MSc Project Report

COMP0117 Research Project – Literature Review

Mobile Virtual Annotator

William Herbosch

A review submitted in part fulfilment of the degree of

MSc (Hons) in Computer Graphics, Vision and Imaging

Supervisor: Prof Niloy J Mitra

Department of Engineering / Computer Science

University College London

2018 - 2019

Declaration / Disclaimer

This report/review is submitted as part requirement for the MSc Degree in "Computer Graphics, Vision & Imaging", at University College London.

It has been substantially prepared on the basis of my own work. Where other published and unpublished source materials have been used, these have been explicitly acknowledged in the text.

It may be freely copied and distributed provided the source is explicitly acknowledged.

Word Count: 4797

Student Name: William Herbosch

Date of Submission: April 11th 2019

Signature:

Table of Contents

This literature review's objective is to introduce the reader to the problem(s) presented by this project, inform them on current trends and any present-work done regarding it and to provide key foundations for how this project's proposed work will either attempt to solve or further aid in the understanding of said problem. It provides the goals that are to be achieved, the motivation behind taking on this project and lists all material initially surveyed at the time of writing. It inscribes the following subtopics:

- The Problem
- Aims and Goals (Program Development and Project Documentation)
- Project Motivation
- Theory and Related Literature (Current State of the Art and Resource Survey)
- Personal Approach (Implementation and Consideration of Risks)
- Final Proposition (Discussion of Evaluation Methods and Potential Conclusions)
- Summary of Deliverables (for both Report and Programs)

1. The Problem

The primary goal of this project is to develop an annotation system for smartphone devices that will allow the user to 'make notes' within an AR environment. The secondary goal would then be to allow the system to work across multiple smartphones and potentially make it readily available on mobile app storefronts. This is an independent project and is therefore not being produced as a requested deliverable for third-party developers; it is a project of pure academic interest and achievement.

Aims:
To develop a virtual annotator for mobile devices using neural networks (NNs), depth images and point cloud environment scans (PCEs).

Deliverables:
1. Show familiarity with neural networks and annotation-based systems and their implementations in Python.
2. Develop a basic depth map neural network that trains on set of depth images (where each pixel has a depth value) and use it to convert fed-in images into their own depth images. Ideally, this is to work with live-fed video input.
3. Once the depth image is produced, provide the user tools to obtain information from the image, i.e. given two points on an image, compute the length/size spanned between them.
4. Investigate and become acquainted with smartphone app development, particularly with IOS's ARKit, Swift and how to incorporate Python.
5. Develop a gesture-based AR neural network that is able to read simple hand-gestures and from that deem what the user is attempting to achieve. The prime example is to allow the user the ability to 'annotate in AR space'.
6. Optional: design a few models through Unity or Blender than can be imported to the system and viewed in AR. It should also be simple to make any new models be readily accessible to the system itself.
7. The final program will be a software app that uses both depth-mapping and gesture-based annotation through AR. It will have a full object-oriented design, with a full implementation life cycle using modern software engineering principles. This product will have a graphical user interface and be useful for quality of life or work purposes.
8. The final report will describe the theory behind any algorithms used for the neural networks, the implementation issues necessary to apply relevant theory, the software engineering process involved in generating depth images and gathering information from them and the integration process to mobile devices and its potential deployment.

2. Aims and Goals of the Project

This section provides an overview of the various aims and goals for the project. It describes the program development process as well as the report and documentation writing made in relation to time, scheduling and the project's life cycle.

2.1 Program Development

For the initial few weeks of the project, not much effort is planned to be put towards actual project development. Instead, it will be to concentrate on two main factors that will contribute to every NN made for the project: to find a suitable integrated coding environment (IDE) to write the programs and to spend a considerable amount of time building the required skillset and researching on neural networking principles. This also includes seeking and understanding potential software development tools and external libraries that would further aid to the development of the networks to be produced. Simple proof-of-concept programs such as a single-neuron network and a basic multi-layered network will be developed as initial tests of self-understanding for having learned these principles. These proof-of-concepts will later be further developed not only towards the first major deliverable of the project (a depth map NN) but also as a means of determining optimal network parameters and internal settings needed to produce effective and precise outputs. While scheduling of the project's content will ultimately fluctuate based on progress made at any given moment during development, this initial research on NNs as well as the deliverance of the depth map NN is planned to be done by the end of June 2019.

