COMP 4905 - Honours Project

Sneaker Size

Author:
Jacob Perks

Supervisor:
Dr. Tony White

December 15th, 2017
Abstract

With the boom in sneaker culture and online sneaker shopping, the market is becoming more and more competitive. Companies now sell sneakers in limited supply releases, with a very high demand. It has become increasingly difficult for consumers to purchase sneakers online, let alone be confident in such a simple decision as shoe sizing. In this project I have set out to help sneaker enthusiasts by creating an incredibly simple to use, centralised platform for sneaker sizing. This consisted of two main pieces of functionality: being able to view authentic user submitted sizing advice for specific sneaker models, as well as submitting your own advice, and being able to determine your own shoe size using a custom camera view. One of the main focusses of my work was exploring mobile design principles to ensure the user interface/user experience was clean, refreshing, and above all user friendly. I also explored current solutions in shoe size measuring and created two working methods. My final implementation involved simply taking a photo of a user’s foot with a quarter next to it, and an accurate shoe size being generated based on the foot’s length and width.

Acknowledgements:

Firstly, I would like to thank Dr. Tony White for supervising this project and helping me to develop my idea into a finished product. I would also like to thank my close friends and family for constantly providing feedback on my application, and editing my report. Without them, it would not have been possible.
Table of Contents:

1. Introduction ...........................................................................................................5

2. Background ..........................................................................................................7

3. Development Process ..........................................................................................8
   3.1. Materials/Environment ..................................................................................10
   3.2. Design .............................................................................................................11
       3.2.1. Application Flow ....................................................................................12
       3.2.2. Design Elements ....................................................................................16
   3.3. Development ..................................................................................................22
       3.3.1. Database ................................................................................................22
       3.3.2. Shoe Size Calculator ............................................................................24

4. Results ..................................................................................................................32
   4.1. Design ..........................................................................................................32
   4.2. Shoe Size Calculator ....................................................................................34

5. Conclusion ............................................................................................................35
   5.1. Next Steps ...................................................................................................35

6. References ............................................................................................................36
Table of Figures:

FIGURE 1: Mobile Development Process…9
FIGURE 2: Tab Bar View…13
FIGURE 3: Popup View…14
FIGURE 4: Navigation Controller…15
FIGURE 5: Sneaker Model Page…16
FIGURE 6: Colour Palettes…17
FIGURE 7: Roboto vs San Francisco fonts…18
FIGURE 8: User Profile Page…19
FIGURE 9: Login Page…20
FIGURE 10: Model Selection Page…21
FIGURE 11: Model View Controller Diagram…22
FIGURE 12: Brannock Device…24
FIGURE 13: Paper Method - Shoe Size Calculator…26
FIGURE 14: Coin Method - Shoe Size Calculator…28
FIGURE 15: Contour Detection…30
FIGURE 16: Rectangle…31
FIGURE 17: Rotated Rectangle…31
1. Introduction:

The beginning of the sneaker, as we know it today, dates back to the 19th century, when a canvas upper was moulded to a rubber lining to create a plimsoll. Although the exact origin and purpose varies on location and story, the general idea was to create something much more comfortable and practical for activities, than the more traditional completely leather shoe. Throughout the 20th century the rise of sneakers began to increase. The rubber soled plimsoll grew into an athletic shoe model that was used for sports, such as basketball, tennis, running etc. Major brands such as Keds, Adidas, Nike, Puma, Converse and many more emerged as the market grew and demand increased. Soon after in the 1950s children began wearing sneakers for everyday use and they were popularised in the film industry, but for the most part sneakers were primarily for physical activities and sports. It wasn’t until the 1970’s that there was a much greater focus on individuality and cultivating a personal identity. Whilst they still very much had a specific athletic purpose, sneakers began expanding into popular culture. Then, in North America in the 1980s, Michael Jordan would propel sneakers much further than they had ever been before. Jordan signed with Nike and wore his first model, “the 1’s”, in the 1984/85 National Basketball Association season. Due to his sneakers’ colour not conforming with the Chicago Bull’s uniform, he was fine for every game for wearing them (starting at $1,000 and rising as high as $5,000 a game). This controversy became national news and served as an ad campaign that began Nike’s gross revenue increasing over $8 billion from 1984 to 1998.\(^1\) With this incredible surge in sneaker interest and consumerism, sneaker culture was truly born.

Now, introduce the internet and again the sneaker world changes incredibly. All of a sudden sneaker lovers, commonly referred to as “sneakerheads”, can communicate and share their passion with other sneaker lovers, not only instantly and from the comfort of their own home, but across the world. The individual sneaker cultures in different areas of the world now have a place where they can all come together and form a global community. Ideas and inspiration can be drawn from one another, ultimately just fuelling the industry even more. This makes it so easy for companies to instantly reach the sneaker audience with online ad campaigns. The ads can be spread throughout the online sneaker community, generating a global buzz around a product and/or brand with thousands of people immediately talking about it. And on top of all of this, the most important

advancement is that these sneakers can be sold directly to the customer through the internet. It becomes so effortless to be a consumer that once again the sneaker industry grows unbelievably. You simply log on to the internet, buy some shoes, and receive them in the mail a week later. But, unfortunately, it became much more more difficult than this. As more and more companies plugged into this online sneaker industry, consumers began having more and more choice. Companies and retailers put up thousands of products and stock online. Consumers could now start waiting for sales, discounts and new and exciting releases. While it obviously wasn’t to the point where the brands and retailers were doing poorly financially, supply did go up and demand did go down. This is where the online sneaker market became very competitive. Brands had to come up with internet strategies to stay ahead of their competitors. So similarly with Michael Jordan, brands created partnerships with people in popular culture, but instead of primarily professional sports people it was with singers, actors, internet personalities and generally people in popular culture with a strong internet presence. This combined with exciting and long ad campaigns created online hype around a product, building up to a specific release date. With the online sneaker community’s interest, the brands could now control the market with limited product releases. This created a frenzy among internet users and ultimately a fear of missing out. Now, it’s not so easy to go online and buy a pair of sneakers that you like. You have to be quick and knowledgeable beforehand. Something as simple as figuring out what size you should buy has become a difficult task. You have to spend hours scouring different websites and reviews beforehand and hoping that you can trust them. You then spend 4 hours in an online queue waiting to buy a pair of sneakers, only to receive them and find out that you should have bought half a size bigger. And, as these sneakers are limited or on sale, the companies do not offer refunds.

