Protein Visualisation and Design
in
AR iOS Application

A dissertation submitted to the National University of Ireland, Cork
in partial fulfilment of the requirements for the degree of
Master of Science in Interactive Media
in the College of Science, Engineering and Food Science

2020

By
Thao Phuong Le
School of Computer Science & Information Technology

Contents

List of Tables

List of Figures

Abstract

Dummy text

Declaration

No portion of the work referred to in this dissertation has been submitted in support of an application for another degree or qualification of this or any other university or other institute of learning.

Acknowledgements

I would like to thank...

Chapter 1

Introduction

In recent years, along with the advancement of technology, there are major advances in molecular biology. Technology has become a great help for scientists and biologists aiding their research and make things easier to study. This project focuses on protein structure displaying and protein structure design.

Protein is not a single substance. There are many different proteins in an organism or in a cell that comes in every shape and size, doing a unique and specific job (Khan Academy, 2015). Proteins are considered as the "ultimate players in the processes that allow an organism to function and reproduce" (Stephenson, 2016, p.375).

Proteins are formed by linear chains of amino acids. A linear chain of protein is called a polypeptide. Each protein is formed by one or more of polypeptide chains, linked together in a specific order (Khan Academy, 2015). Protein are the fundamental components of all living cells (Wiesława, 2013). Protein has a countless number of functions in a cell or organism that are extremely important in the biology of many organisms. They form enzymes to speed the reactions up by break-down, link-up, or rearrange the substrates (Khan Academy, 2015). They from hormones to control specific physiological processes such as "growth, development, metabolism and reproduction" (Khan Academy, 2015). To maintain these roles, the shape of a protein is critical. If the shape changes, the protein will lose its functionality. There are four levels of protein structure: primary, secondary, tertiary, and quaternary (Khan Academy, 2015). Knowing the structure of a protein makes understanding how that protein works much easier. By being able to manipulate a structure of a protein, scientists can create hypotheses about how to affect, control modify them to, for example, design mutations to change functions.

This year of 2020 has once again proven the importance of molecular biology study. As of this year, we all have experienced the Severe Acute Respiratory Syndrome Coronavirus-2 (SARS-CoV-2) as is it "a newly emerging, highly transmissible and pathogenic coronavirus in humans that cause the global public health emergencies and economic crises" (Mittal et al., 2020). The number of infections worldwide had reached millions, including thousands of deaths. To find a cure, much researches have been carried out. Some research developed on the protein structure of SARS-CoV-2 has provided some insights into its evolution. As Wiesława has pointed out: "The chief characteristic of proteins that allows their diverse set of functions is their

ability to bind other molecules (proteins or small-molecule substrates) specifically and tightly." (Wiesława, 2013, p.15). the characteristic of SARS-CoV-2 is the protein spikes that cover the surface. The virus uses this to bind with and enter human cells (Wrobel et al., 2020). The spike of SARS-CoV-2 is very stable and thus help to bind to human cell tightly. Therefore, analysing the structure of theses spikes could provide clues about the virus's evolution. The study of the structure of the spike protein can aid with drug discovery and vaccine design. Understanding the new importance of implementing IT in Biology research, , this project aims to aid with protein structural study and raise interest in protein design. Due to the shortage of time and lack in experience, this project only provides the first step into bringing the visualisation of protein into AR-display and combine simple protein structure. The main goal of this project, however, is to visualise protein structure on AR using an iOS App. There are various previous studies on protein visualisation on 3D and VR, however, there has not been much on AR, especially AR app on iOS. This project's application is suggested to be displayed on AR so that it can be considered as a trial for future study and research as it might make displaying more appealing than simple 3D and cut down on the side effects of VR. All the protein models that are to be displayed are retrieved from RCSB Protein Data Bank.

Besides visualising protein, the app aims to let users design their simple protein structure. This function is suggested so that this project's app is not only appealing to biologists and scientists but also can be used by anyone curious about biology. Being able to construct a protein structure as a mini-game might make it easier for users to understand more about protein structure.

In this project, a mobile application for iOS system was developed: ProteinAR. ProteinAR has two main categories: Education and Mini-game. The "Education" category assumes that the users have already known about proteins, they can input the name of protein and get the 3D visualisation of the protein structure in AR. Users can study the protein by zooming in, turning, flipping the protein structure. This will come in handy because under a microscope, with the complexity of a protein structure, it is hard to look at not to mention to be able to interact. Being able to interact with the protein structure, and being able to discuss the structure with other fellows will help a lot in study and research on proteins. The "Mini-game", on the other hand, is user-friendly to users who are not familiar with proteins or biology in general. Users can add the polypeptide chains namely: Flex Coil, Rig Coil, Helix, Sheet onto each other to create a protein. This might make the concept of protein sounds more appealing to the user and thus, motivate the wish to study more about protein from the user who is not familiar.

Last but not least, this project will evaluate the pros and cons of displaying protein on an iOS AR app for future study research based on: - The feasibility of retrieving and uploading contents - The usability of the app (AR) - The system preference - The difficulties of implementation - The limitations of the study

Some important technical notes about the project: ProteinAR was designed on XCode 11.6, based on Swift, on MacBook OS version: Catalina 10.5.5. There is no support for AR on MacOS, thus, the built-in simulator will not be able to display the AR function and can cause some other errors. The project was run and tested on an iPhone. The attached demo video is

recorded on iPhone X, iOS version 13.6. Other versions of XCode or macOS or iOS might not be able to get ProteinAR running and thus might generate some unwanted errors.

