Design and Development of an MRI-Compatible Tactile Stimulation Device

By

Alexander Hay

M.S. Northwestern University, 2020

Project Submitted in Partial Fulfillment of the Requirements for the
Degree of Master of Science in Robotics
in the
Department of Mechanical Engineering

Executive Summary

The Robotics and Sensorimotor Control Lab investigates how humans perceive somatosensory information in their upper arms. The aim of the project is to design and develop a mechatronic system for delivering mechanical stimuli at an individual's fingertip. The greater aim of this work is to use the designed system to assess whether deficits in tactile perception occur in survivors of a stroke due to the sensory signal not reaching the brain versus ineffective perceptual processing within the brain. The system consists of an actuator to direct airflow, an analog controlled pressure regulator, a controller based on the PIC32, and a Honeywell pressure sensor. The pressure sensor provides feedback with the pressure regulator, creating a closed-loop system.

Table of Contents

Introduction

The purpose of this device is to deliver a tactile stimulus at an individual's fingertip in a controlled manner in an MRI environment. The greater aim of this work is to use the apparatus to assess whether deficits in sensorimotor function are caused by the sensory signal not reaching the brain, or altered processing within the brain.

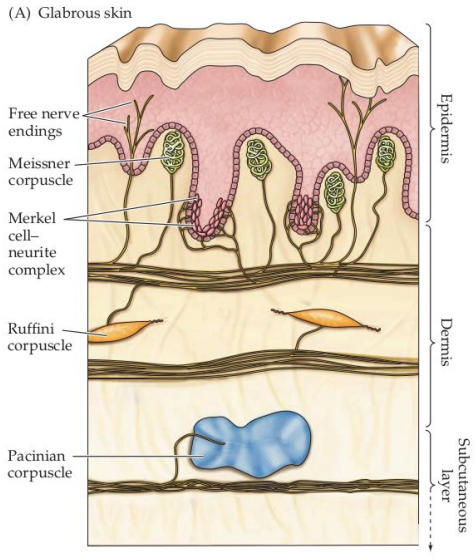
1.1 Scope

The scope of the project is to develop a way to deliver a tactile stimulus to a fingertip in a controlled manner in an MRI environment. The pneumatic device consists of six (6) main components, which are connected via hoses and cables. An air supply is also required. The system was designed to control the actuation of a plastic cylinder. A PIC32 controls a pressure regulator, which in turn controls the actuator; simultaneously, multiple sensors provide feedback.

1.2 Somatosensory System

To deliver tactilestimuli to the finger it is important to understand how the stimuli will be received. In the human finger there are four main mechanoreceptors; Meissner corpuscles, Pacinian corpuscles, Ruffini endings, and Merkel cells. All four together encode touch sensory information that is sent to the brain.

Meissner and Pacinian corpuscles are both rapidly adapting cells, meaning they don't respond to a sustained stimulus. Meissner's corpuscles are sensitive to light touch while Pacinian corpuscles are sensitive to pressure and vibrations such as the sensations involved when handling an object. Pacinian corpuscles are sensitive to higher frequency vibrations, peaking at 50 Hz and 200 Hz respectively.



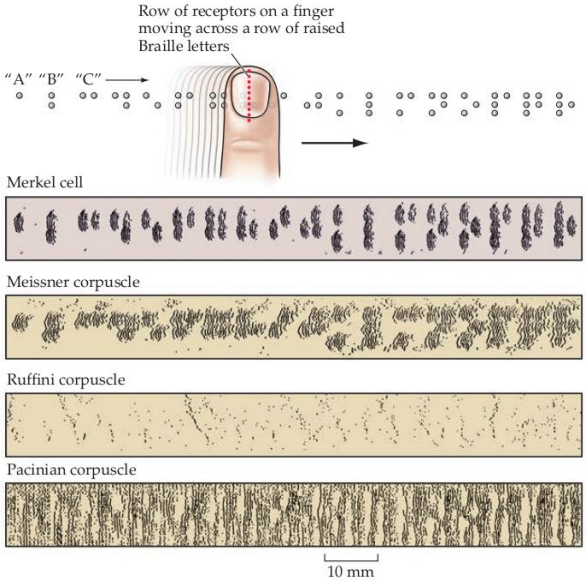


Figure 1.2.1 - Position of receptor cells in the fingertip (*left*); Activity patterns of Braille stimuli (*right*)[1]
Merkel cells are sensitive to deep, static touches, and low vibrations (0-100Hz). They have a small receptive field and transduce detailed information about the surface with which they are interacting. Merkel cells signal the static aspect of a touch stimulus, such as pressure, whereas the terminal portions of the Merkel afferents in these complexes transduce the dynamic aspects of stimuli.[1]

Once a tactile stimulus is applied, sensory information travels to the brain via the dorsal column-medial lemniscal pathway, or via the anterolateral column (noxious/thermal)[2]. After traveling up the spinal column and reaching the medulla, the sensory information decussates, continuing up through the thalamus, and terminating in the somatosensory cortex.