Whereas the first-half of the project's growth will be used to get everything properly initialised as to create the foundations of a camera-oriented piece of software, the progress made during the second-half will be more focused on adapting this work and incorporating it into a functional AR-based smartphone application. As before, some time will initially be spent developing simple proof-of-concept programs (i.e. touch-screen communication, utilization of ARKit [19], etc) to justify a means of self-comprehension of concepts before attempting to build an app around the previously made NNs. Careful reconsideration and in some cases re-evaluation of some deliverables might be made at this point to ensure that the final submission displays a substantial amount of effort and progress spent on the project as a whole. The application itself will attempt to combine both the NN and AR methodologies learned throughout development and incorporate a graphical-user interface that is both suitable and presentable for professional use. Furthermore, submitting a GUI-based deliverable would provide a more practical application as a final product, obtaining a certain level of interactivity as to engage with clients during demonstrations. This app is planned to be given a name and be fully functional by the beginning of August; this will allow the remainder of that month to be spent on user testing, adjusting the app through feedback and potentially implementing some additional features, should time allow.

The programs are to be safely stored in a GitHub [20] repository made explicitly for this end-work. This will allow a means of monitoring the project's growth while also permitting my personal supervisor to track progress made and to view when deliverables are uploaded.

2.2 Project Documentation

The purpose of project documentation is to document the requirement details of the final product. These documentations will consist of a collection of reports, each one representing either the development of a particular component of the NN or app or a small researched-based case study on some concept, theorem, architecture or algorithm related to the topics presented. Once a program is fully developed and commented, time will be set aside to examine the notes made during the development period and to write a summarised report

on the project milestones that were accomplished, including what the end product was (i.e. what issues there were while testing it and whether the end product differed from expectations). This is done so that each component of the final product can have its development recorded in such a way that the documentation is flexible if changes occurred through development while still remaining relevant when the project is eventually completed. Ultimately however, these documents are essential to displaying the gradual evolution of the project's growth, while also providing good content for self-reflection.

3. Project Motivation

The motive behind this project is to produce an AR annotation software that benefits from best NN practices by developing a network capable of observing and interpreting the dimensions of its environment. Once the basics have been learned, multiple reports will be written to provide a more in-depth look at the deep learning and app design aspects of the topic. All of this will be carried out by studying various NN schemes and structures, building these networks from the ground up, documenting their development cycles according to software engineering practices and measuring their efficiency using some score-ruled system before adapting them to an AR framework.

This project plans to first explain what NNs are and how they operate. Once a general understanding has been given, this will be followed up with a full report that documents the NNs developed for this project that will describe in detail the exact features and operations that are implicit to the project's end goal, while also providing real-life case examples of such networks, potentially as a means of providing examples of how others resolve issues that were encountered during the development phase. Once complete, focus will shift towards the production of an IOS application that layers the produced NN in conjunction with ARKit to produce a virtual annotator, which will also be documented in a second report in a similar manner. Both reports shall also contain information about any third party tools and software used in the project's development. Once the application has been developed, more documentation will be provided on case-testing, which would consider user-feedback to adjust the design of the app as to increase its functionality and to improve the network's ability to successfully label observations.

For the external tools and libraries to be used, the decision to write the majority of the project in Python [21] was a major motivation towards undertaking this project as it represents the language that was most proficient by this student; having certificates from the British Coding Society for it and having written two previous master-theses on artificial intelligence planning and neural network algorithms, which heavily demanded the use of Python for working on relatively large datasets, only further cements this.

What hopes to be achieved from working on this project is a deeper understanding of how NNs are developed and given purpose through appropriate application use, as well as how they can be used for everyday tasks. Furthermore, there is potential to implement some ideas gathered during time spent at UCL as this allows for the demonstration of new skills and knowledge gained throughout the year. There is also the matter of app development, which presents the opportunity to meaningfully work on something that had always intrigued this student, particularly with learning the Swift language. As for the motivation of this project, it would have to do with a personal interest in artificial intelligence; from designing blueprints for Lego robots, to writing domain and problem files for planning systems, to working with datasets for hand-written digit recognition. Using knowledge gained from previous experience in deep learning from earlier higher-education experience, as well as the two theses mentioned, this project hopes to not only improve upon existing knowledge of AI systems, but to also aid in future career choices as an aspiring AI Developer.

