Final Report

ARCANE: A Navigational Aid for the Visually Impaired

Alex Davis | Anastasia Dodd | Abeer Javed
Lindsay Langford | Joseph Song | William Xie

Submitted: May 4th, 2014

Senior Design
Rice University
Spring 2014

Table of Contents

Executive Summary

Team ARCANE will design and create an intuitive, user-friendly device that will aid in the navigation for the visually impaired. The needs for this device are as follows:

- Visually impaired people face enormous challenges when navigating alone, and strongly rely on visual aids to avoid obstacles and dangerous situations.
- Visual aids such as the white cane do not provide sufficient information to the user on his or her surroundings to respond to obstacles in a timely and appropriate manner
- In 2012, there were 40 hospitalizations and 5 deaths of visually impaired people caused by motor vehicle crashes.

The visually impaired require an intuitive user interface that will effectively communicate essential information of their surroundings. To that end, Team ARCANE must adhere to the following constraints in order to build a useful product, which includes:

- Cardinal directions to aid in orientation and the construction of mental maps
- Long range identification of obstacles at waist or head height for obstacle avoidance
- Notification to the user of the distance and location of an object through haptic feedback
- Low power consumption for extended periods of use
- Lightweight and comfortable for everyday use as well as durable and water-resistant
- Easily attachable as a supplement to the white cane

To address these objectives while meeting the constraints imposed, Team ARCANE has identified three major components that should go into the device: sensors, haptic feedback, and form factor. The sensors will be pointing in the directions of interest of the user, and develop depth maps to pinpoint nearby obstacles. This information will then be relayed to the haptic feedback, which will trigger certain vibration modules by intensity based on the location and proximity of the object. Finally, this will go into the form factor of the individual, which will interact directly with the user to provide information on these obstacles.

Team ARCANE designed and built its first prototype in December 2013, showcasing the integration of the sensors, haptic feedback, and form factor. This was tested on blindfolded human subjects, identifying major inconveniences and possible design flaws evident to the user. An improved prototype was created in February 2014, focusing on faster feedback to the user. This second prototype was IRB-approved to begin testing with visually impaired individuals to test the accuracy, ease of use, and maneuverability of the device. The final design was made as

light and durable as possible and showcased in the Engineering Design Showcase in April 2014.

Introduction

While taking a nice walk, it's easy to enjoy your surroundings and get lost in your own thoughts, mindlessly wondering towards your destination. Now, imagine taking this exact same route with your eyes closed. Every single step would have to be carefully thought out, with obstacles in your way presenting challenges you have to face. For the 285 million visually impaired across the world as of 2012, this is a day-by-day reality. Team ARCANE is creating a device that will improve the navigation experience for the visually impaired by providing haptic feedback to translate the surroundings to the user.

Navigational Needs

Visually impaired people lack the ability to easily navigate by themselves, and usually require assistance for daily mobility. Obstacles easily avoided by sighted people become much more problematic for those with limited vision. Visually impaired people are also endangered by trip hazards such as cracks in the sidewalk, curbs, and changes in pavement. Transitioning from cement to gravel, for example, may trip a visually impaired person who doesn't anticipate the upcoming change. Curbs without wheelchair ramps feel like sudden drops and make it all too easy to step out into a street without realizing it. Overhead obstacles such as low-hanging branches and signs pose a risk of head injuries. While all of these scenarios may seem trivial, they significantly affect the mobility and independence of the visually impaired.

Highways and crosswalks also present a constant stream of danger for those who cannot see the flow of traffic. While the white cane indicates that an individual is visually impaired, it can be nerve-wracking to hold drivers and pedestrians accountable for their safety. While there are only 40 hospitalizations and 5 deaths of visually impaired people caused by motor vehicle crashes per year [Hogan, 2008], the loss of confidence from an accident or even a near miss can be crippling when added on to the already enormous challenges of navigation.

Additionally, while navigating, visually impaired individuals rely heavily on cardinal directions to orient themselves. Most build mental maps based around cardinal directions and the sounds and smells that they sense relative to these [Texas School for the Blind and Visually Impaired, 2010]. Because of this, they rely on familiar landmarks, other individuals, or electronic devices (e.g. navigation apps) to get from one point to another.

First solutions

Visually impaired individuals require assistance in daily navigation; the common solutions are the white cane and guide dogs. While they are the predominant visual aids, they have several shortcomings which limit their overall effectiveness.

White Cane

One of the most common and inexpensive navigation aids used by the visually impaired is the white cane. At a price of $30, a white cane allows the user to navigate independently in both indoor and outdoor environments and alerts other people that he or she is blind. However, there are multiple drawbacks to the white cane. It has a limited range for detecting obstacles and requires the user to physically tap objects to identify them which in turn limits reaction time. It also completely misses objects higher than the waist, which can be dangerous in an outdoor environment. Additionally, while some canes do fold up or telescope into a smaller size for easier storage, the joints in those canes disrupt the vibrations travelling through them.

Figure 1: An individual using a white cane to detect changes in terrain

Guide Dogs

Another common navigational solution for visually impaired individuals is the guide dog. Guide dogs can similarly steer visually impaired people away from obstacles and dangerous situations and are another frequently used navigational aid. However, the dogs must go through an extensive training program as shown in Figure 2, which can cost up to $20,000 per dog. These costs typically do not fall under insurance coverage, making guide dogs a poor solution for visually impaired individuals with limited incomes. Furthermore, it takes a few months to adapt to each new dog and learn to command it effectively. Because guide dogs are most effective in peak physical condition, they generally retire after 6-8 years of service. [Guide Dogs of America, 2013]

Figure 2: A guide dog leads a woman through

State of the art solutions

Recent commercial, academic, and hobbyist solutions utilize technology to address the shortcomings of the traditional white cane and guides. Most use distance sensors to detect objects and then provide that information to the user through some method of haptic feedback. However, these solutions have several unaddressed shortcomings.

Ultracane

The Ultracane is a commercially available product that uses ultrasonic sensors integrated onto the handle of a cane to detect approaching hazards. The device notifies the user through two vibrating transducers: one to signal hazards at waist level, and the other for hazards at head and chest level. The Ultracane is sturdy and resistant for every day, outdoor use. Furthermore, the device has only a small learning curve; users simply need to toggle between two operating modes and receive feedback from two vibration sources.

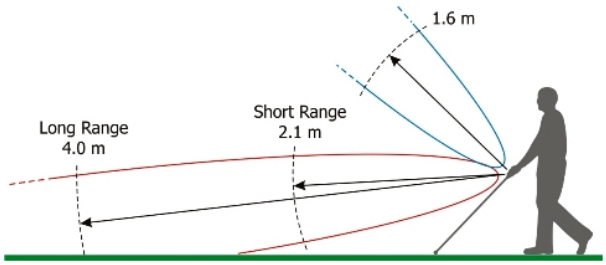
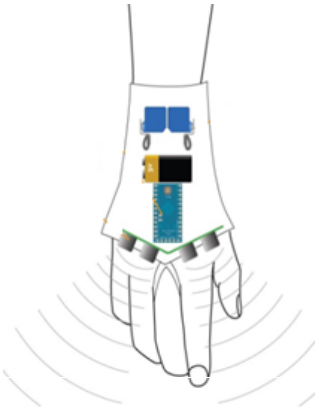


Figure 3: The UltraCane detects obstacles at several heights and ranges

Unfortunately, the Ultracane is bulky, straining the user's hand and, specifically, their index finger. The placement of the vibrating transducers also makes shifting hand positions difficult for some users. Furthermore, it costs more than $1000 and is not covered by insurance. The device is powered by AA batteries and is not rechargeable, inconveniently forcing the user to purchase replacement batteries. Furthermore, its audible tones indicating low battery life are soft and cannot be distinguished in noisy outdoor environments, making this feature practically useless. [Héiligh, 2012]

Tacit

Tacit is a glove-like device that uses two ultrasonic sensors to detect objects. The user waves the glove around to understand his or her surroundings. The information is then processed in an Arduino and relayed to the user via two servo motors, which apply varying amounts of pressure to the back of the hand. The pressure exerted by the glove increases as an object gets closer to the corresponding sensors. [Hoefer, 2011] Figure 4 outlines the basic design of the haptic glove.



The Tacit is small, uses simple and relatively cheap hardware, and responds in real time. Unfortunately, two ultrasound sensors only sample two points and hence provide low resolution

output. In addition, entering a room and waving a hand around can be obtrusive and inconvenient for the user.

Kinecthesia

The Kinecthesia is a student project that uses a Microsoft Kinect as the main sensor. As shown in Figure 5, it is meant to be worn around the waist like a belt. There are numerous vibration sensors spaced around the waist to indicate locations of the obstacles ahead of the user through haptic feedback. The intensity of the feedback is scaled to the distance from the object using depth data gathered by the Kinect. [Berdinis, 2012]

Figure 5: A Kinect, belt, microcontroller, and vibrating modules compose the Kinecthesia.

One of the primary benefits of the Kinecthesia is its relative discretion. It is meant to be worn like a belt, freeing the user's hands for other tasks, and draws minimal attention. Multiple vibrating feedback modules provide directional information allowing ease of navigation. Unfortunately, the Kinect drains a large amount of power, thus limiting the operating time of the belt. The fundamental flaw of the device lies in its limited detection range: it cannot detect objects below or above the waistline. Additionally, it does not work outdoors, which is crucial to the navigation of the visually impaired.

Brainport

Brainport is a device commercially sold by Wicab Inc. that captures visual information via a 1.5cm diameter camera placed on a pair of sunglasses worn by the user. This data is then processed and transformed into electrical pulses, which are relayed to a matrix of electrodes placed on the tongue, as shown in Figure 6. The visually impaired user is able to interpret each electrode as the image in front of him or her. [Kendrick, 2009]

The tongue is used as the receptor due to the high density of nerves located close to its surface; this differs from outer skin, which is covered by a layer of dead skin cells and hair. In addition, the Brainport system can be used in tandem with the traditional cane. However, a pair of wires entering the mouth makes the device strange to wear and may deter visually impaired individuals from using it at a daily basis. Furthermore, the tongue has a minimal usable surface area, limiting the electrode grid size and thus output resolution.

Problem statement

Visually impaired people need to easily and intuitively navigate in new environments to prevent potentially dangerous situations and avoid obstacles. Current solutions such as the white cane return only a single output informing the user if an object is present or not. These visual aids also require the user to physically tap an obstacle, which limits reaction time to one or two steps. Furthermore, they do not warn users of overhead obstacles or surface irregularities such as drop-offs, potholes, and other trip hazards. While state of the art solutions do solve some of these issues, many of them have their own unique shortcomings. To improve the navigation abilities of visually impaired people, Team ARCANE aims to create a lightweight, comfortable, and reliable device that will supplement the white cane. This device will provide haptic feedback to furnish the user with a high-resolution mental map of upcoming obstacles and the surrounding area.

Our navigational aid will give users the confidence to move assertively, without fear of unknown objects ahead and with additional time to react to any obstacles in their path. Because our navigational aid is a supplement rather than a replacement of the white cane, users will still be able to indicate their disability to people around them and will never be left stranded because of a dead battery or other technological failures, providing an added sense of security. The rest of the binder will outline Team ARCANE's process to creating this visual aid, one step at a time.

Design Strategy

In Fall 2013, Team ARCANE was challenged to create a prototype for an intuitive, user-friendly device that will aid in the navigation for the visually impaired The Design Strategy will explore how Team ARCANE is approaching this problem and why. Before diving deep into each of the design's components, it is important to understand how the overall design will work.

First, two RGB cameras attach to the user's head will take an image of the user's surroundings. This image is then processed into a depth map, which will detect the location and

proximity of the obstacles of the user. This depth map is relayed in the form of a serial code to the LED driver. When an obstacle is reasonably close to the user, the driver will then strategically turn on a motor located on the user's form to indicate location, and intensify in amplitude as the obstacle gets closer. **Figure 7** outlines the data flow of this information.

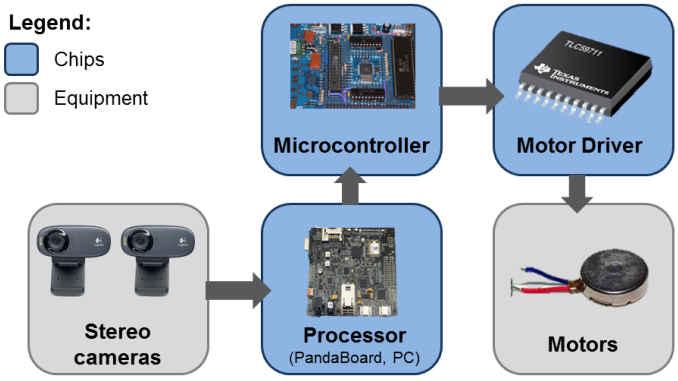


Figure 7: Design Flow for Prototype

The rest of this document will break down the strategy into its three main components along with the reasoning behind each: sensors, haptic feedback, and form factor. Additionally, it discusses risks associated with the project, and any regulations that the team must keep into account. The design focuses on meeting the design criteria that have been set on this project, which can be found in the **Design Criteria** section of this binder. The costs for each corresponding components can be found in the **Expenses** section of this binder.

