
**School of Computer Science and Engineering**

**Faculty of Engineering**

**The University of New South Wales**

# Smart Home

by

Jake Palandri

Thesis submitted as a requirement for the degree of

Bachelor of Engineering in Software Engineering

Submitted: April 2024

Supervisor: Prof. Claude Sammut Student ID: z5313097

# Abstract

This document describes the requirements to theses submitted for the Bachelor of Engineering in Software Engineering degree at the School of Computer Science and Engineering. Requirements described are that of both of context and layout of the theses. The document is written using the LaTeX template provided by the school.

# Acknowledgements

This work has been inspired by the labours of numerous academics in the Faculty of Engineering at UNSW who have endeavoured, over the years, to encourage students to present beautiful concepts using beautiful typography.

Further inspiration has come from Donald Knuth who designed TEX, for typesetting technical (and non-technical) material with elegance and clarity; and from Leslie Lamport who contributed LATEX, which makes TEX usable by mortal engineers.

John Zaitseff, an honours student in CSE at the time, created the first version of the UNSW Thesis LATEX class and the author of the current version is indebted to his work.

# Abbreviations

**ECHO IV** Electronic Computing Home Operator

**USD** United States Dollars

**CSA** Connectivity Standards Alliance

**AI** Artificial Intelligence

**ML** Machine Learning

**IoT** Internet of Things

**CV** Computer Vision

**IR** Infrared

**Yolo** You Only Look Once

**AAL** Ambient assisted living

**VCR** Videocassette Recorder

**DVD** Digital Video Disc

**BMS** Behaviour Monitoring System

**PIR** Passive Infrared

**ROS** Robot Operating System

**WSL** Windows Subsystem for Linux

**Distro** Linux distribution

**VM** Virtual Machine

**IAI** Institute of Artificial Intelligence, University of Bremen

**ASCII** American Standard Code for Information Interchange

# Contents

# List of Figures

# List of Tables

# Chapter 1

# Introduction

Having a set of clear requirements to their thesis is important to student finalising their BE, or other, degree. Such requirements are both in relation to the physical appearance of the thesis, as well as the writing style and organisation. The present document tries to concisely state the theses requirements while appearing in layout and structure as a thesis itself.

# Chapter 2

# Literature Review

## 2.1 Background

### 2.1.1 History of the Smart Home

The evolution of smart homes can be traced back to the early 1900s when the introduction of various electric household appliances promised to reduce the time spent on mundane household chores and free up more time for leisure, revolutionising domestic life. Devices such as the electric mixer and iron, the world's first refrigerator in 1913, and the pop-up toaster in 1919, emerged in this period, bringing about the beginnings of a more automated home environment and the following couple of decades followed suit.

In 1966, there was a significant leap forward in home automation technology when Jim Sutherland, an engineer from Pittsburgh, Pennsylvania created the ECHO IV. The Electronic Computing Home Operator (ECHO IV) was the first device designed specifically for home automation and was hand-crafted with surplus electronic parts. With the ability to compute shopping lists, control home temperature, limit children's television time with questionnaires and even tell the weather, this was the first glimpse at a whole-home computerised system that was capable of automating every element

of home life.

The following year saw the introduction of Honeywell's Kitchen Computer. A computer designed for housewives to use in the kitchen as a recipe storage device with a built-in chopping board. At $10,000 United States Dollars (USD) in 1967 (nearly $100,000 USD today when accounting for inflation), this recipe storage device which had no display, was well described as "amazingly beautiful and hopelessly impractical". No units were ever sold. Despite its failure on the open market, the device received a lot of attention and the public began to imagine a future where homes would be interactive.

Eight years later, in 1975, a group of engineers from Scotland released the X10 protocol. Capable of controlling up to 256 devices on a single circuit, X10 sent messages to devices through a property's existing AC electrical wiring. At the time, this was an efficient way to send basic signals through large spaces before the widespread adoption of wireless technology in all electronic devices. t later advanced to be controllable from a computer, enabling users to schedule events and run decision-based sequences. The protocol formed the basis for many domestic control installations for several decades and was one of the first protocols to completely cover the home automation spectrum, with power, security and lighting.

Throughout the 1980s and 1990s, the idea of having robots as companions was solidified in popular culture through science fiction movies. During this same period, advancements in battery technology and the rapid decrease in the size of microprocessors meant that home robots became achievable, and by 1991, the Electrolux Tribolite was released, the world's first robot vacuum. This class of device is now a quintessential smart home product and was a turning point for the industry. The robot vacuum cleaner heralded the integration of the first wave of smart home devices that were not hard-wired into the building. Along with this, the launch of the first-ever internet-connected refrigerator from LG, in the year 2000, demonstrated the increasing integration of technology into everyday household appliances.

This brings us to the current century, where, in the early 2000s, technology began to boom. The home computer had become commonplace and the internet had become

more accessible and understandable to the general public. More smart devices, like speakers that speak to you about news headlines and weather or act as an alarm, began to appear on store shelves at affordable price points and home automation became a realistically achievable goal.

Today, smart homes are now a reality, they are not exclusively for the eccentric and wealthy. They provide homeowners comfort, security, energy efficiency and convenience at all times, regardless of whether they are home or not. Through the use of innovative technology, homeowners can turn their homes into state-of-the-art machines that can be controlled and monitored from anywhere in the world.

It is clear to see that with every iteration of the development of increasingly quasi-intelligent smart devices, early adopters were promised a more comfortable lifestyle with menial tasks being delegated to machines, saving more time for leisure. With the beginnings of electric home appliances reducing the time required to dedicate to cooking and cleaning, through to interactive devices capable of holding simple conversations with users, this rapidly growing technology shows no signs of slowing down and its future capabilities are near limitless.

### 2.1.2 Problems with Smart Homes

This is not to say that smart home technology is flawless. There are several factors that limit the advancement of the industry and the effectiveness of smart devices as a whole. Most of the problems with modern smart homes can be grouped into three main categories:

- Interoperability
- Lack of true intelligence, and
- Requirement for internet access

#### 2.1.2.1 Interoperability

There are many different competing standards in the world of smart homes, including devices that use Zigbee, Z-Wave, Bluetooth and Wi-Fi protocols. This means that any given smart home device may not be compatible with another device, even if they are from the same manufacturer. This can be frustrating for consumers who want to create a customised smart home environment that meets their specific needs and preferences when selecting their products and discourages their entry into the market.

There is currently a new protocol, called Matter, in development by the Connectivity Standards Alliance (CSA), in partnership with some of the leading smart home companies, that aims to unify each of these standards into a single, open-source standard and allow legacy devices to be able to communicate with one another. This is a step in the right direction, but if history is anything to go by, this may just end up being an additional competing standard, with lengthy certification processes driving away smaller manufacturers, and, in turn, competition in the market.