This modern day challenge of purchasing footwear online has led me to creating an application that aims to help the online consumer in the sizing of sneakers. My application centralises authentic user sizing advice in one incredibly user friendly platform. Users are able to view sizing advice for particular shoe models, as well as submit their advice for shoes they own, stored in their virtual closet. They are also able to determine their own foot size with the shoe size calculator, which works by simply placing a Canadian quarter to the left of their foot and then taking a photo through the app. The shoe size it generates can then be stored to your user profile. For my application to solve this problem facing the consumer in the sneaker industry today, it is paramount that the user interface is simple, clean, and incredibly user friendly. The main objectives of my honours project is to solve this issue, while exploring IOS application design and aspects of
computer vision. In this report I will be discussing the design and development processes involved when creating Sneaker Size, before analysing my results and looking at the limitations and next steps.

2. Background:

Because the inability to try sneakers on when purchasing online is a real issue to consumers, there are applications and tools that already try to help consumers in the sneaker community and other audiences to achieve similar things as my application. So, in this section I will provide a brief insight into other applications and alternatives to shoe sizing advice and shoe size calculators.

Unfortunately, but unsurprisingly, very few retailers and brands help consumers determine what size of a particular model to purchase when shopping online. And, there is no current website that acts as a central hub for sneaker sizing advice. So, consumers are forced to look towards social media platforms, such as Facebook, Reddit, Youtube. While this may yield the results a consumer is looking for, it definitely will take time and effort, and often is unsuccessful and with conflicting information. For example, to find this information on Facebook, a sneaker head would have to join or already be a part of a sneaker community group. Once in, they would then have to search through the previous posts looking for the information and if its not there then post on the group asking for advice, hoping someone else has that model and is able to give the correct advice. For example, to find this information on Youtube, a consumer would have to find a video for the specific model, watch the whole video and hope that they actually do discuss sizing (which very often they do not). The case is very similar on Reddit and other social media platforms. While it is possible to find the information you’re looking for, a lot of the time you’re spending a lot of effort and time and end up only finding one source. This makes it very difficult to be confident in purchasing sneakers online, and as mentioned earlier, if the product is limited or on sale you unfortunately won’t be able to get a refund.

Moving onto the calculator, there are a few applications that currently exist to determine a person’s shoe size. They are all mobile applications as they make use of the builtin camera. I will be discussing one paid and one free IOS application that in total use three different methods. The
first one is called Right Shoes and is completely free to use.\textsuperscript{2} It gives users two options: a 2D and a 3D scan. The 2D scan is done by drawing an outline of your foot on a piece of paper and then taking a picture of it. The 3D scan is done by having someone else take an almost panoramic view of your feet by hovering your phone just above the ground and capturing your feet with the camera 360 degrees. This scan actually generates a 3D model of your foot so it takes into account not only your foot length and width, but also your instep and foot arch. The second application is called Feet Meter and costs $2.79 on the Canadian App Store.\textsuperscript{3} It uses an A4 or letter piece of paper as a fixed length measuring tool to compare to your feet. To use it, you must find a flat based wall (one without a skirting board) and place your feet against it, along with the piece of paper. The 3D scan used by Right Shoes is the most detailed and thorough scan out of the three methods, and out of any other applications I could find. However, it is a much more involved process, requiring another person to take the scan. For something simpler, the 2D scan of drawing the outline of your foot on a piece of paper works but again does require some effort. And, if you have gone to the trouble of drawing the outline, it wouldn’t be very difficult to just measure the outline yourself and determine your shoe size. The utilisation of paper in Feet Meter makes it the easiest to use out of the three methods, but finding a wall without a baseboard could potentially be difficult and the $2.79 price tag is not very user friendly. In section 3.3. I will be discussing the various methods and technologies I researched and implemented, and then compare them to the final method I implemented, which uses a coin as a measuring anchor.

3. Development Process

In this section I will focus on the development and design processes I used in creating Sneaker Size. I will discuss and explain the methods and approaches I took as well as ones that were considered and possibly implemented/tested, but ultimately discarded. The two main areas I will concentrate on are the user interfacer and user experience (UI/UX) design and the algorithm for the

Right Shoes was last updated on November 29th, so it is very much a current application.

shoe size calculator. But first, I will briefly go over the mobile development process, as a whole, from conceptual idea to a final deliverable. An overview of this can be seen in figure 1 below. Once I realised the need for an application like Sneaker Size, I began by doing all the necessary research. This included both market and technology research. To be able to deliver a product to the intended customer, I had to fully understand the target market and understand their requirements. Because I am a part of the target audience, it made it a lot easier to grasp but it was still essential to make sure I understood their needs as a whole. Once this information had been collected, I did the necessary research into the technology available to me. Next, I began the design process. This started with rough mockups to try to capture the overall design theme of the application and how it would flow and interact with the user. After these mockups were analysed and edited, next came the design stage. This involved actually designing the concept, graphics, icons etc. within the Xcode main storyboard. Because the UI/UX is one of the key features of this application, it was essential that I got feedback from potential users, and reviewed the design. Then I began the actual development, consisting of both backend and frontend components. And again, I retrieved user feedback to make the necessary improvements to development. Next, I began the final stage of revising and final edits. This mostly consisted of making changes from the additional feedback and adding smaller final revisions, before completing the process with a final product.