Sensors

Our design relies on sensors to accurately generate a depth map of the environment in front of the user. This depth information is processed and relayed to the user via the haptic feedback modules.

Selection

Category	Weight	DepthSense 325	DepthSense 311	PrimeSense 3D Sensor	Asus XTion
Size	30%	5	3	2	2
Power	30%	5	1	5	5
Range	20%	1	5	4	4
Price	10%	2	1	3	4
Field of View	10%	5	1	3	3
Weighted Sum		3.9	2.4	3.5	3.6
Rank		1	4	3	2

Figure 8: Pugh Matrix for Depth Sensors

Using the decision matrix in **Figure 8**, we decided to use the DepthSense 325 as the depth sensor for our device. As pictured in **Figure 9**, the DepthSense 325 is a portable, IR-based, time-of-flight sensor that can output depth map directly from the camera module. It was chosen primarily due to its compact size and low operating power. The sensor will need to be located on our final overall design, and as such form factor is very important. A large, bulky sensor would be uncomfortable not only due to its weight, but also because it would impede movement; if the sensor is placed on the cane, body parts such as the users free hand would repeatedly collide with the aid; if the sensor is placed on the body, the device would limit the body part it is attached to. The second factor, power, is crucial as our design is meant to be portable and thus will need to operate on battery power. Low operating power is a prerequisite in ensuring convenient use in daily outings.

Figure 9: The DepthSense 325

After ordering and testing the sensor, we found its limitations made it an inadequate option for our design. First and foremost, the DS325 is an infrared time of flight sensor; it detects distances by measuring the time between emitting an IR wave and detecting its reflection. Sunlight, filled with IR, overloads the sensor and effectively blinds it. As such, the sensor has limited applicability outdoors in sunny weather. Furthermore, the DepthSense has a limited range of up to one meter, required to be supplemented by additional sensors to meet our

design goal of detecting objects up to 12 feet away. Lastly, the DS325 currently does not have ARM drivers, which are required to interface with the processing board (e.g. BeagleBoard, PandaBoard) we are using to relay the information to the user; these drivers are currently under development.

We concurrently began prototyping with using computer stereo vision as an alternative method to generate a depth map. Computer stereo vision is the extraction of 3D information from digital images, such as obtained by a webcam. 3D information can then be extracted by observing and comparing the relative positions of two objects in distinct panels. Using stereo vision addresses the issues plaguing the DS325, such as outdoor usability. Furthermore, the algorithm has been an active area of research and implemented code is available in the OpenCV library. By using OpenCV functionality, we aim to minimize implementation errors on our part.

Category	Weight	Logitech C310	Logitech C920	Creative Live! Cam
Size	40%	5	1	3
Power	30%	5	1	5
Range	20%	1	5	1
Price	10%	5	1	5
Weighted Sum		4.2	1.8	3.4
Rank		1	3	2

Figure 10: Pugh Matrix for RGB Cameras

The same factors guiding the purchase of the DS325 weighed in on deciding which RGB cameras to use. As shown in **Figure 10,** the Logitech C310 is small, low power, and cheap, making it an ideal camera for our design.

Design & Features

Using OpenCV library and two Logitech C310 webcams, we successfully generated depth maps in real time. The depth map is generated using four steps: lens un-distortion, image

rectification, stereo correspondence, and depth estimation via triangulation. The first two steps are part of the calibration that only has to be done once given that the relative distance and angle between the two cameras do not change. The latter two steps we compute in real time.

We calibrated the cameras using an object of fixed geometry; we decided to use an “8 x 6” chessboard due to its simplicity and for its widespread use. We employed algorithm that takes the positions of the corners of the chessboard to fix the distortion and calculated the focal length in pixels for each camera. Then, in the image rectification step, the translational and rotational relations between the two cameras were calculated from the same calibration data. Using this information, the resulting left and right images were row aligned and rectified. This was an important preparation to finding the correspondence. According to epipolar geometry, in order to find corresponding features on two images, the features must lie on the same epipolar line. With our side-by-side parallel cameras set up, the resulting images were row aligned with each row representing an epipolar line.

After the calibration, using the rectified maps, we decided to use block-matching algorithm to find correspondence. This method was chosen due its relative light computation load as opposed to other higher accuracy algorithms such as graph-cut. This property is highly desirable for real time depth map generation. Block-matching algorithm divided the two images into same number of blocks. For each block, the sum of absolute differences (SAD) was calculated. Using the SAD value, for each block on the left image, the corresponding block was found by finding the block with the best matching SAD along the epipolar line to the left of the block's coordinate. The differences in pixels for corresponding blocks created a disparity map.

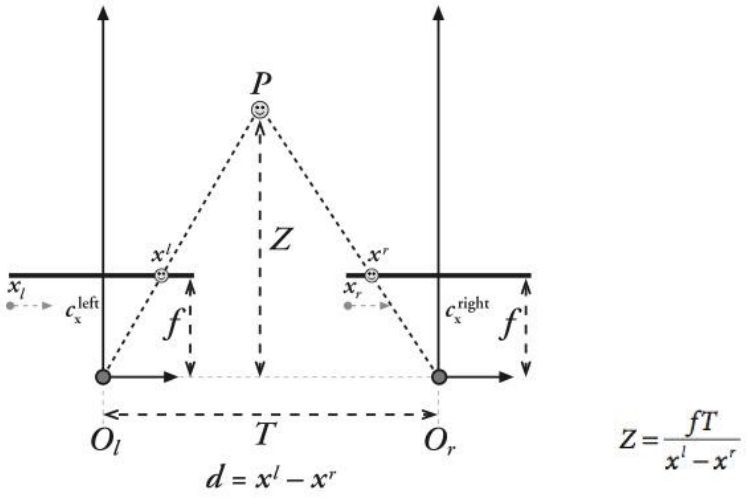


Figure 11: Distance estimation using similar triangles

Finally, for the last step, using the disparity values in pixels (x_l - x_r), calculated focal length in pixels (f), baseline distance between the two cameras (T), the distance at every point was estimated using the equation derived using similar triangles in **Figure 11**. In **Figure 12**, the top portion is a pair of rectified images from the result of calibration. The green lines are selected epipolar lines used for the purpose of demonstration. On lower right corner is a normalized depth map created from disparity map with darker color corresponding to a lower value (black = 0). On the lower right is the rectified left image with nine overlay sampling points. For each point, we took the average value of 25 distances. Because our setup deviates from the idealized setup, we compared real world measurements and calculated value to apply an offset to each value. The coordinate and the distance values are displayed under each sampling point.

Figure 12: Stereovision implementation

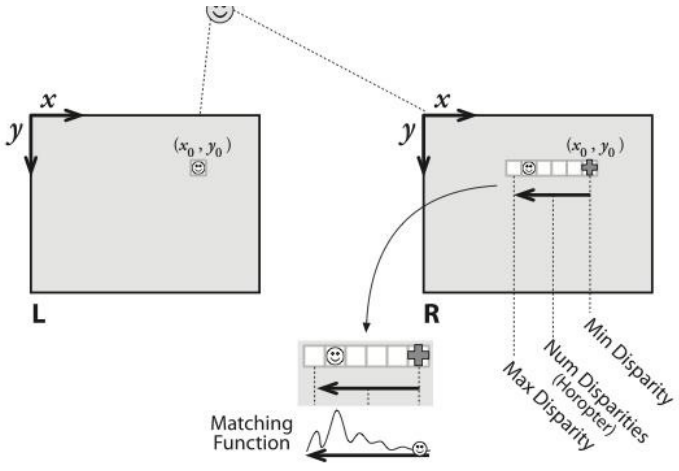


Figure 13: Block-matching algorithm for finding corresponding features

Our program has a few key features. It has calibration built-in. The user can choose to calibrate the cameras at the beginning of the program. By loading the calibration data, the user can skip the calibration step for future use. Our program is designed so that the user can tweak important variable settings such as sampling size, block-matching parameters, and camera IDs easily. Our program also has two variants of the block-matching algorithm implemented and the user can easily switch between the two. Our program communicates with the haptic feedback microcontroller using serial protocol and transmits a string containing normalized data for each sample point. The program uses minimal dependencies (only OpenCV and boost library for serial communication) to ensure portability across all platforms for development and deployment.

Hazards

Electric shock from faulty connections is our primary potential hazard. In particular, we should be fully proactive in avoiding this situation when the user washes his/her hands or there is rain. This is particularly a concern relating to the wires that would connect the sensor to the haptic feedback modules placed elsewhere. In order to address this issue, we plan to first, place the sensor electronics in a waterproof casing with its own power supply. No wires will leave the sensor box. We plan to communicate to the haptic feedback modules using wireless communication, most likely Bluetooth. This solution will be explored in depth in Cycles 3 – 5.

Haptic Feedback

Selection & Hazards

We decided to use vibration to provide tactile feedback to the user of our device because it is safe and widely used. Because vibration feedback is so common, there are a multitude of options for us to choose for our solution, from low cost pager motors which can be controlled the same way as other brushed DC motors to high cost but very refined and high resolution piezoelectric motors.

In addition to vibration, we considered a wide variety of other feedback types. One of our options was thermal feedback, which changes the temperature the user feels, but felt that the safety concerns with placing a heat source on a person were too great. We also looked into

electrovibration, which is a way to stimulate textures on screens, but disliked that the user had to constantly move in order to feel the texture changes. Our fourth option was pneumatic feedback, which involves using puffs of air to provide tactile feedback; there were significant drawbacks to this particular approach, specifically the large size and weight of the necessary equipment and how complicated such a feedback mechanism would be to design. The Pugh matrix below provides a succinct look at our haptic feedback possibilities and why we chose to use vibration.

Category	Weight	Vibration	Thermal	Electrovibration	Pneumatic
Perceptibility	**25%**	4	2	2	3
Safety	**25%**	5	2	3	3
Controllability	**20%**	3	2	3	2
Size and Weight	**15%**	3	2	3	1
Cost	**15%**	3	4	2	1
Weighted Sum		**3.75**	**2.3**	**2.6**	**2.2**
Rank		**1**	**3**	**2**	**4**

Figure 14: Pugh Matrix of Haptic Feedback

By choosing to use vibration feedback instead of something like thermal feedback, which has a significantly higher chance of harming the user due to its heating elements, we are practicing the first level of hazard mitigation, which is removing the hazard of serious harm from our feedback device. When testing with vibration motors, we found that there is still a possible hazard for the user, namely that vibrations which are too strong will become uncomfortable and may even cause temporary numbness. Luckily, this hazard is very easy to mitigate; our team has written software which allows us to control how strongly the motors vibrate, meaning we can easily set limits on the maximum vibration amplitude which is lower than the average threshold for discomfort and far lower than the level which causes numbness. Because every person has a slightly different response to haptic feedback it is very unlikely that we have completely reduced the possibility of discomfort. However, we have designed our control scheme to significantly reduce the chances that discomfort and numbness will occur, which is the second level in the hazard reduction sequence.

As mentioned earlier, there is a wide array of choices even after we

Figure 15: A pancake-style pager motor

chose vibration. We elected to use multiple low cost pancake-style pager motors for a number of reasons: they are small and can be run at low voltages and currents, our complete solution is meant to be low cost, pager motors are easy to control, and being able to afford multiple motors means we can test multiple configurations and make feedback patterns.

However, these motors often have very small and delicate leads attached to them, and the motors we purchased are no exception. For that reason we had to design some coupon boards, which are a way for us to attach the motors to a mechanically rigid surface and then provide a way for us to easily connect the motor to outside devices. This provides mechanical security for the delicate leads and makes the motors easier to handle. The schematic and layout for those boards are pictured below. We also designed in a flyback diode to prevent back EMF into our driver circuit, which is described in the next paragraph, and a capacitor to limit noise in the motor lines.

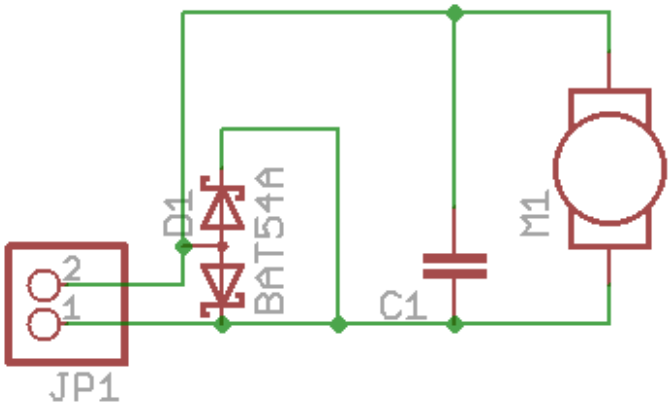


Figure 16: Schematic for motor coupon board

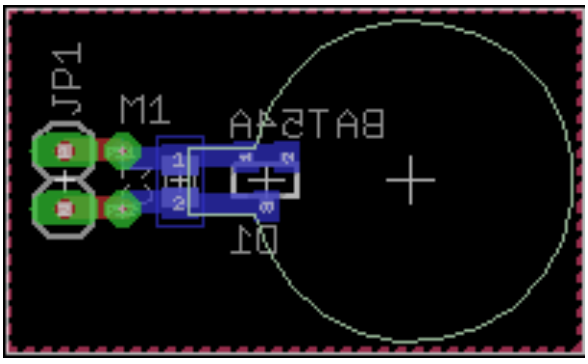


Figure 17: Layout for motor coupon board

Design & Features

In order to control a significant number of motors, our team needed to design a motor driver solution. We looked into a variety of motor drivers, but found that most of them either controlled only one or two motors or were meant for high power applications. After testing the current draw of one of our pancake motors and finding it to be 40 mA at a constant voltage of 3.3V, we determined that we could use the TLC59711 from Texas Instruments, which is a constant-current 12 channel LED driver. It can sink up to 60 mA and is capable of operating at 3.3V, which fits our desired specifications. We successfully designed a printed circuit board (PCB) around this chip and fabricated it on the OEDK's T-Tech milling machine. The fabricated design and the schematic for it are displayed below.