Chapter 2

Analysis: Field Review, Literature Review and Existing Products Research

2.1 Introduction to Protein

As mentioned, proteins are "the most important macromolecules in all living organisms" (Rashid, 2015). Sequences of amino acids that bounds into linear chains create proteins. These chains have a specific folded three-dimensional (3D) shape, which enables the protein to perform a certain task (Rashid, 2015). The shape of the protein defines the tasks of it, thus, knowing the protein structure is very important. There are four different levels of protein structures: Primary Structure, Secondary Structure, Teritiary Structure, and Quaternary Structure. A sequence of amino acids in a chain form a *primary structure*. These chains, then, would fold into three different shapes of Helix, Coil or Sheet where the alpha helix, the beta sheet, and the random coils are positioned, which is called the *secondary structure*. The combinations of these formed chains of helix, coil and sheet (a polypeptide chain) would form a 3D structure – the *tertiary structure* of a protein. The *quaternary structure* is a large assembly of multiple polypeptide chains. To understand a protein's function, understanding the structure of the protein is necessary. In the same way, designing a protein from the structure will help to design its function. In this app, users will get to design protein structure by combining different protein secondary structures of helices, coils, and sheets to form tertiary structures.

2.2 Existing Solutions to Protein visualisation

"Proteins are three-dimensional (3D) objects" (Ratamero et al., 2018). The key to understanding protein functions is to understanding protein structure. Computer models for protein has become very popular for a long time. Many projects were developed to make 3D viewing of protein possible such as PYMOL, CHIMERA, VMD, ISOLDE, etc.

2.2.1 Protein Visualisation in Mobile Applications

There are numerous of mobile applications in which protein are visualised in 3D. The RCSB Protein Data Bank (the single worldwide repository of protein data) also provide a mobile appto provide data access and visualization. Basically, the protein can be downloaded directly from the PDB from RCSB and displayed in 3D. This app is based on the open-source molecular viewer NDKmol. However, NDKmol can only be used on Android and not iOS. Jmoll is another Andriod app that connects to the RCSB PDB, visualising the protein in 3D once the protein name is typed in. There are some molecule Viewers that can run on iOS devices. Unfortunately, most of them are no longer in used or was having troubled, thus, being removed from the Apple App Store. iMolview can still be used, however, the interface is not very user friendly.

2.2.2 Protein Visualisation in VR

The advancement of implementing VR in Protein Display

Visualizing protein on computer in 3D has been a great step, however, it still lacked the immersion and a true feeling of 3D presence, leads to limitation in analyzing protein structure. Virtual Reality (VR) provides a wide field of view on an immersive display and a better perception of the protein structure by head-tracking. Furthermore, VR enables user to have the freedom of hand controllers for simple manipulation and interaction with the protein instead of the conventional manipulation on 2D using trackpad, mouse and keyboard (Goddard et al., 2018). This makes VR entrance into the world of protein visualizing/molecular biology more than welcomed. HMDs[1] are commonly used because they are easy to use and becoming more and more common and affordable. VR games have become popular, thus the tools for programming software that are compatible with HMD are better and cheaper. Project such as REALITYCONVERT, AUTODESK, MOLECULE VIEWER are well developed, providing good resource for further development on protein display in VR (Ratamero et al., 2018). UNITY is largely used with the combination of HMDs such as OCULUS RIFT and HTC VIVE to display and manipulate protein (Ratamero et al., 2018).

There have been many advanced projects of implementing VR in molecular biology. The MOLECULAR RIFT – an open source tool that creates a virtual reality environment steered with hand movements, incorporate OCULOUS RIFT as the display to create the virtual setting (Norrby et al., 2015). The combination of a virtual reality experience with natural acts such as hand movements creates a much better experience for the users than just experiencing the 3D (Norrby et al., 2015).

Other research shows that the technology in displaying Protein in VR is advanced, however, tools that are designed to be installed on desktop systems are often tedious (Xu, 2019). The configurations might be different with systems and therefore causing errors and sharing between system is difficult. With the help of Web Graphics Library (WebGL), web-based applications such as JMOL , ASTERVIEWER are more straightforward as VR experiences can be directly accessed with common web browsers. However, there are many limitations for these web-based applications because it would only support a few file types and cannot perform complex