On top of this, there are many competing smart home platforms and hubs, such as Amazon Alexa, Google Home, Samsung SmartThings and Apple's HomeKit. Even if the consumer does manage to choose devices all running on the same communication protocols, they may still be unable to control them all from a single app or interface. Apple and Google are both notorious for having lengthy and expensive certification processes for third-party developers to integrate their devices into their platforms, which can be a significant barrier for smaller manufacturers, splintering the market further.

This usually leads to one of two possible outcomes for the consumer. Either consumers end up locked into a single ecosystem, unable to switch to a different platform without replacing all of their devices, which can be a significant financial burden. Or, they end up with a mix of devices from different manufacturers that are unable to communicate with one another each with its own dedicated app, which can be inconvenient and more time-consuming than not having smart devices at all.

#### 2.1.2.2 Lack of True Intelligence

Despite their name, smart homes are not actually all that intelligent in their automation abilities. Even in some of the most advanced smart homes, the devices are limited only to simple routines and schedules, and basic decision trees relying on primitive data from a limited number of sensors in the home.

With the recent advent of Artificial Intelligence (AI) and Machine Learning (ML) technologies, it is theoretically possible to create systems that can learn from user behaviour and adapt to their preferences. However, most smart home devices are not yet capable of this level of predictive control, and instead rely on the user to program them to perform specific tasks at specific times.

#### 2.1.2.3 Requirement for Internet Access

Finally, the requirement for constant internet access may be one of the most significant inhibitors of the expansion of the smart home industry. Many smart devices sold today often needlessly process all of their commands remotely, meaning that they require constant internet access to communicate with their respective cloud services in order to be able to control the devices, even from within your own home. There are very few options on the market for devices that allow for local processing of commands. For example, Google, Amazon and Apple's respective voice assistants, Google Assistant, Alexa and Siri, all require an internet connection to process voice commands so without one, they are unable to control any devices in the home.

So if your internet connection goes down, you may be left with a house full of smart devices that are suddenly very dumb. Not only will you not be able to access the devices remotely to control them or monitor your house, but you may also lose the ability to control them from within the same network as they cannot connect to their company's servers.

## 2.2 Review of Existing Literature

### 2.2.1 Computer Vision for Smart Home Automation

In a paper published at the 2019 International Conference on Communication and Signal Processing, Mohammad Hasnain R. et al. set out to "develop a smart [Internet of Things] (IoT) based light control system" using computer vision (CV) and artificial intelligence (Hasnain, S, P, & G, 2019). Their objective was to reduce the wastage of electricity "due to [the] negligence and forgetfulness" of people using the environment.

The authors correctly identify one of the biggest issues with presence detection in common home automation deployments. Most setups use an array of infrared (IR) sensors to detect movement in a room but this poses a few challenges. Firstly, any movement in view of the sensors will trigger an action that is intended only for human presence. So if a book falls off a shelf or an animal runs past and interrupt the IR beams then the sensor will report back a false positive result. Additionally, another limitation of this method, which was not outlined in this paper, is that if a person remains stationary in the room, while sitting on a couch for example, then there will be a false negative result suggesting that there is no human presence in the room.

To solve these downfalls of existing implementations, they took a different approach, utilising AI to identify people through a camera in a living space, enabling them to differentiate between objects and people. However, there are a couple of areas that they did not explore that were within reach using the setup that they had implemented and that this thesis intends to expand upon.

For starters, the authors only used the sensors as a toggle for LEDs in the aim of reducing unnecessary energy use. This thesis aims to implement a system capable of executing more complex tasks such as controlling other smart devices in the house like a television or blinds with more potential states than just off and on. Furthermore, to identify people in the camera, they used You only look once (Yolo), a computer vision AI model for used object detection, classification, and segmentation, which also has a

built-in "skeletonisation" feature, allowing you to track a person's posture and make more intelligent decisions based on a person's movements. These two ideas present an opportunity to fill a potential gap in the market making more intelligent decisions in the home using existing technology and real-world implementations.

Aside from this paper, there has been extensive research into the use of computer vision in smart home deployments. Nonetheless, very few of these discuss the integration of computer vision with smart home devices, and even fewer attempt to implement this integration. There is lots of discussion around computer vision in homes but it is mostly regarding intrusion detection and security (Cucchiara, Grana, Prati, & Vezzani, 2005; García, Meana-Llorián, G-Bustelo, Lovelle, & Garcia-Fernandez, 2017; Nandhini, Rabik, Kumar, & Brahma, 2020; Patchava, Kandala, & Babu, 2015; Sefat, Khan, & Shahjahan, 2014; Zhang, Yi, & Saniie, 2019).

V. Patchava et al. implemented a front end for a smart home with the ability to toggle devices and stream the video feed from a camera with computer vision capabilities, while C. González García et al. implemented a system that could detect the presence of people and measured the rate of true positives, false positives, true negatives and false negatives (Patchava et al., 2015; García et al., 2017). Others implemented presence detection with CV and were able to identify intruders and any weapon that they are carrying (Cucchiara et al., 2005; Nandhini et al., 2020). These papers discuss facial recognition software and the ability to alert home owners of potential intruders. They even discuss integration with custom smart home dashboards and IoT platforms, but they do not discuss the integration of these systems with smart home devices and automation, which is the primary focus of this thesis.

### 2.2.2 Human Action Recognition for Smart Home Automation

Another paper, from the Department of Computer Technology, University of Alicante, covers some potential methods of person tracking, human action recognition and behaviour analysis using cameras in the home (Chaaraoui et al., 2013). Here, there is more of a focus on using this technology to support the elderly and the disabled, through

ambient assisted living (AAL) to "improv[e] their quality of life and [maintain] their independence."

A. Chaaraoui et al. discuss their deployment with both RGB cameras collecting information and recognising "key poses" and hand gestures by creating silhouettes of the person, and also by using RGB-D cameras using the depth information to more accurately track movements. They were able to achieve this using a Microsoft Kinect camera and depth sensor "with low-cost and real-time performance", which is the same sensor that was used for preliminary testing in this thesis as will be covered in more detail in Chapter 4. This existing deployment shows promising signs that the proposed setup should work smoothly in a home environment.

While they were able to get these systems working in real-time, recognising primitive human actions such as standing, walking, sitting and falling, there was no mention in the paper of actual integration with home devices.

In 2000, J. Krumm et al. followed very similar processes, using background subtraction to create silhouettes of people and overlaying depth data over the top of these cutouts through stereo images (Krumm et al., 2000). However, due to their early adoption of the technology they were somewhat limited by the computational power of computers during this period and had to run far more complicated and unwieldy setups that would not be suitable for a real home deployment today. This meant that their trackers ran at only 3.5 Hz, even with the computational load shared between three computers, and often had trouble tracking more than 3 people at a time or people wearing similarly coloured clothing.

They were able to integrate their system with some home devices with custom programs such as a wireless mouse that could be carried to any table in the room and the clicks and movements of the mouse would be redirected to the nearest computer display. Another program "automatically start[ed] and stop[ped] a [Videocassette Recorder] (VCR) or [Digital Video Disc] (DVD) movie when a person sits on or stands up from a couch." The film would also be "automatically rerouted to different displays in the room, depending on where the person [sat]."