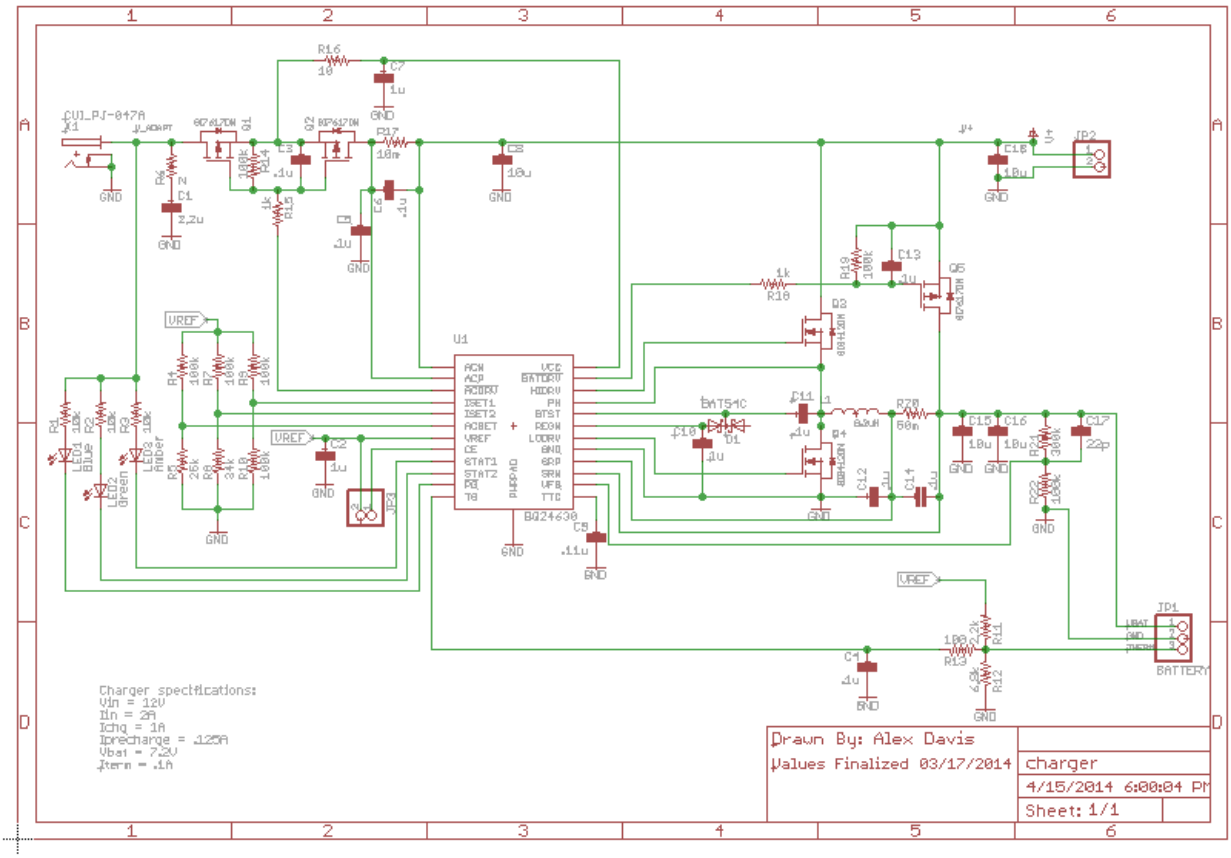


Figure 18: Schematic for haptic control board

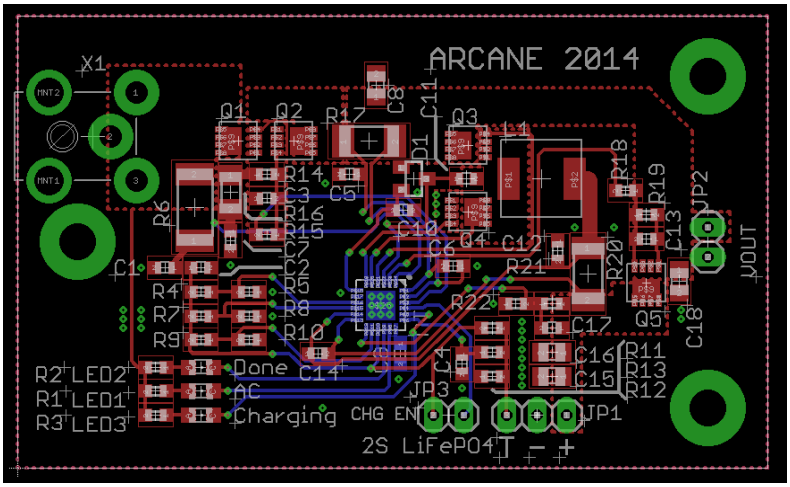


Figure 19: CAD drawing of layout for haptic control board

The only drawback to our use of the TLC59711 is its somewhat complex register structure. Since the chip is designed for use in LED displays and RGB lighting applications, it provides per-output PWM control as well as a global Red, Green, and Blue current scale. A handful of register bits control operation of the device, including PWM clock source (internal or external), pattern repeat, display blanking, and others. A complete map of the 218 bits of registers is included in the Appendix. The first 12 16-bit registers control PWM output on each of the 12 output channels. The following 3 7-bit registers control maximum current available to each of the red, green, and blue groups. The remaining 5 bits control the internal operation.

The TLC59711 sends this register data over an SPI-like 2-wire data bus consisting of a clock and data line. The device latches data on the clock signal's rising edge and may be operated at clock frequencies up to 10MHz. For transmitted data to be latched into the device's internal registers, an initial 6-bit write command must be sent ahead of the MSBs of the transmitted data.

To relieve some complexity from the code running on the stereo vision processor, as well as to make our system maximally platform-agnostic, code to control the TLC59711 was developed for the Parallax Propeller microcontroller. The Propeller was selected due to its processing power, memory, and prior experience with its use. In order to maximize compatibility across platforms, a TTY-like terminal was implemented on the Propeller, connected over a USB to serial converter to a controlling device (e.g. a PC running terminal software).

Terminal input is expected to be a single line (ending with a newline) containing the command and up to two arguments separated by spaces. An example would be “s 65535 1”. This command is to set (s) channel 1 of the TLC59711 to level 65535 (full duty cycle).

Input to the terminal is parsed into three separate strings by replacing spaces within the string with null (0) characters and noting the substring start addresses. So, for the previous example, “s 65535 1\n” becomes “s”, “65535” and “1\n” with \n denoting the newline character sent by the terminal. The last two of these strings are converted to integers using the C string library’s atoi conversion function. At present, the first string containing the command is assumed to be a single character long and is simply passed on to a switch-case statement to handle various command types, at which point the two (or fewer) arguments are handled appropriately.

As of the present, only the set (s) command is fully functional. Set must be provided with two arguments: the desired level from 0 to 65535 (2^16-1) and the channel on the TLC59711 to affect from 0 to 11. This will be extended to support a single-argument version of set which will affect all channels simultaneously. Upon receiving and parsing the command string, data to be sent to the TLC59711 is created by modifying the selected channel’s intensity value based on the provided arguments. All other data is unmodified. This new data is then transmitted (along with existing unmodified channel states and configuration data) MSB first at around 10 kbps.

The serial output protocol itself is bit-banged by a simple loop which shifts each successive bit of data out on the data line, waits for about 50us, pulls the clock high, waits for about 50us, then repeats the cycle with the next MSB to be transmitted. After transmitting the entire 224-bit register data and write command to the TLC59711, the microcontroller stalls to wait for input on the terminal once again.

This code has been tested to be functional both for testing the haptic feedback system by manually entering values from a PC and for running the system with input from the stereo vision system.

Form Factor

The first and most important consideration of the form factor is the placement of the haptic feedback. When considering where to place the haptic feedback, the sensitivity, the amount of surface area, and the accessibility of the location were all factors.

We considered the back: with a large surface area, the back had the potential to be a great option. Unfortunately, the back is not terribly sensitive; the necessary spacing of two inches or more severely limit the large surface area. Additionally, because the haptic feedback needed to be directly against the skin, we decided that the back would not be a convenient place to take on and off haptic feedback. The large variation in back size would also require customizable haptic feedback application, which increases the cost of our device.

Another place we considered was the hand. Although the hands have relatively little surface area, they are very sensitive, allowing the haptic feedback to be placed closer together than in other areas of the body. The hands are very accessible as well, easy to take on and off the haptic feedback. Unfortunately, the hands are used quite a bit in day to day activities, like opening the door, carrying a bag, and even using the white cane. Because of this, we have decided to limit the amount of haptic feedback we place on the hands to perhaps one or two motors on the back of the hand.

In the end, we have decided to place our haptic feedback on the forearm. The forearm seemed to be the perfect balance between sensitivity, accessibility, and surface area. While a smaller surface area than the back, the forearm is more sensitive, allowing the haptic feedback to be placed closer together. Additionally, using the forearm leaves the hands free for the user to continue doing their day to day activities without interference. Taking the haptic feedback on and off of the forearm is relatively easy and mostly one size fits all.

The layout will be determined through a haptic feedback test we will conduct early in January. We have three possible layouts we will test. They are shown in the figure below. Testing procedures are described in the **Testing and Results** section.

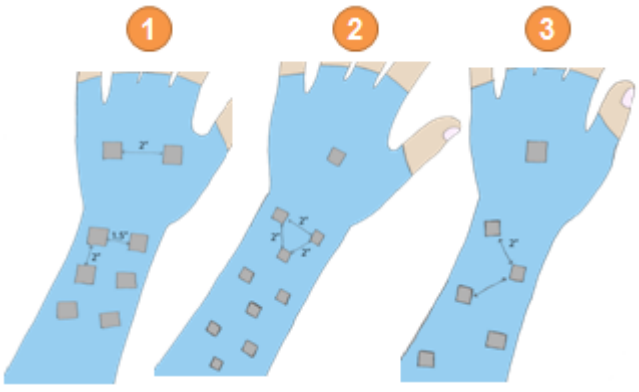


Figure 20: Potential layouts for haptic feedback

Regulatory Requirements

If our team wishes to eventually make our device a commercially available product, we will have to ensure that the system complies with all restrictions imposed on it. Besides those hazards and mitigations outlined previously, this section will explore the specific restrictions relevant to our design.

Restriction of Use of Hazardous Substances (RoHS)

To make our device a commercially available product, our first step is to comply with Restriction of Use of Hazardous Substances (RoHS) regulations, which limits or bans certain six toxic materials which are lead, cadmium, mercury, hexavated chromium, polybrominated biphenyl (PBB), and polybrominated diphenyl ether (PBDE) from new electronic devices.

The criterion we will have to pay the most attention to is the exclusion of lead from our product - being fully RoHS compliant will require that we do not use leaded solder when we manufacture our device and that we ensure that all components used in our device are RoHS compliant themselves. We currently ensure that the majority of our components are RoHS compliant by buying products from well-known companies with proven track records in compliance, such as Texas Instruments, Panasonic, and Parallax. In the future we will have to keep RoHS in mind when we choose a supplier for our PV batteries, professional PCBs, and motors.

International Organization for Standardization (ISO)

From an interface standpoint, we must take into consideration regulations set on assistive technology for the blind and vision-person. While there is no section that specifically touches upon the design of assistive technology such as the white cane, ISO 23559:2012 goes into tactile walking surface indicators - an important feature to understand for a successful design.

An important point that this document discusses is the importance of a simple, consistent layout that the user can easily understand. This would allow the user to navigate through environments that he has gone through the first time, in addition to those which he is extremely familiar with. Additionally, the device should be unobtrusive both for the user himself, and for other users in their surroundings. Finally, all electronics should be tightly enclosed in a casing so that the user may not directly interfere with it.

The rest of regulations and considerations made have been encompassed in the specific components of the overall design.

Final Design

Team ARCANE has successfully created an early prototype that serves as a supplement to the white cane by detecting upcoming obstacles that would not be identified otherwise. Simply put, two sensors are attached to the user, which will identify upcoming objects based on their distance and direction. This information is then translated into a depth map, which will serve as inputs for the haptic feedback strategically placed on the forearm of the user and will vibrate

Sensor

Our final design used two RGB cameras to estimate distance through triangulation. We used an Intel NUC as the main processor to generate depth maps in four steps: lens distortion, image rectification, stereo correspondence, and distance estimation through triangulation. The resulting depth data was then sent over to the haptics drivers via serial protocol over Bluetooth.