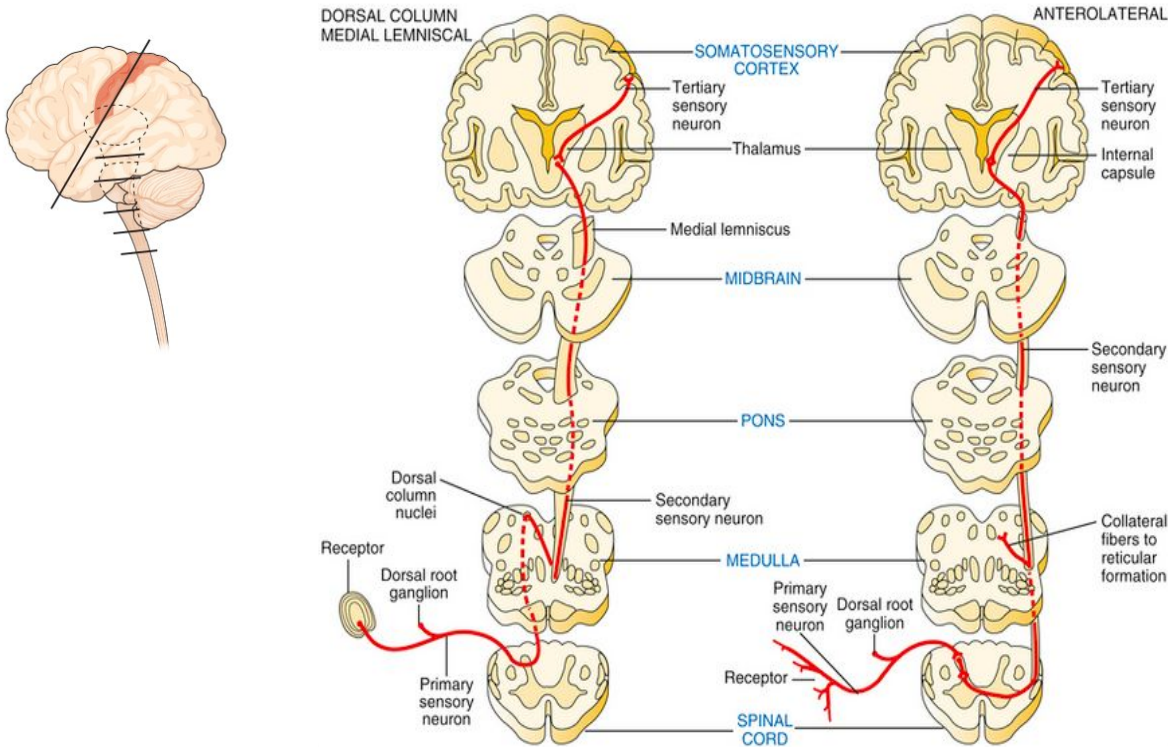


Figure 1.2.2 - The Dorsal Column-Medial Lemniscal and Anterolateral sensory pathways

1.3 Requirements

The following requirements were placed on the project for the following justifications:

MRI Compatibility - The lab will use fMRI imaging during stimulation. The device must be able to operate within an MRI environment and should be made with non-magnetic materials such as brass, rubber, or plastic. Typical MRI scanners range in strength from 1.5T to 3T with research scanners reaching up to 8T. RF heating is also a concern and should be taken into consideration if conductive materials (including the body) are used.

Documentation - Information in this report will serve as a manual for the device. When appropriate, links to source files, CAD files, datasheets, and other supporting documents will be provided and cited. Rationales will also be provided for design decisions.

Fabrication - Due to the accessibility constraints placed by Covid-19, the prototype must be able to be fabricated locally, limiting specialized tools and equipment such as those found in a machine shop. That said, anyone with sufficient access to a lab or fab shop should be able to fabricate this device.

Controller - The controller will be based on the PIC32 microcontroller. The PIC32 provides a platform to exercise firmware level control. Anyone who is comfortable programming in *Arduino* should reasonably be expected to understand the functions of the controller. Any electrical components or sensors should be capable of interfacing with the PIC32 as it will act as the hub of the device.

System Design

2.1 High Level Design

Five (5) methods of actuation were considered for this project; pneumatic, hydraulic, piezoelectric/ultrasonic, mechanical, and electromotive. MRI-compatibility shaped most of the decision process, followed by ability to fabricate the device.

The necessity of the actuator operating in an MRI environment rules out many conventional methods of actuation. Traditional actuators like DC or stepper motors cause electromagnetic interference. If placed near enough to the scanner they will cause EMF noise. Even if placed outside the scanner room there is still nonzero electromagnetic interference caused by the motors. Piezoelectric and ultrasonic motors are more common for this application, however a drawback to using such motors is that the motors can't be actuated while imaging is taking place, otherwise there will be artifacts in the image.