These are similar objectives to what this thesis partially aims to achieve, however, the technology has advanced significantly since then and the proposed setup should be able to run on a single computer with a single camera and depth sensor, making it more accessible to the general public.

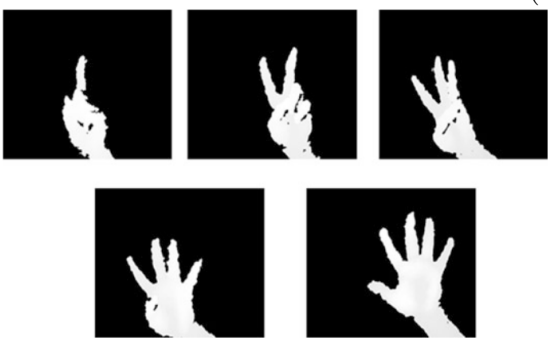
### 2.2.3 Gesture Recognition for Smart Home Automation

In 2017, S. Desai and A. Desai, proposed an algorithm for gesture recognition for home automation, also using the Microsoft Kinect (Desai & Desai, 2017). Their proposal was specific to only recognising hand gestures, no full-body tracking. This involved segmenting the hand from the image at a specified depth range of 250-650mm, removing the noise and background from the image and converting the RGB into a solid binary black and white image, as shown in Figure 2.1. The finger positions were then compared against a set of predefined gestures and then classified. The classification then defines the action that the system should take, sending a signal to an arduino board to control different home appliances such as a television, charger or fan.

Using this technique, they were able to achieve an 88% accuracy rate in recognising the gestures. This is not really a high enough accuracy rate for a real-world deployment, as users would have to repeat the gesture multiple times at least one in every 10 attempts in order to get the desired result. Additionally, this only worked when users were giving gestures within 65cm from the camera, and only within a 40cm range. This is not practical for day-to-day use as this would either require dozens of cameras per room, pread out every 1.3m, or it would require users to get up from wherever they are in the room and move to the camera in order to control their devices. In which case, they may as well control their devices manually. Due to these limitations, tracking gestures only via hand movements does not appear to be a viable solution for smart home automation, and full body tracking may be required to make this technology more practical for everyday use.

This paper, among others, refers also to computer vision technology recognising gestures through wearable devices (Krumm et al., 2000; Starner, Auxier, Ashbrook, & Gandy,

Figure 2.1: Extracted Hand Gestures from Microsoft Kinect (Desai & Desai, 2017)



2000; Volety & Geethanjali, 2022). Implementing a system to track wearable devices opens up many possibilities for gesture recognition as well as the potential to integrate medical monitoring tools with it. In fact, using the wearable device, T. Starner et al. were able to track the user's gestures against control gestures, which measure continuous input from the user, with an accuracy of 95% and user defined gestures with an accuract of 97% (Starner et al., 2000).

Using a device that interacts directly with the user's body, is bound to significantly increase the accuracy of gesture recognition, as the device is able to measure it's own movements, and hence the user, as opposed to a potentially distant camera that attempts to infer the user's movements based on depth data. However, this creates a significant hurdle as the user must be wearing the device at all times in order to control their devices, which significantly impedes the user experience and reduces the inconvenience that this technology is supposed to alleviate.

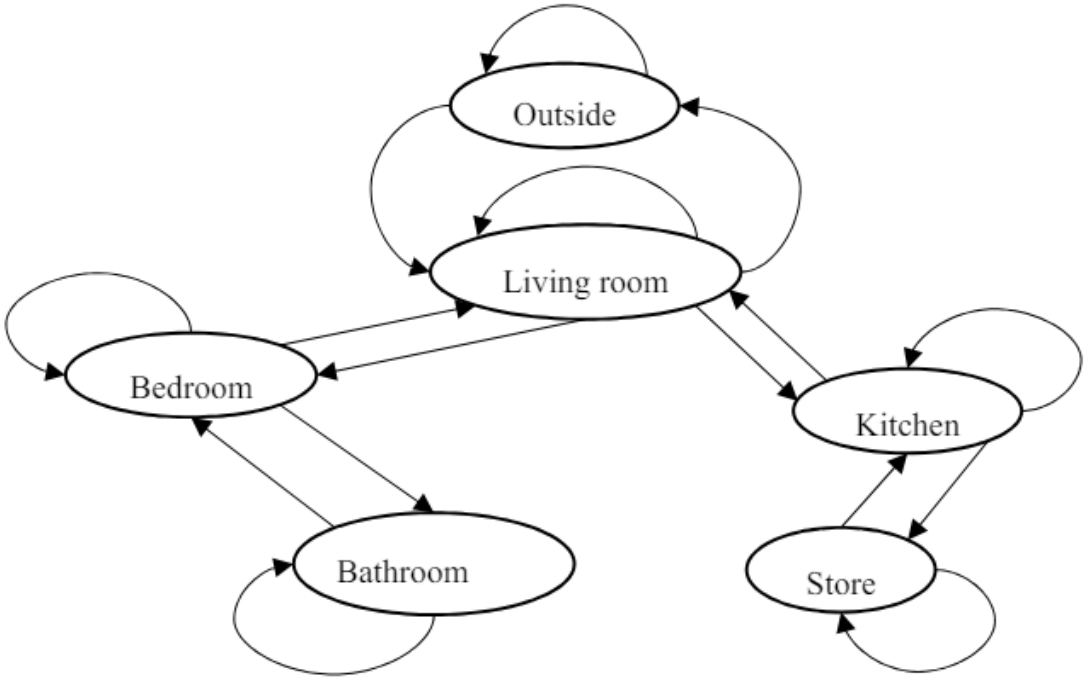
### 2.2.4 Behaviour Monitoring for Elderly Care

In a 2017 paper, "A Behaviour Monitoring System (BMS) for Ambient Assisted Living," S. Eisa, and A. Moreira constructed an environment that could monitor the behaviour of elderly people in their homes (Eisa & Moreira, 2017). The system used passive infrared (PIR) sensors to monitor the movement of the elderly person between rooms and detect any deviations from their normal routine. The theory behind their practice was that "[a]n elderly person remains in good health as long as he or she can carryon

his/her daily activities as usual with no significant deviations from the normal daily routine." This meant that if the system could identify any irregularities, then it could alert a carer or family member to check in on the user to ensure they are okay and potentially catch any health issues early.

This was implemented by treating the user's location in the house (by room) as a state machine, with the user able to transition states by moving between rooms, as shown in Figure 2.2. From this, a transition matrix is created for a given time of the day, which represents the probabilities of the user either staying in the same room or moving from any room to another. Training an AI model to recognise the user's regular behaviour, the system could then recognise any deviations from this behaviour. If the behaviour was deemed abnormal, it would then be classified to determine possible causes of the deviation, such as the user oversleeping if they are in their bedroom for too long, Table 2.1 shows the abnormal behaviours that the authors listed.