4. Theory and Related Literature

This section is written with the dual purpose of informing the reader of current state of the art technologies that have produced similar end-works to this project's deliverables and to gloss over the collected resources that will aid the project. This is achieved by observing published key papers, projects and programs that present themselves as solutions to the problem issued, analysing any algorithms or approaches they utilize and how they might offer insight as to how this project's programs are to be written, tested and evaluated.

4.1 State of the Art

Before we can even attempt to understand what a virtual annotator means, it is best to identify what an annotator is in the context of this report. The Cambridge English Dictionary defines annotation as a means "to add brief yet concise notes, explanations, comments or opinions to a text or diagram"[24]. Broadly speaking, they are primarily used for revision of the work they are addressing or as feedback used for further improvement; they represent a form of organisation, both on an individual and general sense. The ability to apply this to the virtual domain would be an equivalent of digitally-augmented sticky notes that not only would be personal to the user but also discards the needs for physical space and/or contact.

The most prominent project examined is the Gesture-Based AR Annotator by Columbia University[15]. This HoloLens-based system acts as a virtual sketcher that allows for augmented "air-drawing" within a 3D space, capable of labeling and referencing objects of significance, i.e. a means of annotating the real world. This system combines depth-mapping with hand-recognition by assuming that the user's hand will always appear within certain distance/range from the camera, thus the lens scans this region, identifies the hand, interprets its gesture and annotates the corresponding background environment accordingly.

A more recent study in 2018 showed how NNs could be used for visual depth mapping for flight and automobile navigation systems[9]. This project proposed the use of recurrent convolution NNs to compute depth images (seen in Fig. 1), where the darker a pixel is, the closer the object represented in that pixel is to the camera. It uses Microsoft's AirSim UAV simulator to produce a vast dataset of images, from which 80% was used for training and the remaining 20% for testing. These images were then fed through RCNNs to produce results, as they "achieved state-of-the-art performance in various image translation tasks"[9].

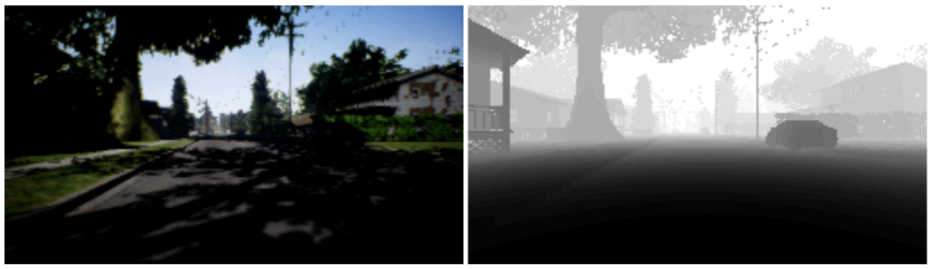

Fig 1. Example comparison of camera-image (left) and corresponding depth map (right) [9]

It is also worth addressing the more practical applications of a virtual annotator and how it might benefit both clients and industries as a whole. The Daqri Smart-Helmet is a form of AR workspace intended to improve productivity by "overlaying an image of a computer onto the real world"[14] via a head-based display, even acting as a means of allowing users to view typically non-visible features. The significance of this product is that it allows for multi-system viewing across various devices, meaning that multiple users can view, adjust and create notes on a 'shared' annotation system, thus allowing for a team-based project management framework. Another project even suggests the use of such a system to annotate crime-scene environments through AR to aid forensics in collecting evidence[4].

Fig 2. Promotional material for the Daqri Smart-Helmet [14]

One final case study to be considered as a major influence over the project's direction is IKEA's Place app for smartphone devices. Acting as a "virtual room-designer consultant"[13], it is able to place life-size models of furniture from IKEA's vast catalogue within a given space. This is mainly done to remove the hassle of clients carrying out measurements of the space they wish to furnish before shopping or the off-chance of users accidently realizing that the furniture does not fit after already purchasing and bringing it home. The app initially scans the environment to produce a unique ground-mesh, where it can then place 3D models in a suitable aspect. Here, users can translate, rotate and orientate around the model. Scaling of said models occurs with distance between the model's positioning and the smartphone camera, creating a sense of perspective. EON Reality's 3D Annotator[2] takes this one step further by providing vivid descriptions and visual cross-sections for individual components for complex model, such as the design of a turbine engine.