Description:

The Intel NUC D54250WYK is a portable, lightweight Intel Core i5-4250U based development platform. We used it for all stages of image processing and communication with the haptics drivers. The Logitech C310s were selected as the main sensors for the system. They were attached to a fixed platform to lock the relative orientation and distance between the two cameras. The C310s required no external power source and were connected directly to the

NUC through two USB ports. The Intel NUC was powered by the main battery of our system.

Leveraging OpenCV, an open source computer vision library, our Linux running processor first represents raw image data as matrices. Then, lens distortion is removed by performing a transform on the image. The actual transformation is calculated by calibrating the cameras using an object of fixed geometry; a “8 x 6” chessboard was used for this purpose. The OpenCV calibration algorithm locates the corners of the chessboard and uses their locations to fix the distortion and calculate the focal length of each camera. This data is furthermore used to rectify the images; the translational and rotational relations between the two cameras are calculated from the same calibration data. The resulting left and right images are thus row aligned and rectified, an important step in identifying the correspondences between points in each image.

These rectified images are then used to calculate the correspondence map via semi-global block-matching algorithm. This method was selected due its relatively light and parallelizable computational load, an important factor in attaining a real-time algorithm. The block-matching algorithm divides the two images into same number of blocks. For each block, the sum of absolute differences (SAD) was calculated. Using the SAD value, for each block on the left image, the corresponding block was found by finding the block with the best matching SAD along the epipolar line to the left of the the block's coordinate. To make the estimation more accurate, Mutual Information cost calculation was added to make the algorithm less sensitive to recording and illumination changes. In addition, constraints such as smoothness were added to penalize disparity changes and noise. The final costs were then used to select the corresponding blocks. The differences in pixels between each corresponding blocks form a disparity map.

This disparity map is used to calculate a depth map; using the disparity values in pixels (x_l - x_r), calculated focal length in pixels (f), baseline distance between the two cameras (T), each depth point was estimated using the equation presented in Figure 5. In **Figure 22**, the top portion is a pair of rectified images from the result of calibration. The green lines are selected epipolar lines used for the purpose of demonstration. On the lower right corner is a normalized depth map, with darker colors corresponding to lower distance values (black = 0). On the lower right is the rectified left image with nine overlaid sampling points.

Each of the nine samples is labeled by the average of multiple depth points surrounding it. The mapping between calculated and actual distances was completed by comparing measurements to calculated values. A best fit line was then used to model the relationship (as shown in **Figure 25**) and the gradient and intercepts of this line were used to calibrate the depth output.

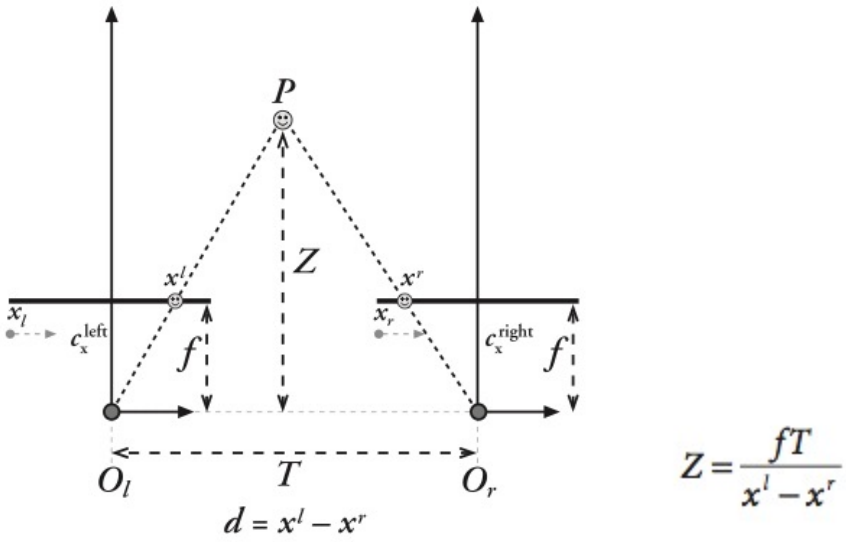


Figure 21: Distance estimation using similar triangles

Figure 22: Stereovision implementation

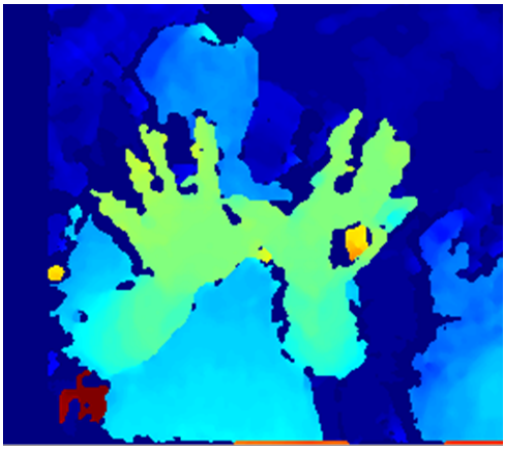


Figure 23: Color depth map

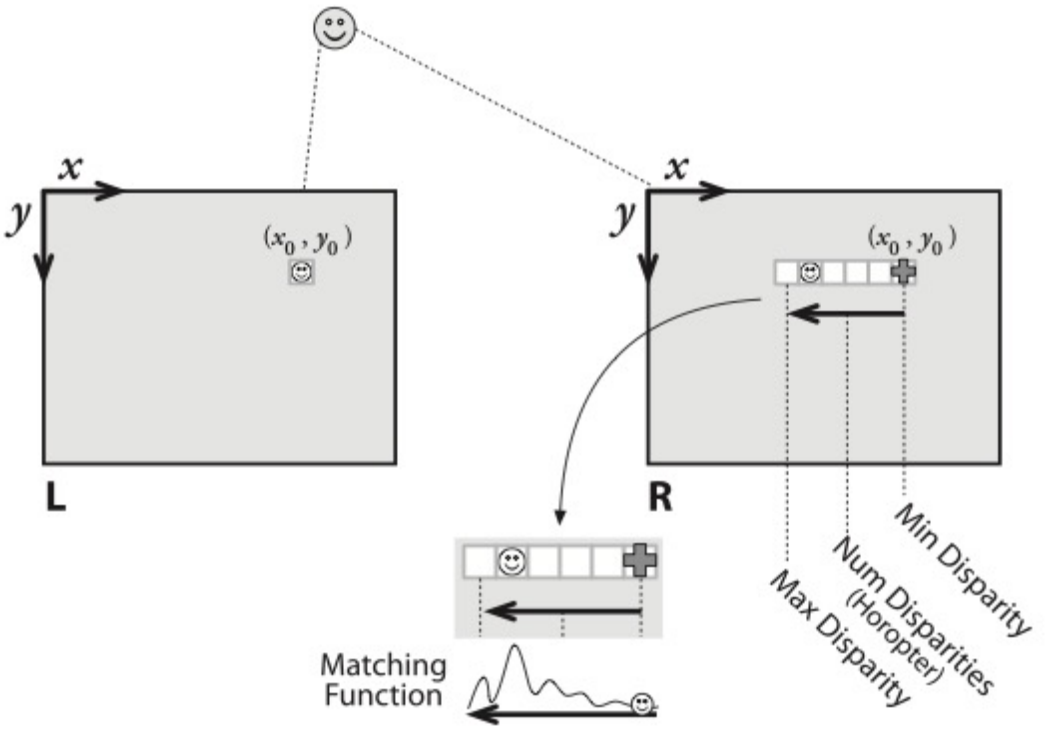


Figure 24: Block-matching algorithm for finding corresponding features

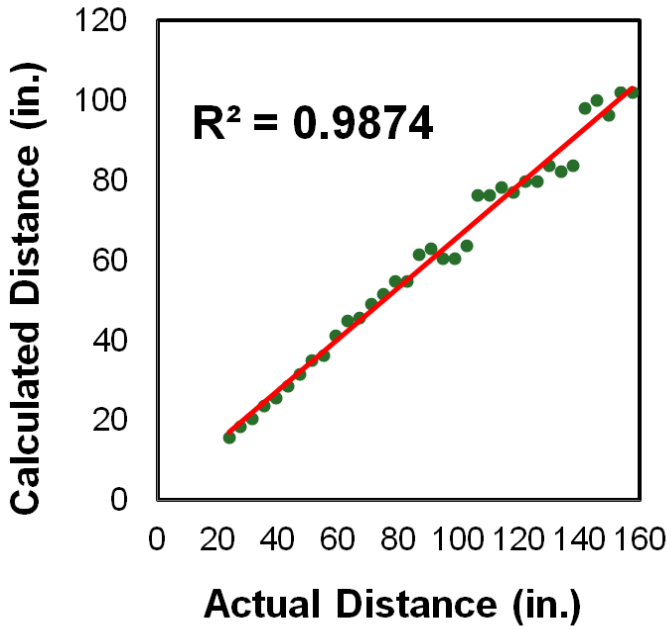


Figure 25: Best fit line modeling the relationship between the actual distance and the calculated distance

Camera Mount

While several locations were considered for the placement of the sensors, the final placement will be on the head of the user. The head was determine to be one of the body parts that moved the least while walking, and image stability is crucial for extracting depth maps from the user's environment.

The cameras are placed in a weather sealed case, which has two holes in the front to expose the lenses, and two in the back for wiring. This is then attached by Velcro onto the brim of a hat, which is worn by the user. It was determined that the weight of the camera case alone did not obstruct with the structure of the hat, and could be worn like any other cap. This allows for maximum control of the cameras, and does not obstruct with the user's daily tasks.

A CAD model of the camera mount can be found in **Figure 26**:

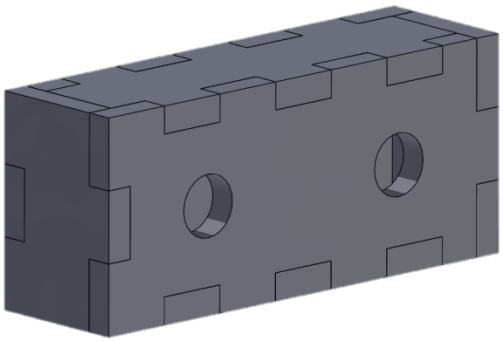


Figure 26: Casing for the two cameras

Haptic Feedback

The goal of the haptic feedback portion of our design is to receive commands from the sensor subsystem's more powerful processor, converting and relaying them to the various vibrating motors. The motors are arranged in a 2x3 grid on top of the user's forearm. There is a specific motor numbering scheme which corresponds to the block matching points used in the depth map algorithm.

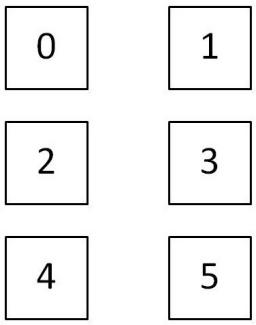


Figure 27: Top down view of the motor scheme

The haptic subsystem consists of four major components: the vibrating motors which provide the feedback, a motor driver which provides these with power, a compass module for navigation, and a microcontroller to handle communications between the various components.

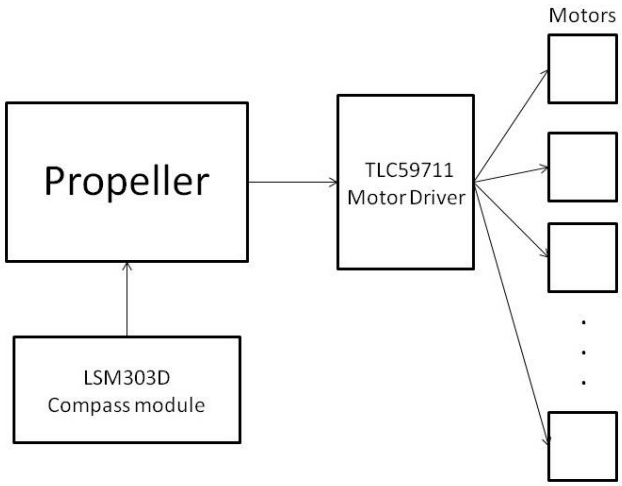


Figure 28: Components of haptic subsystem

Motor Driver:

The motor driver for our system is the TLC59711 from Texas Instruments. Its intended function is as an LED driver, but our motors have low enough current draw and voltage requirements that the TLC59711 is more than adequate. At 3.3 V, each pager motor requires only 40 mA of current to operate at a 100 percent duty cycle. Each channel on the motor driver can drive a current of up to 60 mA at 3.3V.

The only drawback to our use of the TLC59711 is its somewhat complex register structure. Since the chip is designed for use in LED displays and RGB lighting applications, it provides per-output PWM control as well as a global Red, Green, and Blue current scale. A handful of register bits control operation of the device, including PWM clock source (internal or external), pattern repeat, display blanking, and others. The first 12 16-bit registers control PWM output on each of the 12 output channels. The following 3 7-bit registers control maximum current available to each of the red, green, and blue groups. The remaining 5 bits control internal operation of the chip.