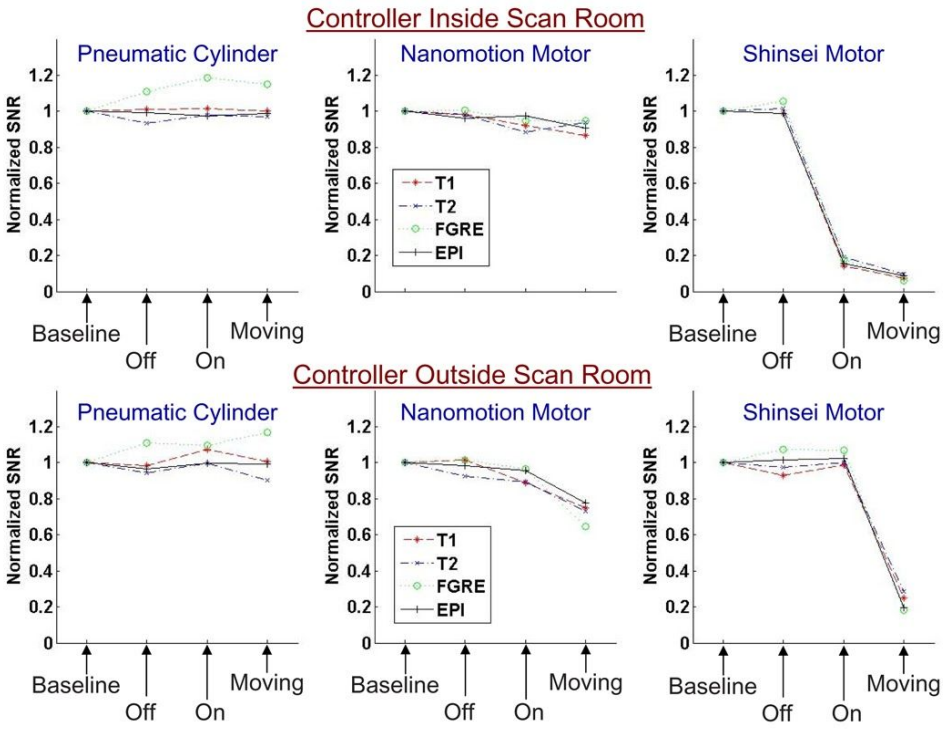


Figure 2.1.1 - Signal-to-Noise ratio using ultrasonic motors[3]

Pneumatic and hydraulic systems provide an alternative means of actuation. Often there is an air source present in or near the scanner room. Air can be delivered using plastic or rubber tubing, and pneumatic fittings are commonly available in plastic. A pneumatic system can potentially be an electromagnetic interference free system. One of the issues that pneumatic systems face is the nonlinear behavior of compressed air. Hydraulic systems work around this issue at the cost of complexity and risk of leakage.

Given the requirements placed on the project and the tradeoffs associated with each method, a pneumatic system was determined to be best suited. Figure 2.1.2 illustrates the decision process.

	Ability to Control	Literature / Support	MRI Compatibility	Construction Difficulty	SCORE
Weights	4	3	5	5	--
Pneumatic	3	4	5	4	69
Hydraulic	4	4	4	3	63
Piezoelectric	4	3	3	1	45
Mechanical	5	5	2	1	50
Electromotive	5	5	2	1	50

Figure 2.1.2 - High level decision matrix

Using a pneumatic system, a controller receives the desired input and information from the touch and pressure sensors, and calculates the necessary output for the pressure regulator. The regulator has a built-in controller to convert the analog voltage signal to the desired pressure output.

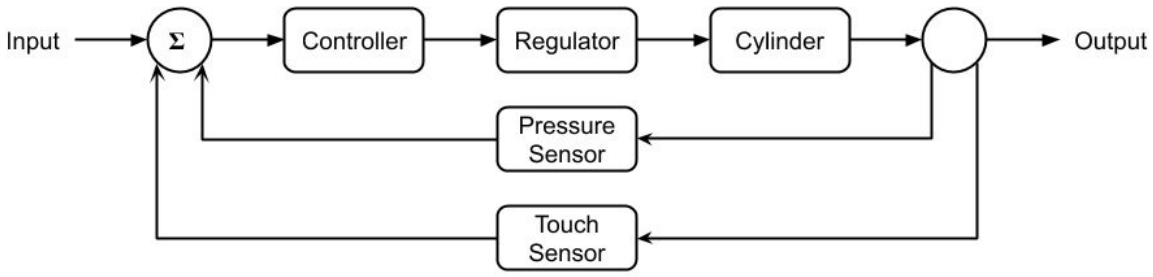
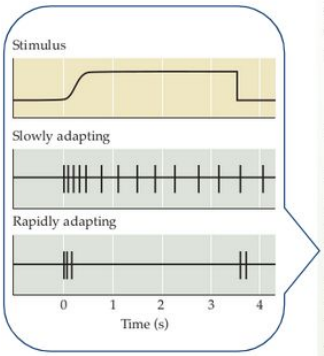


Figure 2.1.3: High-level control loop

2.2 Design Process

Somatosensory Considerations

As mentioned earlier, Merkel cells respond to fine pressure related stimuli. Referring to Figure 2.2.1, information on the properties of each type of receptor cell is provided.



	Small receptive field		Large receptive field	
	Merkel	**Meissner**	**Pacinian**	**Ruffini**
Location	Tip of epidermal sweat ridges	Dermal papillae (close to skin surface)	Dermis and deeper tissues	Dermis
Axon diameter	7–11 μm	6–12 μm	6–12 μm	6–12 μm
Conduction velocity	40–65 m/s	35–70 m/s	35–70 m/s	35–70 m/s
Sensory function	Shape and texture perception	Motion detection; grip control	Perception of distant events through transmitted vibrations; tool use	Tangential force; hand shape; motion direction
Effective stimuli	Edges, points, corners, curvature	Skin motion	Vibration	Skin stretch
Receptive field area[a]	9 mm^2	22 mm^2	Entire finger or hand	60 mm^2
Innervation density (finger pad)	100/cm^2	150/cm^2	20/cm^2	10/cm^2
Spatial acuity	0.5 mm	3 mm	10+ mm	7+ mm
Response to sustained indentation	Sustained (slow adaptation)	None (rapid adaptation)	None (rapid adaptation)	Sustained (slow adaptation)
Frequency range	0–100 Hz	1–300 Hz	5–1000 Hz	0–? Hz
Peak sensitivity	5 Hz	50 Hz	200 Hz	0.5 Hz
Threshold for rapid indentation or vibration:				
Best	8 μm	2 μm	0.01μm	40 μm
Mean	30 μm	6 μm	0.08 μm	300 μm

[a]Receptive field areas as measured with rapid 0.5-mm indentation.
(After K. O. Johnson, 2002.)