All cutting-edge research mentioned are considered as such due to their impact on the topic and for improving understanding of relevant theories. They have each offered solutions to the problem at hand and shall be taken as viable and inspirational guides as to how this project progresses and its accomplishment.

4.2 Resource Survey

The project's research consists of neural network development, 3D-object modelling, IOS app development (including interface design), cooperation methods between these and any relevant topics that contain algorithms and constructs that helped during production. Most sources used to determine what was the current state of the art were found through online searches, which provided up-to-date developments on current projects, research, products and operations. The remainder of resources, primarily ones that revolved around the more theoretical intricacies behind such programming, were discovered through UCL's Library, both through physical visits and by searching key-phrases via their online catalogue.

Python was the chosen language agreed upon with the project supervisor to be used to develop the NNs as it offers a simple, mathematical-oriented syntax, as displayed in Python for Data Analysis[22] and Test Driven Development for Python[23]. It could also be beneficial to incorporate a Python GUI using the PIL and TKinter libraries[25] as this shows how the project could be visually explained to people researching similar topics, as well as having something to demonstrate the project's findings to supervisors.

A research project based on 2D annotation[1] offered a basic understanding of two core components that an annotation system should possess; labelling and highlighting. Furthermore, 3D models that will be incorporated into the program are to be modelled in Unity, which also demands an understanding of how to import said models into AR by means of ARKit and IOS's unique coding language, Swift[6][7][8]. A paper on finger-tracking could also prove to be useful when designing the sketcher component[10]. Lastly, optimisation methods were considered to further improve the NNs and annotation system as a whole[3][5][11][12].

The professional issues section of the report will mainly contain contributions from online research via articles, video documentaries and interview transcripts. A more descriptive analysis of these sources will be present in the final report.

5. Personal Approach

This section is devoted to describing the personal approach taken to produce a virtual annotator, or at the very least a system capable of labeling and emphasizing objects within a 3D AR environment. This project is to be achieved using a mixture of personal knowledge from previous experience, knowledge gained during time spent at UCL and knowledge gained by analyzing current state of the art technologies.

The three utilities as specified in the Aims component of section 1: The Problem, demand an adequate understanding of both machine learning, 3D model design and mobile app development practices, so the software tools and languages planned to be used in the project have been chosen due to their optimal and preferable application range.

5.1 Implementation

Before work even began, it was decided on what programs and pieces of software were to be used in order to fulfil the requirements and deliverables specified. This would provide a better sense of the project's learning outcomes and allow the application of different integrated development environments (IDEs) and end-systems to varying degrees.

Python is an interpreted, high-level, general-purpose programming language designed to work with large and/or complex datasets, which makes it highly coveted for neural network development and deep learning practices. Libraries such as Tensorflow, Scikit-learn and Keras all contribute to this. While the ultimate goal is to get this working for live-images, the initial testing and development phases will utilize a small collection of still images personally taken by the student, which will allow for a more comprehensible understanding of how far certain objects are from the camera. It also has access to the numpy and open3D libraries, which will be needed for data analysis and mesh/point-cloud visualisation respectively. The coding environments chosen to be worked with for this project will be a mixture of PyCharm[16] and Spyder[17], two IDEs by companies JetBrains and AnacondaInc respectively that provide suitable Python interpreters and debuggers needed for the project.

Although not its main focus, one feature of the project is the ability to import 3D models and examine them in the AR space. These simple models are planned to be designed in both Unity and Maya[26] from reference (provided with appropriate texturing) and will showcase skills attained from both the Virtual Environments and Computer Graphics courses at UCL. Apple's ARKit[19] is also planned to be heavily utilized throughout the project as it offers the best means of producing AR-based apps via its application programming interface. This presents a requirement for an understanding of not just basic Apple app development (done primarily through Swift, Apple's general-purpose compiling language based off Objective-C) but also augmented-reality production. Furthermore, ARkit has been made more readily available now than before since it came with IOS 12 devices, effectively making smartphones one of the biggest AR platforms.