[1]Head Mounted Display

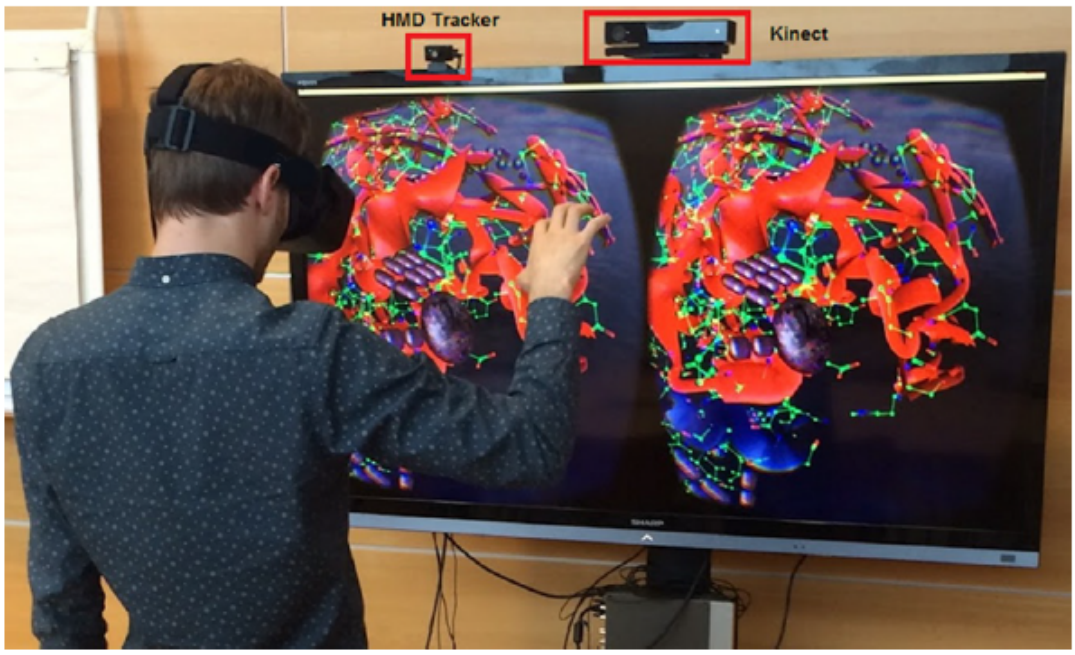

Figure 2.1: Oculus Rift (HMD) and Kinect v2 sensor placement used during Molecular Rift development

tasks for analytical purpose (Xu et al., 2019). A few solution of an integrative cloud-based system that can directly access databases and uses VR technology to visualise and analyse macromolecular structures were proposed, such as VRMOL. This might be the new direction for protein visualising in VR.

The limitations of using VR in Displaying Protein

Even though the VR implementation in displaying protein has come far and will still go further in the future, there are still some inevitable limitations. First, the limitations in the associated hardware, software may lead to an unsuccessful application of VR, which leads to the inaccuracy and imprecise in the results of using the application. With the increasing development of VR techniques and the popularity that VR games are gaining, software and hardware to be integrated with VR are becoming more compatible, but not without limitations. They are still costly and need to be increased in fidelity (**liu˙using˙2018**) Secondly, it is the unnatural feeling of using VR. Even though VR offers a realistic view, the users have to be wearing goggles which are not transparent and thus, blocking the vision of the real world. Furthermore, the head movements are unnatural because users will have to try to move their heads in order to see things. New HMDs are better because they are much lighter but mostly VR devices are still quite bulky and not that easy to use yet. Thirdly, most VR users claim to have motion sickness. This happens because of the difference the bodies and the eyes experience at the same times. The actual physical actions and the actions that are carried out in VR might be different and therefore, causing motion sickness to the users. Due to this problem, when using VR, users

cannot use it for a long time.

2.2.3 Protein Visualisation in AR

Similar with Virtual Reality, Augmented Reality (AR) generate the realism by displaying the 3D models in a real-world context. However, unlike VR where the whole vision of the users is taken away and replaced by another completely different scene, AR's defined characteristic is that it added a layer onto the vision. While VR creates an immersive experience for users by shutting out the real physical world, AR maintains the realism of the world, allowing users to see whatever they are seeing plus more. With AR, the users have free movements while projecting images. Commonly speaking, there are the most two well-known types of AR technology implementation. The first one is implemented on AR smart-glasses such as the Microsoft HoloLens, Google Glass, Apple Glass. Contrast to VR goggles, AR glasses looks just like sunglasses or even normal glasses, thus, causing no bulky feelings to the users. The second type of implementations are on AR apps such as Pokemon Go. In this type of implementation, camera's phones are used to track the surroundings environment as well as adding a layer on top of the screen to show external information.

As AR is gaining popularity, more projects are being done but not much as it is an extension of VR, it is still very new. Some studies show that AR being used in science such as molecular displayed has yielded in good results for students, as it takes less imagination and makes things more easy to understand (Cai et al., 2014). However, there are not many AR apps available to support education, specifically in visualizing molecules.

As mentioned, there are not many projects done on visualization molecules on AR. Unlike VR, where there are a various number of HDMs incorporated software and app for protein visualization, on AR, apps are more commonly used. There are only a few apps that can be found. BiochemAR is one of those. BiochemAR was released in 2019 and are available on both App store (for iOS) and Google Play (for android). According to the developers, Sung and her team, the idea of the app is to create a simple, easy-to-use teaching tools for both teachers and students in the class room (Sung et al., 2020). The main function is to display protein in AR by scanning a QR code, thus the design is very basic. Figure 2.2 the landing screen of the app.

When a QR code is scanned, the app will use the phone/ tablet/ smart devices' built-in camera to bring the protein structure into life through VR as shown in Figure 2.3.

As the main purpose is to make things simple and easy to use for teachers and students, there is no other functions as well as interactions between users and the protein. Those are just visualized and users can move the phone around to look at the protein in different angles and size.

Having the same idea, another app called AR ASSISTED VISUALISATION was developed in 2020 to visualize protein. These proteins are not written under QR code form but instead printed out on paper as in Figure 2.4.

