - 75 Hz
% from an MMG signal with the goal of identifying patterns
% corresponding to voluntary muscle activity or intended
% movement.
%
% The MMG data (assumed to have been sampled at 8 kHz) is
% downsampled before extracting the frequencies of interest.
%
% The Fourier Transform of each signal is normalised in order
% to facilitate comparison.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

clear;
close all;

pkg load signal;

% Sampling frequencies
fs = 500; % desired
fsorig = 8e+3; % original
% Nyquist frequency
fNy = fs/2;

% Bandstop filter (remove 50 Hz noise)
fl = 49;
fh = 51;
% Convert frequencies from Hz to radians
%wc = [fl/fNy fh/fNy];
wc = [fl/fNy fh/fNy];
[b,a] = butter(3, wc, 'stop');

% Read in MMG data
rest(:,1) = wavread('rest1.wav');
flex(:,1) = wavread('plantar-flexion-1.wav');
flexMax(:,1) = wavread('max-contraction-1.wav');
rest(:,2) = wavread('rest2.wav');
flex(:,2) = wavread('plantar-flexion-2.wav');
flexMax(:,2) = wavread('max-contraction-2.wav');
rest(:,3) = wavread('rest3.wav');
flex(:,3) = wavread('plantar-flexion-3.wav');
flexMax(:,3) = wavread('max-contraction-3.wav');

% Get the duration of the signal (in seconds)
dur = length(flex(:,1)) / fsorig;

% Downsample from original frequency to 500 Hz
% TODO: Should an anti-aliasing filter be used here?
```

```
for i=1:3
  restTmp(:,i) = resample(rest(:,i), fs, fsorig);
  flexTmp(:,i) = resample(flex(:,i), fs, fsorig);
  flexMaxTmp(:,i) = resample(flexMax(:,i), fs, fsorig);
end
rest = restTmp;
flex = flexTmp;
flexMax = flexMaxTmp;

% Get the Fourier transforms and remove the DC offset
for i=1:3
  rest(:,i) = filter(b,a,rest(:,i));
  flex(:,i) = filter(b,a,flex(:,i));
  flexMax(:,i) = filter(b,a,flexMax(:,i));

  flexMaxFT(:,i) = fft(flexMax(:,i));
  flexFT(:,i) = fft(flex(:,i));
  restFT(:,i) = fft(rest(:,i));

  flexMaxFT(1,i) = 0;
  flexFT(1,i) = 0;
  restFT(1,i) = 0;

  % Normalise the Fourier trnasforms
  % Normalisation in this context means rescaling
  % the signal such that the total sum or area is
  % equal to 1.
  flexMaxFT(:,i) = flexMaxFT(:,i)/sum(flexMaxFT(:,i));
  flexFT(:,i) = flexFT(:,i)/sum(flexFT(:,i));
  restFT(:,i) = restFT(:,i)/sum(restFT(:,i));
end

% Plot the results
fCutoff = 70;
binCutoff = fix(fCutoff*dur);
binRange = 1:binCutoff;
freqRange = linspace(1, fCutoff, binCutoff);

for i = 1:3
  subplot(3,3,i);
  plot(freqRange, abs(restFT(binRange,i)));
  axis([0 70 0 0.05]);
  ylabel('Amplitude');
  xlabel('Frequency (Hz)');
  title('Frequency content of MMG for gastrocnemius at rest');
  subplot(3,3,i+3);
  plot(freqRange, abs(flexFT(binRange,i)));
  axis([0 70 0 0.05]);
  ylabel('Amplitude');
  xlabel('Frequency (Hz)');
  title('Frequency content of MMG for gastrocnemius during plantar flexion');
  subplot(3,3,i+6);
```

```
  plot(freqRange, abs(flexMaxFT(binRange,i)));
  axis([0 70 0 0.05]);
  ylabel('Amplitude');
  xlabel('Frequency (Hz)');
  title('Frequency content of MMG for gastrocnemius during maximum contraction');
end

A = zeros(binCutoff,10);
A(:,1) = freqRange';
for i=2:4
  A(:,i) = abs(restFT(binRange,i-1));
end
for i=5:7
  A(:,i) = abs(flexFT(binRange,i-4));
end
for i=8:10
  A(:,i) = abs(flexMaxFT(binRange,i-7));
end

dlmwrite('results.dat', A, '\t');
```

# D Scilab simulation

Listing 5 shows the Scilab code used to simulate our closed-loop control system.

Listing 5: Simulation in Scilab