Figure 2.2: Room-to-room State Transition Model (Eisa & Moreira, 2017).




The model they implemented also self updated once every week so that it could adapt to the user's "behavioural changes that are not nevcessarily abnormal behaviours" over

Table 2.1: Abnormal Behaviours—Descriptions and related health-declines (Eisa & Moreira, 2017).

| **Behaviour** | **Description** | **Health-Declines** |
|---|---|---|
| OverSleeping | An extended stay at bedroom; longer than usual. | Mobility problems, strokes. |
| LessSleeping | Detected motion at one of the rooms, not a bedroom, during the usual sleeping time. | Having sleepless time due to anxiety, depression or may be developing Alzheimer's diseases. |
| NotBackHome | Monitored person stayed outside longer than usual. | Having trouble coming back home or get lost or wondering outdoors. |
| Dead | No movement and long stay at one of the rooms, not bedroom nor outside; | Death. |

time and account for seasonal changes in behaviour. This meant that they were able to implemet a behaviour monitoring system that learned to adapt over time and consistently monitor the user's behaviour for any irregularities with primitive PIR sensors. If this was achievable with such simple sensors, then the possibilities with more advanced depth sensors and computer vision should allow for more advanced behaviour monitoring and prediction.

There is also some overlap in this field with the computer vision and gesture recognition technologies that have been discussed. M. Oudah et al. implemented a hand gesture recognition system, similar to that of S. Desai and A. Desai, but with the aim of assisting the elderly and disabled in their daily lives by allowing them to use gestures to get support if communication is a problem for them (Oudah, Al-Naji, & Chahl, 2021). With working deployments of both of these technologies in the AAL field, this shows promising signs for potential in the home automation field.

## 2.3 Gaps in Literature

After a thorough review of existing literature in the space that is being explored by this thesis, it is clear that there has been extensive research into many of the facets of the proposed system in consideration. Computer vision is a widely researched field and has been applied to many different areas of smart home technology, from security to

gesture recognition. Human action recognition and gesture recognition have both also been studied for decades.

Despite this, there is a significant gap in the literature when it comes to the actual integration of these technologies with smart home devices and automation. Most of the literature either discusses the technology in isolation or discusses implementation into smart homes on a theoretical level. For example there are many papers that discuss the use of computer vision for intrusion detection and security in a smart home, but none of them consider the possibility to use this technology for automating tasks and controlling devices in the home. Almost none of the papers have actually implemented the technology into a real-world smart home environment and tested the results. This presents a significant opportunity for this thesis to contribute to the field by providing a real-world implementation of these technologies and evaluating their performance and user experience.

There is also an opportunity to dive into a new area of research by combining these technologies with behaviour prediction to create a truly intelligent smart home environment. Behaviour prediction and monitoring has been studied in the context of care for the elderly and disabled for ambient assisted living, but not in the context of smart home automation. If done correctly, combining these technologies could allow for a smart home environment that can predict a user's behaviour and automate tasks in the home based on these predictions, without the need for user input.

## 2.4 Problem Statement

Smart homes today promise enhanced convenience, safety, and security through the connectivity of more and more devices and the automation of regular tasks. However, several critical challenges prove to be a hindrance to their effectiveness as well as widespread adoption and, in turn, the growth of the industry.

Interoperability issues between devices and existing platforms, the lack of genuine intelligence in these automated systems, and the constant reliance on internet access pose

significant obstacles to realising the true potential of smart home technology.

In an attempt to combat these challenges, this thesis aims to develop an entirely custom and customisable, intelligent environment where a user's movements will be able to accurately predict how to control devices within the home, through gesture recognition and behaviour prediction. Utilising local processing and sensor data, the system will enhance convenience, safety, and security without relying on internet connectivity or support from third-party companies to allow interoperability.

Moreover, with several gaps in the existing research and literature into this area, this thesis aims to be able to integrate computer vision techniques and depth sensor technology into a smart home environment, utitlising this data to assist in automation, not just home monitoring and security. This will provide a real-world implementation of these technologies and a live demonstration of the home's capabilities, as opposed to the theoretical implementations that are currently discussed in the recent literature. The implementation of behavioural prediction technology, commonly used in AAL systems, will also be explored to predict user behaviour and automate tasks in the home based on the user's regular routines.

Overall, this research endeavours to contribute to the evolution of smart homes by introducing innovative approaches to enhance their intelligence and autonomy, ultimately improving the quality of life for users.

## 2.5 Aims and Outcomes - DRAFT

The research objectives of this thesis include designing and implementing artificial intelligence and computer vision for movement prediction and gesture control, integrating these into existing smart home infrastructure, and evaluating the performance and user experience of the environment.

With entirely locally processed data, the system will be able to control devices within the home without the need for internet connectivity, enhancing convenience, safety,

and security for users. Computer vision techniques will be used to identify users in the home, and track their movements and gestures to control devices, while AI will be used to predict user behaviour and automate tasks in the home.

The anticipated outcomes of this research include advancements in smart home technology, improved user experiences, and insights into addressing the broader challenges in the field as outlined in the literature review. While the research focuses on addressing specific aspects of smart home functionality, certain limitations, such as device compatibility and privacy concerns, will be acknowledged and considered throughout the study.

# Chapter 3

# Project Plan

## 3.1 Methodology and Approach

## 3.2 Timeline

## 3.3 Required Training and Upskilling

# Chapter 4

# Project Preparations

The first phase of this thesis consisted mostly of many attempts at getting a working environment set up on a Linux machine, running Robot Operating System (ROS). The main goal was to be able to allow depth cameras to communicate with ROS 2 and control simple devices through this system. ROS was chosen as the platform to control devices as it is already used by many of the devices set up in the lab and robots on the university campus. The lab also had Microsoft's Kinect v2 depth cameras already accessible and was known to have previously been successfully set up and used with ROS so this became the sensor of choice.

While this all seemed simple in theory, having had previous work prove that the work was possible, in reality, the Kinect v2 integration with ROS was not regularly maintained and ROS had a new major version release in this period so it required a significant amount of work to get the device to communicate with a ROS 2 distribution.

## 4.1 Progress and Roadblocks

The first step was to get a ROS 2 distribution running on the machine and learn the inner workings of the operating system. The ROS website and documentation suggest that an installation on a Windows machine is possible and is supported but

the overwhelming response from members of their community and in their forums was that the Windows installation was not an easy task and often performed worse. The consensus was that a Linux installation was the most reliable and safest option. This was the first of many times that the environment setup had to pivot and try some alternative method.

Running ROS 2 on Windows Subsystem for Linux (WSL) felt like the next logical step as this would allow ROS 2 to run on a Linux installation without a significant change to the machine that this was being run on. It was quickly discovered that WSL would not be a suitable operating environment as the visual tools of ROS would still be required for debugging and testing. Soon after, a successful installation of ROS on native Windows was achieved but with very little documentation or support from the community, it was clear that this was not the best option and was only really suitable for small, simple deployments.