Similar with BioChemAR, AR Assisted Visualisation only display protein structure in 3D, without any interacting elements.

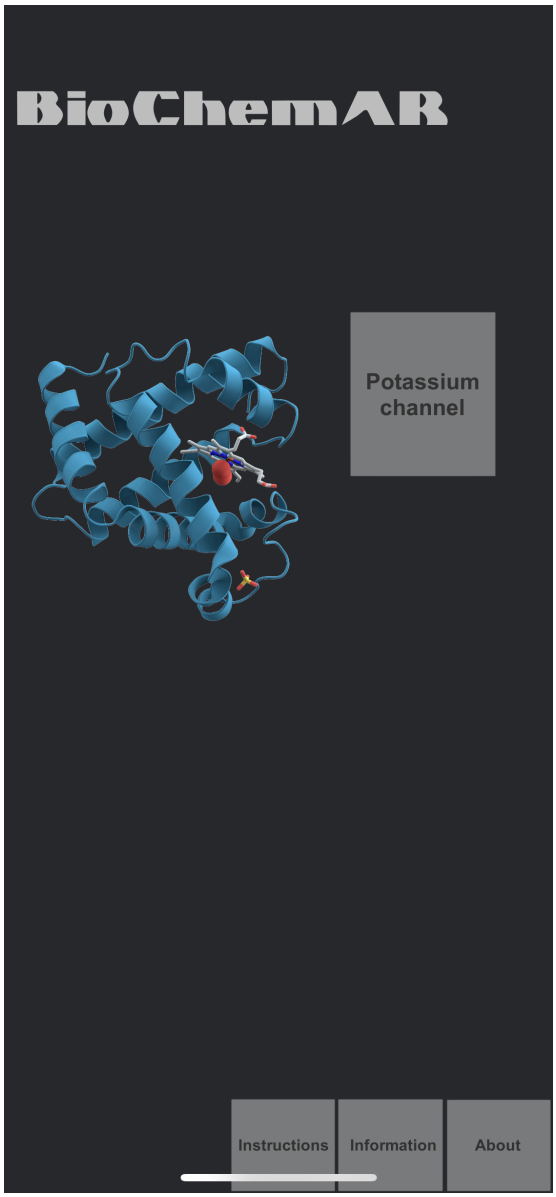

Figure 2.2: BiochemAR app landing screen

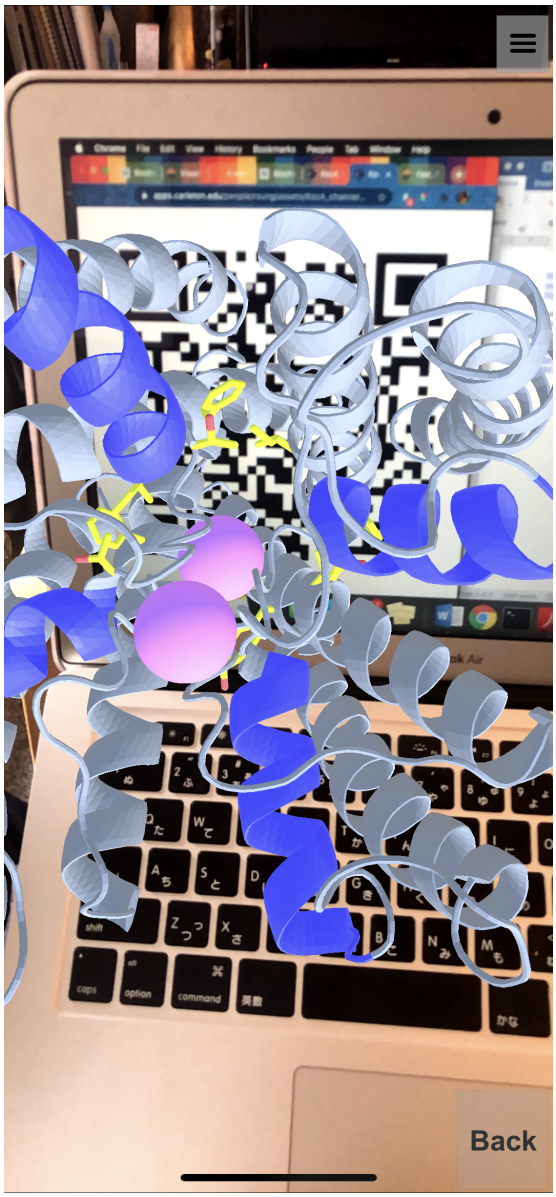

Figure 2.3: BiochemAR app displaying protein in AR

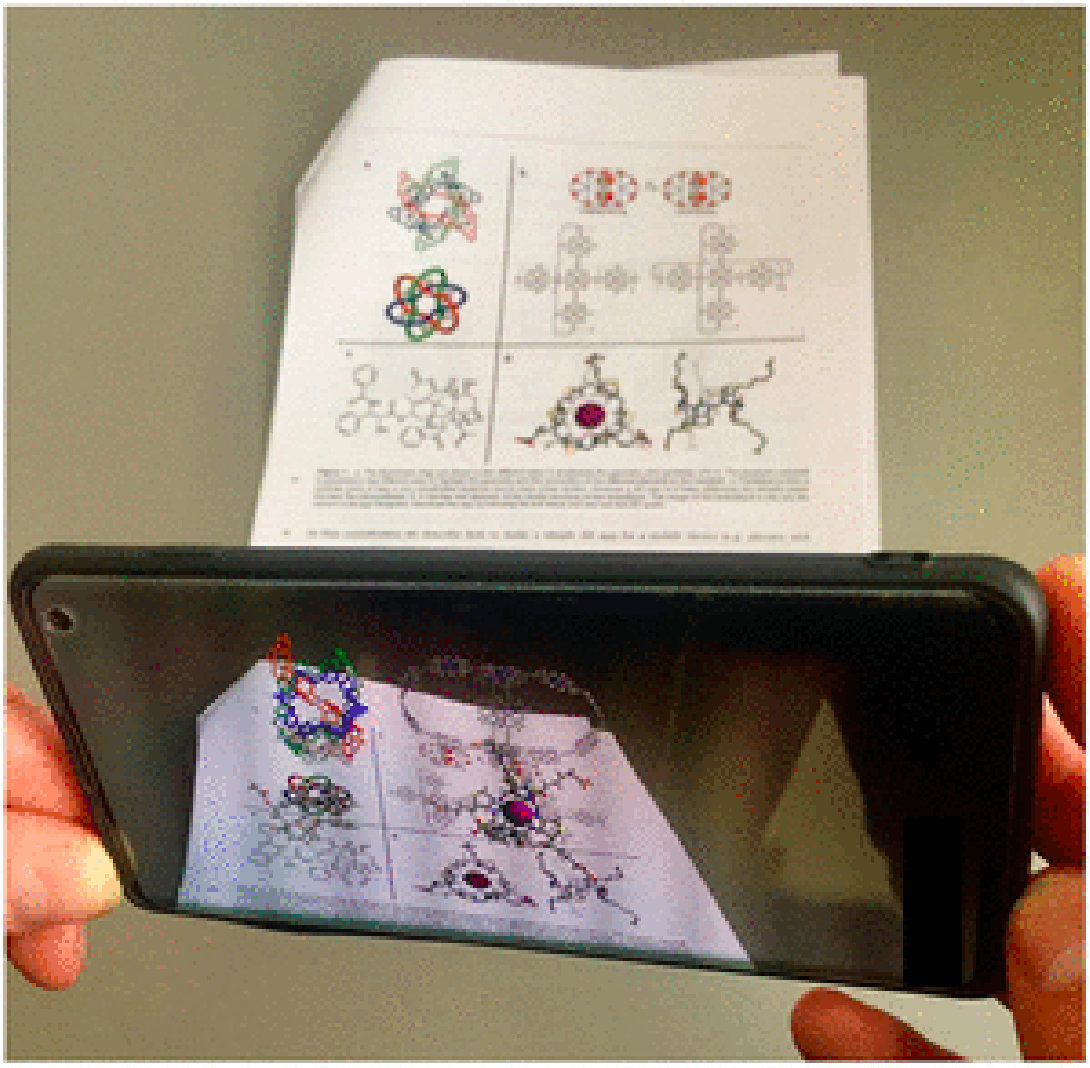

Figure 2.4: AR Assisted Visualisation App (Eriksen et al., 2020)

2.2.4 About this project

In a recent research on American Chemical Society and Division of Chemical Education, it seems that when undergraduate students get to create their own AR protein visualisation, they were enthusiastic in doing so and thus, their learning was enhanced when the AR module was inserted to their upper level biochemistry class (Arguello & Dempski, 2020). With the trend of online learning, the application of AR would promise a better curriculum for biochemistry.

Integrating protein visualisation on Mobile Apps is a good solution because of its availability. Most students have access to a mobile phone and it is handy to bring around as it is not bulky or need specific customisation.

The AR apps on protein visualisation are still new and young (released in 2019 and 2020). Thus, there is not much user interactions and functions to it. To use these apps mentioned in this thesis, a certain document with information of the protein, whether it be a figure of a protein or a QR code has to printed in order to get the AR visualisation. Moreover, the proteins can be viewed but cannot be interacted with in anyway. Furthermore, these apps are one-side oriented as users can only view protein but cannot create any.