```
// Motor parameters - Armature
Ra = 12; // armature resistance (Ohms)
La = 1.8e-3; // armature inductance (Henries)
// Motor parameters - Shaft
k = 3.9e-3; // Back EMF constant [V/rad/s] = (in magnitude) Motor torque constant [Newton metres per
    Ampere]
B = 3.5e-6; // motor damping coefficient [ Newton metres per radian per second]
J = 1e-6; // moment of inertia [ Newton metres per radian per second squared]
r = .05; // radius from to plantar flexor to axis of rotation of tibial joint [ metres ]
p = .01; // leadscrew pitch [ metres ]

s = poly(0,'s');

pulleyGain = p/(2*%pi*r);
motorGain = k/(s^3*J*La + s^2*(Ra*J + B*La) + s*(Ra*B + k^2));
plant = pulleyGain*motorGain;

kp = .35; // Controller gain

forwardLoop = kp*plant;
sysClosed = forwardLoop/(1+forwardLoop);
```

```
t = 0:0.05:11;

out = csim('step', t, sysClosed);

plot2d(t, out);

csvout = [t' out'];
```

# E CAD models

With the exception of the actuator design — for which OpenSCAD[4] was used — the open-source FreeCAD software[5] was used for used for the majority of the CAD modelling. This uses the OpenCascade geometry kernel internally[6]. OpenCascade has strong capabilities for 'freeform' modelling (e.g. Bezier curves, splines, etc.), which were really valuable in this context. Constructive solid geometry (CSG) modelling techniques alone are of limited use in biomedical applications — the resulting parts tend to appear overlay 'square'.

Figure 22 shows the result of a 'loft' operation performed on a series of Bezier splines.

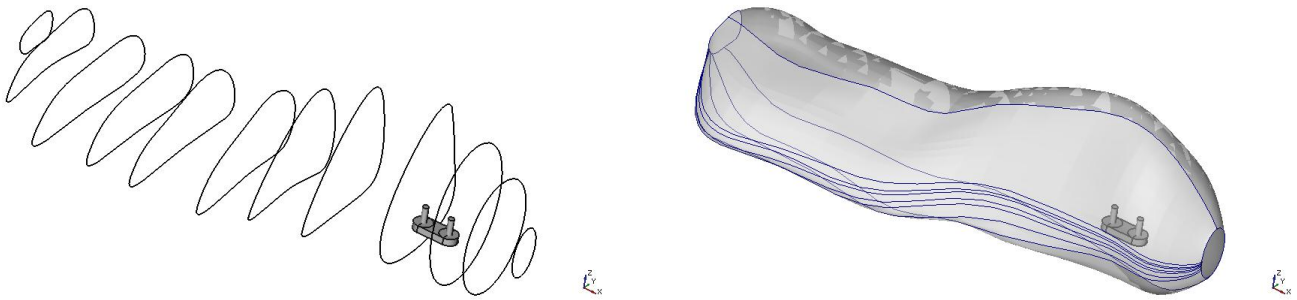

Figure 22: A loft operation performed on a series of Bezier spline curves

Models in FreeCAD can be scripted in Python, created using the graphical user interface, or using a combination of both methods. Another noteworthy feature of FreeCAD is that it is a parametric modeller. Thus, if a composite object is created by performing a Boolean operation on other objects, this composite object will be updated automatically when any parameters of the underlying objects are modified.

[4] See http://www.openscad.org/ for more information (accessed May 24 2014).
[5] See http://www.freecadweb.org/ (accessed May 24 2014).
[6] See http://www.opencascade.org/ (accessed May 24 2014).

## E.1 Linear actuator

An example of the OpenSCAD code used to generate the actuator components is presented in Listing 6. A CAD view of one the actuator components itself is presented in Figure 23.

Listing 6: OpenSCAD code for linear actuator

```
screw_distance = 25;
screw_hole_diam = 3.2;
motor_mount_rad = 17;
inner_radius = 20/2;
wall_thickness = 2;
outer_radius = inner_radius + wall_thickness;
slider_height = 11;
clearance = .7;
motor_clearance = 1.5;
overlap = 2;
base_offset=[motor_mount_rad+motor_clearance+outer_radius,0,0];
rod_x_offset = 9.75;
inter_rod_distance = 14.25;