While the entire scope of the project has yet to be finalized at the time of writing this review, one of the most essential deliverables for this project is the development of a depth-mapping or "depth estimation" software, which is to be achieved by designing an appropriate neural network. The program would train on a collection of pre-made images with corresponding depth-images as a validation set; i.e. the predicted output of the network for

a given image is considered correct if it matches the true output. The program would take an input image, create a single vector of all its pixels and feed this into the network as its input layer, where each element of this layer represents a node or reference point for the image. The network itself is planned to be a multi-layer, convolution network that attempts to classify certain aspects of the image, where every proceeding layer tries to obtain more precise results before outputting a final solution. This is possible thanks to pixel-correspondence, where the output vector of the network is simply restructured to the original dimensions of the input image. It is worth nothing however that it is paramount that the outputted images are non-normalized across all outputs as the range of distances for one image might differ from others. The network would also adjust its internal parameters for optimization upon each iteration of learning.

An NN should be able to learn from multiple examples in order to better improve itself; the more examples the network is given, the more accurate it becomes as it now has more points of references for it to classify new observations. This would require an adequate, readily-available dataset that is approved by the community. After conducting some sufficient research, an appropriate distance-image dataset does exist in the form of the NYU Depth Dataset V2[18]. Available as both short video-sequences and still-images, the depth-images go towards brighter colours for objects that are further away. An example image from this dataset can be seen below in Fig. 3.

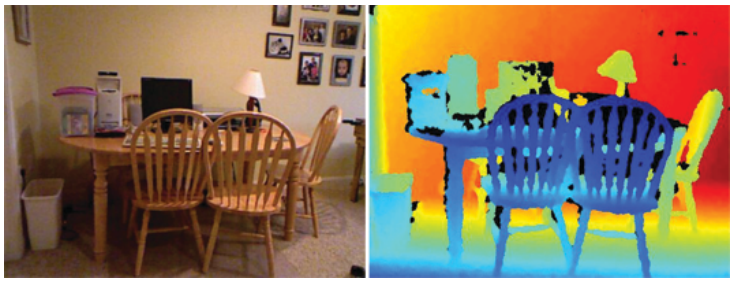

Fig 3. Example from distance-image dataset comparing raw (left) and depth (right) images [18]

What's important here is that the produced depth-image can essentially be converted to create a Point-Cloud mesh representation of the environment by considering the position at which the image was taken as an origin, and spanning from there as a form of projection. With this environment being made, certain interactive elements can now occur within it through the use of ARkit such as model placement and the actual annotation of objects within said space. This will allow a demonstration of knowledge gained during the Aquisition of 3D Geometry course at UCL, as this also deals with the application of 3D PointClouds and meshes.

It is at this point in development where the project should consider features that separate it from other virtual annotators. Such ideas for features include the ability to compute the length spanned between two selected points within the environment, being able to retrieve previously annotated (i.e. saved) data, and a model store that can be updated and modified to import new models, allowing for a loose database-structured system (hosted locally for simplicity). All of this is in consideration of a single image of the environment however; should the project demand more immediate results, the NN would have to adapt to accept live-fed fields of view taken by the smartphone camera.

The methods used here to describe the approach this project will use to tackle the problem presented relates back to previous work as mentioned in section 4.1 State of the Art, in the sense that it looks towards projects such as the *Visual Depth Mapping from Monocular Images using Recurrent Convolutional Neural Networks* (i.e Fig. 1) and extensive papers like *Studying gesture-based interaction on a mobile augmented reality application for co-design activity*[5] and aspires to achieve a similar output, albeit on a smaller scale.

5.2 Risks and Mitigations

This section is devoted to addressing all the potential risks that could be encountered during the project's development. This list will continuously receive updates for every new obstacle that needs to be challenged whenever they present themselves. For every mention of a risk, a potential solution was derived so that the issue could be resolved quickly and efficiently.

1. **Potentially spending too much time studying neural networks or app development rather than actually developing them.**
   Likelihood – Moderate
   Download and install tools ahead of schedule to save time devoted to working on the project, managing time wisely and focusing only on the fundamentals of what needs to be discussed. Frequently ask project supervisor for advice, tips and guidance during project meetings and discussions.

2. **Limited knowledge about the development and structure of smartphone apps and related coding languages to begin with.**
   Likelihood – Low
   Become familiar with IOS app development as quickly and efficiently as possible such that work on planning UI and software integration can begin on an immediate level.

3. **Available resources might be out-dated or too recent to garner enough credibility.**
   Likelihood – Moderate
   Tend to favour more present-day resources, though older papers and articles might still provide useful background info, as AR annotation is still a very modern discussion.