Figure 2.2.1 - Afferent systems and their properties[1]

Because the apparatus will press on the user's fingertip, it's important to understand how force (or pressure) travels through the finger. Figure 2.2.2 illustrates what happens to the tissue in the fingertip when pressure is applied and, consequently, veins in the tissue are compressed. This tissue compression is how the Merkel cells are stimulated.

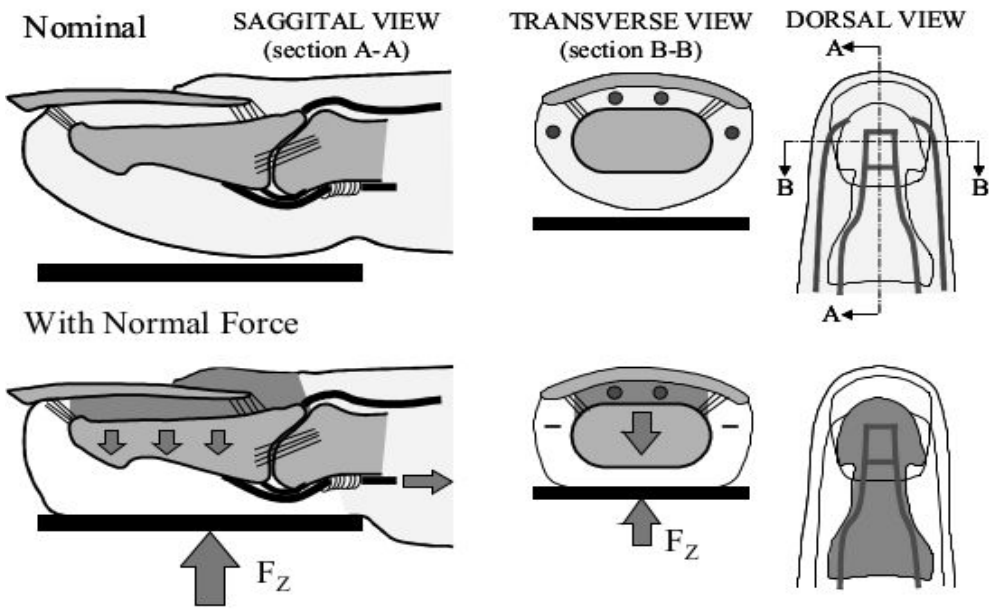


Figure 2.2.2 - Illustration of how fingertip forces travel through finger[4]

The fingernail is intrinsically part of the force-load structure of the fingertip; “note that the compression extends to the area around the bone as well as beneath it. This is because the nail bed fibers are in tension and pull the nail down with the bone, compressing all the tissue that is around the bone but beneath the nail...” “...slowly adapting type I afferents from the sides and end of the finger respond to stimuli on the center of the fingerpad.”[4] The experimental apparatus used by Birznieks et. al has a similar function and caps the force applied to the fingertip at 4N.

It’s important to know the pain/damage thresholds that exist in the human body. Brennum et. al. investigated the pressure-pain threshold in 30 individuals [5] and found that pain can first be experienced when a pressure of ~400kPa is applied to the fingertip. It should be noted that there’s some nuance to this, such as speed [6] or probe shape.

Hardware Considerations

Inputs - The system requires two (2) inputs; (1) a desired actuation and (2) feedback signal from the pressure sensor. A second feedback signal, touch, can be incorporated as well, taking advantage of the CTMU peripheral of the PIC32.

Pressure Sensor - The Honeywell pressure sensor was chosen from the available sensors on Mouser and Digikey, with a priority placed on accuracy and operating pressure. Accuracy should be within 0.25%. Due to the proportions of the actuator, a lower accuracy comes at a cost of actuation precision, which could potentially result in damage to the user. Both analog and digital (I2C) sensors were considered. Ultimately, the analog sensor was chosen because it requires fewer PIC32 pins, which became a premium resource as development progressed. More on the topic can be found in the Discussion & Future Work section.

Touch Sensor - The PIC32 has an onboard peripheral called the Charge Time Measurement Unit (CTMU). It's designed to facilitate capacitive touch-based applications and only requires one pin on the PIC. It works by supplying a constant current to the pin and measures the voltage using the ADC. More information can be found in Microchip's CTMU Documentation.

Calibration is needed before each use, and this procedure is described below:

1. Turn on unit, wait for it to boot
2. Read capacitance display, note value
3. Touch the probe with your finger, note the value
4. In main.c, set cap_calib accordingly

Simple insulated copper wire can be used in the MRI room as the CTMU probe, minding any RH heating that may occur. The CMTU probe can be sensitive; if the probe's wires have thin insulation, skin contact with the wire can change the CTMU value. This is either a feature, or a bug, but any insulated wire larger than a typical jumper wire is enough to avoid the issue.

Outputs - The only output of the control system is to the QB1X pressure regulator.

Actuator - While the proportions could be changed, keeping the overall cylinder size small allows faster actuation speed and requires less airflow. It's made with a PLA polymer using a 3D printer. O-rings were purchased from McMaster Carr.

2.3 Circuit Design

The control system was designed around the PIC32 microcontroller. An FTDI-USB cable provides a voltage source for the L4931 voltage regulator and facilitates serial communication with a computer. The voltage from the regulator typically measures 3.36V. A separate 6V source is needed for the MIC2940 regulator, which supplies the relay and pressure sensor. The solenoid and QBX pressure regulator have their own power supplies.