FIGURE 1: flow chart of the development process
3.1 Materials/Environment

SneakerSize is developed for IOS devices, coded in Apple’s Swift 3 programming language. And, as a result developed in the Xcode environment. When I came up with the idea for this project, I looked into both Android and IOS app development. While in the future it may support both, the original platform it was developed for was crucial. This is because even though the storyboards and mock ups were designed independent of the platform it was intended for, it would obviously evolve and take inspiration from it. And thus, the application’s overall image would be determined by the original platform it was developed for. Taking all of this into consideration, creating it for IOS devices was the clear choice. Apple is renowned for paying meticulous attention to detail not only in their own operating systems and software, but also in the apps that they publish on the App Store. When an app is submitted for approval to the App Store it is very common for it to be denied at least once, but often more than this. Apple has a page on their developer website that explains the most common reasons for app rejections\(^4\). They even have an entire Human Interface Guideline and UI Design Do’s and Don’ts to help their developers submit applications that are cohesive with Apple’s overall theme. In the next section I will expand further on this, when I discuss the design elements and choices made in creating SneakerSize. To further solidify my decision, I looked at an application in the same market targeting the exact same audience as mine: Frenzy. Frenzy is an IOS application, developed by Shopify, that releases limited quantity sneakers at specific times and sometimes specific geographical locations, directly through the app. Shopify has announced that they plan on releasing an Android version in the future, but for now has decided to remain with their original IOS platform. With the positive reception and incredible use and support Frenzy has received, I think it is a reasonable assumption that a very large percentage of the online sneaker community, and thus my target audience, uses an IOS mobile device versus Android or Windows.

Now that I had chosen a mobile platform to develop with, I had to decide on a backend framework to use. The amount of data I need to store for each user and for each shoe model is fairly small. For the server I wanted to choose something that was going to be simple to integrate into my application and that would allow me to pull and push data efficiently. So I used Parse Server and

\(^4\) Common App Rejections. Retrieved from: [https://developer.apple.com/app-store/review/rejections/](https://developer.apple.com/app-store/review/rejections/). Eleven common rejection reasons are listed but one of the top ones is a “substandard user interface”.
IOS software development kit (SDK), hosted on an Amazon Web Services EC2 instance. Internally the data is stored as JSON and uses MongoDB. It provides an online environment for viewing the data using Parse Dashboard. As well as this, it facilitates the creation of user accounts very easily with the builtin PFUser, which is a subclass of the native object: PFObject. After testing it with some dummy data, it was more than sufficient. In terms of scaleability, because it is stored on an Amazon EC2 instance, it is completely elastic. Its currently running at the free tier as t2.micro with 1GiB of memory, but can be upgraded to 384 GiB of memory\textsuperscript{5}. So obviously, in the future if the workload was to increase significantly, there wouldn’t be a need for having to change or redo the backend.

The final technical decision was choosing a computer vision framework that could be easily integrated into my application developed in Swift and would provide me with powerful image analysis tools. I chose to use the framework OpenCV because swift is based on Objective-C, and OpenCV offers a C++ interface. With the framework integrated into my application, a .mm file can be created that allows the mixed use of C++ and Objective-C, creating a bridged Objective-C++. A bridging header file can then be created to expose the Objective-C++ code to Swift, thus allowing the OpenCV tools to be implemented in the .mm file and used throughout the rest of my application. As well as it being incredibly easy to integrate into my Swift code, OpenCV allows for easy UIImage conversion to its native image container: Mat, and offers great image analysis and contour detection tools. In section 3.3, I will be elaborating on how these were used in creating the shoe size calculator.

3.2 Design

As described earlier, the target audience for the application is the online community of sneaker enthusiasts. While this gives a general idea of the consumer, its important to look more specifically at the demographic of this community. This will heavily influence the design process and overall theme of the application’s user interface and user experience. Because the online sneaker industry is a relatively new market, there isn’t any real data or statistics on the demographic. So, the best way of establishing this was to rely on my own personal experience.

\textsuperscript{5} Amazon EC2 Instance Types. Retrieved from: https://aws.amazon.com/ec2/instance-types/. The example given is a m5.24xlarge instance.
Having been a part of the community for nearly two years now I can safely conclude that the average “sneakerhead” is male, ranging from 15-30 years old. While there are a few females in the community, it makes more sense to target the larger group of the intended audience. Initially I considered making it useable for both males and females, however I felt that the added functionality would clutter up the interface and ultimately negatively impact the primary user’s experience. This was another important factor when considering the design requirements. The ultimate goal of the application is to solve a real life consumer issue whilst ensuring that this is done in the most efficient way in order to minimise the users time spent on the app. Therefore, the UI/UX design has to be simple, clean and decluttered. Adding in functionality that wouldn’t be used regularly takes away from this.

3.2.1 Application Flow

Regardless of stylistic elements, the application flow was one of the most important elements in designing Sneaker Size. How the user navigates throughout the app had to be seamless and intuitive. No matter how well each individual page was designed, if manoeuvring from one page to another was awkward and clumsy it would make the entire application feel the same way. The first and only access point to Sneaker Size is the login/sign up page. Once a user has logged in or signed up, it navigates directly to the main page view. However, Parse does save the current user’s session so if a user leaves the application and then returns, it will remain logged in. A lot of applications nowadays provide an introductory walkthrough or information slideshow once a user creates an account and logs in for the very first time. It is intended to give a brief overview of how the application works and flows. While I am sure they are beneficial for very complex applications with a wide variety of functionality, I am a firm believer that a user interface for most applications should be designed so such that a walkthrough is not necessary. And, even if there is a wide variety of functionality and/or some “exploration” is involved, I still think that the user experience should be as natural as possible. So once logged in to Sneaker Size, the user goes straight to the main tab bar view. The tab bar view controller splits the application into three subviews accessible via the icons on the bottom bar. I chose to use the tab bar view because it lays out the subviews in a very clean and direct way. It is also very popular in today’s market. Most functional or lifestyle applications on the App Store use a tab view. For example, Instagram, Twitter, Facebook, Facebook Messenger, Youtube, and many more all use it. Millions of people use these applications everyday so it obviously works very well. But what is even more important is the fact that since so many
popular applications use it, consumers are very familiar with it. Upon logging into the application, users immediately understand the flow of the application and can begin navigating and interacting with the UI without a walkthrough or tutorial.