4. **Limited or prevented access to example or testable neural networks either online or via the computer science department, or at least their source code.**
   Likelihood – Low
   Look for alternative code samples, research more potential resources from the library or ask guidance from professors and advisors, perhaps search for some via outside sources.

5. **Allowed access to examples from the department and accidently alter its code or prohibit it from running as it used to, effectively rendering it useless.**
   Likelihood – Low
   Maintain some form of back up of the original copy so that the clone version can be examined and experimented on, thus keeping a copy intact.

6. **Insufficient or lack of library and online sources.**
   Likelihood – Low
   Seek out third party sources for potential solutions. Perhaps visit libraries from other universities.

7. **Not knowing enough knowledge about neural networks or mobile app development that certain forgoing risks cannot be predicted.**
   Likelihood – Moderate
   Be prepared for anything and know the best approach on how to tackle it both efficiently and swiftly. Study neural networks and app development to the best effect during the first few weeks of the project's lifecycle rather than encounter an issue further down the pipeline.

8. **The test data for the neural network is significantly too large for the final program to operate on.**
   Likelihood – Moderate
   Split the data accordingly; so long as the general 80% Training and 20% Testing ratio remains constant.

9. **The development of one or more program features encounters some issues, thus hindering or halting the progression of the final product.**
   Likelihood – High
   Once the main program has been developed, functionalities should be treated as independent horizontal slices to ensure each ones gets completed before another is considered (i.e. the final program's applicability will depend on how many features were successfully implemented into it). This will ensure no time is lost during the development period, and features that could not successfully be developed could be referenced as their own segment in the final report.

10. **The neural network is of poor quality or too complex to read, train and produce data properly (performance issues).**
    Likelihood – Moderate
    Improve some of the features of the network to make it more efficient to solve the problem at hand.

11. **Designing the general UI for the mobile app might consume a lot of time and resources.**
    Likelihood – High
    Use time effectively and stick with a simple design during the development process; worrying about the visual presentation should be considered only towards the final submission, as functionality is more important. Prioritise on ensuring the app works as intended rather than how efficient it is.

12. **Cannot complete the report, a program or task in time due to unforeseen circumstances (cannot hand in work on time).**
    Likelihood – Moderate
    Research for a possible solution and implement it into the program while noting that I am referencing it according to standards. Work extra during time off if necessary to catch up on any missed work.

13. **Final program(s) unable to execute properly or does not work as intended (productivity issues).**
    Likelihood – High
    Learn trouble-shooting methods and create test cases to break down the problem. If problem persists, research some sources online or in the library. Only email project supervisor as a last resort for an expert's opinion on the matter.

14. **Unable to understand key concepts introduced by the project.**
    Likelihood – Moderate
    Ask for assistance from personal adviser or search for guides online. In the extreme case, perhaps even consult a tutor, or book an appointment with a leading expert.

15. **Using existing code written by other people.**
    Likelihood – Low
    Always give credits to other code developers as to avoid copyright and plagiarism.

16. **Not enough time understanding and developing the neural networks or app, thus the design process is rushed.**
    Likelihood – Moderate
    Rushing the design process is harmful to the development of the final program as it could lead to significant design flaws. It is best to spend time towards perfecting architectural design rather than waste programming hours as designing is a crucial part of any software development.

17. **Loss of code, progress, hardware, etc. by forms of theft, carelessness or actions outside my control.**
    Likelihood – High
    Loosing any progress, regardless of quantity, would be detrimental to the project as a whole. It is thus a necessity to use GitHub repositories and cloud-storage services such as DropBox to frequently store and update any progress made on the project. Maintain a form of 'project diary' and frequently document general notes. Keep a close eye on hardware at all times and backup data on an external hard drive.

18. **Certain software and plugins only work or compile properly on specific OS systems or languages.**
    Likelihood – Moderate
    Be sure to include some form of 'set-up' file in the project submission that describes in detail the step-by-step procedures and system requirements for allowing the programs to run. Be sure to initialise them from scratch personally to see whether they operate as intended.

19. **Final Program does not work as intended during demonstration.**
    Likelihood – Low
    Produce and upload/submit video recordings of me running the demo successfully and include it in the submission.