// resolution
/*
Circles are actually stored as polylines (approximations of circles)
rather than true circles.
The $fn parameter dicates the closeness of the approximation.
Higher values correspond to smoother parts - but larger file sizes (and
possibly slower prints.)
*/
$fn = 32;

module motor_mount_base(ht,rad,irad,rrad,hrad) {
  poly_points = [
    [rad*.53,-rad*.8],
    [rad,-irad],
    [rad+motor_clearance+irad,-irad-wall_thickness],
    [rad+motor_clearance+irad,irad+wall_thickness],
    [rad,irad],
    [rad*.53,rad*.8]
  ];
  union() {
    difference() {
      intersection() {
//        translate([0,-rad,0]) cube([rad,rad*2,ht],center=false)
        translate([0,0,-1]) cylinder(r=rad,h=ht+2,center=false);
        translate([0,-rad,0]) cube(size=[rad,rad*2,ht],center=false);
      }
      translate([0,0,-1]) cylinder(r=irad,h=ht+2);
    }
    union() {
```

```
      translate([0,irad+rrad,0]) cylinder(r=rrad,h=ht);
      translate([0,-(irad+rrad),0]) cylinder(r=rrad,h=ht);
    }
/*    translate([rad,-rad,0])
      cube(size=[motor_clearance+wall_thickness,rad*2,ht],center=false);*/
    linear_extrude(height=ht)
      polygon(points=poly_points);
  }
}

module motor_mount() {
  ht = 4; // height
  rad = motor_mount_rad; // radius (outer)
  irad = 7.75; // inner radius
  rrad = (rad-irad)/2;
  screw_offset = screw_distance/2;
  hrad = screw_hole_diam / 2;

  difference() {
    motor_mount_base(ht,rad,irad,rrad,hrad);
    translate([0,screw_offset,-1]) cylinder(r=hrad,h=ht+2);
    translate([0,-screw_offset,-1]) cylinder(r=hrad,h=ht+2);
    translate(base_offset+[0,0,-1]) cylinder(r=inner_radius+clearance,h=slider_height+2);
//    cube([20,20,5]);
  }
}

module slide_rail_slot_left() {
  slide_rail_slot([rod_x_offset,-inter_rod_distance/2,2.5]);
}

module slide_rail_slot_right() {
  slide_rail_slot([rod_x_offset,inter_rod_distance/2,2.5]);
}

module slide_rail_slot(offset) {
  rod_hole_diam = 3.6;
  hrad = rod_hole_diam /2;
  brad = hrad+2.5;
  ht = 6;

  translate(offset) difference() {
    cylinder(r=brad, h=ht);
    translate([0,0,-1]) cylinder(r=hrad, h=ht+2);
  }
}

module base_mount_screw_clips(ir,or,co,ht,th,offset) {
  rad = ht/2;
  hrad = screw_hole_diam / 2;
  overlap = .2;
  clip_base_height = 6;
```

```
  boffset = [clip_base_height,0,0];
  hoffset = [clip_base_height+.5,0,-1];

  translate(offset) rotate(v=[1,0,0],a=90)
  difference() {
    union() {
      translate(boffset) intersection() {
          cylinder(r=ht/2,h=th);
          translate([0,-rad,0]) cube(size=[rad,rad*2,ht],center=false);
        }
      translate([0,-ht/2,0]) cube(size=[clip_base_height,ht,th],center=false);
    }
      translate(hoffset) cylinder(r=hrad,ht+2);
    }
}

module base_mount() {
  cl=clearance; // clearance
  ir=inner_radius+cl;
  th=2;
  or=ir+th;
  ht=11;
  motor_mount_outlay=3;
  overlap=th;
  co=3;
  offset=base_offset;