![Tab Bar Controller](image)

FIGURE 2: tab bar controller with the three icons on the tab bar

The three tabs of the application are the sneaker page, the calculator page, and the profile page. Originally I made all three of them single pages, with the only further navigation being popup views. The profile page really just needs to display the current user’s information and allow the user to logout. So, it definitely didn’t need any further navigation. The calculator page uses a built-in camera view, so a photo of the user’s foot can be taken. The only extra information that I needed to display were the instructions on how the photo should be taken for best results. To do this I used a popup view. I felt that this method was best because it contained all the calculator functionality. When giving instructions, it is often the case that a user skims through them as quickly as possible,
attempts the task unsuccessfully, and then has to return to the instruction guide to re-read them. In this case, having a popup view made it much easier for the user. They could quickly flick back and forth between the camera view and the instruction guide inline. They didn’t have to navigate to a completely different page, only to have to ultimately return to that page later on.

FIGURE 3: shoe size calculator information page, displayed as a popup view

The last page of the application, the sneaker page, requires a lot more information to be displayed, as well needing user input. The user must be able to select a shoe brand and model. Then the sizing advice is retrieved and displayed based on the user’s selection. Then the user must be able to select and submit their own sizing advice, if they choose to. In an attempt to make the application’s navigation as cohesive as possible I wanted to follow the same structure as the profile
and calculator pages, displaying all the information in one single page with a popup view. To do this I used two picker views in the lower half of the screen for receiving a brand and model selection from the user. Then, the sizing advice was displayed in the upper half of the screen. To submit sizing advice, a user clicked a button that displayed a popup view. All information relating to submitting the advice was contained within the popup view, and once the user was finished, they could simply close it, revealing the sneaker page again. Just in the popup view alone there was a lot of information on one page. I felt it was possible to organise the information in a much better way to make it less cluttered and easier to understand and use. To validate this, I conducted a small survey of friends and family that would be future users of the application, for feedback. Whilst there were some mixed views, the general consensus was that the page didn’t blend well with the rest of the application. They felt it would be “passable”, but lessened the overall app and consequently could be improved. So, I decided to redesign the sneaker page using a navigation controller. I split the brand choice and the model choice into two separate views, and then had a final view for the specific model chosen.

FIGURE 4: sub navigation controller of the sneaker tab.

To keep the choice pages as simple as possible I used near full page table views, with a small text box at the top. Because there was zero clutter on these pages, the user would only see the available selections and be able to manoeuvre very quickly through the pages. Once they selected a brand it would segue immediately to the model selection and after a model selection it would again segue immediately to the model page. Since I had now freed up a large percentage of space, I could
remove the popup view for submitting size advice and add that to the model page. This worked very well because it allowed me to contain all of the information pertaining to that particular model to one specific page. All functionality associated with the submitting size advice was now contained in the lower half of the page. And, the display of the current sneaker model and its sneaker advice was contained in the top half of the page. Figure 5 beneath illustrates this.

![FIGURE 5: the sneaker model page](image)

### 3.2.2 Design Elements

One of the very first considerations when designing anything is the colour scheme. In applications, the colour immediately communicates the tone and mood of the entire app to the user.
It invokes certain feelings and influences the user’s overall experience. As I have previously noted, I wanted to ensure the app was simple, clean and easy to use. Initially I chose a subtle mint and off-white colour scheme to achieve this. Whilst this worked well, I wanted to get feedback on how it connected with the rest of the application and how it was received by the target audience. The result of this stage of user feedback was that it felt much too soft and feminine. It was aesthetically pleasing but overall confused what target audience it was aimed for, as well as the purpose of the application. After implementing and testing a few different colour combinations, keeping my target audience of males aged 15-30 in mind, I decided to go with a simple but much bolder colour scheme of blue and white. The shade of blue I chose had a very slight purple tone which made it deep and rich. Combining this with white created a balance of darker and lighter tones. While the darker shade of blue felt masculine and connected very well with the target audience, I felt that it came across to the user as too heavy, and the balance with the white was too harsh. So, I decided to lighten the shade of blue and darken the white to a shade of off-white. This made it feel much more refreshing, while maintaining a contrasting colour scheme. Also, and crucially it still allowed the white text to stand out against the blue.

![FIGURE 6: the original and final colour schemes as colour palettes](image)

Next, I had to decide on a font. My only real constraint was choosing one that was sans serif. I felt for this application serifs added an unnecessary flair and disturbance. I wanted to maintain the direction of my target audience of males. Apple’s system font is San Francisco. It fitted very well with Sneaker Size, and would make it cohesive with the rest of Apple’s default apps. However, I found the letters to be too compressed. Whilst this concern was only minor, San Francisco has a sister font called Roboto that is available on Google Fonts. It is almost identical, except that the lettering was stretched out slightly. In my opinion this made it noticeably neater and easier to read.
Overall, a lot of companies and developers use bold colour schemes and fonts to help their application stand out. This definitely does work, however for Sneaker Size, I felt that the goal for my target audience was subtlety, rather than a visually more obvious approach.

The next element that became an integral part of my design was the use of rectangular geometric shapes. Boxes and rectangles have sharp edges and clean lines. It was essential to keep the pages organised and information well defined and contained. Using boxes allowed me to create structure throughout my application without having to draw too much attention to the actual division of the page. The most subtle and effective way of incorporating this was through colour blocking backgrounds. Information that I wanted to be separated or defined as its own entity I colour blocked a blue view against the off white of the app’s background. A good example of this is on the profile page. The virtual closet is contained by a solid blue box, spanning the width of the page. This separates it from the rest of the profile information, displaying it as its own entity. But the key factor is that it does not disturb or impact the entire page. The rest of the profile page is equally emphasised with the off white background. It merely looks like a splash of colour but serves a much greater purpose.