This project's application: ProteinAR's purpose is to not only let users directly view the shape of protein in AR, interact with the protein by gesture touch on the screen but also allow users to design and create their own proteins. The majority of mobile apps to visualise protein are only in 3D, and most of the times on Android. Therefore, the open-source API for protein visualisation directly from the PDB files are limited. This project will have to start from little availability in pre-developed techniques.

With further work being put into, ProteinAR can be applied to be used in teaching to make lessons more interesting and understandable for students as well as motivate students to do higher level in Biochemistry.

Chapter 3

Methodology

ProteinAR is an app designed to run on an iOS system. It was written in Swift 5, on Xcode. The dataset in which protein files are downloaded from is directly connected to RCSB PDB. There were some other sources of protein data websites such as Protein Parameter or Protein Structure Function and Prediction I-TASSER Server were used to test out the application during the process of making. The app only runs fully on an iOS device, not a built-in simulator due to the requirement to use the camera to achieve the AR function.

3.1 Softwares used

3.1.1 Xcode

Xcode is an integrated development environment (IDE) for MacOS. It was first released in 2003, enables developers to create apps for Apple platforms. Xcode supports sources codes for various programming languages including C, C++, Objective-C, Swift, etc. Xcode has a built-in Interface Builder to construct graphical interfaces. During the making process, Xcode has a few version upgrades. The latest update was Xcode version 12. With every version updates, there are few changes in codes and functions as the main goal is to build more compact and user-friendly interfaces.