This then led to a pivot to running Linux on a virtual machine within Windows. Using the Linux distribution (distro) Ubuntu 22.04, Jammy Jellyfish, was the first environment setup where the ROS 2 distribution Humble Hawksbill was installed and ran smoothly, with tools like RViz and the built-in turtle simulator working as expected. After getting ROS running, and also throughout the remainder of the setup process, there was a significant amount of time spent learning the ROS environment and how to use the tools provided with it, including sending messages between node topics and creating custom messages and services. This was both in the terminal commands and through the Python client library for ROS.

The next step was to get the Kinect to communicate with ROS 2. To do so, the Kinect needed to communicate with the virtual machine (VM) through USB passthrough. Out of the box, VirtualBox, the VM software used to run the Linux distro, only supports USB 1.0 and USB 2.0 devices, while the Kinect requires a USB 3.0 connection to send the large amount of data that it captures. Installing Oracle's VirtualBox Extension Pack allows USB 3.0 devices to connect to the VM through the host machine, however, it was later discovered that the Kinect v2 drivers for Linux do not support connection

through a virtual machine.

Again, this meant pivoting to a new environment setup and dual-booting the machine to run Linux natively. This way it was possible to get OpenKinect's Libfreenect2 drivers for the Kinect v2 working and it was now possible to capture colour and depth data from the sensors using software provided with the drivers.

It was soon realised that there is no existing integration for the Kinect v2 with ROS 2 as the hardware at a decade old is now slightly outdated, having been succeeded four years ago by the Azure Kinect DK. Since this was the only depth camera available on hand, it was decided that the best course of action would be to attempt to get the Kinect working on the known solution of ROS 1 and then use the ROS Bridge to forward the data to ROS 2. Using the most recent version of ROS 1, Noetic Ninjemys, the Kinect was able to send depth and colour data to ROS at several resolutions using a fork of the Kinect bridge from the Institute of Artificial Intelligence at the University of Bremen (IAI), that supported OpenCV 4.0, an open source computer vision library.

Open Robotics provides a bridge between ROS 1 and ROS 2 that allows for communication between the two systems and translation of the topics, services, and actions between nodes in either direction. Using ROS Humble, ROS Noetic and the ROS 1 bridge, it was possible to send data between nodes in ROS 1 and ROS 2 across the bridge. When using the Kinect bridge at the same time, there were struggles getting all parts of the system to work stably and consistently. There appeared to be much more documentation and support for a setup using a previous distribution of ROS2, Foxy Fitzroy so this was the next change needed to be made for the project.

Unfortunately, ROS Foxy does not run on Ubuntu 22.04, so the distro had to be downgraded to Ubuntu 20.04, Focal Fossa. This was the final roadblock encountered in preparations trying to get the Kinect working with ROS 2. On Ubuntu 20.04, with ROS Foxy, ROS Noetic, Libfreenect2, the IAI Kinect Bridge, and the ROS 1 bridge, it was finally possible to read colour and depth data from the Kinect into ROS 2 and into Python using the ROS Client Python library.

This leads to a rather convoluted setup to allow communication as shown in Figure 4.1.

Figure 4.1: Kinect to ROS 2 Layers

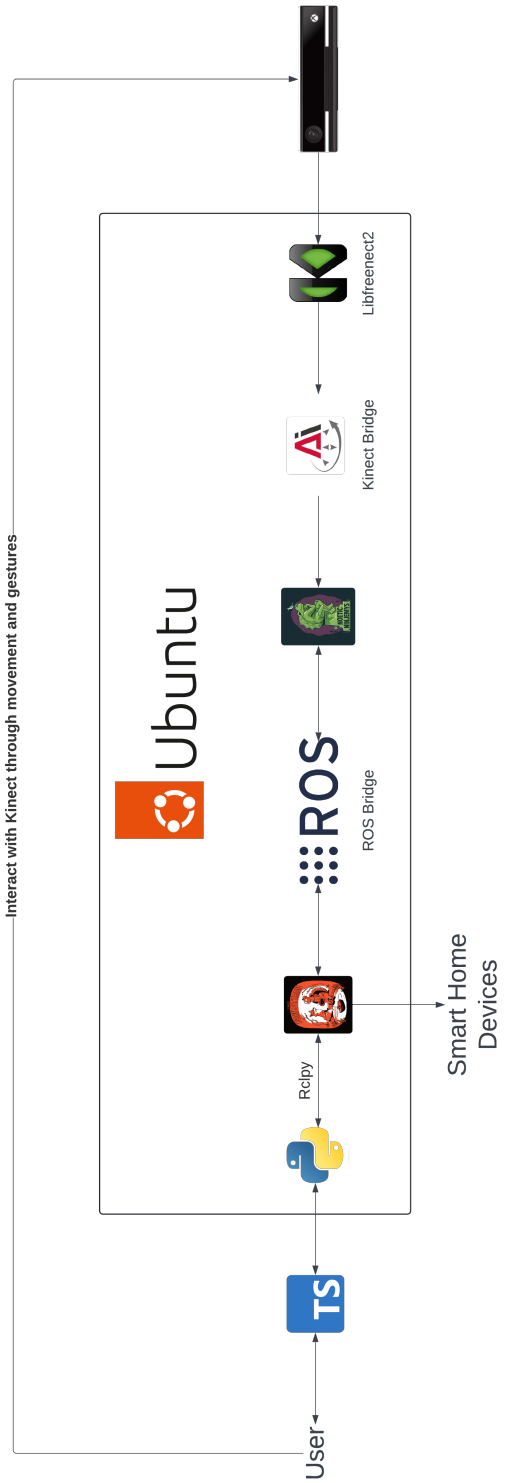

The Kinect data can be read using the Libfreenect2 drivers from Open Kinect. This is then sent via the IAI Kinect Bridge to ROS 1. Using the ROS 1 bridge, the data is then sent to ROS 2 where it can finally be read by the ROS Client Python library and more meaningful operations can be done with it.

## 4.2 Technology Stack

Using this setup for ROS and the Kinect, the final proposed tech stack for continuing development is as shown in Figure 4.2.

The setup outlined previously will run on an Ubuntu machine within the home and connect to a Kinect v2 depth camera. External smart home devices will be controllable through ROS Foxy either via data from the Kinect or through scripts written in Python. Additionally, there will be a TypeScript frontend provided as well that will be accessible within the network to allow users to control devices manually. This is because any smart home system should only ever add functionality to a home, never replace existing functionality. There should always be the option to control the devices manually. The web interface will also allow users to add new gestures to be recognised by the Kinect and to control devices in new ways. Users of the environment will be able to interact with both the front end and with the Kinect, via movement and gestures, to control devices in the home.