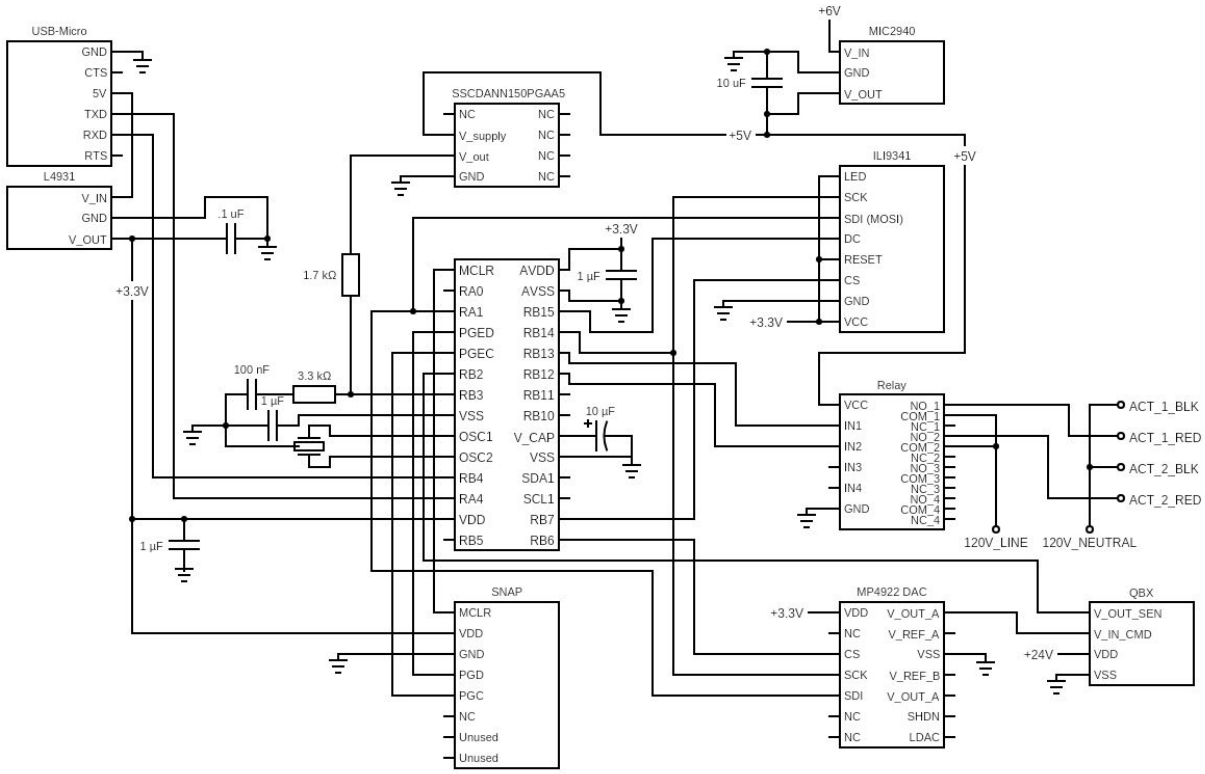


Figure 2.3.1: Overview of the system

Apparatus

3.1 Major Components

Air Supply

The California Air Tools P1060S provides 1.20 CFM @ 90 PSI, has a one gallon tank, and runs at 56dB. It's lightweight and can be easily carried by one person. The Makita MAC100Q was also considered, but was passed over for being 2dB louder.

Solenoid

A 3-position solenoid from AutomationDirect is used to direct the airflow to/from the chambers of the pneumatic actuator (Figure 3.1.1). Air is exhausted to the environment through mufflers. It uses 2 coils to direct the air, each requiring 120V and a command signal, necessitating the use of a relay. Air supplied to the actuator is also supplied to the controller received by the pressure sensor.