Advantages of using Xcode

ProteinAR is written on Swift, a native language for iOS apps, released by Apple and since Xcode is the native IDE of Apple, the compatibility is perfect, making the app and tests run faster and less errors. Xcode is a highly intuitive IDE where there is the main storyboard interface, visualising the designs elements of an app, with various built-in function to customise the design, from background colours to framing and a built-in library for easy adding and changing elements such as icons, pictures, text labels, etc (Introducing Xcode 12, n.d).

Disadvantages of using Xcode

ProteinAR used the built-in ARKit package. As this requires camera accessibility, tests cannot be run on the built-in iPhone simulators but instead, a real iPhone device. This creates a great disadvantages as iPhone iOS version keeps on updating, and thus, being incompatible with Xcode if Xcode is not the up-to-date version, which means MacOS should always stay as the latest version. Xcode's disk size is large, thus, downloading takes a great amount of disk space and time. Moreover, in some updates, the packages supports change, meaning there might be some errors that needed to be fixed with the newer version.

3.1.2 UCSF Chimera

UCSF Chimera (or Chimera) is developed by the University of California. This program allows interactive visualisation of protein data. Once a PDB file is downloaded. Chimera can open the files in a 3D form and allow users to interact with it.

3.2 Language used: Swift

Swift is a powerful programming language for Apple platform. Apple released Swift from 2014, taking ideas from various other languages (Rust, Haskell, Ruby, Python, C, etc.,) but it bares most similarities to Objective-C (Swift, n.d).

Advantages of using Swift

Swift was always considered as one of the *Most Loved Programming Language* on Stack Overflow for many years as it is highly interactive, with concise and expressive syntax which runs fast. There are several improvements comparing to other languages: there is no need for semi-colons, UTF-8 based encoding is used, Strings are Unicode-correct, etc. It is also designed for safety as by default, Swift objects can never be *nil*. As a successor to C and Objective-C, Swift includes low-level primitives such as types, flow control and operators as well as object-oriented features such as classes, protocols, and generics (Apple, 2020). Overall, Swift is a simple and straight-to-the-point coding language.

Disadvantages of using Swift

As mentioned above, there were a few version updates of Xcode during the programming process. Swift is a new language, thus, are being changed constantly to reach perfection. Therefore, the syntax and packages might change from times to times. It is relatively new, therefore, solutions to coding problems might be too new to have an answer, which was the most challenging in using Swift as the main coding language.

3.3 ARKit API

The technology to develop Augmented Reality was ready for mobile devices, however, it is too complex as algorithms for detecting objects in real world and displaying virtual object need

to be created, and these are very complex for developers. This is why Apple released ARKit in 2017 as a software framework, making developing an AR iOS app so much easier. It is an API that supplies numerous and powerful features to handle the process of building Augmented Reality apps and games for iOS devices.

Apple has been acquiring many AR companies, thus, the ARKit is built on all of these acquisitions. One of the major ones was the German company Metaio, which IKEA initially used to let customers display IKEA furnitures in their own home. Ferrari also used Metaio's technology to allow customer changing colours of cars in showroom, and looking at car's internal features. In 2017, Apple acquired SensoMotoric Instrument, a company specialized in eye tracking technology to use in AR. Other companies that specialized in other parts of AR technology are being acquired by Apples throughout the year. By doing this, the features of ARKit on iOS devices are frequently newly added and updated. ARKit is continuing to grow, making the creating of AR apps easier than ever (Wang, 2018)

3.3.1 Basic understanding of the ARKit

There are three layers that works simultaneously in ARKit (To, n.d). **Tracking** is the key