Figure 4.2: Proposed Technology Stack

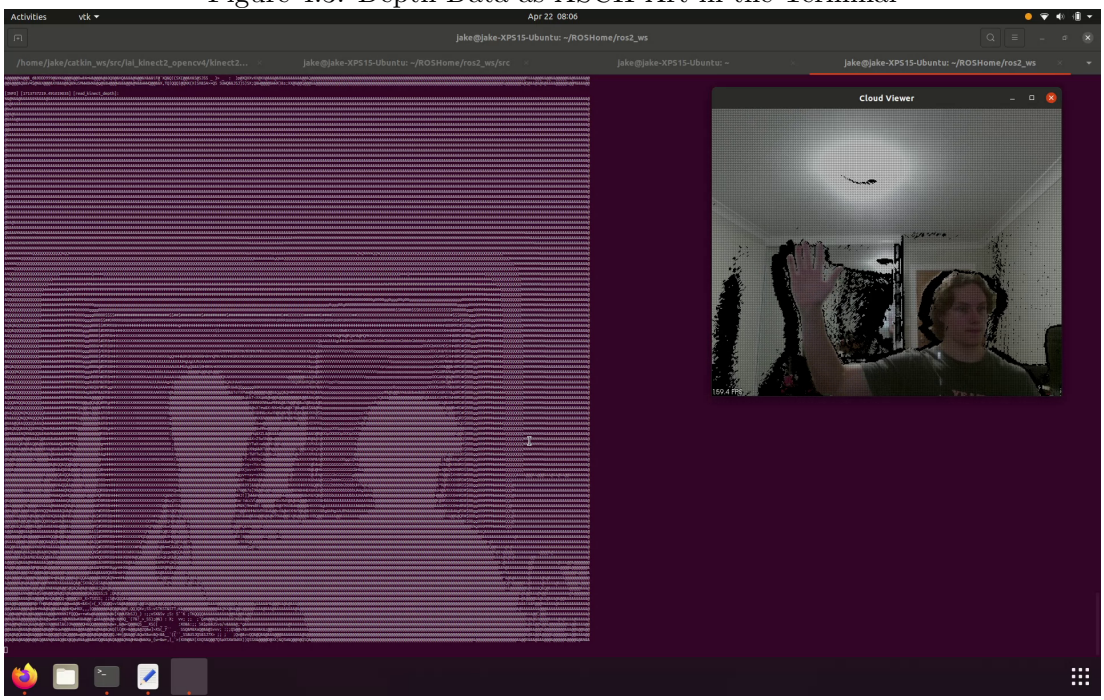


## 4.3 Demonstration and Viability / Preliminary Results

After the ROS environment was set up, a couple of demonstration programs were written in Python to test the functionality and as a proof of concept, to show that external devices could be controlled.

The first was a simple program that extracted depth data from the Kinect and displayed this in the terminal in the form of American Standard Code for Information Interchange (ASCII) art. This is achievable by taking a scale of the depth data and mapping this to a range of characters of increasing density of pixels. The more pixel-dense characters represent objects that are closer to the sensor and less dense characters represent objects that are further away. As shown in Figure 4.3, this is a simple way to visualise the depth data that the Kinect is capturing as the user's outline is visible in the ASCII art image.

Figure 4.3: Depth Data as ASCII Art in the Terminal

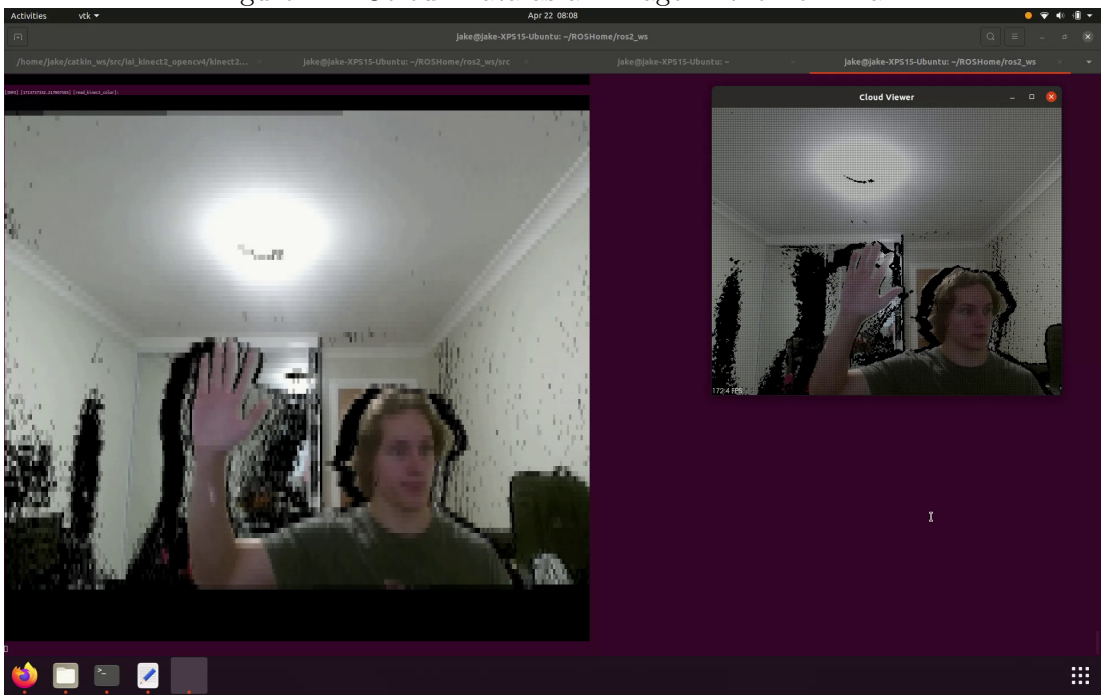



On the right of the image is the provided Kinect viewer tool showing a live feed from the camera, and on the left is the ASCII art representation of the depth data in the terminal.

The second program extracted the colour data from the camera and also printed this

to the terminal by using space characters (“ ”) with its background colour set to the colour of the pixel in the camera. The resolution of the image had to be decreased in order to fit the entire image on the terminal screen, so an average colour of surrounding pixels was taken. As shown in Figure 4.4, the colour data was able to be displayed in the terminal in real-time as read from the Kinect camera.

Figure 4.4: Colour Data as an Image in the Terminal




On the right of the image is the provided Kinect viewer tool showing a live feed from the camera, and on the left is the colour data printed as coloured space characters in the terminal.

These first demonstrations were initially only implemented for the sake of testing to ensure that the Kinect was working as expected and that both the depth and colour data could be read from the camera. The third and final demonstration was a simple program that allowed the user to control a turtle in the ROS turtle simulator using hand gestures.

By moving both hands in front of the Kinect, the user could move the turtle forwards, backwards, or stationary and turn the turtle left, right or stationary. The direction the turtle moved was dependent on the distance of the closest hand to the camera and the direction the turtle moved was dependent on the difference in the distances of the two

hands to the camera.