---

FIGURE 8: the user’s profile page, where the distinct colour blocked background can be seen.

Another way I used rectangular shapes to this effect was on the login screen. The username and password fields were placed within a table view. Each field is created as its own row in the table. This groups the two entry fields together and contains them within the table’s outlining structure. With an off white background, it also gives the effect that the fields are setback within the table, placing more emphasis on them. Figure 9 beneath shows the result of what I was trying to achieve with the geometric shapes and contrasting.
Similar to the application’s navigational flow, it’s interface guides a user’s eye movement and thus their experience and interaction. This means just like the overall navigational flow, each page needs to be designed in such a way that provides an intuitive and seamless flow of eye movement. To do this I used markers that places emphasis on initial areas of flow. Then once the user gravitated towards the initial marker, they would instinctively flow throughout the rest of the page. An example of this is on the brand and model choice pages. Initially I wanted to strip everything except for the bottom bar away and just have a full page table view displaying the brands or models. However when I did this the user had no primary anchor to look towards. Obviously it was still a very simple display, but I found that the large list of data would almost inundate the user with text. So, to ensure the user’s eyes had a safe landing point I placed a simple
coloured label box at the top of the screen and reduced the size of the table. Once they saw the label, their eyes naturally flowed down the list. Figure 10 below has a closer view of one of the model selection pages.

![Pick a sneaker model](image)

**FIGURE 10**

Another important aspect of eye movement is using whitespace to give the user’s eyes a break. If a page is full top to bottom and side to side, it can be overwhelming. This is something I made sure to incorporate throughout the app. Even on the model page, which required a lot of information to be displayed, I added whitespace gaps vertically between each element, and restricted their width so horizontally the space opened up.
3.3 Development

In terms of software patterns for my development process, Apple has designed Swift and Xcode to work best with the Model-View-Controller pattern. So, this was the obvious choice. The main storyboard in Xcode hosts the views. They are then connected to controller files, which are triggered through user interaction. The controllers communicate with the model before updating the view. In Sneaker Size, the main model object is the OpenCVWrapper class. It is given a UIImage by the calculator view controller and then returns the foot’s dimensions. This model layer encapsulates most of the logic involved. In terms of the data layer, the controllers update the data storage and are notified of the current information stored in the database. The following model from Apple’s developer website illustrates exactly how this works.7

![Diagram](https://developer.apple.com/library/content/documentation/General/Conceptual/DevPedia-CocoaCore/MVC.html)

3.3.1 Database

As mentioned previously, for my backend framework I used Parse Server. Parse Server provides an online Dashboard that allows for the easy creation of data. On the cloud this data is stored in classes. Through the dashboard, class objects can be created and class data types can be

---

added. Sneaker Size needs to be able to store sneaker brands and models. And each model needs to store the user submitted sizing data. Each brand has one or more models, and each model belongs to only one brand. Therefore, firstly I created a brand table, where each tuple would contain the brand name as a string, being the primary key. Then I made a single model table, where each tuple would contain the brand table’s primary key as a referencing key, a model name as string, and an array of numbers storing the frequency of sizing advice submitted by users. In an ideal world, users would only submit sizing advice for a particular model once, but this isn’t always the case. So I needed a user table to keep track of this. Having a user table also becomes helpful because it allows me to store each user’s shoe sizing information, once they use the shoe size calculator or manually enter it in themselves. Conveniently, the Parse dashboard comes preloaded with a user table containing username, email address, and password attributes. To keep track of the sneaker models that a user has submitted advice for I created an array of model primary keys.

The data tables mentioned above, stored on the Parse cloud, are locally represented by PFObjects. In swift, a PFObject can be queried using a PFQuery. Various constraints can be set, such as setting a key value in the query, but ultimately to perform a PFQuery all that is needed is the table’s name. The first time my application interacts with the database it displays the brands in a UITableView for the user to select a sneaker brand. This simply queries all tuples in the brand table and populates the UITableView with the primary key: the name. The second is displaying the models in a UITableView for the user to select a model after they have just selected a brand. This queries the model table with the constraint that the foreign key, brand name, is equal to the brand previously chosen. It then populates the UITableView with the model names belonging to that brand. The third time is on the final page of the navigation controller, where a sneaker model has been selected. Again, the model table is queried to find the tuple with the same primary key as the model chosen, and the array of sizing advice is retrieved, interpreted and displayed as plaintext feedback. As mentioned earlier, the sizing advice is stored by frequency of submissions, by unique users. The feedback that is given to the user is a breakdown of percentages of each submission. For example, it might say 80% of users said go true to size, and 20% of users said go half a size down. So, when a user wants to submit their own advice, first it checks the user table to ensure they have not already submitted advice for that particular model and then it pushes to the model table incrementing the count of the advice chosen. The last interaction with the database is on the user’s profile page. The user’s virtual closet, which is essentially just all the shoes they have submitted sneaker advice for, shoe size, and username are all pulled from the user table.
3.3.2 Shoe Size Calculator

To determine someone’s shoe size the industry standard Brannock measuring device looks at three main factors: length, width, and arch length. The standard US shoe size system most importantly looks at the length of the foot, but also the width. Unless a shoe is made specifically as a wide model, or is particularly narrow, arguably all that is needed to determine the correct shoe size is the length. In the case where a shoe is particularly wide, but the length is standard, taking a shoe size that matches the shoe size measured in a Brannock device would be perfectly fine. Most of the time, the extra width could be reduced by simply tightening the shoe. In the case where a shoe is particularly narrow, but the length is standard, this could cause discomfort on the sides of a user’s foot. So, taking half a size up would widen the shoe to the correct width, only sacrificing the length by no more than 1.3 inches (by the US system). But, to be as accurate as possible it is best to obtain the user’s foot length and width. This way we can firstly look at the user’s foot length to determine a base shoe size, and then look at the user’s foot width to determine if any adjustments need to be made. The arch height isn’t as important because even though it affects the width of the foot at that particular point, it isn’t the widest part of the foot. And, regardless of the size of the foot arch, how a shoe sits against the arch is most often personal preference.