  translate(offset) union() {
    difference() {
      cylinder(r=or,h=ht,center=false);
      translate([0,0,-1]) cylinder(r=ir,h=ht+2,center=false);
      translate([ir-.1,-co/2,-1]) cube(size=[th*2,co,ht+2]);
    }
    base_mount_screw_clips(ir,or,co,ht,th,[or-th,-co/2,ht/2]);
    base_mount_screw_clips(ir,or,co,ht,th,[or-th,co/2+th,ht/2]);
  }
}
```

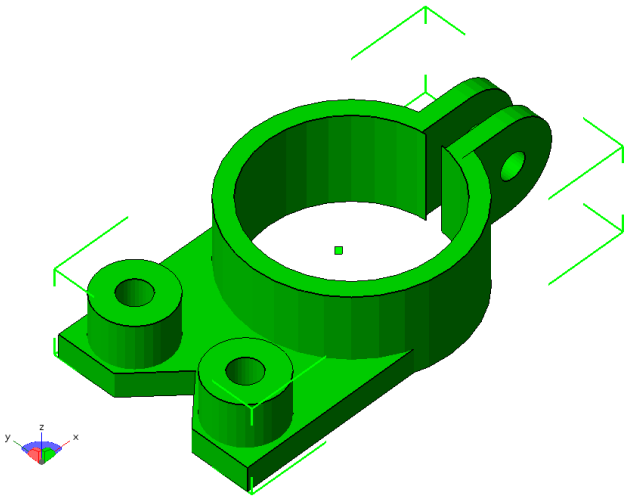

Figure 23: Linear actuator (lower part) designed in OpenSCAD

# F Strain-gauge-based flexion sensor

An idea for a flexion sensor based on a strain gauge which might be useful in this application is presented. In summary, the idea is that a strain gauge mounted on a suitable elastic element (a strip of a suitable elastomer) might prove more versatile than conventional methods used in this context (e.g. displacement sensors or — in the case of rotrary displacement — potentiometers). The elastic element could be run parallel to the linear actuator or tendon in a prosthetic or robotic application. A numerical example in which an appropriate value for Young's modulus is calculated follows.

Let $l_0$ be the length of the elastic element and $\Delta l$ be the maximum change in length experienced in normal operating conditions. The strain, for values that might be used in an application such as this, can be calculated as

$$l_0 \approx ..2\text{m}, \Delta l \approx ..005\text{m},$$
$$\epsilon = \frac{\Delta l}{l_0} = \frac{.005}{.2} = .025 = 2.5\%.$$

A force of no more than approximately .5 N should be necessary to plantar flex the foot. Such a force exerted on an elastic element in the longitudinal direction would result in a tensile stress, which we can determine quantitatively according to

$$\sigma = \frac{F}{A},$$

where $F$ is the longitudinal force and $A$ is the cross-sectional area of the element. For an element with a

cross sectional area of $1\text{cm} \times .2\text{cm} = 2 \cdot 10^{-5}\text{m}^2$, we would have

$$\begin{aligned}\sigma &= \frac{.5\text{N}}{2 \cdot 10^{-5}\text{m}^2} \\ &= .25 \cdot 10^5 \text{N}\,\text{m}^{-2} \\ &= 2.5 \cdot 10^4 \text{Pa}.\end{aligned}$$

Thus, we require a material with a Young's modulus less than or equal to

$$\begin{aligned}E = \frac{\sigma}{\epsilon} &= \frac{2.5 \cdot 10^4\text{Pa}}{.025} \\ &= \frac{2.5 \cdot 10^4\text{Pa}}{2.5 \cdot 10^{-2}} \\ &= 1 \cdot 10^6\text{Pa} = 1\text{MPa}.\end{aligned}$$

This value is very low compared to most engineering materials. Indeed, the Young's modulus for common engineering materials is generally measured in GPa rather than MPa. Even natural rubber has a modulus of approximately 10 MPa which, according to the foregoing analysis, is too high for this application.

Care would need to be taken to ensure that the strain gauge is not subject to variations in length of greater than 4% — unless custom, extra-flexible gauges are used.

In summary, three constraints would need to be satisfied if using strain gauges.

1. The elastic element must be wide enough to accommodate mounting of the gauge.
2. The strain must be less than 4 %.
3. The Young's modulus of the material must be sufficiently low that the force required for plantar flexion is not significantly greater than that required in natural human locomotion.

Laslty, strain gauges require the use of a Wheatstone bridge, which introduces non-linearity in to the measurement system. Care would need to be taken to handle this non-linearity either in software or by other means.

# G Microprocessor communications and debugging

The microprocessor was used in what is referred to as a 'headless' configuration — it had no monitor, keyboard or mouse attached. Thus, communication with the microprocessor had to be performed over a serial port or over a TCP network connection. A serial connection was established in this case, using the

picocom serial program.[7] Files were transmitted and received using the Z-modem protocol, which can be invoked by use of appropriate key-bindings while logged in to the Linux operating system over the serial channel. Thus, source code files to be uploaded and downloaded easily and without terminating the serial connection. This capability was highly useful, and facilitated a faster development cycle than would have been otherwise possible.

A typical means of invoking picocom from the command, for example, might be

```
picocom --send-cmd="sz" --receive-cmd="rz" -b 115200 /dev/ttyUSB0
```

where the 'b' parameter is the baud rate and the other parameters denote the file transfer protocol to be used (Z-modem in this case).

[7] picocom is very similar, in its command-line interface at least, to the well-known minicom. See http://code.google.com/p/picocom/ (accessed May 24 2014).