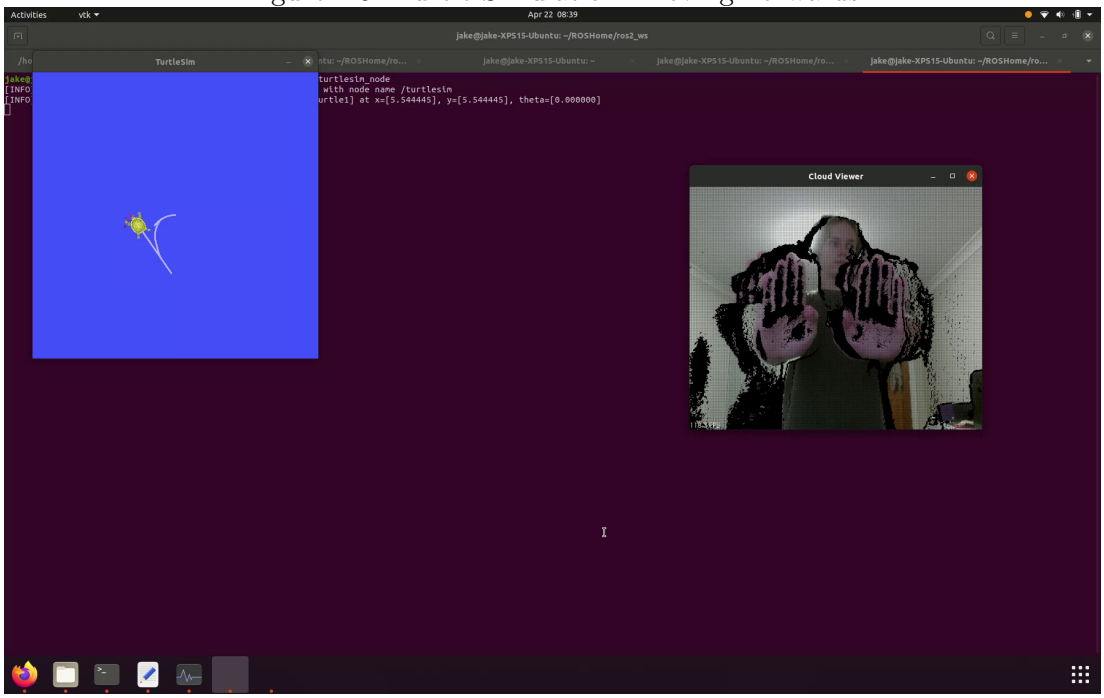
If the user's closest hand was closer than 200mm to the camera, the turtle would move forwards.
If the user's closest hand was further than 500mm from the camera, the turtle would move backwards.
If the user's closest hand was between these two distances, the turtle would remain stationary.

If the user's left hand was more than 200mm further from the camera than the right hand, the turtle would turn left.
If the user's right hand was more than 200mm further from the camera than the left hand, the turtle would turn right.
If the difference in distance between the two hands was less than 200mm, the turtles would not turn.

This can be observed in Figure 4.5 and Figure 4.6 where the turtle is moving forwards as the user's hands are close to the camera and the turtle is turning left as the user's left hand is further from the camera than the right hand.

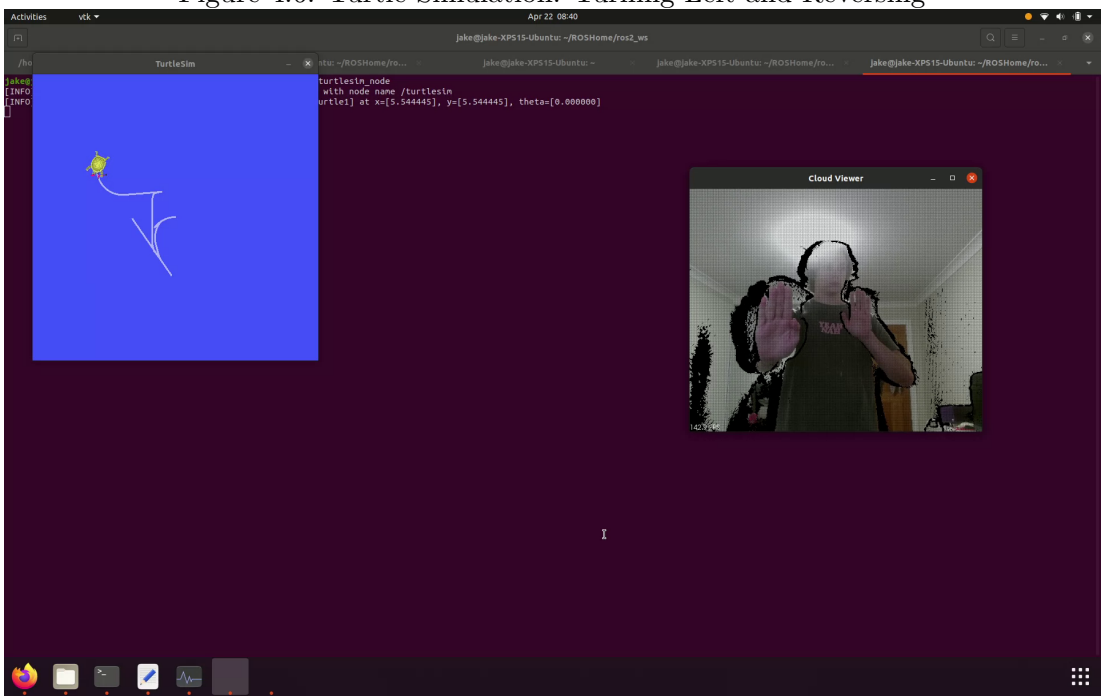
This system, however, did not actually use any hand gesture recognition, it simply measured the distance of the closest object on the left and the right half of the Kinect's sensor. Despite this program's primitive control methods, this demonstration was a proof of concept, showing that the Kinect could be used to control external devices through ROS and that the system was viable for further development.

Figure 4.5: Turtle Simulation: Moving Forwards




The user's hands are closer than 200mm to the camera so the turtle is moving forwards.

Figure 4.6: Turtle Simulation: Turning Left and Reversing




The user's hands are further than 500mm from the camera and their left hand is more than 200mm further from the camera than their right hand so the turtle is moving backwards and turning left.

# Chapter 5

# Conclusion

A thesis requirements/template document has been created. This serves the dual purposes of giving students specific requirements to their theses — both style and content related — while providing a typical thesis structure in a LaTeX template.

## 5.1 Future Work

Extract the requirements from the template in order to have very concise requirements.

# References

Chaaraoui, A., Padilla-López, J., Ferrandez, J., García-Chamizo, J., Nieto-Hidalgo, M., Romacho-Agud, V., & Flórez-Revuelta, F. (2013). *A vision system for intelligent monitoring of activities of daily living at home* (Vol. 8277). doi: 10.1007/978-3-319-03092-0_14

Cucchiara, R., Grana, C., Prati, A., & Vezzani, R. (2005, 4). Computer vision system for in-house video surveillance. *IEE Proceedings: Vision, Image and Signal Processing*, *152*, 242-249. doi: 10.1049/IP-VIS:20041215

Desai, S., & Desai, A. (2017). Human computer interaction through hand gestures for home automation using microsoft kinect. In N. Modi, P. Verma, & B. Trivedi (Eds.), (p. 19-29). Springer Singapore.