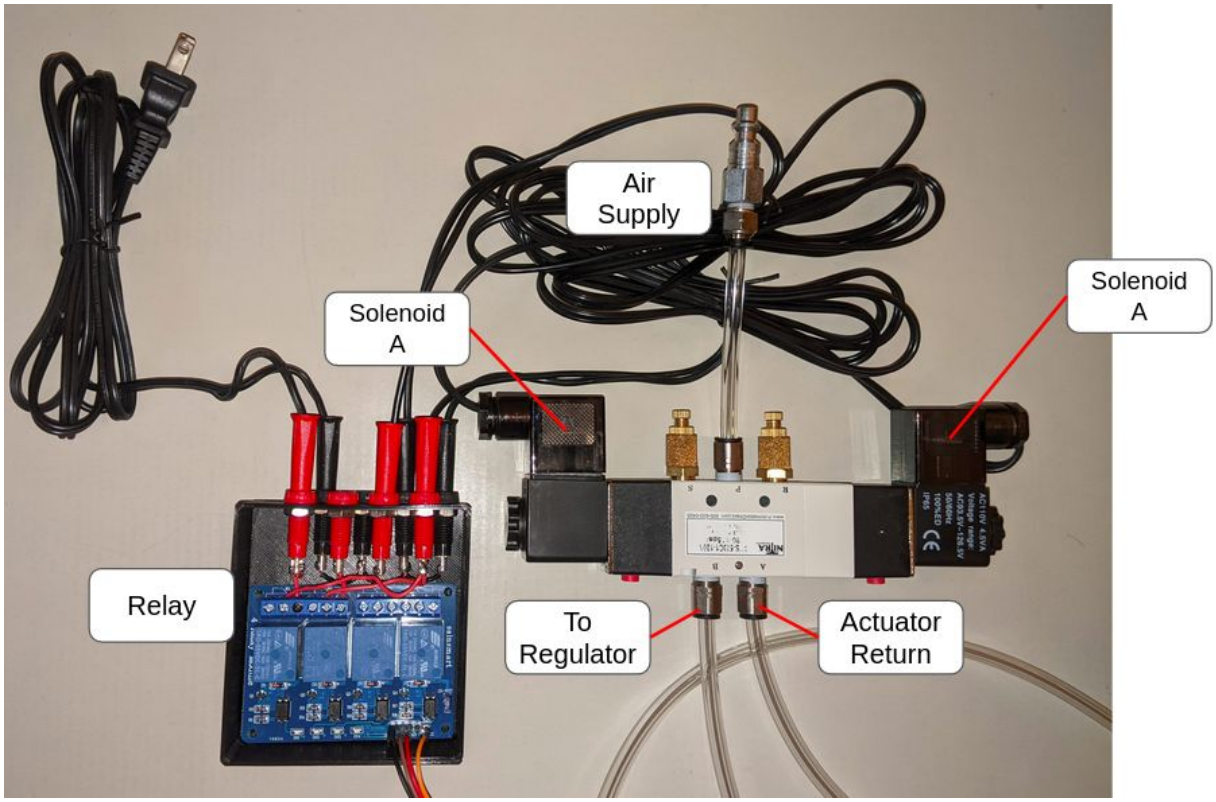


Figure 3.1.1 - Solenoid diagram

Regulator

The QBX pressure regulator receives air from the solenoid and a command signal from the PIC32, and supplies air to the actuator. Commanding the QBX alters the speed and force in which the actuator moves. There is also an onboard pressure sensor that should corroborate with the Honeywell pressure sensor, as they both measure the actuator's feed pressure.

Relay

The PIC32 operates at 3.3V. A relay is needed to *relay* the command signal from the PIC32 to the 120V solenoid. The relay circuit module draws ~72mA/coil, which is more than what the L4931 can provide while also supplying the PIC and ili9341 screen. A separate 5V supply rail was developed (Figure 2.3.1). Because the solenoid requires 120V, a protective shell was designed and a banana wire-clip system was implemented. More on the topic is in the Discussion & Future Work section.

Actuator

The actuator is a 3D-printed, PLA pneumatic actuator. ¼" NPT holes need to be tapped on either side of the piston for the plastic tubing fittings. Rubber o-rings from McMaster provide a seal between the piston rod and cylinder walls. The cylinder head is sealed using a polymer epoxy.

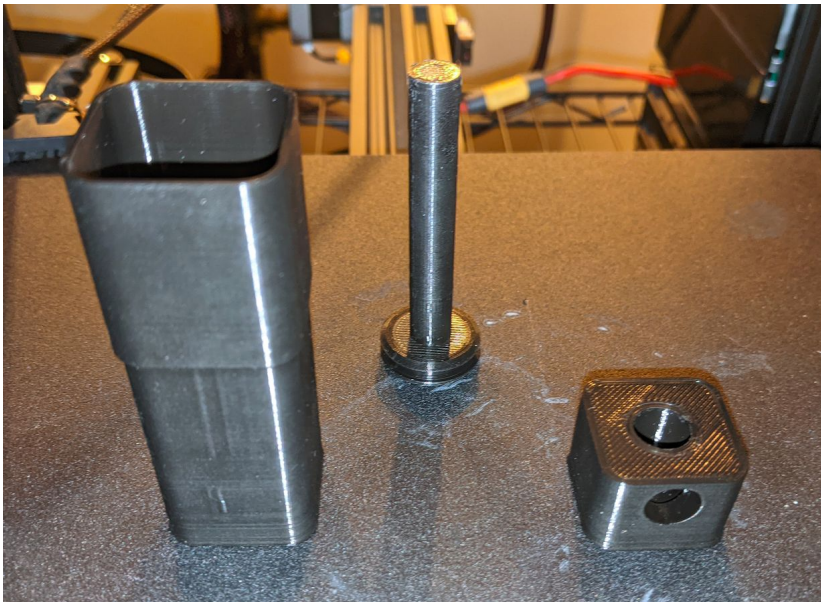


Figure 3.1.2 - Testing new cylinder/piston dimensions

A spring return variant was also developed and tested. Though promising, it was difficult for the actuator to return to its initial position, primarily due to the rough finish of the 3D printer and getting caught on itself. The pneumatic return variant was used moving forward.

Leakage is a recurring problem with the design. In addition to O-Rings, piston rod rings were incorporated into the design to alleviate the issue. Though not unsuccessful, it was ultimately not utilized in the final design. Future work focusing on surface finish of the actuator would go a long way in addressing leakage issues.

Controller

The control system was designed around the PIC32 microcontroller. An FTDI-USB cable provides a voltage source for the PIC and facilitates serial communication with a computer. The voltage from the regulator typically measures 3.36V. A separate 6V source is needed for the MIC2940 regulator to power the 5V peripherals. System status is displayed to a 320x240 LCD touchscreen. It monitors system pressure, sensor frequency, serial communications, and touch sensitivity, and can graph sensor data in real time. More information can be found in the Software section. Refer to Figure 2.3.1 for the circuit diagram.

3.2 Assembly

Connect a ¼” airline from an air source to the middle port on top of the solenoid. The ports on either side should have a muffler. On the bottom, the left port feeds into the QBX regulator, the right port is the return air from the actuator.

Each solenoid has red and black leads that plug into the right ports of the relay. The left-most red and black ports supply the 120V and have their own cable. Smaller leads from the controller plug into the input pins of the relay.