The TLC59711 receives this register data over an SPI-like 2-wire data bus consisting of a clock and data line. The device latches data on the clock signal's rising edge and may be operated at clock frequencies up to 10MHz. For transmitted data to be latched into the device's internal registers, an initial 6-bit write command must be sent by the Propeller ahead of the MSBs of the transmitted data.

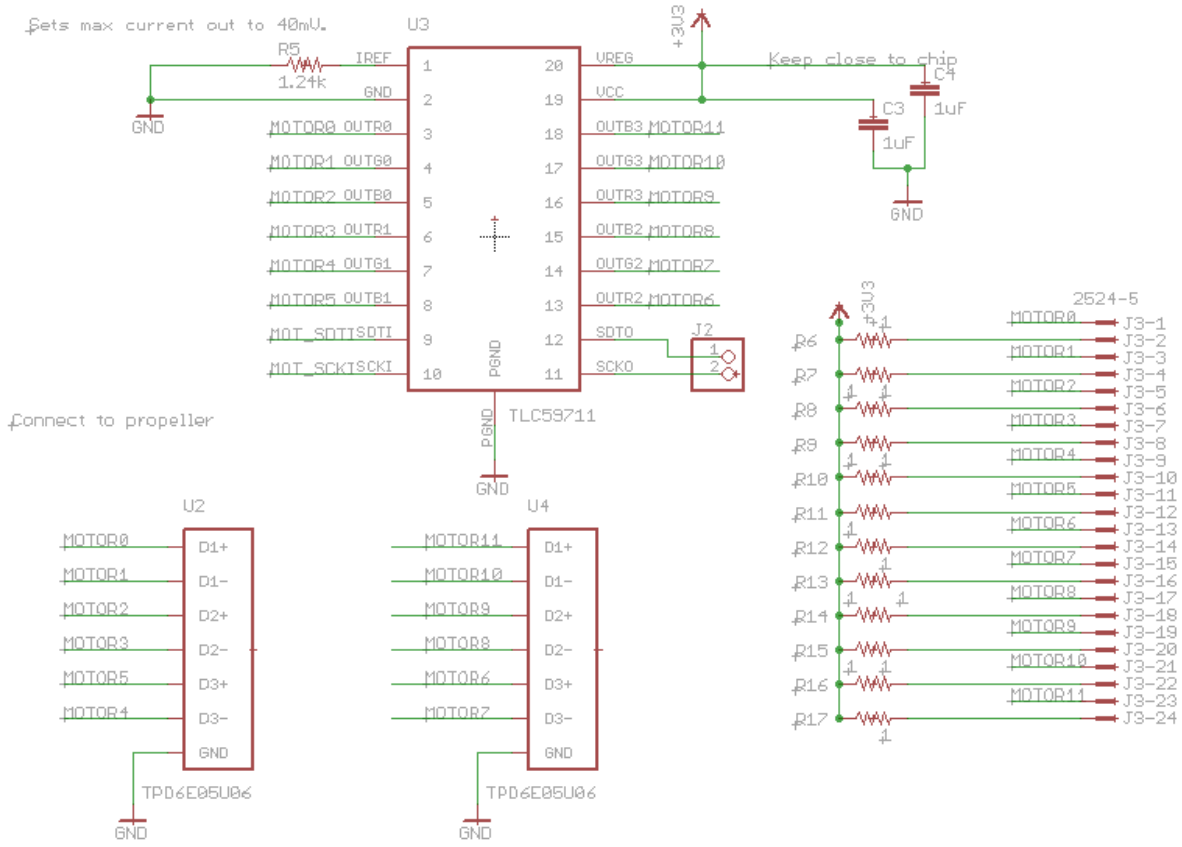


Figure 29: Schematic of motor driver

Processor:

To relieve some complexity from the code running on the stereo vision processor, as well as to make our system maximally platform-agnostic, C code to control the TLC59711 was developed for the Parallax Propeller microcontroller. The Propeller was selected due to its processing power, memory, and team members' prior experience with its use.

While Parallax does not provide native support for C code on the Propeller, there are a few open source compilers which do, Ross Higson's Catalina project and gcc being two examples. For this project, we chose Catalina due to its ease of use, integration with a full-featured IDE (CodeBlocks) generally good documentation, and compatibility with the Windows operating system.

Multicore Architecture:

At present, we are utilizing 7 of the Propeller's 8 cores to run code. At startup, a main function is launched in core 0. This in turn launches C code to run in four other cores before entering a continuous loop. Within this loop is code which facilitates inter-core communication between the other running cores.

The first core launched by the main function is a terminal driver described in greater detail later on. This driver receives characters from a separate assembly-language RS232 driver running in another core, allowing the device to be controlled by a separate computer. The terminal code also provides a mutually-exclusive printf implementation allowing other cores to print debug information without contention.

The next core to be launched runs the compass code. This loops, checking a messaging variable which enables or disables compass updates. If compass updates are required, the compass code will compute the current heading, placing it into a second messaging variable to be read by the main loop. The compass code relies heavily upon floating-point math, so an additional core is launched to run an assembly-language floating-point math engine to accelerate computation. Additional information about the compass code is provided later in this document.

A sixth core is launched to run the motor driver code. The core of this code simply bit-bangs the SPI-like protocol of the TLC59711. This is done by a simple loop which shifts each successive bit of data out on the data line, waits for about 50us, pulls the clock high, waits for about 50us, then repeats the cycle with the next MSB to be transmitted, giving an effective data rate of 10kbps. The driver polls a status variable in memory, waiting for an update. When an update is indicated, the driver transmits the new values to the TLC59711.

The final core simply polls the front-panel buttons, indicating their state to the main loop. Dedicating a core to this task was necessary due to the Propeller’s lack of hardware interrupts.

The main loop polls the terminal for updates. If updated data is present, it is copied locally. If the user has pressed the compass button, the compass activates for approximately 10 seconds, causing the main loop to merge both terminal-set motor levels and compass data before transmitting the updates to the TLC59711. Wherever possible, wait statements are used to reduce power consumption of cores, giving a power reduction over a naive busy-waiting implementation.

Terminal:

In order to maximize compatibility across platforms, a TTY-like terminal was implemented on the Propeller, connected over a USB to serial converter to a controlling device (e.g. a PC running terminal software).

Terminal input is expected to be a single line (ending with a carriage return character) containing the command and up to two arguments separated by spaces. An example would be “s 65535 1”. This command is to set (s) channel 1 of the TLC59711 to level 65535 (full duty cycle).

Input to the terminal is parsed into three separate strings by replacing spaces within the string with null (0) characters and noting the substring start addresses. So, for the previous example, “s 65535 1\r” becomes the strings “s”, “65535” and “1\r” with \r denoting the carriage return character sent by the terminal. The last two of these strings are converted to integers using the C string library’s atoi conversion function. At present, the first string containing the

command is assumed to be a single character long and is simply passed on to a switch-case statement to handle various command types, at which point the two (or fewer) arguments are handled appropriately.

At present, only three commands are supported, as follows:

Set- Usage: “s <intensity> [<channel>]”. Set can take one or two arguments. If followed by a single numeric argument, it sets all channels on the motor driver to the specified level (“s 0” will turn all motors off, for instance). If provided with two arguments, the command will alter the intensity of the channel specified by the second argument to the value specified by the first argument.

Mode - Usage: “m <live/blind>”. Mode allows the user to enable or disable the command line interface provided in the code. By typing “m live” feedback on commands will be provided. For example, when in live mode, typing “s 32767” results in the Propeller echoing back “All channels set to level 32767”. In cases when command-line feedback is undesirable (when the haptic

controller is connected to the sensor system, for example), typing “m blind” will suppress these statements.

Two commands, “h” and “?” are provided to display a help message documenting all of these commands. Both of these override the mode set by the “m” command, printing an output regardless of whether command-line mode is enabled.

Upon receiving and parsing the command string, data to be sent to the TLC59711 is created by modifying the selected channel’s intensity value based on the provided arguments. All other data is left unmodified. This new data is then transmitted (along with existing unmodified channel states and configuration data) MSB first at around 10 kbps.

Vibratory Motor Coupons:

The coupon boards provide a mounting point for our vibrating pancake motors, providing strain relief for their fine (~28ga) wire leads. Each board is about an inch long and half an inch wide and has four components on it: the pager motor itself, a schottky diode, a .1uF ceramic capacitor, and a right-angle header. The diode is reverse-biased across the motor's power leads and is intended to dissipate any inductive spikes caused by the motor commutating. The capacitor helps with this as well, smoothing out fast transients and switching noise which would otherwise cause problems for the motor driver. A right angle header provides a quick disconnect point for prototyping while keeping the coupon's overall profile low.

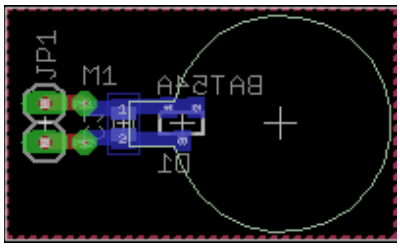


Figure 30: A coupon board

Compass Operation

The compass module we are using is the actually a combination of an accelerometer and magnetometer in a single chip, the LSM303D designed by ST Microelectronics. This combination can be used to create a tilt-compensated compass, which allows us to give the user accurate compass heading information while they are wearing the device, regardless of how their arm is positioned.

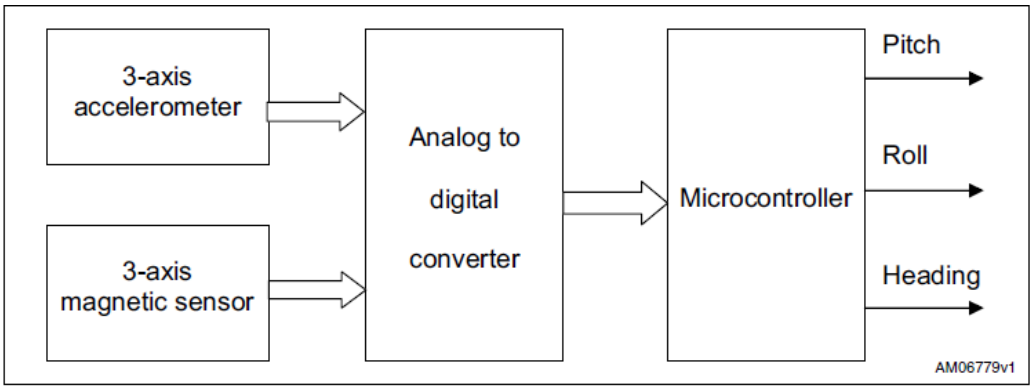


Figure 31: An overview of getting a compass reading from the LSM303D

The LSM303D already contains an analog to digital converter, so we can simply transmit data from the chip to the Propeller. We chose to use the standard 4-wire SPI protocol due to the ease of implementing it in software, its stability, and the fact that we have a plethora of free pins on the Propeller. The LSM303D is an SPI slave with a fairly simple communication scheme. SPI transmissions are 16 clock pulses long, with each pulse corresponding to a bit, and follow a set pattern. The first half of each transmission occurs over the master-out slave-in (MOSI) pin and is consistent for both reads and writes: The first bit indicates whether a read or write is taking place, the second is used to determine if the address should be incremented in multiple read/ write commands or not, and the next six bits are used to indicate which register the operation is accessing. If the operation is a write, the next eight bits will be written by the Propeller to the LSM303D over MOSI, and if it is a read the compass will send the next eight bits over the master-in slave-out pin (MISO) to be received by the Propeller. A diagram of the overall protocol

is below, with the MISO line being referred to as SDO and MOSI being called SDI. The CS pin is held low to indicate that a transmission is in progress.