Eisa, S., & Moreira, A. (2017). A behaviour monitoring system (bms) for ambient assisted living. *Sensors*, *17*. Retrieved from `https://www.mdpi.com/1424-8220/17/9/1946` doi: 10.3390/s17091946

García, C. G., Meana-Llorián, D., G-Bustelo, B. C. P., Lovelle, J. M. C., & Garcia-Fernandez, N. (2017, 11). Midgar: Detection of people through computer vision in the internet of things scenarios to improve the security in smart cities, smart towns, and smart homes. *Future Generation Computer Systems*, *76*, 301-313. doi: 10.1016/J.FUTURE.2016.12.033

Hasnain, M. R., S, R., P, M., & G, S. (2019). Smart home automation using computer vision and segmented image processing. In (p. 429-433). doi: 10.1109/ICCSP.2019.8697997

Krumm, J., Harris, S., Meyers, B., Brumitt, B., Hale, M., & Shafer, S. (2000). Multi-camera multi-person tracking for easyliving. In (p. 3-10). doi: 10.1109/VS.2000.856852

Nandhini, M., Rabik, M. M., Kumar, K. S., & Brahma, A. (2020). Iot based smart home security system with face recognition and weapon detection using computer vision. *Regular*.

Oudah, M., Al-Naji, A., & Chahl, J. (2021, 6). Computer vision for elderly care based on deep learning cnn and svm. *IOP Conference Series: Materials Science and Engineering*, *1105*, 012070. doi: 10.1088/1757-899X/1105/1/012070

Patchava, V., Kandala, H. B., & Babu, P. R. (2015). A smart home automation technique with raspberry pi using iot. In (p. 1-4). doi: 10.1109/SMARTSENS.2015.7873584

Sefat, M. S., Khan, A. A. M., & Shahjahan, M. (2014). Implementation of vision based intelligent home automation and security system. In (p. 1-6). doi: 10.1109/

ICIEV.2014.6850818

Starner, T., Auxier, J., Ashbrook, D., & Gandy, M. (2000). The gesture pendant: a self-illuminating, wearable, infrared computer vision system for home automation control and medical monitoring. In (p. 87-94). doi: 10.1109/ISWC.2000.888469

Volety, R., & Geethanjali, P. (2022). Smart home automation using wearable technology. In G. D. Gargiulo & G. R. Naik (Eds.), (p. 259-279). Springer Singapore. Retrieved from `https://doi.org/10.1007/978-981-16-5324-7_11` doi: 10.1007/978-981-16-5324-7_11

Zhang, X., Yi, W. J., & Saniie, J. (2019, 5). Home surveillance system using computer vision and convolutional neural network. *IEEE International Conference on Electro Information Technology*, *2019-May*, 266-270. doi: 10.1109/EIT.2019.8833773

# Appendix 1

This section contains the options for the UNSW thesis class; and layout specifications used by this thesis.

## A.1 Options

The standard thesis class options provided are:

| | |
|---|---|
| undergrad | default |
| hdr | |
| 11pt | default |
| 12pt | |
| oneside | default for HDR theses |
| twoside | default for undergraduate theses |
| draft | (prints DRAFT on title page and in footer and omits pictures) |
| final | default |
| doublespacing | default |
| singlespacing | (only for use while drafting) |

## A.2 Margins

The standard margins for theses in Engineering are as follows:

| | U'grad | HDR |
|---|---|---|
| `\oddsidemargin` | 40 mm | 40 mm |
| `\evensidemargin` | 25 mm | 20 mm |
| `\topmargin` | 25 mm | 30 mm |
| `\headheight` | 40 mm | 40 mm |
| `\headsep` | 40 mm | 40 mm |
| `\footskip` | 15 mm | 15 mm |
| `\botmargin` | 20 mm | 20 mm |

## A.3 Page Headers

### A.3.1 Undergraduate Theses

For undergraduate theses, the page header for odd numbers pages in the body of the document is:

Author's Name *The title of the thesis*

and on even pages is:

*The title of the thesis* Author's Name

These headers are printed on all mainmatter and backmatter pages, including the first page of chapters or appendices.

### A.3.2 Higher Degree Research Theses

For postgraduate theses, the page header for the body of the document is:

*The title of the chapter or appendix*

This header is printed on all mainmatter and backmatter pages, except for the first page of chapters or appendices.

## A.4 Page Footers

For all theses, the page footer consists of a centred page number. In the frontmatter, the page number is in roman numerals. In the mainmatter and backmatter sections, the page number is in arabic numerals. Page numbers restart from 1 at the start of the mainmatter section.

If the **draft** document option has been selected, then a "Draft" message is also inserted into the footer, as in:

| | | |
|---|---|---|
| | 14 | **Draft:** April 28, 2024 |

or, on even numbered pages in two-sided mode:

| | | |
|---|---|---|
| **Draft:** April 28, 2024 | 14 | |

## A.5 Double Spacing

Double spacing (actualy 1.5 spacing) is used for the mainmatter section, except for footnotes and the text for figures and table.

Single spacing is used in the frontmatter and backmatter sections.

If it is necessary to switch between single-spacing and double-spacing, the commands `\ssp` and `\dsp` can be used; or there is a `sspacing` environment to invoke single spacing and a `spacing` environment to invoke double spacing if double spacing is used for the document (otherwise it leaves it in single spacing). Note that switching to single spacing should only be done within the spirit of this thesis class, otherwise it may breach UNSW thesis format guidelines.

## A.6 Files

This description and sample of the UNSW Thesis LaTeX class consists of a number of files:

| | |
|---|---|
| unswthesis.cls | the thesis class file itself |
| crest.pdf | the UNSW coat of arms, used by `pdflatex` |
| crest.eps | the UNSW coat of arms, used by `latex` + `dvips` |
| dissertation-sheet.tex | formal information required by HDR theses |
| library.bib | reference details for use in the bibliography |
| sample-thesis.tex | the main file for the thesis |

The file sample-thesis.tex is the main file for the current document (in use, its name should be changed to something more meaningful). It presents the structure of the thesis, then includes a number of separate files for the various content sections. While including separate files is not essential (it could all be in one file), using multiple files is useful for organising complex work.

This sample thesis is typical of many theses; however, new authors should consult with their supervisors and exercise judgement.

The included files used by this sample thesis are:

| | |
|---|---|
| definitions.tex | mywork.tex |
| abstract.tex | evaluation.tex |
| acknowledgements.tex | conclusion.tex |
| abbreviations.tex | appendix1.tex |
| introduction.tex | appendix2.tex |
| background.tex | |

These are typical; however the concepts and names (and obviously content) of the files making up the matter of the thesis will differ between theses.