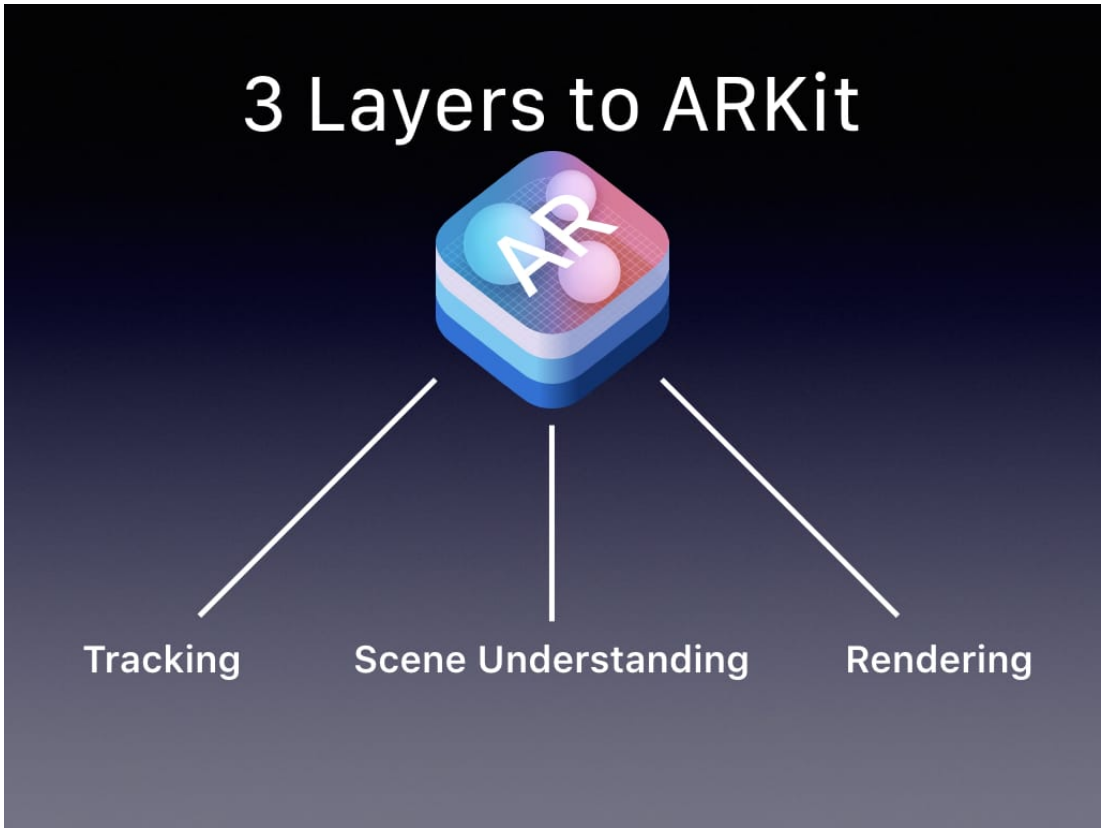

Figure 3.1: Three Layers to ARKit

function of ARKit. Without ARKit, it would be very complex for developers to write algorithm

to track a device's position, location and orientation in the real world. **Scene Understanding** is the layer that allows ARKit to analyse the environment presented by the camera's view to adjust and provide information in order to put place a virtual object on it. **Rendering** is the process where ARKIt handles the 3D models to put them in a scene such as SceneKit, Metal, RealityKit.

3.3.2 Language and System Requirement for ARKit

As mentioned in this thesis, since augmented reality requires access to cameras and high resolution display, ARKit apps can only be run on modern iOS devices:

- iPhone SE, iPhone 6s and later
- iPad 2017 and later
- All iPad Pro models

To develop an iOS app, Xcode is the best IDE to be used as it also has the built-in simulator program to mimic different iPhone and iPad models. However, with ARKit integrated, the app cannot be tested on the simulators but has to be on a real iOS devices listed above, connecting through its USB cable. Both Swift and Objective-C can be used to create an ARKit app. This project chose Swift as the language because it is much easier to learn and run faster. ARKit framework allows developers to be able to focus on the features of the app rather than on the AR required technologies such as detecting, displaying and tracking virtual object in the real world.

3.4 Database used: RCSB

ProteinAR downloads PDB files directly from RCSB.

PDB (Protein Data Bank) file format provides a standard representation for macromolecular structure data. These are obtained from X-ray diffraction and NMR studies (About RCSB PDB, n.d). RCSB was the first open access digital data resource for Protein Data Bank. It provides access to 3D structure data for all biological molecules. RCSB is a global archive where PDB data are available for free (About RCSB PDB, n.d). The data acquired on RCSB are data submitted by biologists and biochemists around the world. On the website, users can search for any protein name and the 3D structure will be displayed and can be interacted with. Information about the protein will also be displayed, and PDB files can be simply downloaded. During the process of making ProteinAR, some other sources for protein data were used including I-TASSER and ProtParam. ProtParam displays all the parameters for protein once the amino acid sequence is entered. ProtParam is a simple designed encoded website, allowing the GET method to get information from the server to the app easier, however, the PDB files contains 3D structure information of the protein is not available, therefore, it was used as a test to see if the POST and GET method work well in the app for similar website. I-TASSER predicts protein structure and function after users enter the sequence of amino acids. Similar with RCSB, I-TASSER allow free downloading of the PDB files, where the structure of protein is

already created in 3D and can be opened using UCSF Chimera. The cons of using I-TASSER is that the data cannot be downloaded in real-time because user need to enter their emails into the server and get the PDB files back a few hours later. As the goal of ProteinAR is to visualize protein structures and display them instantly, RCSB was chosen for the database as it fits the goal.

Chapter 4

Analysis: Thesis Problems and Solving

4.1 Main problems of the project

ProteinAR is an iOS application to visualise the three-dimensional (3D) structure of protein. The project was set with two main goals:

(1) Provide **educational experience** for users: download and visualise in real time protein structure with data from RCSB PDB, using the user-typed input protein name. As mentioned in chapter 2, there are a few existing apps on visualising protein on AR. However, these apps need to scan a code/ an image to display the protein which lead to the limitation in displaying the protein as the protein needed to be prepared in some form already (QR codes, images). The apps that allow direct protein structure viewing in 3D by entering proteins names also are available but not in AR. Thus, the **first main problem** to solve is to make it possible for the app to connect to RCBS PDB server, download the protein model, and display it on AR after the user type in the name of the protein.

(2) Provide **entertaining experience** for users: user can put together the polypeptide chains (coils, helix, sheet) to create new protein. As the existing apps on protein visualisation are more focused on just displaying the protein, this project is set to bring some entertaining element by adding the mini-game function in which users can create new proteins. The *second main problem* to solve is to enable users to create new proteins from the combination of coils, helices, and sheets.