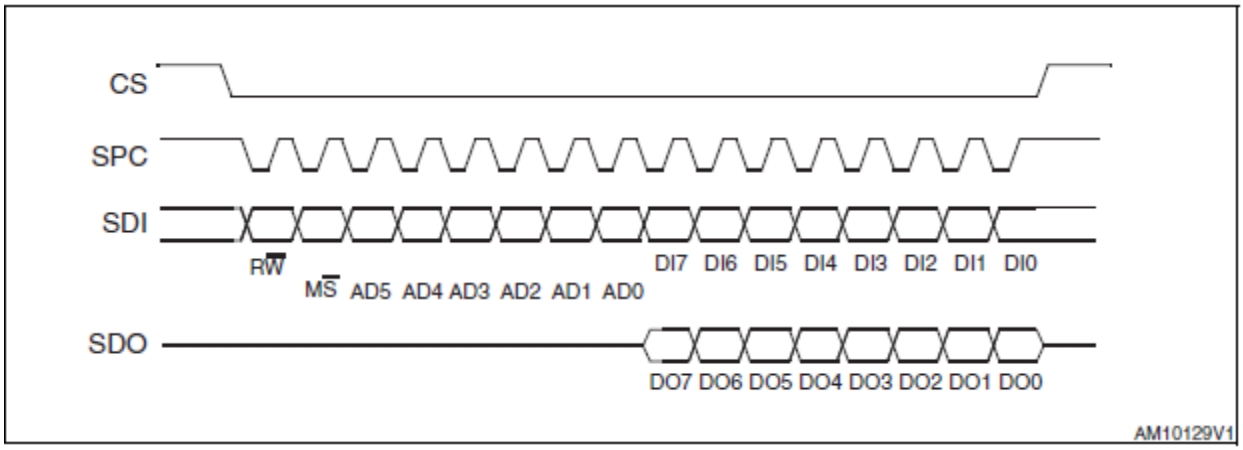


Figure 32: Four line SPI on the LSM303D

After obtaining raw accelerometer and magnetometer data in the X, Y, and Z directions, the Propeller has to transform them into useful readings for pitch, roll, and heading, the definitions of which can be seen in the diagram below. Its equations can be found in the **Appendix:**

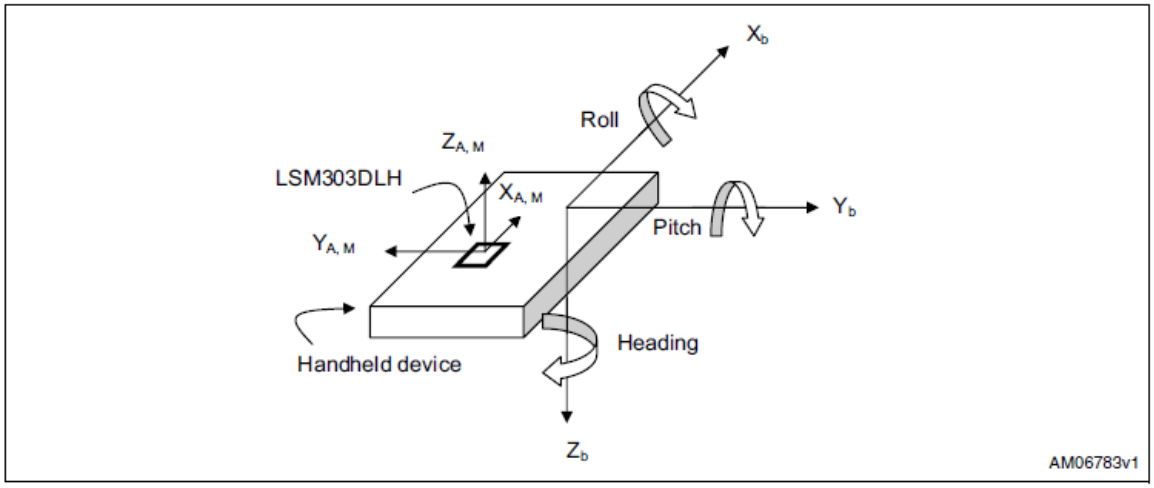


Figure 33: Coordinate system for compass readings from LSM303D

Integrated Haptic System

In order to make our product wearable, we combined many of the individual components of the haptic feedback system into a single custom printed circuit board.

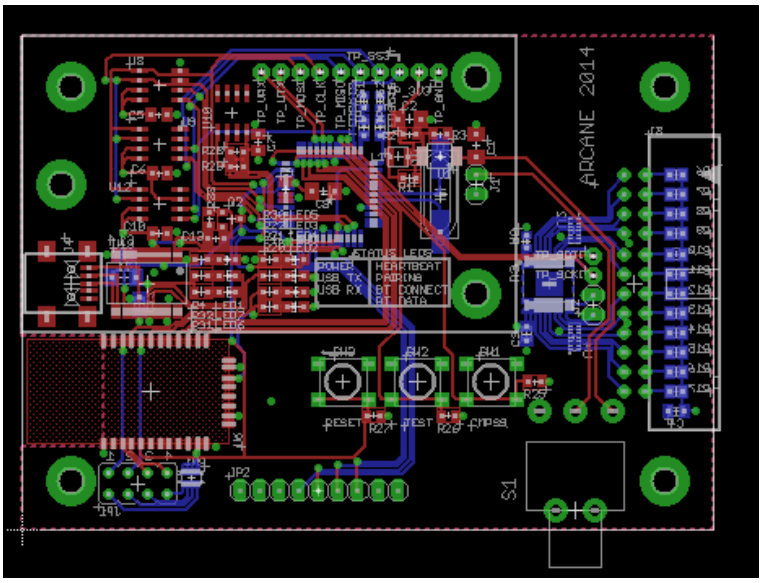


Figure 34: Integrated board

The Propeller and TLC59711 are both included on this board, and there is a 9-pin header for the pre-assembled compass module. The Propeller has the required 128 Kbit EEPROM for program storage, and we also include two 256 Kbit SRAM chips and a 4 Mbit Flash memory chip in order to provide additional storage should it become necessary. The SRAM and flash memory chips are addressed using 4-wire SPI protocol like the compass, and the EEPROM is addressed using I2C.

Our custom PCB also gives up two options to communicate serially with the NUC used by the sensor side of the system - over a USB cable or via Bluetooth. An FT232RL USB-to-serial converter sits between a USB mini B port and the Propeller, which allows for serial communications between the Propeller and a given computer over USB. An RN-42 Bluetooth module contains the entire Bluetooth stack within it, meaning that the Propeller can interface with it the exact same way it does with the FTDI chip - the only change we have to make to use Bluetooth is changing the pins used for serial communication. The code currently supports using

only one of the serial communication methods at a time, and the decision about which to use must be made when compiling the embedded code for the Propeller.

Local power management for the haptic PCB is provided by a TPS62160 buck converter from Texas Instruments. We chose this particular IC because it can accept an input voltage in the range of 3 - 17 V, which means our battery voltage of 6.4V falls well within spec, and because it can output up to 1A of current, which is suitable for our system's needs. The motors are the most current-demanding aspect of our project, and they only demand 40 mA of current each, or 240 mA total. Each cog in the Propeller typically draws currents on the order of uA, and the other ICs in our system typically draw tens of mA at maximum. For example the FT232RL draws 15 mA in normal operation, the RN-42 draws 3 mA when connected and 30 mA when transmitting, and the TLC59711 draws under 32 mA. We set the TPS62160 to output a constant 3.3V.

Battery Management

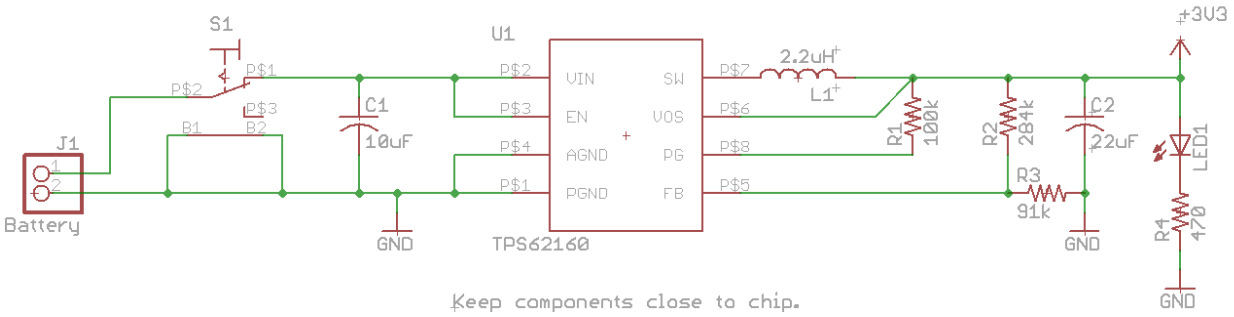


Figure 35: Power management schematic

To simplify use of the device and prevent the user from having to change batteries on-the-go, an integrated lithium-iron-phosphate battery charger was designed. Lithium phosphate batteries were selected primarily for safety reasons, as their less-volatile chemistry is more tolerant of abuse than the slightly more compact and more energy-dense lithium-polymer chemistry.

To give ample runtime without causing the 3.3V regulator on the integrated haptic board to enter a brownout condition, a 2-cell (6.4V working voltage) Lithium Phosphate battery was chosen. This pack is capable of supplying 1200mAh at a peak discharge rate of 4.2A and is internally protected against overcharging (Vbat > 7.6V), over-discharging (Vbat < 4.46V), and short circuits. Even at our worst-case peak current demand of 500mA, a 1200mAh battery should last a healthy 2.4 hours. Under real-world usage, this can easily stretch much farther.

The core of the battery charger is the Texas Instruments bq24630 stand-alone synchronous switch-mode lithium phosphate battery charger IC. The bq24360 was chosen due to its support for lithium phosphate packs of up to 4 series cells with no need for supervisory circuitry or microcontroller monitoring. This allowed us to design a fully independent battery management board, segregating our design into two easily debuggable sections. The only real drawback to the bq24630 is its lack of internal MOSFETs, instead requiring 3 discrete P-channel SI7617DN MOSFETs and 2 discrete N-channel SIS412DN devices. While these are bulky, they do support ~35A continuous drain currents, adding a short-circuit safety margin to our design.

The bq24630 is essentially a buck-boost regulator with sensitive current monitoring. Typical battery charging for lithium phosphate batteries consists of two phases: an initial constant-current phase and a final constant-voltage phase. During the constant-current phase, pack voltage is monitored. Once the pack voltage reaches its final value, the charger enters the

second phase, maintaining a constant voltage across the pack and monitoring the charging current. Once this current drop below a set threshold, the battery is considered charged and the charge cycle is terminated. In our application, the constant-current phase is limited to 1A, the maximum pack voltage is set to be 7.2V, and charging is terminated once charge current drops below 100mA. These values were selected to be well within safety margins of our chosen battery pack. During the entire charging process, pack temperature is monitored via an attached thermistor. If the pack reaches an unsafe temperature, charging is terminated to prevent fire or pack damage.

The bq24630 is able to switch attached loads between wall power and battery. When an external power supply is connected, the bq24630 disconnects the battery before switching to external power. This break-before-make switching prevents inadvertent damage to the battery from excessive shoot-through current and is mirrored when the charger is disconnected from external power (i.e. MOSFET for external power is turned off before battery MOSFET is turned on). The bq24630 is also able to limit current draw from the external supply, throttling its own charging current if necessary to do so. Our charger is designed to operate from a 12V 2A wall-wart type supply, so a 2A limit was set on the bq24630 to remain within the wall-wart's capabilities.

PCB Case

To house the PCB boards and battery, Team ARCANE has created a 3D printed PCB case which will be worn on the bicep of the user. This case was 3-D printed to maximize the precision of our design, since we are dealing with a PCB that requires a near perfect fit. The case is divided into two compartments – one for holding the PCB into place, and another for isolating the heat of the battery. Additionally, there are four holds for components of the PCB: the connector, power switch, USB port, and DC port.

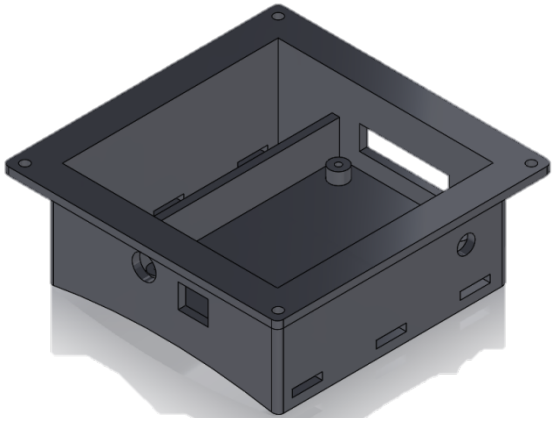


Figure 36: PCB Case

Arm Sleeve

The arm sleeve is made of flannel, to minimize coupling between motors, and Velcro, to make it adjustable to all arm sizes and levels of comfort as well as to allow the easy removal and replacement of pager motors if necessary. Additionally, the soft material allows for the vibrations of the individual motors to be felt in specific parts of the forearm. This is unlike a rigid sleeve, which would make the entire forearm vibrate with a slight vibration in one specific area. The layout for the sensors was determined through thorough testing, as described in the **Testing and Results** section. A picture of the arm sleeve can be found in **Figure 37:**

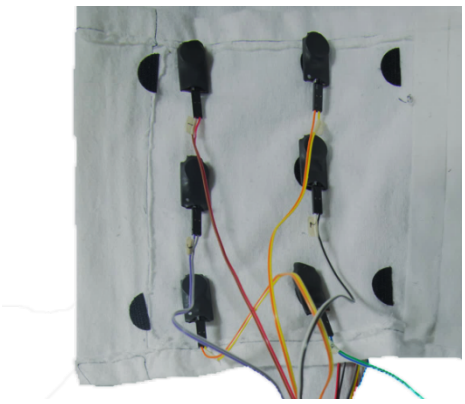


Figure 37: Arm sleeve with a 3x2 motor layout

Cost

The current cost of our device is $1,614.36. The most expensive components in our device are the NUC, the custom PCBs, and the parts for the custom PCBS. The NUC comes in at $379.99, with accessories such as an SD card reader, RAM, a flash drive, Bluetooth + WiFi chip, a battery and battery charger adding an additional $225.79. The custom PCBs run about $421 for 3 each of the motor driver board, battery charging board and compass module. The parts necessary to build these boards add another $442.21. Other miscellaneous parts make up the difference. Unfortunately it is difficult to determine how much mass production would lower the cost for these components in order to find the final cost of a commercialized device.