20. **The original scope of the project is too big to deliver all assigned deliverables in time for the September 1st due date**
    Likelihood – Low
    Monitor the work that has currently been made in the project's development cycle and from that, determine what other aspects or deliverables can be made within the given timeframe. This will allow the project specification to evolve and adapt to what is currently being, and what will be developed, until it eventually culminates in an end product that reflects the personal work and effort done to reach this goal.

6. Final Proposition

6.1 Evaluation

The desired outcome for this project is an improved understanding of NNs and app development and being able to present the final work as an example of what this student can achieve as an individual. Seeing as the project possesses multiple layers of functionality, a measurement of success would be either how many features are ultimately implemented or how efficient the program runs as a whole. The project itself will be organised with a SCRUM development cycle, where weekly meetings will be made with supervisors to explain what was achieved and what is to be worked on next.

6.2 Conclusions

To conclude, this student is highly optimistic about the prospects for working on such a project, as this will allow for a display of all expertise learned throughout higher education. This includes the capability to work independently by self-teaching the knowledge required for comprehending relevant theories and concepts and the refining of programming skills. Technical decisions will be made such as how to present findings in a manner that would allow for showcasing knowledge learned through software design and engineering modules. All of this work will be composed and completed in a report, ranging on topics from historical implication to modern-day experimentation, all while using scheduling methods such that progress on the project does not falter. Areas that failed to deliver will be learned from by experiencing their consequences first hand.

All aspects listed will ultimately make this student more aware of what goes into the development of programs and software, and will make the student understand what is required of a computing professional, as the skills learned are very valuable to one who wishes to further their knowledge on the wide applications of machine learning.

7. Summary of Deliverables

The final report will be written with a dual-purpose; to display an understanding of how the NNs operated and to explain their design process and to detail its further development from a simply program into a fully functional smartphone app. This would then lead into a basic comprehension of how the network was constructed, methodologies used to optimise its learning processes and the formulation of a suitable interface with tools that fills the project's requirements. A summary of the report sections and programs can be read below.

7.1 Sections of the Report

1. **Introduction**
   Present the problem at hand and provide the motivation to go about solving it.

2. **Background**
   Provide an adequate understanding of relevant topics and current developments within the field.

3. **Analysis and Design**
   Give a high-level overview of the project's design process, including any change made throughout the project's lifespan.

4. **Implementation**
   Explain written code used for the programs and discuss important aspects of it.

5. **Testing**
   Verify that the system works as envisioned.

6. **Results**
   Details of the experiments used during testing, their preference over other experiments and the importance of internal parameters and their affects on results.

7. **Conclusion, Evaluation and Further Work**
   Show what was achieved and what could have been improved upon, plus self-critique.

8. **Professional Issues**
   Consider the ethical and social implications that might influence the final product and provide suitable countermeasures.

7.2 The Programs

1. **Depth Map Neural Network**
   Using Python and neural network development skills learned over previous years, this program will convert raw images (potentially life-video) into unique depth maps.

2. **Annotation Software**
   This program would build off of the depth map neural network to offer additional layers of functionality, primarily to obtain information about the environment the system is attempting to map (annotate, obtain space dimensions, model placing, etc).

3. **Final Product: Integration to Smartphone Devices**
   The end-work of the project, where all previous work is constructed into a fully functional smartphone app, which includes the ability to view the same annotations across multiple devices.

Bibliography

[1] *A. von Kapri, Annotating 2D Objects in Augmented Reality, Columbia University of New York, 2007, video showing AR 2D circles fixing their positions on edges of a leaf.*

[2] *EON Reality, AR 3D Annotation, EON Reality, 2017, video showing models in AR and how they can be applied for training/educational purposes.*

[3] *S. White, L. Lister and S. Feiner, Visual Hints for Tangible Gestures in Augmented Reality, Columbia University, 2017, paper investigating gesture-based communication between the user and AR system, their meanings within the system and which are most optimal.*

[4] *A. Gee, P. Escamilla-Ambrosio, M. Webb, W. Maylo-Cuevas and A. Calway, Augmented Crime Scenes: Virtual Annotation of Physical Environments for Forensic Investigation, University of Bristol, 2010, conference proceedings that describes a system for annotating objects within a crime-scene environment through AR to aid forensics in collecting potential evidence.*