4.2 Functional requirements to solve problems

4.2.1 Visualising Protein from RCSB PDB server

There are a few problems that needed to be solved in order to visualise the proteins. Firstly, the app needs to be able to send request to the RCSB PDB server. Secondly, the app needs to be able to download the files from the server. Then, the app needs to be able to track the

downloaded files' location. Finally, the app should be able to pull the files out and display them as an AR layer on the screen.

Send request and download the files

As mentioned, the app needs to be able to send information (user input) to the server and get the files back. Based on this thinking, the first try was to use the *Post* and *Get* method. This can be achieved by using HTTP Request in Swift.

HTTP POST Request allows the app to post information onto the destination URL where the specified embedded method is "POST". The way this can be achieved is firstly, to go on to the website that needed to be post on, inspect its element to find the action method as well as the parameters needed to be used in this method.

Similarly, HTTP GET Request allows the app to get information from the destination URL where the method is specified as "GET". The approach is the same with POST, usually the parameters can be found by inspecting the source code of the website, most of the times under "form action". To test out the function, hrefhttps://web.expasy.org/protparam/ProtParam was a good start as the website only consists of string type data. The URL for both POST and GET are the same and the methods are in the form action, which was no trouble to find.

However, since there is no PDB files on ProtParam, RCSB PDB has to be the data source. On RCSB PDB, the methods of POST and GET do not exist in the form function. The PDB files are directly downloaded by a separate URL in which only the only changing part (parameter) is the name of the protein. Understanding this, ProteinAR uses URLSesssion and downloadTask(). URLSession makes network transfers easy and downloadTask() fetches the contents of a specified URL, saves it to a local file and calls a completion handle. The URLSession tracks the storing place of the download task while it happens. This will be explained more with codes in chapter 6

Display the file

Interactive gestures with the models

To be interactive, the apps need to enable gesture.

4.2.2 Create new proteins from combination

Add polypeptide chains to screen

The app needs to be able to display individual polypeptide chain when the user clicks the buttons. There are four types of polypeptide chains to be added: Flex Coil, Rig Coil, Helix, and Sheet. Each polypeptide chain is input into the project as a *.dae* model. In order for these models to be loaded on ARKit, it must be converted into *.scn* files. Each model consists of different nodes: the model, lighting, camera, etc. By using the pre-defined function of SCNScene, the 3D models can be loaded into the AR view. By passing on the name of each models as a parameter, only one function of adding is needed to add four polypeptide chains using four different buttons.

All of the models are loaded on screen at the same location as if the location is not specified, the models might go off-screen. However, this caused a problem because if the same model is added twice, they will lay on top of each other, causing misunderstanding for the users as they can only see one model on screen. To solve this problem, the app randomises the orientations of the models every time a new model is added to screen by using the pre-defined function of *eulerAngles* to specify the *SCNVector3* with random x, y, and z.

Combining polypeptide chains

After adding individual polypeptide chains on screen, ProteinAR must be able to combine these chains into proteins. The combinations might be successful and might not be. For this to happen, successful combinations of these chains are pre-loaded into the apps in a "Combinations" folder.

In the code, an empty string array for protein name is created. Every time user adds a polypeptide chain to the screen, the name of the protein is added using *append* to the array. After user clicking the "Try" button to combine the polypeptide chains, the names in the array are joined using the *array.joined()* function. The models' name in the "Combinations" folder have a naming convention so that when the array are joined, the name it generated matches with the name of the models in the "Combinations" folder. See more chapter 6 for further understanding.

4.3 Non-functional requirements

Core Data

Constraint

Protein Combination Models Database

Chapter 5

Project Design

5.1 Application Design

The structure of ProteinAR is very simple. It consists of four main screens including the landing screen.

To achieved the set main goals and still create a user-friendly app, the two goals are separated to be achieved on two different views. From the landing view (first view), user can choose to go to the "Introduction" about the app, or go to "Education" where visualization of protein happens or "Mini-game" where new proteins can be created by joining the coils, helix and sheet in different order. This app was created based on single view iOS app and then the other views are added later with segues to create the connection between them.

User interface

Introduction to user interface

In order to ask the computer/smart devices to do anything, users need to communicate with them. The way users (human) can communicate with the product (software, app, website) is through interacting with the user interface (UI) of that product. The purpose of a UI is to enable users to control a computer or a device they are interacting with, by giving orders and receiving feedbacks in a chain to complete a task. The user interface of any software or website does not only create the first impression to the products which makes user decide instantly if they want to use the product, but it also plays a big role in keeping the users. With a complicated or not efficient UI, users would not want to keep using the product because it takes memory efforts. Therefore, an UI should be *intuitive* – be kept simple where no training should be needed to operate, *efficient* – functions are precise and on point, and *user-friendly* (User Interface, n.d) Currently, there are three formats of user interfaces (What is User Interface Design, n.d?): • **Graphical User Interfaces (GUIs)** – interactions happens through visual representations on digital control panels such as a computer desktop, a website interface. • **Voice-controlled interfaces(VUIs)** – interactions happens through voice representation such as Siri, Google Home or Alexa. • **Gesture-based interfaces** – interactions happen through physical motions in 3D spaces in VR games.

ProteinAR's user interface

Chapter 6

Project Implementation