![FIGURE 12: the original patent for the Brannock measuring device.](image-url)
The goal of my shoe size calculator is to make it quick and easy to use. Time and efficiency is essential in the online sneaker world. Realistically, a large percentage of my target audience would already have a general idea of what shoe size they take. But, having said this, a large percentage of them would also only buy sneakers online in limited releases. As a result they are buying varying sized sneakers and very easily could end up purchasing and wearing a shoe size that isn’t their actual true fit. So, providing a user with a quick and easy way to verify their shoe size is very useful, practical and potentially saving the consumer money as well. The best way of doing this was obviously using the IOS device’s built-in camera and then analysing the photo to determine the length and width of the foot. To calculate the size of an object in a photo, I needed the ratio of pixels to centimetres or inches, but in order to get this I needed to calculate the distance of the camera to the object. Whilst it probably would be relatively similar for most users, to get an accurate measurement of the size the distance has to be very accurate as well. The formula for calculating the distance is defined by the pinhole camera model:

\[ \frac{x}{f} = \frac{X}{d} \]

And thus calculating the distance is impossible, as we need the size of object, which is what we were trying to calculate in the first place. So, the next best option is using some common object as a measuring tool. Using a household item such as a piece of paper is the basis for my first method. Since the exact width and height of a piece of paper is known and fixed, whether it be A4, letter or ledger, it is ideal as a measuring tool. My first idea for calculating a user’s shoe size was simply getting a user to take a photo of their foot on a piece of paper and then determining what percentage of pixels cover the piece of paper. This percentage could then be multiplied by the fixed width and length of the piece of paper yielding a length and width of the user’s foot. However, because the piece of paper was such a large measuring tool, the angle at which the photo was taken would make a massive difference. A slight lift in the camera upwards would cause the foot to cover a much bigger portion of the piece of paper, throwing off the length considerably. So, to ensure the camera was level as much as possible I implemented a mask on the camera view. This blurred the

---

8 Wu, Y. Image Formation and Camera Calibration. Retrieved from: [http://users.eecs.northwestern.edu/~yingwu/teaching/EECS432/Notes/camera.pdf](http://users.eecs.northwestern.edu/~yingwu/teaching/EECS432/Notes/camera.pdf). Where \( x \) is the size of the object on the sensor, \( f \) is the focal length of the lens, \( X \) is the size of the object, and \( d \) is the distance from the nodal point to the object.
live camera feed around a rectangle with the same ratio as, for example, an A4 piece of paper. Then, the user could line up the paper with the rectangle and take the photo. This also allowed me to simply crop the image to the size of mask and detect the contours of the foot. I could then find the max height and width of the contour in pixels, find the ratio of this to the height and width of the image and then multiply by the fixed length and width of the piece of paper used, thus giving me an exact length and width of the foot. After testing it with my own foot it worked quite well, giving a length -0.3 inches compared to my actual foot size.

Figure 13 shows an example of the first shoe size calculator method, using a piece of letter paper. In practice, it was a little difficult to line up the piece of paper perfectly with the mask. This can be seen on the left and bottom portions.
Whilst I was pleased with the results, I felt that the main goal of my application, to make life easier, might not be completely fulfilled using this method. The most standard piece of paper in North America would be letter, which is 11” long. Using this size of paper the calculator could only measure feet size US 11 and under. Therefore I would have to introduce multiple options for the paper size, either A4 which is 11.7”, legal which is 14” or ledger which is 17”. Firstly, the added options begin to clutter the user interface and require more work for the user. Whilst it may seem trivial, the simple user experience is an integral part of the application. But, most importantly I’m now requiring a measuring tool that isn’t necessarily a common object. With everything being online nowadays, the need for paper is declining, let alone the need for obscurely sized pieces of paper. And so I began to look towards other methods. The idea of using a common object measuring tool seemed to be the best option, however instead of using something bigger than a foot to measure against, why not use something small with a fixed size to create an inches per pixel value? This was the foundation of my second method.

Using a variable to model the relationship between inches of an object and number of pixels solved my issue of not knowing the distance from the foot, because as the distance of the device from the foot changed, so did the object’s number of pixel cover in the image. This would be my indicating factor of distance from the foot. Depending on the height of the camera, I could calculate exactly how many pixels the object took up in the image. Therefore I would know that the size of the object corresponds to this specific number of pixels, based on the photo taken and current distance to the object. The decision for what object to use was simple. I needed something with a fixed size, very common and accessible, and most importantly small that could be placed next to the foot. The answer was a coin. A coin also provides the added bonus of having equal length and width. So while it doesn’t make a massive difference, it does allow the user to place the coin on the ground in no particular orientation. In the paper method, the angle of the device made a big difference in determining the foot length. This was because the foot was measured directly against the large piece of paper. So even a slight positive change in degree projected the foot much larger against the paper. And thus changed the size significantly. Using a small object like a coin made a less significant impact when the angle was slightly off.
Figure 14 shows an example of the final shoe size calculator method, using a coin. In practice, reducing the distance of the camera to the foot is best, as it reduces the possibilities of image noise and makes the coin and foot have much larger pixel spread.