The QBX regulator has 3 or 4 leads that go to the controller, depending on if you want to take advantage of the onboard pressure sensor. It corroborates with the controller’s pressure sensor as they both measure air pressure supplied to the actuator.

Air from the regulator flows to both the controller and actuator using the T-connector.

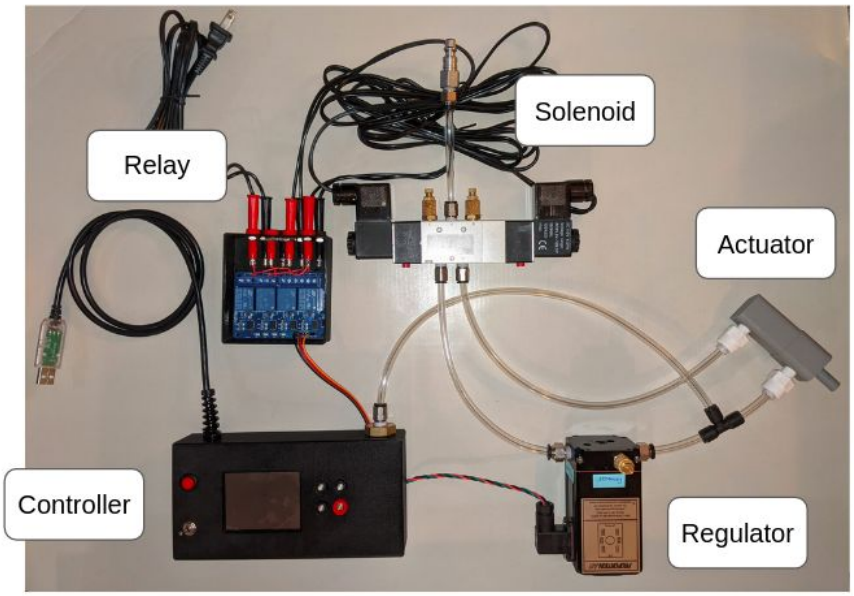


Figure 3.1.3 - Complete system setup

Data sheets for the solenoid, regulator, and controller components can be found here.

Software

4.1 IDE & Debugger

The following software is required to program the PIC32:

- Native C Compiler
  - MinGW (Windows)
  - gcc (Linux)
- Make
  - Obtained via MinGW (Windows)
  - make (Linux)
- Microchip XC32/32++ Compiler
- FTDI Virtual COM Port Driver (Windows)
  - All FTDI devices now supported in Ubuntu 11.10, kernel 3.0.0-19
- Terminal / Serial Interface
  - PuTTY (Windows)
  - PuTTY (Linux)

4.2 Control Library

Version Control

A git repository was created to develop the controller and to store component data sheets. Each controller feature has its own dedicated branch, and once a feature was developed it was merged with the master branch.

Source Files

adc.c - Contains the necessary functions to initialize and use the ADC and CTMU peripherals.

`void adcConfigureManual()`
Input:
N/A
Function:
Configures the ADC for manual sampling, sampling rate configured here

`void adcConfigureManual()`
Input:
`adcPINS`: pins to use for scan mode
Function:
Configures the ADC for automatic sampling. **Double check AD1CON2SET**! The buffers need to be set manually, a consideration for future updates.

`int val = analogRead(char analogPIN)`

Input:

`analogPin`: PIC32 input pin

Output:

`val`: integer value from ADC buffer between 0-1023, sampled manually

`int val = analogRead_auto()`

Input:

N/A, pins that are automatically sampled are pre-assigned

Output:

`val`: integer value from ADC buffer between 0-1023, sampled automatically

`void ctmu_setup()`

Input:

N/A

Function:

Configures the Charge-Time-Measurement-Unit (CTMU) module on the PIC32. The module needs time to charge its capacitor.

`int val = ctmu_read(int pin, int delay)`

Input:

`pin`: PIC32 input pin

`delay`: time delay, on order of 10e5

Output:

`val`: integer value from ADC buffer between 0-1023, sampled manually

`int val = av_cap(int pin, int delay, int win)`

Input:

`pin`: PIC32 input pin

`delay`: time delay, on order of 10e5

`win`: window of samples to average over

Output:

`val`: integer value from ADC buffer between 0-1023, sampled manually, averaged over the window provided. This is a workaround to calling the ctmu_read function recursively.

ili9341.c - Initializes the screen, as well as initializes SPI communication. There are also helper functions to display characters, draw, and clear the screen. (0,0) on the screen is the upper left corner unless otherwise defined/oriented.

`void SPI1_init()`
Input:
N/A
Function:
Initializes SPI module on PIC32. **This is where SDI/SDO are set**

`void LCD_init()`
Input:
N/A
Function:
Sends commands via SPI to initialize the screen

`void LCD_drawPixel(unshort x, unshort y, unshort color)`
Input:
`x`: x screen coordinate
`y`: y screen coordinate
`color`: color, as defined in ili9341.h

Function:
Draws a pixel at (`x`, `y`) location in `color`, per ili9341.h.