Testing & Results

Sensor Testing

For our sensor testing, we need to ensure that we can accurately translate the environment into a depth map. Then that depth map needs to be converted into inputs to the haptic feedback system of our design. To that end, we will test our depth map generation in a number of different environments to ensure that they are portrayed correctly in the depth map. For each of these different environments, we will change one thing after the initial depth map, to ensure that the depth map can detect changes when most things in the environment stay constant. Further, to ensure that the sensor correctly identifies objects at all the sizes and distances we set out in our design criteria, we will place objects of these sizes and at these distances in all of our testing environments. This means that we must include obstacles as low as a foot off the ground and as high as seven feet off and obstacles as far away as twelve feet.

We tested our sensors by shrinking a piece of brown, square packaging paper down an inch at a time to see if the sensors would still identify it as a distinct object.

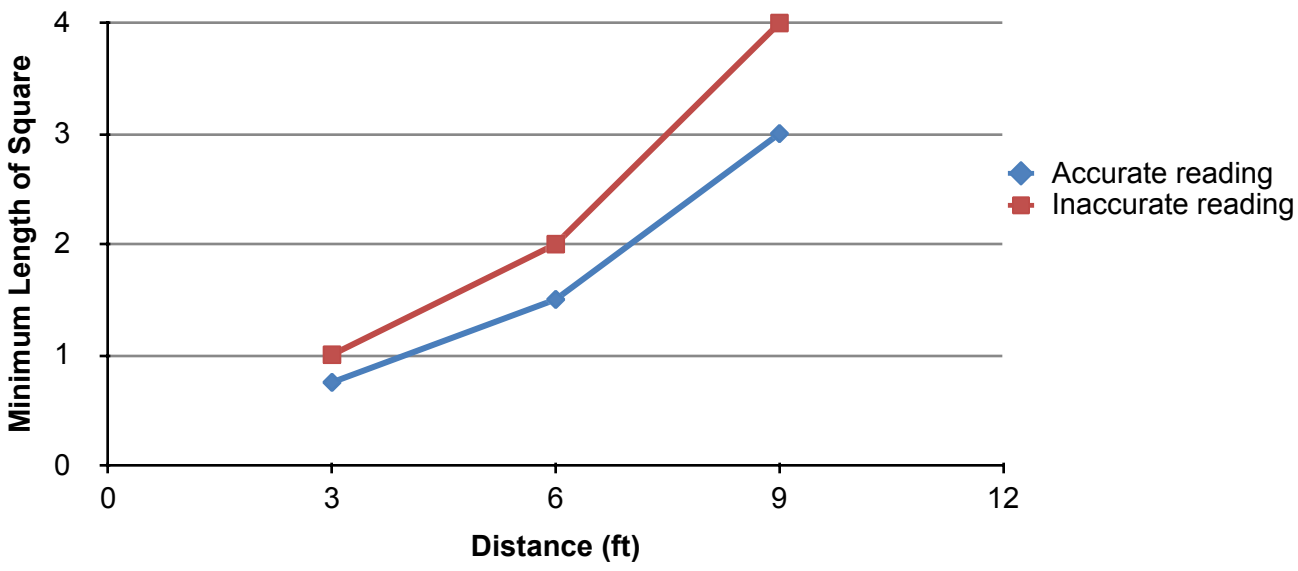


Figure 38: Minimum length of square with reading versus distance from sensors

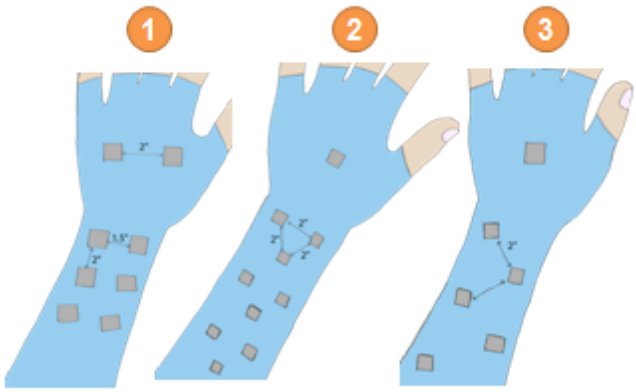
With a block size of 3x3 pixels giving a resolution of 160 x 120 pixels, the farther the object is from the sensors, the smaller the area of the object. Only when the object is very large, or very close to the 3x3 pixel size is the distance accurate. Anything smaller causes the value to change, but with the cost function and other noise making the data point inaccurate in finding the corresponding distance.

Haptic Testing:

There are two main focuses for the haptic testing. Firstly, we need to determine the best layout for our haptic vibration on both or only one of the hands or arm. We need to ensure that each vibration is felt as distinct from the others and that the intensity at each location is comfortable and safe for the user. Secondly, we need to ensure that the vibrations we feed back to the user can be interpreted into meaningful information about the environment. For example, the vibrations corresponding to a tree branch need to be understood as indicating a tree branch instead of a wall or some other obstacle.

“Virtual” Obstacle Course

Figure 39: Three unique layouts for haptic feedback



To ensure that we have successfully accomplished these two goals, we tested a number of layouts and run the user through a virtual obstacle course. Because there will be a learning curve for each layout, we will run each virtual obstacle course multiple times in order to get an accurate read on how intuitive the layout really is. We will average the percentage of obstacles correctly identified for the last three of five obstacle courses for each layout and compare the accuracy count and learning curve for each layout in order to determine our best layout option.

In order to meet the design criteria we set for ourselves in the middle of the semester, our virtual obstacle course must include the haptic feedback cues for an obstacle to the left, to the right, overhead, down low, and moving closer. To ensure that our test subjects are learning the haptic cues for each layout instead of memorizing the order of the virtual obstacles, we will vary the order of the obstacles in the virtual obstacle course. If at any point the vibrations become uncomfortable or indistinguishable, we know that we need to scale back the intensity of the vibrations or that the layout is unusable.

As we began running our tests, we discovered that the three by three layout of pager motors (Layout #2) was an overwhelming amount of information and impossible to distinguish distinct motors' locations. The motors were too close to identify which one was vibrating, which further confused the mental image of the environment we were trying to create, especially when multiple motors were running at the same time. Keeping this in mind, we decided to rule Layout #2 out, and continued testing with Layouts #1 and #3.

From there, the purpose of the rest of the testing was to determine whether the first or the third layout would be clearer: whether it is easier to understand the haptic feedback when the right and the left are even with each other or offset from each other.

The results from the four users are as follows:

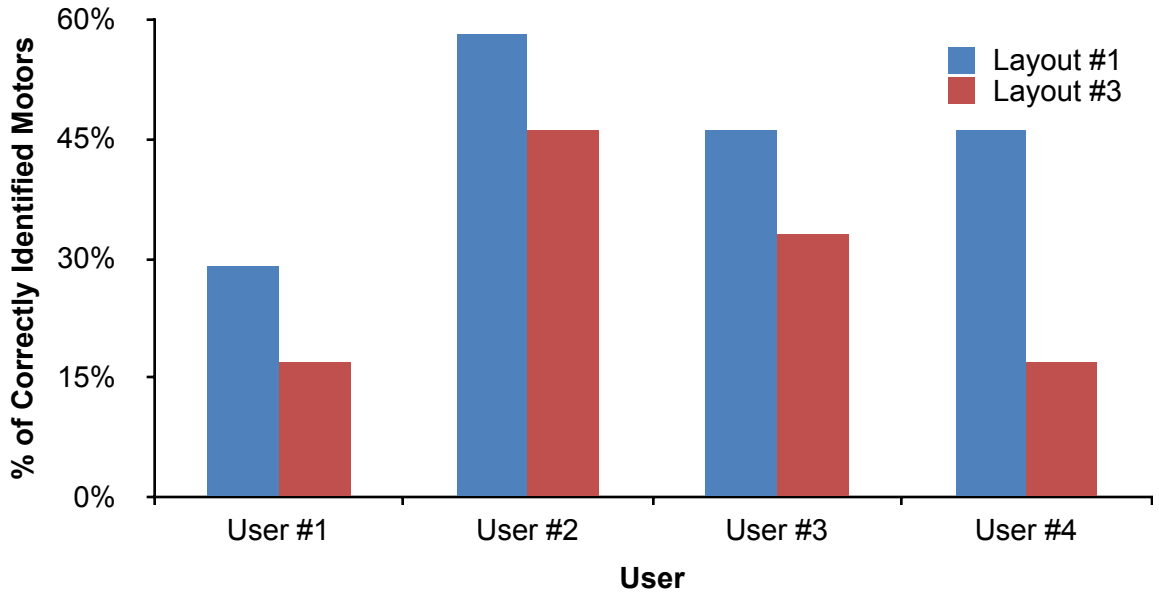


Figure 40: Comparison in pager motor differentiation between Layouts #1 and #3

From a comparison of the average success rates, Layout 1's accuracy rate of 45% was much better than Layout 3's 28%. From this preliminary testing, it appears that Layout 1 is a better choice for our purposes.

Integrated Prototype Testing

When we completed our integrated prototype, we began testing on blind-folded individuals. The individual was put into the arm sleeve according to how tight was most comfortable for them while the sensors were left attached to a monitor so we could see which of the 6 regions detected an obstacle. Then, before the individual was blindfolded, we explained the layout of the motors: which would indicate an obstacle to the right vs. left, overhead, waist-height or ground-level. We explained that only one would be on at a time, so if more than one was on at a time, it was our mistake and not to bother trying to figure out which motors were vibrating. Additionally, we reassured test subjects that the inability to identify the motor vibrating or a poor score was not a reflection on them, but on us and our ability to communicate information to them as a user. The individual was then blindfolded and the testing began. The only training our test subjects received was that we stimulated each of the six regions of the arm sleeve and told them which region was being triggered in order for them to feel where each of the six regions were. This only lasted around ~2 minutes.

We stimulated specific regions of the arm sleeve by holding a box lid in the region of the sensors that would trigger the specific region we wanted. We did each of the six regions four times for each test subject in a random order. The results were as follows:

		Motor Accuracy (Correct / Total)						Overall Accuracy
		Top Left	Top Right	Middle Left	Middle Right	Bottom Left	Bottom Right	
User #1	Correct	5	4	4	4	3	3	92%
	Total	5	4	4	4	4	4	
User #2	Correct	3	3	4	3	3	3	73%
	Total	4	5	4	4	5	4	
User #3	Correct	4	3	2	1	1	4	63%
	Total	4	4	4	4	4	4	
User #4	Correct	4	2	4	1	3	1	60%
	Total	4	4	4	4	4	5	
User #5	Correct	4	4	3	3	1	4	79%
	Total	4	4	4	4	4	4	
User #6	Correct	3	4	3	2	4	4	83%
	Total	4	4	4	4	4	4	
User #7	Correct	3	5	5	2	4	4	92%
	Total	3	5	5	4	4	4	

In order to interpret these results more easily, we can also look at an accuracy map:

Figure 41: Accuracy of vibration layout by user

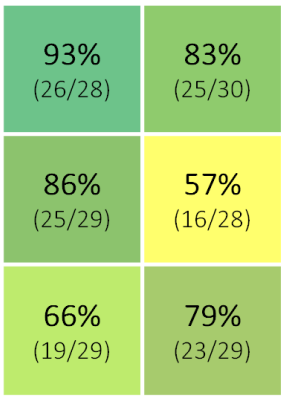


Figure 42: Percent accuracy of vibration layout

Our vibration layout resulted in a 78% success rate of correctly identifying the location of an object in front of the user. Many users commented that it became easier to identify the motor vibrating over time, as they became more familiar with the device. A few commented specifically that it was more difficult to identify the motor vibrating when it had been vibrating for a long time as they felt that they got used to the vibrations and didn't notice them anymore.

The sensor and computer processing has a battery life of about 5 hours, more than enough to sustain a user throughout their daily movement. We tested the battery life of the sensor and computer processing by timing how long it took for the battery to die; thus we have a very accurate measurement of its battery life. Unfortunately, it was not possible to test the haptic's battery in this manner. However, we measured the average power load of the system at 0.88W. The battery has a 7.2Wh lifespan, which we calculate to give us about an 8 hour expected battery life. While we have not been able to actually test the battery life of the haptic system, we believe that the haptic battery life is enough more than the sensor and computer processing battery life that the sensor and computer processing battery life will always be the limiting factor.

Summary & Recommendations

As of 4/20/2014, the ARCANE is a fully functional prototype.

The current status of our design:

- Sensors are fully functional and housed in an acrylic case placed on the brim of a hat.
- NUC is fully functional with our stereovision algorithm and housed in a backpack.
- The haptic controller is fully functional and housed in a 3D printed case fastened on the bicep of the user. Additionally, the haptic controller receives Bluetooth information from the NUC without error.
- The pager motors are fully functional and correctly receive information from the haptic controller.

Overall, our design accomplishes its major objectives. We are able to detect objects at a distance of 9 feet as well as at a range of heights. While our final product does not resemble what we originally believed it would look like, it is entirely functional and better suited to the needs of the visually impaired.

Things we might have done differently include:

- Using a different microcontroller than the Parallax Propeller, since it is more of a development board rather than a final product
- Implementing and experimenting with a different form factor, such as the hand
- Increasing the frequency of pulses of vibrations rather than intensifying vibrations to indicate distances

While the core of our design works, there are changes we would make in hindsight to make the work easier and the product more intuitive for visually impaired users. Many visually impaired users felt that a constant vibration became difficult to understand after a time, recommending instead that we change the frequency of pulses of vibrations instead of intensifying a constant vibration.