Now I will discuss the actual implementation details of this method and the tools and functions used in Swift and OpenCV. First the user needed to be able to take the photo of their foot with a coin next to it, to be analysed. So I set up a custom camera view within the application with AVFoundation using a simple video preview layer. When the photo is taken it is captured and

---

saved as Swift’s native UIImage. As mentioned earlier, OpenCV uses a specialised image container called a Mat. However, OpenCV’s C++ framework provides a very simple function to convert the UIImage to Mat format. Obviously it would be ideal for the foot and coin to be captured on a completely plain background. This would make the objects distinct and easily captured. However, in practice this is sometimes hard to achieve. So first, I converted the Mat to greyscale. This creates a greater level of distinction between the background and the coin and foot, and removes any unnecessary and distracting colours. Then to remove any noise in the image I performed a slight blur. Before attempting to find the contours in the image, I performed one final operation to help accentuate the coin and the foot against the background. This was a threshold. OpenCV offers a variety of thresholding types to segment an image into objects. Since the image was already in grayscale I separated it using a binary threshold type. If a pixel had a value greater than 100, it was set to 255 and if it was under 100 then it was set to 0. Now I could locate the contours much more easily. OpenCV provides a simple to use function that gives a Mat and contour hierarchy, finds all contours and stores them as Mats. Since all I need is the outline of the foot and the coin, I used the RETR_EXTERNAL contour hierarchy, which only retrieves the outermost contours. This is very helpful because if the user was wearing a patterned sock, it would not retrieve any of the contours created by this pattern. For example, in figure 15 below, you can see where the external contours have been detected. And despite the fact that I was wearing patterned socks, nothing inside the foot’s contour was detected. It is also important to note that this photo was not taken for determining a shoe size, but purely to illustrate the contour detection of a foot. And as a result, there is no coin and the angle of the photo is not horizontal to the ground.
The problem with creating contours around a foot is that there’s a leg attached to it! If it were a singular object, it would have clearly defined external bounds. With a foot I could only create a contour up to the heel. To solve this issue I used the edge of the image to complete the contour. If a user could line up the bottom of the camera view with their heel, it would complete the contour. Whilst this sounds relatively easy for the user, when you are actually taking a picture of your foot, its very hard to determine where the base of your heel is. So as a guide, it is recommended that the user places their heel against a straight line, such as the edge of a carpet or the base of a wall. Then use this to line up the bottom of the camera. This worked very well and didn’t require an excess amount of time and effort from the user. Since my application used a tab bar view, the custom camera view could not be stretched full screen. AVFoundation takes an image that is full screen regardless of the custom view displayed to the user. There are resizing options, however they stretch and distort the image. To correct this before sending the UIImage to the OpenCV model, I
converted the UIImage to a CGImage and simply cropped out the portion of the photo that would be covered by the tab bar view, before converting it back to a UIImage.

Now that I had the contours of the user’s foot and coin, I needed to calculate their widths and heights in pixels. Originally, I converted each contour to a Rotated Rectangle using the minAreaRect function. This function finds the outermost points and connects them using a best fit rectangle. This means if necessary, the rectangle could be titled. In OpenCV, Rotated Rectangles store a size value containing both the width and height, which was very convenient. However after testing it using rotated rectangles I ran into a few problems. If a user’s ankle bone protruded out further than the width of their foot or if they had a very wide bunion, this part of their foot would become a corner point of the rectangle. And, since not vertically inline with the big toe (usually the longest part of the foot), it would generate a rectangle titled on its side. Therefore, the length of rectangle would be slightly longer than the actual length of the foot. To solve this issue I used the boundingRect function to transform the contours, as Mats, into rectangles that were restricted to always being perpendicular to the bottom of the photo.

![Figure 16](image1.png) ![Figure 17](image2.png)
Figure 16 shows the rectangles created from a user’s foot and coin using Rects and the `boundingRect` function, whilst Figure 17 shows the rectangles created from a user’s foot and coin using a Rotated Rect and the `minAreaRect` function. Note, both images have been cropped from their original frames to show the rectangles more clearly.

Similarly to Rotated Rectangles, Rects store a height and width value. If any of the rectangles had an area under 2500, I simply discarded them as image noise that made it through the grayscale, blurring and thresholding. Next the only thing left for me to do was to divide the coin’s pixel length or width by the actual coin’s length or width in inches to calculate a pixel per inch value. Then divide the foot’s pixel length and width by this pixel per inch value to obtain a final length and width in inches for the user’s foot. To convert this to a shoe size I calculated a range that the size would fall under. For any given size, I found the mean of itself and the next half size or full size up and the mean of itself and the next half size or full size down, all in inches. These two numbers then defined the lower and upper bounds for the range that that particular shoe size would fall under, based on length in inches. Now that I had a base shoe size for the user base solely on length, I could look at the width to determine if it required me to adjust the shoe size. For example, if a user’s width fell in the range for half a size to a whole size up, the user’s shoe size would need to be adjusted to the next half size. However, if it was half a size narrower, I could simply leave the shoe size as it was. It was important to consider that extra space in the shoe was acceptable, but having a shoe too small was not. And finally the size generated could be saved and stored to the user’s profile.

4. Results

4.1 Design

From the early conceptual stages of creating Sneaker Size I placed a lot of emphasis on making the UI/UX a top priority. I knew it had to be clean, professional, and above all incredibly user friendly. And I tried to achieve this as much as possible. Throughout the design process I faced certain challenges and as a whole learned a lot about designing a mobile application geared towards a specific problem for a specific target audience. When first designing the navigational structure and
page layouts, in the sneaker sizing advice functionality of the application, I placed too much emphasis on simplicity and lost focus on the user experience. I ultimately confused a very important feature in an attempt to reduce the amount of navigation. When, if I had taken a step back to see the application as a whole, it was quite clear that it just didn’t work nor blend well with the rest. This, among other aspects of the design process, showed me that whilst less can be more, it isn’t always the case and must be done in moderation. The pages still needed substance and a level of “pop” to draw the user’s attention. Similarly, I discovered just how important colour and eye movement were and the overall impact they had on the user interface and user experience. The structure of each page and the elements within it, directly controlled the user’s eye movement, allowing them to rest at certain points. Thus, the placement of views and elements were essential in defining the flow of the entire experience. The colour scheme played a much more important role than I had anticipated. Originally I chose a mint colour and white background. However, after the feedback I realised that the softness of the mint and white colour scheme did not mesh well with the target audience. I needed something a bit bolder that would appeal to 15-30 year old males. But after implementing a considerable number of colour combinations, it was clear how challenging something as simple as choosing a colour could be. Another result of my design experience was how different the process is when targeting a male audience. With every decision I made, I had to keep at the forefront of my mind that males are drawn towards specific visual elements.