[5] *L. Gul, Studying gesture-based interaction on a mobile augmented reality application for co-design activity, Springer International Publishing, 2018, documentation for a series of user-studies for understanding behaviour while using mobile AR applications as a means for how to better design such systems and their interfaces.*

[6] *J. Glover, Unity 2018 augmented reality projects, Packt Publishing, 2018, book on how to model and integrate creations in Unity for develop AR applications for ARKit.*

[7] *S. Shekar and S. Haney, Swift Game Development, Packt Publishing, 2018, book for understanding AR game development by learning IOS 12's Swift coding language for ARKit.*

[8] *J. Linowes and K. Babilinski, Augmented Reality for Developers, Packt Publishing, 2017, book for building AR applications through Unity and ARKit.*

[9] *J. Mern, K. Julian, R. Tompa and M. Kochenderfer, Visual Depth Mapping from Monocular Images using Recurrent Convolutional Neural Networks, Stanford University, 2018, article that presents methods to estimate object distances for flight-based systems by producing depth maps using deep neural nets.*

[10] *W. Hurst and C. van Wezel, Gesture-based interaction via finger tracking for mobile augmented reality, Springerlink, 2012, article that investigates finger tracking for gesture-based AR for mobile device touch screens.*

[11] *J. Shim, Y. Yang, N. Kang, J. Seo and T. Han, Gesture-based interactive augmented reality content authoring system using HMD, Springer-Verlag London, 2016, article on the proposition of a gesture-based AR system applied to virtual objects intended to introduce AR creation and usage on a simplified level.*

[12] *L. Ng, S. Oon, S Ong, A. Nee, GARDE: a gesture-based augmented reality design evaluation system, International Journal on Interactive Design and Manufacturing, 2011, article that displays the benefits of both physical and virtual prototyping in design evaluation using AR.*

[13] *IKEA, Place, Ikea, 2017, smartphone app used for visualising true-scale Ikea products within a real environment.*

[14] W. Sherman and A. Craig, Understanding Virtual Reality: Interface, Application, and Design, Morgan Kaufmann Publishers, 2002, book on VR where one section highlights the fundamentals of AR annotation systems and provides some real-life examples such as the Daqri Smart Helmet.

[15] Y. Chang, B. Nuernberger, B. Luan, T. Hollerer and J. O'Donovan, Gesture-Based Augmented Reality Annotation IEEE VR 2017 Demo, University of California, 2017, series of videos that demos some features of a developed annotation system that allows for 'air-drawing'.

[16] JetBrains, PyCharm, JetBrains, 2010, a Python IDE

[17] Spyder Project Contributors, Spyder, AnacondaInc, 2009, a Python IDE

[18] N. Silberman, P. Kohli, D. Hoiem and R. Fergus, NYU Depth Dataset, New York University, 2012, dataset of distance images that will be used to train the networks for this project.

[19] Apple, ARkit, Apple Inc, 2017, application for using AR systems on IOS devices.

[20] P. Hyett, Github, Microsoft, 2008, web-based source-code management system.

[21] G. van Rossum, Python, Python Software Foundation, 1990, an interpreted, high-level, general purpose programming language.

[22] W. McKinney, Python for Data Analysis, O'Reilly Media, 2012, book on Python syntax

[23] H. Percival, Test-Driven Development with Python, O'Reilly Media Inc, 2014, book on Python syntax

[24] Cambridge, Cambridge Advanced Learner's Dictionary, Cambridge University, 1995, proper definition of the term 'annotator'

[25] J. Grayson, Python and TKinter programming, Manning Publications, 2000, book on Python GUIs

[26] Alias Systems Corp, Maya, AutodeskInc, 1998, 3D computer graphics application

README:

(Submitted as part of the document as I hear Turnitin has issues dealing with zip files)

Might I ask for a digital sticker from the Disabilities department to be considered for this coursework. Please feel free to get in touch with me personally and I shall forward my Summary of Reasonable Adjustments form. Many thanks.

Effort was made to stay within the 5000 word limit, which is why the following texts were not taken into consideration for this word count:

- Cover page
- Declaration / Disclaimer page
- Contents page
- Page headers / footers
- Titles and subtitles in general
- The bullet-points and the Aims and Deliverables sections on page 3
- Figure descriptions on pages 6 and 9
- Risks (in bold) and likelihood measurements from pages 10-12
- Annotations within text (example [1])
- Bibliography