Future work on the ARCANE might include:

- Compass functionality-currently the compass does not work
- Object recognition- for emergency alerts about fast moving objects or stepping off a curb
- Use of wider angle cameras- wider field of view and more information
- Use of wireless cameras or a different form factor such that there are no cords between sensors and the NUC
- Use of lighter, more compact components

Despite our progress this past year, there is much left to do in order to make a fully fleshed out supplement. Many visually impaired users recommended object recognition for stepping off a curb and an alert to warn the user about a fast moving object such as a bicycle will increase the usefulness of our product. Additionally, our current prototype is in four parts that the user must put on separately. Future work might condense these four parts into fewer parts and make them lighter and more wearable. Other improvements would increase the durability, wearability, and usefulness of our product for visually impaired users.

References

[1] R. A. Jarvis, “A Perspective on Range Finding Techniques for Computer Vision,” IEEE Trans. Pattern Anal. Mach. Intell, vol. 5, no. 2, pp. 122-138, Mar. 1983. http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=4767365

[2] S. Birchfield and C. Tomasi, “Depth Discontinuities by Pixel-to-pixel Stereo,” Interational Journal of Computer Vision, vol. 35, no. 3, pp. 269-293, Aug. 1999. http://www.ces.clemson.edu/~stb/publications/p2p_ijcv1999.pdf

[3] D. Scharstein and R. Szeliski, “High-Accuracy Stereo Depth Maps Using Structured Light,” Microsoft Corp., Redmond, WA, 2003. http://research.microsoft.com/pubs/75606/scharstein-cvpr03.pdf

[4] A. Levin et al, “Image and Depth from Conventional Camera to Coded Aperture,” M.I.T. Comp. Sci. and Arti. Intell. Lab, Cambridge, MA, 2007. https://graphics.stanford.edu/courses/cs448a-08-spring/levin-coded-aperture-sig07.pdf

[5] M. Grosse, “Coded Aperture Projection,” ACM Transaction on Graphics. vol. 29, no. 3, pp. 1-19, 2010. http://web.media.mit.edu/~gordonw/publications/CodedApertureProjectionTOG.pdf

[6] L. Jones et al. Vibrotactile pattern recognition on the arm and back. Perception. 52-68, 2009.

[7] V. Hayward and K. MacLean. “Do-It-Yourself Haptics, Part I.” IEEE Robot. Automat. Mag. 88-104, 2007.

[8] K. MacLean and V. Hayward. “Do-It-Yourself Haptics, Part II: Interaction Design.” IEEE Robot. Automat. Mag. 104-119, 2008.

[9] L. Jones and M. Berris “The Psychophysics of Temperature Perception and Thermal-Interface Design.” Proc. of the 10th Symp. Haptic Interfaces for Virtual Environments and Teleoperator Systems, 2002 : IEEE Computer Society. http://midas.inrf.uci.edu/groups/bats/uploads/Xsense/Jones_HIfVEaTS_2002.pdf

[10] R. Sodhi et al. 2013 “AIREAL: Interactive Tactile Experiences in Free Air” ACM Trans. Graph. 32, 4, Article 134 (July 2013), 10 pages. DOI = 10.1145/2461912.2462007 http://doi.acm.org/10.1145/2461912.2462007

[11] S. Kim et al. 2013 “Tactile Rendering of 3D Features on Touch Surfaces” Proc. 26th annua. ACM symp. User interface software and technology pp.531-538. DOI = 10.1145/2501988.2502020 http://doi.acm.org/10.1145/2501988.2502020

[12] G. Robles-De-La-Torre. "International Society for Haptics: Haptic technology, an animated explanation". [Online]. Available: http://Isfh.org.

[13] C. Hogan. “Analysis of blind pedestrian deaths and injuries from motor vehicle crashes, 2002-2006.” Direct Research, LLC., Vienna, VA. 2008 http://files.meetup.com/211111/analysis-blind-pedestrian-deaths.pdf

[14] A. Mandal. “What is visual impairment?” [Online]. Available: http://www.news-medical.net/health/What-is-visual-impairment.aspx

[15] World Health Organization (WHO). (2013, October). “Visual impairment and blindness.” [Online]. Available: http://www.who.int/mediacentre/factsheets/fs282/en/

[16] D. Héiligh. (2012, June). “Review of the ultracane.” [Online]. Available: http://www.digitaldarragh.com/2012/06/20/review-of-the-ultracane/

[17] S. Hoefer. (2011, August). “Meet the tacit project: it’s sonar for the blind.” [Online]. Available: http://grathio.com/2011/08/meet-the-tacit-project-its-sonar-for-the-blind/

[18] E. Berdinis. (2012, October). “Kinecthesia: haptic belt for the blind.” [Online]. Available: http://www.kinecthesia.com/

[19] M. Kendrick. (2009, August). “Tasting the light: device lets the blind ‘see’ with their tongues.” [Online]. Available: http://www.scientificamerican.com/article.cfm?id=device-lets-blind-see-with-tongues

[20] Texas School for the Blind and Visually Impaired. (2010, February 19). “Visual Impairment - What is it Like?” [Online] Available: http://www.tsbvi.edu/instructional-resources/1912-visual-impairment-what-is-it-like

[21] ST Microelectronics, “Ultra compact high performance e-Compass

3D accelerometer and 3D magnetometer module,” LSM303D datasheet, Nov. 2013.

[22] ST Microelectronics, “Using LSM303DLH for a tilt compensated electronic compass,” Application Note AN3192, Aug. 2010.

Images: *(Figures not listed were created by Team ARCANE)*

Figure 1: http://dsq-sds.org/article/viewFile/1373/1537/3631

Figure 2: http://en.wikipedia.org/wiki/File:Geleidehond_testparcours.jpg

Figure 3: http://theory.fastsecureservers.com/~ultrac27/image/data/rangefig.jpg

Figure 4: http://grathio.com/wp-content/uploads/2011/08/haptic_glove_diagram.jpg

Figure 5: http://1.bp.blogspot.com/-nwNft7jAmxI/UIWif2ejlbI/AAAAAAAAbnc/Q0bd7REzyVA/s1600/2010-04-16+03.04.11.jpg

Figure 6: http://www.scientificamerican.com/article.cfm?id=device-lets-blind-see-with-tongues

Figures 7 and 8 cited in References

Figure 9: https://catalog.precisionmicrodrives.com/uploads/files/media/308-102_iso.3699.280.280_.jpg

Figure 10: http://cdn.slashgear.com/wp-content/uploads/2013/05/aireal_0-580x349.png

Figure 11: http://robotica.dc.uba.ar/public/papers/ras2012.pdf

Figure 13: http://217.92.194.243/fws2010/presentation/
FWS16_SimGeoColCamCalib_Kapusi_ZBS_Slides.pdf

Figure 15: http://www.ebay.com.au/itm/20-x-Pancake-Vibrating-Vibration-Cell-Phone-Vibrator-Pager-Coin-Motor-10mmx2-7mm-/181116369561

Figure 21: http://robotica.dc.uba.ar/public/papers/ras2012.pdf

Figure 24:

http://217.92.194.243/fws2010/presentation/
FWS16_SimGeoColCamCalib_Kapusi_ZBS_Slides.pdf

Figure 31:

http://www.st.com/web/en/resource/sales_and_marketing/presentation/product_presentation/Smart_street_lighting_marketing_pres.pdf

Figure 32: http://www.st.com/web/en/resource/sales_and_marketing/presentation/product_presentation/Smart_street_lighting_marketing_pres.pdf

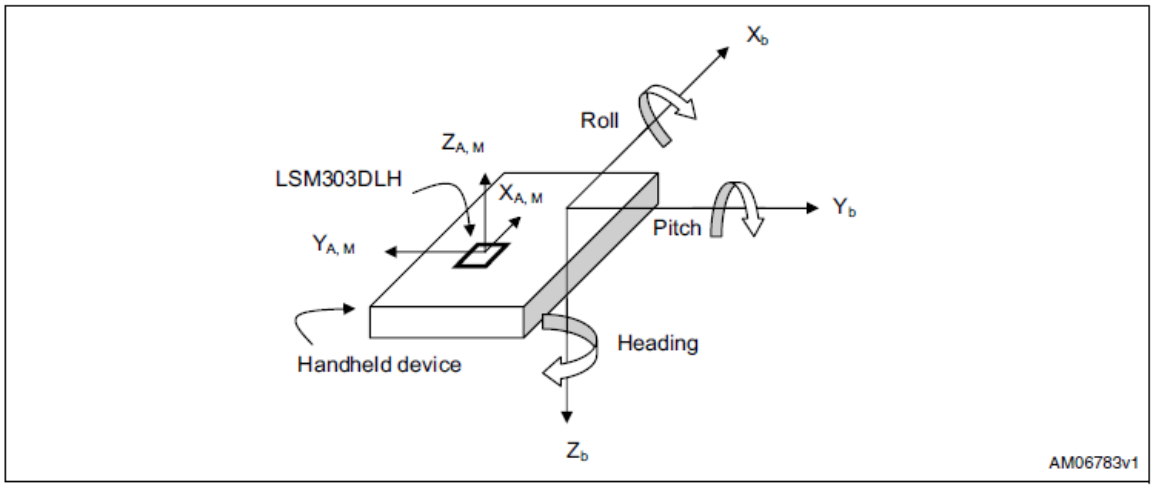
Appendix

Appendix A: Table from the datasheet of the TLC59711, detailing its register structure

Table 5. Data Latch Bit Assignment

BIT NUMBER	BIT NAME	CONTROLLED CHANNEL/FUNCTIONS
15-0	GSR0	GS data bits 15 to 0 for OUTR0
31-16	GSG0	GS data bits 15 to 0 for OUTG0
47-32	GSB0	GS data bits 15 to 0 for OUTB0
63-48	GSR1	GS data bits 15 to 0 for OUTR1
79-64	GSG1	GS data bits 15 to 0 for OUTG1
95-80	GSB1	GS data bits 15 to 0 for OUTB1
111-96	GSR2	GS data bits 15 to 0 for OUTR2
127-112	GSG2	GS data bits 15 to 0 for OUTG2
143-128	GSB2	GS data bits 15 to 0 for OUTB2
159-144	GSR3	GS data bits 15 to 0 for OUTR3
175-160	GSG3	GS data bits 15 to 0 for OUTG3
191-176	GSB3	GS data bits 15 to 0 for OUTB3
198-192	BCR	BC data bits 6 to 0 for OUTR0-3
205-199	BCG	BC data bits 6 to 0 for OUTG0-3
212-206	BCB	BC data bits 6 to 0 for OUTB0-3
213	BLANK	Constant-current output enable bit in FC data (0 = output control enabled, 1 = blank). When this bit is '0', all constant-current outputs (OUTR0-OUTB3) are controlled by the GS PWM timing controller. When this bit is '1', all constant-current outputs are forced off. The GS counter is reset to '0', and the GS PWM timing controller is initialized. When the IC is powered on, this bit is set to '1'.
214	DSPRPT	Auto display repeat mode enable bit in FC data (0 = disabled, 1 = enabled). When this bit is '0', the auto repeat function is disabled. Each constant-current output is only turned on once, according the GS data after BLANK is set to '0' or after the internal latch pulse is generated with the TMGRST bit set to '1'. When this bit is '1', each output turns on and off according to the GS data every 65536 GS reference clocks.
215	TMGRST	Display timing reset mode enable bit in FC data (0 = disabled, 1 = enabled). When this bit is '1', the GS counter is reset to '0' and all constant-current outputs are forced off when the internal latch pulse is generated for data latching. This function is the same when BLANK is set to '0'. Therefore, BLANK does not need to be controlled by an external controller when this mode is enabled. When this bit is '0', the GS counter is not reset and no output is forced off even if the internal latch pulse is generated.
216	EXTGCK	GS reference clock select bit in FC data (0 = internal oscillator clock, 1 = SCKI clock). When this bit is '1', PWM timing refers to the SCKI clock. When this bit is '0', PWM timing refers to the internal oscillator clock.
217	OUTTMG	GS reference clock edge select bit for OUTXn on-off timing control in FC data (0 = falling edge, 1 = rising edge). When this bit is '1', OUTXn are turned on or off at the rising edge of the selected GS reference clock. When this bit is '0', OUTXn are turned on or off at the falling edge of the selected clock.

Appendix B: Calculating the Compass Readings from an LSM303D



Let Mx, My, and Mz be magnetometer readings, and Ax, Ay, and Az be accelerometer readings.

It can be shown that in order to get pitch and roll from the raw accelerometer readings, the following calculations can be used:

Pitch = arcsin(-Ax)

Roll = arcsin(Ay/cos(pitch))

One can then solve for the X and Y components of the heading (Hx and Hy) by utilizing:

Hx = Mx * cos(Pitch) + Mz * sin(Pitch)

Hy = Mx * sin(Roll) * sin(Pitch) + My * cos(Roll) - Mz * sin(Roll) * cos (Pitch)

The heading can then simply be found as arctan(Hy/Hx).