`void LCD_clearScreen(short color)`
Input:
`color`: color, as defined in ili9341.h

Function:
Makes entire screen `color`

`void print_char(unsigned short x, unsigned short y, char ch)`
Input:
`x`: x screen coordinate
`y`: y screen coordinate
`ch`: ASCII character

Function:
Draws ASCII character `ch` at location (`x`, `y`).

```
void clear_space(unsigned short x, unsigned y, unsigned short
    end)
```

Input:

x:	x screen coordinate
y:	y screen coordinate
end:	length to clear

Function:

Clears ASCII chars

```
void write_screen(int x, int y, char *msg)
```

Input:

x:	x screen coordinate
y:	y screen coordinate
msg:	message to write to screen

Function:

Writes message to screen by calling print_char

init.c - Contains the PIC's configuration settings and contains functions to run through the initialization routines.

```
void init_pic()
```

Function:

Initializes core PIC32 configurations. **sysclk is configured here.**

uart.c - Initializes UART serial communication and contains the read/write functions.

```
void initUART(int desired_baud)
```

Input:

desired_baud:	Baud rate for serial communication

Function:

Configures the UART module of the PIC32. **Tx/Rx pins are set here.**

```
void writeUART(char * string)
```

Input:

string:	Message to be sent

Function:

Sends message via serial transmission

`void readUART(char * message, int maxLength)`
Input:
`string`: Message to be sent
`maxLength`: Maximum length of message
Function:
Receives message via serial transmission

main.c - Contains the main loop function, the transfer function for the pressure sensor, and some helper functions

`void ui()`
Function:
Displays a decorative header

`void heartbeat(int freq)`
Input:
`freq`: Flash frequency
Function:
Flashes single pixel at (0,0). Should be rewritten as a timer/interrupt

`double val = transfer_function(int voltage)`
Input:
`voltage`: 0-1023 value as returned by the ADC
Output:
`val`: output value is proportional to the applied pressure and atmospheric pressure.

A proportional transfer function rather than the one Honeywell provided because it provided expected values under known conditions, while the Honeywell did not. The pressure sensor was checked against a pressure gauge on the compressor air supply and the pressure gauge on the compressor tank.

Discussion & Future Work

During development I made it a priority that the device was MRI-compatible, which influenced many of the design considerations and challenges. The decision to use a pneumatic system made it so there is no interference from the apparatus during the MR imaging process. However, air is a compressible fluid which makes it difficult to control. A hydraulic system was considered to address that but was never implemented due to concerns about leakage.

That said, leakage still proved to be a nuisance throughout the project. Pistons of various sizes were printed in an effort to dial in the tolerances, but ultimately will require better surface finishing. In addition to O-Rings, piston rod rings were incorporated into the design to alleviate the issue. Though not unsuccessful, it was not utilized in the final design.

The control loop has two pressure sensors and one touch sensor. An optical linear encoder using fiber optic cables was considered, which would have provided more certain feedback of the piston, but ultimately was shelved due to time constraints.

Future work with this project will primarily focus on characterization of the actuator, as well as making each element more robust and providing the framework for which development can continue. The two largest lessons were in time management, setting realistic expectations and goals, and project management, learning how to juggle multiple facets of a project and navigating the way forward.

The solenoid requiring a relay added complexity to the system that wasn’t anticipated. Working with that voltage is dangerous, and accelerated the design and construction of a protective shell. I needed to address how to safely and reliably make connections and generally handle the apparatus without touching any live wires. I had to learn how to source hardware bits and design around them, making the relay box a tight fit. The control box, which was developed later, has much more space.

Knowing if/when the piston engages with the finger is also important. The controller currently takes advantage of the CTMU module on the PIC32. For it to work, the surface that the fingertip engages with must be conductive (and MRI-compatible, ie. aluminum foil) for it to work. Studies examining the heating effects of MRI scanning have shown that no significant heating occurs with non-ferromagnetic materials.[7] This means the individual won't risk burning their finger. And because the peripheral is non-moving, there's no risk of interference during imaging[8].

References

[1] Kandel, E. R., 2013. Principles of Neural Science. 5th ed.

[2] Purves, D., Augustine, G., Fitzpatrick, D., Hall, W., LaMantia, A., Mooney, R., Platt, M. and White, L., 2018. Neuroscience. 6th ed.

[3] Wang Y, Cole GA, Su H, Pilitsis JG, Fischer GS. MRI compatibility evaluation of a piezoelectric actuator system for a neural interventional robot. Conf Proc IEEE Eng Med Biol Soc. 2009;2009:6072-6075. doi:10.1109/IEMBS.2009.5334206

[4] Mascaro S, Asada H. Understanding of fingernail-bone interaction and fingertip hemodynamics for fingernail sensor design. Proceedings 10th Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems HAPTICS 2002. 2002. doi:10.1109/haptic.2002.998948

[5] Hogeweg JA, Langereis MJ, Bernards AT, Faber JA, Helders PJ. Algometry. Measuring pain threshold, method and characteristics in healthy subjects. *Scand J Rehabil Med*. 1992;24(2):99-103.

[6] Lorusso L, Salerno M, Sessa F, et al. Autoalgometry: An Important Tool for Pressure Pain Threshold Evaluation. *J Clin Med*. 2018;7(9):273. Published 2018 Sep 12. doi:10.3390/jcm7090273

[7] Buchli R, Boesiger P, Meier D. Heating effects of metallic implants by MRI examinations. Magn Reson Med. 1988;7(3):255-261. doi:10.1002/mrm.1910070302

[8] Fischer GS, Krieger A, Iordachita I, Csoma C, Whitcomb LL, Gabor F. MRI compatibility of robot actuation techniques--a comparative study. Med Image Comput Comput Assist Interv. 2008;11(Pt 2):509-517. doi:10.1007/978-3-540-85990-1_61

Birznieks I, Jenmalm P, Goodwin AW, Johansson RS. Encoding of direction of fingertip forces by human tactile afferents. J Neurosci. 2001;21(20):8222-8237. doi:10.1523/JNEUROSCI.21-20-08222.2001