Interpreting these results, designing a mobile application is difficult. There is much more to consider than I had expected. When you see a final product, it is easy to quickly judge the user interface and user experience. In many cases its either very good or very bad. The negative aspects aren’t hard to find, especially after using the application. However, the path designing from a conceptual idea to a finished product is definitely not a straight line. It involves a lot of experimenting, analysing, and feedback from potential users. This becomes even more difficult when the target audience is so specific. So obviously there are limitations to what can realistically be achieved as certain aspects will be subjective to the user. The interface won’t necessarily appeal to everyone. What some users might find clean and professional, others may find boring and bland. What is essential, and in my opinion does shine through in the finished product, is that the overall purpose of the application, the actual real life problem it is trying to solve, and the target audience are the main inspirations in the design process. This includes both considering these aspects when designing, and also getting potential user’s feedback throughout.
4.2 Shoe Size Calculator

Given the diversity of current solutions for calculating a user’s shoe size with a mobile device’s camera, I wanted to test multiple methods that I had in mind and analyse the results. The first method of measuring a user’s foot against a piece of paper, as mentioned previously, was ultimately discarded. Whilst it did provide very accurate results, I felt the cons outweighed the pros. First and foremost, it wasn’t very inventive. Using a piece of paper has been done in multiple ways, as described in the background, because it simplifies the process. By having a completely white background, firstly, it removes any noise that might be in a normal photo. Then, with a contrasting coloured sock, determining a user’s shoe size could even be done by simply comparing a ratio of light to dark pixels. This level of ease does come at a price, however. Firstly, it ran into problems with paper sizing. For any users with over size 11 feet or above they had to find bigger paper than standard letter. And this meant using an object that might not be as common. It was also very important for the mobile devices to be as flat as possible, or else it made a big impact on the sizing. Overall, it was an approach that worked but couldn’t be improved to a level that I wanted for this application. The second method was much neater and impressive. A coin was a much more accessible household object which helped with speed and efficiency. From a user’s perspective, its a lot nicer to abstract the exact method of calculation away from the interface. While its obvious that the coin is being used as a measurement tool, less constraints are being imposed on the user interface. The user simply needs to find a coin, line up their foot at the bottom of the camera view and take a picture. The main issues with the second calculator were to do with disturbances in the photograph. On a dark background with a white sock or barefoot, and with the camera nearly horizontal, the results were very accurate. However, on busier backgrounds, such as patterned carpets, it was much harder to distinctly find the contours of the coin and foot. In some cases it caused distortions in the generated size, but in others it was simply too difficult and required the user to try again.

It’s hard to argue against that shoe sizing will always be most accurate using a standard Brannock measuring device. Without multiple camera views and/or 3d modelling, the arch of a user’s foot alone is impossible to determine and analyse. However, there are a number of recent attempts made by mobile applications to solve this issue without needing any real physical measuring. My work has shown that there are ways to achieve great results using a mobile device’s
camera. It has also shown that there isn’t a single quick solution. It is a complex problem to solve, with a lot of compounding factors to consider. There are many different ways to achieve the goal and some work better than others, whether this means easier to use, more accurate, less work involved etc. For my target audience, but arguably overall, my second method works better than my first.

5. Conclusion

I originally set out to make an IOS application for the online sneaker community that would help them in their online pursuit of sneaker culture. Being a part of this culture, I felt that I had identified a problem and knew exactly what to do to solve it. I believe that my final product has been able to solve this problem and ensure that “sneakerheads” can now shop online with renewed confidence. But, the entire development process has given me an opportunity to explore areas of mobile development and learn a great deal. The first of this is IOS development in general. I spent a lot of time working with AVFoundation and Parse, and discovered more about everything that Swift and Xcode has to offer. I realised the value and importance of the mobile design process in achieving the application’s core goals and connecting with the target audience. I was also able to learn about more detailed aspects of the user interface, such as the use of shapes, and the user experience, such as eye movement. But definitely the area that I learned about and gained the most experience in is mobile computer vision and image analysis. This includes both using the OpenCV library and within Swift. This is a subject that I had no prior experience in and developing the shoe size calculator has given me a great introduction. A central part of everyday mobile phone use is the camera. This development process has shown me how useful and valuable computer vision is in mobile development.

5.1 Next Steps

While I was able to achieve my goals with Sneaker Size, I fully believe there still remains room for improvement and growth. The main area of this is the shoe size calculator. My next steps are to improve its capability of distinguishing the coin and foot in more practical situations. This includes with other objects in the frame, busy floors, the problem of the foot not being aligned with
the bottom of the camera view etc. I also want to explore detecting foot deformities, such as bunions, and how that impacts shoe sizing. In terms of app functionality, I could definitely see a potential need for a user to user interaction feature. Whether this be simply having a comment section for each brand and model, or more of a forum for longer discussions. A large consideration of developing Sneaker Size was the male target audience. I want to explore the same concept but for two different target audiences: women and children. The most exciting one being for children. Children’s feet are always growing and, for parents, checking their shoe size is something that is a chore and needs to be done very often. Not only does it affect what shoe they would buy, but also the sock size. Being able to measure this simply using a mobile phone would be very convenient. It would also allow the parent’s to look into purchasing shoes online, instead of having to visit a store every time and reap the savings. In terms of next steps unrelated to Sneaker Size, my research and work with computer vision and image analysis has definitely propelled me into exploring it for future mobile app development projects and seeing where it can solve problems and make a positive impact to those who use it.

References:


