University of Glasgow | School of Computing Science

In-Car Gestural Interaction

Gerard Romeo

School of Computing Science
Sir Alwyn Williams Building
University of Glasgow
G12 8QQ

Level 4 Project — March 28, 2014

Abstract

Education Use Consent

I hereby give my permission for this project to be shown to other University of Glasgow students and to be distributed in an electronic format. **Please note that you are under no obligation to sign this declaration, but doing so would help future students.**

Name: _____ Signature: _____

Contents

1 Introduction **1**

2 Background and Related Work **2**

2.1 Motivation 2

2.2 What is Gesture Recognition? 2

2.3 Gesture Recognition Devices 3

2.3.1 Microsoft Kinect 3

2.3.2 LeapMotion 3

2.4 Literature Review 4

2.4.1 Previous Research 4

2.4.2 Current Research 5

2.5 Examples In Automotive Industry 5

2.6 Driver Safety 7

2.6.1 Background 7

3 Requirements **9**

3.1 Functional Requirements 9

3.2 Non-Functional Requirements 9

3.3 Determining Device That Meets Requirements 9

3.3.1 Microsoft Kinect 9

3.3.1.1 Supporting Arguments 9

3.3.1.2 Opposing Arguments 9

3.3.2 LeapMotion 10

3.3.2.1 Arguments For . . . 10
3.3.2.2 Arguments Against . . . 10

4 Design **11**

4.1 Motivation . . . 11
4.2 Requirements Gathering . . . 11
4.2.1 Classification of Gestures . . . 12
4.2.2 Main Menu . . . 13
4.2.2.1 Navigating Through the Menu . . . 13
4.2.2.2 Selecting an Option . . . 14
4.2.2.3 Revert to Previous Menu . . . 14
4.2.2.4 Suggested Gestures . . . 15
4.2.2.5 Gestures Defined for ***In-Car Gestural Interaction*** . . . 15
4.2.3 Music . . . 16
4.2.3.1 Play/Pause . . . 17
4.2.3.2 Increasing and Decreasing Volume . . . 17
4.2.3.3 Next and Previous Song . . . 17
4.2.3.4 Suggested Gestures . . . 17
4.2.3.5 Gestures Defined for *In-Car Gestural Interaction* . . . 17
4.2.4 GPS . . . 18
4.2.4.1 Zoom In . . . 18
4.2.4.2 Zoom Out . . . 18
4.2.4.3 Place Marker . . . 18
4.2.4.4 Scrolling . . . 18
4.2.4.5 Suggested Gestures . . . 18
4.2.4.6 Gestures Defined for *In-Car Gestural Interaction* . . . 18
4.2.5 Contacts . . . 18
4.2.5.1 Call Contact . . . 18
4.2.5.2 End Phonecall . . . 18

4.2.5.3 Reject Incoming Call . . . 18
4.2.5.4 Suggested Gestures . . . 18
4.2.5.5 Gestures Defined for *In-Car Gestural Interaction* . . . 18
4.2.6 Air-Conditioning . . . 18
4.2.6.1 Increasing and Decreasing Fan Strength . . . 18
4.2.6.2 Increasing and Decreasing Temperature . . . 18
4.2.6.3 Suggested Gestures . . . 18
4.2.6.4 Gestures Defined for *In-Car Gestural Interaction* . . . 18
4.2.7 Windows . . . 18
4.2.7.1 Opening and Closing the Driver-side Window . . . 18
4.2.7.2 Opening and Closing the Passenger-side Window . . . 18
4.2.7.3 Opening and Closing the Both Windows Simultaneously . . . 18
4.2.7.4 Suggested Gestures . . . 18
4.2.7.5 Gestures Defined for *In-Car Gestural Interaction* . . . 18
4.2.8 Menu Hot-keys . . . 18
4.2.8.1 Suggested Gestures . . . 19
4.2.8.2 Gestures Defined for *In-Car Gestural Interaction* . . . 20
4.2.9 Post Experiment Questionnaire . . . 20
4.3 Designing the User Interface . . . 20
4.3.1 Paper Prototype . . . 20
4.3.2 Storyboard . . . 20

5 Implementation **21**

5.1 Creating the User Interface . . . 21
5.2 Finger Counting . . . 21
5.3 Gesture Recognition . . . 21

6 Motivation **22**

7 Constraint Programming Models **26**

7.1 Model A 26

7.2 Model B 27

8 Solutions **28**

9 Conclusion **30**

Chapter 1

Introduction

As technology progresses, car manufacturing companies are consistently looking for ways to integrate these systems into their new designs in innovative ways. However, such advancements bring safety concerns; if the device is located in an inconvenient location for the driver, attempting to use the device could hamper and detract the drivers attention, thus potentially leading to accidents.

This has led to a deep pool of research analysing how to improve interaction within automotive UI. Various forms of multimodal techniques have been implemented within the interior of vehicles successfully. With the emergence of recent free-gestural motion technology that can accurately determine a user's gesture, the field of Human Computer Interaction (HCI) is pursuing ways in which this could be adapted into the automotive industry. This is especially prevalent considering that research has shown[1] that gestural input maintains a drivers awareness and perception on the road, whilst haptic inputs, such as buttons can provide an additional layer of potential distraction.

Gestures and body language are an integral part of societal interaction – they are natural subconscious reactions that are complimentary to speech. They provide a visual representation of what is being communicated. They allow a form of communication when language becomes a barrier - there are signals that within society feel pre-defined and are instantly recognisable. In cultural anthropology, human body language is the primary contributor to non-verbal communication studies and gestures are often used throughout all cultures. With this in mind I would like to propose using some of these widely accepted gestures to interact with a variety of the vehicle's interior functionality.

The aim of my project, titled *In-Car Gestural Interaction*, is to create a series of quick, intuitive gestures that will allow drivers to interact with systems within their car, providing an alternative form of modality that the typical haptic buttons found in modern day vehicles provide.

Furthermore, as previous research indicates, this should prevent the drivers focus from deviating from the road. Therefore, not only will the development of the gestures provide a useful alternative for drivers, it also has the potential to improve road safety, ultimately leading to less accidents and deaths as a result.

In-Car Gestural Interaction aims to provide a novel and alternative way of interacting with an automobiles multimedia devices and interfaces. Such gestural interaction should complementary to the driver, improving their awareness on the road with the hope of reducing distractions and improving the safety of the road.

Chapter 2

Background and Related Work

2.1 Motivation

In recent years there has been a sudden influx of free-motion gestural technologies that have opened up new horizons in the automotive industry. Car manufactures such as Hyundai and BMW have invested Research and Development into developing concept designs that seamlessly integrate the use of gestures with the motion detection capabilities built into the steering wheel. However, these concepts only use simple gestures such as swiping to navigate hierarchical menus and still require the driver to focus on the central display built into the hub of their car.

Furthermore, free-motion gestural interaction within vehicles has been studied throughout the years. Although the technology has never been reliable to apply in real-world scenarios, their findings from controlled-environment experiments highly suggest that hand gestures could be the optimal form of interaction within automobiles.

This section will consider the pre-existing background research, the current state of gesture controls within the automotive industry and discuss why gesture based interaction could potentially be the safest input modality whilst driving.

2.2 What is Gesture Recognition?

Gesture Recognition in Computing Science is the study of understanding and responding to human movement. Gesture and body language can exist from any part of the body but is most commonly recognised as hand of facial movements, often stemming from emotional responses.

As technology has progressed, so to has our ability to read and interpret these gestures through a series of algorithms. Particularly, Computer Vision is the experimental field that strives to improve the way in which we acquire, process, analyse and understand real-life high-fidelity data in order to produce numerical or symbolic information. With this improvement in Computer Vision, it is now possible to recognise and more importantly, respond to a user's emotions and gestures.

The belief is that with a greater understanding of human body language and gestural interaction, the world of Human Computer Interaction (HCI) will evolve to the state that we are no longer reliant on on input devices such as a keyboard and mouse, but instead users can control devices through Gesture Recognition.

2.3 Gesture Recognition Devices

Gesture technology is typically in the field of research and development and as a result, the technology ultimately becomes overtly expensive, specifically built for one purpose and is never released to public. However the following two devices have not only been released to the masses but are relatively inexpensive, providing future avenues for the adoption of gesture recognition.

2.3.1 Microsoft Kinect

When Microsoft released its Kinect peripheral in 2010 it revitalised the concept of free-motion gestural interaction. No longer did users have to interact with an input device – the Kinect could determine a person's actions and body movements, allowing the user to interact without any other form of input peripheral. This is considered by many to be the first time that full body gestural movements had garnered such public acceptance. The Kinect is not without its technical caveats however.

Figure 2.1: Microsoft Kinect Peripheral

The Kinect operates by sending out infrared beams from the emitter on its base and the reflected beams are then read by the infrared depth sensor. The time deltas between these reflected beams is then used to interpret the depth of field in the area and the distance between the Kinect and objects within its field of view, such as a human. The device then uses skeletal tracking to create a 3D structure of the person and recognise their limbs, hence allowing gestures to be defined within its field of view.

2.3.2 LeapMotion

In 2012 the LeapMotion Controller was released to public, aiming to provide a much higher fidelity and accuracy for finger tracking. As the LeapMotion is capable of tracking all 10 fingers up to 1/100th of a millimeter, it has provided new and unparalleled avenues in gesture recognition.

With a 150 field of vision and 3-Dimensional depth tracking, a users hand can be measured with extreme accuracy. Using two monochromatic infrared cameras and three infrared LEDs, the LeapMotion observes an approximation of a hemispherical area, at a distance of about 1 meter. The LEDs generate a 3D pattern of dots of IR light[14] and the reflected information returns to the device at around 200 frames per second. This data is then analysed and processed by the Leap Motion's patented algorithms to then synthesise a hand and its appendages 3D positional data by comparing the 2D frames generated by the two monochromatic cameras.

As it offers an unprecedented level of precision, it is the most sensitive gestural technology to date.

Figure 2.2: LeapMotion Device

2.4 Literature Review

2.4.1 Previous Research

In-car Gestural Interaction is not a new concept within the automotive industry, but it has yet to be established commercially and lacks the credence of other input modalities. Since the dawn of gestural technology, automotive designers and researchers have attempted to integrate gestural input recognition into vehicles. The especially became relevant when periphery multimedia devices were incorporated into cars, know as vehicle telematics. Developers and car manufacturers have experimented a number of input modalities, seeking to obtain the safest, most practical option for drivers. However, a lack of consensus existed as to what input modality would be the most suitable to control such gadgets.

As technology within vehicles has improved and expanded, this has enforced an increased need for functionality and haptic controls to be built into the vehicle's dashboard or steering wheel.

In 2001, the Institute of Human Machine Communication at the Technological University of Munich []suggested an alternative to haptic controls would be required as the number of physical controls in vehicles were becoming increasingly confusing and demanding on drivers. They proposed using hand gestures to simplify user interaction. This was carried out by creating a defined gesture space to the right of the steering wheel – note that as this experiment took place in Munich, Germany, where vehicles are left hand drive. As such, the 'driver' in the experiment carried out gestures with his right hand, towards the center of the car's interior.

The experiment was carried out in a controlled environment and by allowing participants to explore gestures for specific scenarios, they discovered that certain gestures were culturally independent, such as a mimicking action for answering a phone-call. Such gestures would not have to be learned as they synonymous within society.

However, through experimentation they also discovered that users were only able to recall a small portion of the gesture library and that participants were unsure of which gestures to use in given scenarios, presenting a lack of gesture-conformity. This was overcome by developing a help notification system in the vehicle's User Interface (UI), to develop users learning curve. Given users low threshold to identify gestures it was noted that without an optimization of gesture libraries and their ease of recognition, gestures within vehicles would provide a further layer of distraction. Users would be reliant upon the UI visualisation and not the gestures themselves, resulting in user annoyance. Despite these issues, the study suggested that gestures are a viable multimodal method for in-car interaction, given that the gestures had high-conformity with users.

This research was further analysed by Alpern and Minardo [], whose study looked at developing a car gesture

interface to control secondary tasks. Driving is a cognitively demanding task, requiring the attentiveness of the driver at all times. After iterating through several UI techniques, such as hierarchical menus and virtual objects, Alpern and Minardo developed a heads-up display interface that integrated over the driver's HUD. Gestures were limited to simple swipes and counts based on the number of fingers shown on a given hand.

From their experimental results, they found that the gesture interface for the radio was not only comparable to the haptic radio alternative, that can be found in every car, but that it also increased the driver's attentiveness on the road. This was due to the simple gestures implemented and that the interface could be quickly glanced at whilst still focusing on the road, permitting the driver to be aware of potential hazards.

Although both research papers offer critical praise for gestural interaction within vehicles, neither were able to apply their systems in real world scenarios. The experimental evaluation methodology was a result of the limitations of gesture detecting hardware.

The technology used approximately 10 years ago was impractical and inaccurate to be tested out with experimental findings; as the hardware used was based upon infrared depth cameras, they would be affected by the varying light conditions that drivers will experience. The infrared cameras used to detect the gestures were unable to compensate between the day/night cycle and as such, were not stable and reliable for commercial applications.

The evidence found in these findings however suggested a direct correlation between gestural interaction and the stability of the driver's road awareness.

2.4.2 Current Research

In recent years, there has been an influx of gesture recognition peripherals, that have revitalised the automobile industries interest in gesture input modality to control the plethora of *'infotainment'* functions that vehicles currently have at their disposal.

With this resurgence of free-motion gestures, new research has been carried out and with the improved accuracy, they have been able to carry out more conclusive findings as improved recognition and reliability permits real world field testing.

In this study they were also able to conclude that gestural interaction strengthens attractiveness and hedonic quality significantly compared to haptic interaction and that from their 20 participants that carried out their test, 14 favoured the gesture[5] In addition to this, they also considered what gestures can be viewed as socially and culturally independent, with the aim of minimalising the number of gestures that would be recognised through the device, citing that redundancy of gestures could lead to confusion.

TO DO!!!

2.5 Examples In Automotive Industry

Gestural interaction has often been considered in the industry but as of yet it only exists within conceptual cars. Typically features found in conceptual cars are too far-fetched to exist in commercial vehicles, but as more car manufacturers are investing into gestural technologies, it would imply that both technology and society is ready for the next step in automotive innovation.

Hyundai[] in particular have developed an 'infotainment' concept car that adapts the steering wheel to have both haptic controls and gestural commands on the left and right hand side of the steering wheel respectively. These controls are used to interact with the vehicle's navigation systems, audio controls and other desirable

Figure 2.3: Hyundai's 'infotainment' steering wheel design

functions. This allows for both physical and freehand control dependant on the driver's preference, as they are used to control the same features. [].

BMW have also forayed into this innovative form of HCI through a partnership with Harman, who are developing a new form of interface within vehicles that is primarily gesture-based[]. As BMW have incorporate new features into their cars, so have the number of physical interactive elements – modern cars can resemble aircraft cockpits; over-engineered to the point were new systems are underutilised due to the complexity of the design. BMW, recognising that the automotive market desires sleek, simple design, aim to allow users to control there systems with a small-set of simple gestures. The premise is similar to Hyundai but differs in the gesture recognition technology.

Kia also recently unveiled their User-Centered Driver (UCD) concept at the 2014 Consumer Electronics Show (CES 2014), that alongside an augmented reality display, incorporates gesture recognition to operate various controls in the car. An advanced camera system tracks and mirrors the driver's gestures by mapping movement through a full kinematic spatial hand which, in turn, replicates the motion as a virtual hand on the cockpit's display.

Figure 2.4: Kia's UCD concept design

Although the Korean car-manufacturer have yet to specify when they will integrate their UCD concept into commercial vehicles but have promised consumers it will be released.

Google have also entered the foray of In-Car Gestural Interaction with the recent acquisition of the motion control company Flutter [**?**]. Google aims to secure a patent for their unique idea of accepting free-motion gestures. Using multiple depth cameras that are mounted to the vehicle's headliner and a 3D laser sensor on the dashboard, Google's software engineers aim to create an approximation of the interior of the car.

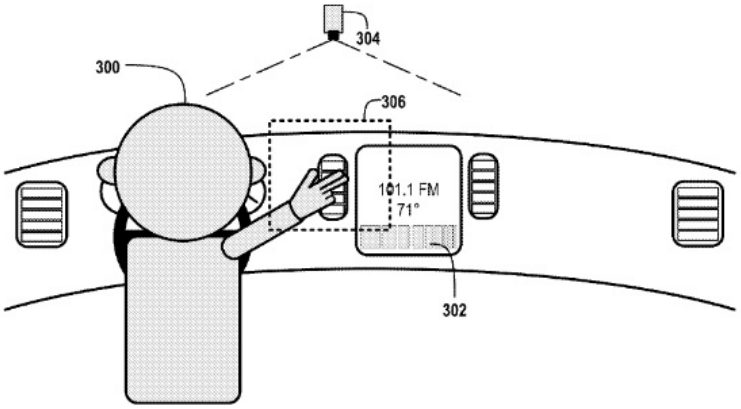


Figure 2.5: Google's gesture based car patent

From this mapped interior, a dedicated CPU will determine the user's gesture, and the location that the gesture has been performed, to carry out an action, such as opening the window, or moving the driver's seat forward[**?**].

These are just some of examples of the concepts car manufacturers have for gesture recognition - ford and Mercedes have also showcased some simple gesture designs at CES. Companies are investing a lot of time and resources to make freehand gesture controls a reality.

Although still mainly conceptual in nature, this the sheer number of concepts shows that some of the largest automotive manufacturers and software powerhouses in the industry are investing Research and Development into gestural interaction. Although these companies are strongly considering and showcasing gesture implementation, consensus is split. Most of these concepts are out with the scope of the research that will be conducted in this project but provide an interesting take on where the future lies. Just as touchscreens have permeated throughout society, so too will gesture controls in the coming years. The belief is that In-Car Gesture Interaction is the next desirable feature; the next evolution of input modality in the automotive industry.

2.6 Driver Safety

2.6.1 Background

Driver distraction has long been recognized as a potential contributor to roadside accidents (Treat et al., 1977). Drivers have always had the opportunity hamper their physical, visual and mental cognition through eating, communicating with the passengers in the car or on their mobile and engaging in secondary activities that are unrelated to the current task of driving. However, over the years the potential for distraction has increased as in-vehicle technology has expanded. In the last decade, distracted driving came to the forefront of public awareness, stemming in large part from the rapid attachment rates and developments in mobile phones and the introduction of built-in peripherals and portable devices that are now widely available in vehicles. These devices allow drivers to engage in activities that were previously unimaginable (for example, setting a location on a GPS or connecting their phone to their car via bluetooth), and have the capacity to absorb drivers attention to a whole new degree.

This concept of 'Distracted driving' has received so much attention in recent years that it was designated the 2009 Word of the Year by Websters New World College Dictionary, becoming a safety-critical buzzword (U.S. DOT, 2010). To try and combat this, The U.S. Department of Transportation has held two summits with industry leading car manufacturers to discuss distracted driving and to identify opportunities for addressing the problem.

One of the primary concerns when implementing a new system within a car is the potential adverse effect it could have on driver safety. As found by National Highway Traffic Safety Administration (NHTSA), approximately 80% of accidents occur due to a driver's negligence to concentrate on the road[**?**]. This compiled with the fact that 65% of rear-end accidents also occur because of the driver interacting with a system within their car highlights the safety concerns that exist within the industry.

The Royal Society for the Prevention of Accidents (RoSPA) also found that in 2012, of the 1,754 in car related fatalities, over 300 deaths can be attributed to driver distraction[**?**].

The Virginia Tech Transportation Institute (VTTI) discovered[**?**] that drivers reaching to interact with an electronic device were 1.4 times more likely to be involved in a crash or near crash than non-distracted drivers. This is caused by the driver reaching for the control or looking at their center console when they should be attentive on the road ahead.

In 2002, the NHTSA conducted a survey [8] on approximately 4,000 licensed drivers to determine the potential distractions and interactions that drivers may face whilst on the road. Most respondents stated that on average drives, they were prone to talking to their passengers (81%) and changing radio stations or looking for CDs (66%). Half of the survey participants (49%) reported consuming food or drink, while fewer reported using their mobile phone (25%), Reading a map or directions while driving (12%) and using telematics such as In-Car Navigation Systems(2%).

As this survey was conducted in 2002, the wave of technological advancement had yet to occur and was nowhere near as prevalent as it is now. Built-in car navigation systems, that at the time of the survey's publication, were considered a luxury, are now viewed as a standardised add-on in modern vehicles and to many, a necessity. Mobile phones can be connected to a vehicle's central console via bluetooth or USB, giving the driver instant access to their contacts and music. Due to the proliferation of mobile phones and advancement of in-car technology, it can be assumed that since the time of the survey publication, these numbers have inflated significantly. These devices are much more commonplace and are more frequently used.

But how do you combat these erroneous mistakes that drivers make? Due to the emphasis and concern of driver distraction, car manufacturers are now looking at how to mitigate this issue when implementing new technology. As previous findings have shown,[**?**] gestural interaction can reduce the level of distraction that drivers face compared to haptic inputs, so if gestures as a form of input modality were globally accepted in vehicles, it has the potential to reduce the number of accidents caused through driver preoccupation. Of course, human error will still purvey, but *In-Car Gestural Interaction* offers a safer alternative than the current haptic input modality.

Based on the survey conducted by Royal [8], some distractions, such as focusing on passengers instead of the road and consuming food, can be attributed to common sense and driver awareness; human factors that are outwith a designer's control and are impossible to mitigate. However, *In-Car Gesture Interaction* can simplify many of the controls that are typically haptic-based. For example, changing to a different music artist whilst driving is visually, mentally, and cognitively demanding through a haptic modality. Gesture recognition within vehicles could remove such complexities and allow the driver to remain attentive to the road whilst interacting with the various systems in the car.

Therefore the requirements of this project is to provide a safer alternative to controlling the various technology within vehicles and ultimately keeping the drivers cognition on the task of driving with the hope of reducing the risk of accidents.

Chapter 3

Requirements

3.1 Functional Requirements

3.2 Non-Functional Requirements

3.3 Determining Device That Meets Requirements

Based upon the Functional and Non-Functional Requirements outlined in the previous sections, I felt that only two gestural recognition devices were suitable for the project. This section will outline the reasons for and against both the Microsoft Kinect and the LeapMotion and ultimately the device I have chosen to use.

3.3.1 Microsoft Kinect

3.3.1.1 Supporting Arguments

Due to the popularity of the Kinect, there was a large adoption rate of developers that created gesture recognition software.

3.3.1.2 Opposing Arguments

However, there are a number of reasons why Microsofts Kinect is not suitable for the project outline. With the first iteration of the Kinect, the device typically requires a full skeletal model of the person to operate efficiently, as defined by its API, to therefore accept any form of gestural input. Obviously, that is a major stumbling block for in-car gestural interaction as the driver will be sitting stationary and unless the Kinect could be angled in such a way to capture enough of their body, there is no assurance that their gestures will be approved by the device.

Furthermore, the skeletal tracking is limited in that it can only captures limbs and joints – appendages such as fingers cannot be tracked effectively. This is a big issue for developing gestures within a car as the idea is to provide and promote a safer method of interacting with a cars systems than the current haptic/ touch screen alternative. With the Kinect, the gestures would have to be profound and would feel obtuse within the confined area of the car. Let alone the fact that such large-scale gestures would potentially be more dangerous to the driver and be far more distracting and frustrating, especially if they were not always approved as mentioned before.

Finally and most importantly, the limitation of the Kinects depth resolution, frame-rate and depth space range really hamper its effectiveness within the car. Within the Kinect API, the DepthImageStream Enumeration determines the resolution and frame rate of the depth stream. At maximum, the resolution of this depth can be

640x480 at 30 frames-per-second. This is why the Kinect is unable to detect small hand movements and gestures. Not only that but the Kinect requires quite a wide view before it will begin to recognise and accepts objects n its field. By default, objects must be at the very least 0.8 metres away before they can be interpreted by the Kinect the infrared that is reflected back from any items or people closer than this range are discarded. Even with Near Range mode enabled, the distance must still be 0.4 metres at minimum. The Kinect has a great range and field of view, up to 8 metres in Normal Range, but distance is not the desire, rather accuracy.

Microsoft recently released the Kinect II with the launch of their Xbox One which greatly optimises all of the issues I highlighted that would affect the objective of the project. However, as of this writing, they have they to announce any plans of releasing the device for PC or their API for development purposes.

Taking these limitations into account and the confined, compact nature of a cars interior, I did not believe that Microsofts Kinect was a viable gestural input device for this particular project. Especially as safety is the primary concern for the driver and from the offset, granted without any development, it seems like the Kinect would be a further distraction rather than a deterrent. The aim is to have the gestures as small-scale as possible and they should not detract from the objective of driving. With the Kinect, that simply would not be feasible.

3.3.2 LeapMotion

3.3.2.1 Arguments For

The smaller observation area and higher resolution of the device differentiates the product from the Kinect, which is more suitable for whole-body tracking in a space the size of a living room.

3.3.2.2 Arguments Against

Chapter 4

Design

4.1 Motivation

4.2 Requirements Gathering

With the objects that the users will interact with in the car defined, the next step in the process was to determine what gestures would be suitable for not only each interaction, but also for the confined space within the car. Classification of gestures has been explored in previous academic research papers but not specifically for in-car interaction. Googles innovative patent conceptually looks at interacting with the cars systems via locality (gesturing your hand down when near a window opens the window, gesturing near air-conditioning turns it on etc.) but given the LeapMotions limited range and infra-red sensitivity, following such a prototype is not applicable.

To find out what gestures were intuitive to users, a gesture elicitation experiment was carried out to determine if the gestures can be standardised. It is important to conclude which gestures feel familiar and intuitive to each action before developing the interactions themselves. One persons interpretation could be vastly different to the majority so it is vital to obtain a cross-section of participants from varying backgrounds and age.

Figure 4.1: Usability Lab Configuration

In order to do this, 13 participants (12 men and 1 woman) took place in the experiment were they were ask to carry out a series of tasks, using the gesture that they felt was natural to each scenario. This allowed not only relations to be found between interactions, but also to classify the types of gestures.

The experiment was carried out in the Usability Lab of the Sir Allan Williams building. Participants were asked to carry out certain gestures whilst focusing on their road awareness in a driving simulator. This driving simulator was provided via OpenDS (insert website at bottom of page).

Using a think aloud approach, users were also encouraged to speak openly regarding their tasks and their overall opinion of interacting with an automobiles systems whilst driving as they took part in the experiment.

With the appropriate ethics approval and knowledge that they could end the experiment at any point if they felt uncomfortable (See Appendix), participants were filmed so that the requirements elicitation could be analysed and documented.

4.2.1 Classification of Gestures

The gestures that participants could use fell into two categories:

- **Small-Scale gestures**, in which either both their hands stay on the wheel or one hand slightly off the wheel. These can be further categorised as:
  - **Finger gestures** – rotating a finger, pointing gestures and counting the number of fingers shown by participant.
  - **Hand gestures** – swiping and open/closed hand gestures.
- **Free-Motion gestures**, which could be a motion towards their head, two handed gestures or a variety of miscellaneous movements. These can also be further classified as:
  - **Mimic gestures** – Hand gesture towards ear, body part or object within the vehicle.
  - **Grandiose gestures** – Two hand gestures or head and body movements.

Users were asked to attempt to primarily use small-scale gestures and if they could not think of a suitable movement to represent the action, then they should opt for a free-motion gesture. For the majority of scenarios, relations existed between the gestures that users felt best represented the action. It is important to note that participants were not informed of the hardware limitations that exist with the LeapMotion the purpose was to find the common, natural gestures for each interaction. This experiment was technology agnostic; finding acceptable gestures was the sole purpose on which software developers could adhere to. As technology progresses and gesture technology becomes more widespread and relevant (the addition of depth cameras and improved tracking technology), the findings of the experiment could be beneficial to off-device interface designers and others aiming for free-motion gesture recognition techniques

The following sections will document each category of the gesture elicitation process and the requisite findings.

4.2.2 Main Menu

Within modern vehicles, a built-in central console allows the driver to navigate to music, GPS, their phonebook and other periphery systems within the car. Naturally, the first step in the requirements elicitation was to determine the gestures the participants would use to navigate such a menu. Participants were provided with a visual representation of a typical menu within a cars central dock.

Four scenarios were presented in controlling a menu and whilst continuity existed between participants for the first three actions, the forth scenario contained a variety of different gestures.

Participants were informed that the gestures they selected for menu interaction would apply throughout the rest of the experiment.

4.2.2.1 Navigating Through the Menu

As gestures are becoming prevalent throughout society, given the worldwide appeal and acceptance of gestural touch screens, people intrinsically know how to interact and navigate with touch and gesture-based interfaces. These are common pre-defined gestures, regardless of the developer or mobile operating system. This was highlighted in the requirements elicitation for navigating as the majority of the participants selected a gesture that is commonplace within mobile devices.

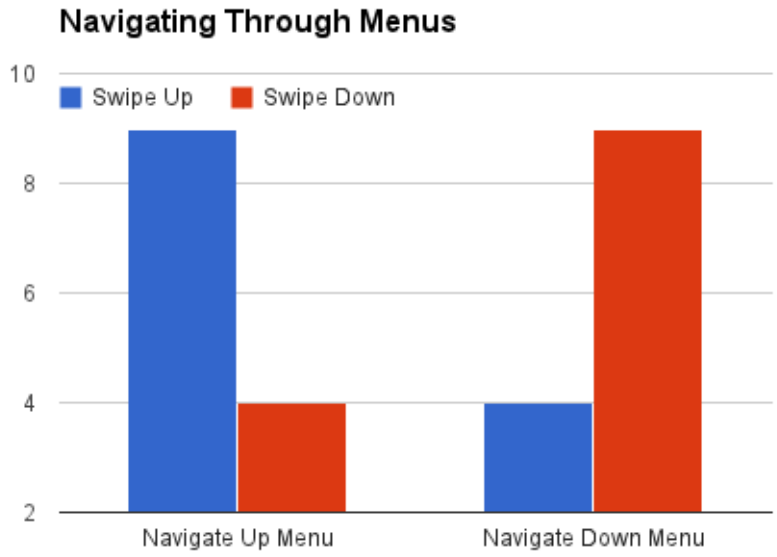


Figure 4.2: Menu Navigation Results

Nine participants swiped in the direction requested whereas a few inverted the gestures, swiping down to go up the menu and the opposite for going up the menu. The reasoning the minority chose to invert the swipe gesture is most likely due their familiarity with mobile phone interface gesture recognition patterns most touch devices invert the motion for swipe movements, making a swipe up on the screen scroll down and vice versa. One participant explicitly discussed their thought process when determining which gesture they would use for navigation:

Participant:	"I guess it depends...if I was using my phone, I would swipe down to scroll up. But if I was using my pc, I would gesture down to the next option."
Myself:	"The important aspect to remember is that you are not directly interacting with a physical object. The gestures you are doing are in the ether."
Participant:	"Yeah. Will there be enough options in a menu for the velocity of the gesture to matter? Like if I swipe my hand quickly will it scroll through the menu faster?"
Myself:	"As this is a proof of concept, it will only have four options Music, GPS, Contacts and Extra features"
Participant:	"In that case I would emulate what I do on my phone and invert the direction down to go up and up to go down"

Table 4.1: Conversation Regarding Gesture Familiarity

4.2.2.2 Selecting an Option

All participants found that a gesture symbolic of a ”screen tap” was the best method to select an option from the menu. Again, this is due to gesture familiarisation with other touch interfaces as it is certainly a clear and evident way to choose one item from a list of options.

It is worth noting that several participants specified that they would swipe to the menu option they desired until it was highlighted and then select it. This was preferred over having than the options existing in a virtual space; the screen-tap gesture should not rely on accuracy or locality in space – especially whilst the car is in motion – but only on the hand recognition.

4.2.2.3 Revert to Previous Menu

The last menu-related task produced a variation of actions from participants, unlike the previous choices. In fact, the participants were evenly split as [] shows. Although the gestures were ostensibly different, they are similar in nature, only in axis of space.

54% of the participants felt that swiping to the left was the best way to return to an option, due again to their pre-existing gesture familiarity, whilst 38% preferred gesturing their hand towards their own body, resembling a come to me gesture. Without a clear preference, I would suggest using a gesture that will be unique throughout. This will be explored in further detail in the next sections.

(a) Swipe Left

(b) Beckon towards body

Figure 4.3: Gestures for Returning to a Previous Menu

4.2.2.4 Suggested Gestures

Given the participants overall agreement and familiarity with these gestures, I would suggest the following gestures for each interaction. Despite the relatively low number of participants, I believe that the same results would appear with a larger demographic. Even if the users did not-have any pre-existing recognition with gesture technology, they would be able to adapt and recognise the natural motions of the gestures selected and what they represent in each case.

Task	**Gesture**	**Classification**
Scroll Up	Swipe Up	Hand Gesture
Scroll Down	Swipe Down	Hand Gesture
Select an Option	”Screen Tap”	Hand Gesture
Return to Previous Menu	”Beckon”	Hand Gesture

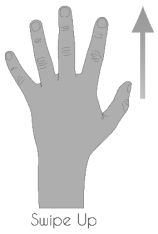
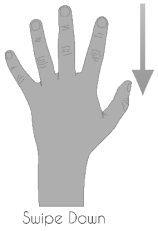
For gesturing to go back to a previous menu, although the majority of participants suggested swiping left to return to the previous set of choices, this may result in gesture confusion which will be further explored in section blah blah, as the gesture is also suited to another scenario. Therefore I would recommend a unique gesture that will not create driver confusion and frustration with the gesture input mechanism. Ultimately, such problems could lead to their attention being removed from driving, which is the adverse aim of the project.

determine which of the two designed gestures users adapt to more naturally and can be detected by the LeapMotion.

4.2.2.5 Gestures Defined for *In-Car Gestural Interaction*

Given the simplicity of the gestures used by participants and most of the gestures are predominately associated with said tasks, I have decided to use the same gesture for all but one, navigating back to a previous menu.

This is to avoid gesture conflict that may occur when zooming in and out a map, which will be discussed in section blah blah, I felt it was best to use a unique gesture for the action. As the LeapMotion in theory can differentiate gestures based on the number of fingers in a swipe, I decided to use a two finger swipe left movement to signify returning to previous set of options. Although this will have a slight learning curve for users, this seems to be the most sensible option in order to ensure that gesture conflict does not occur resulting in the user making an undesired choice.

Task	Gesture	Classification
Scroll Up Menu		Hand Gesture
Scroll Down Menu		Hand Gesture
Select an Option		Finger Gesture
Return to Previous Menu		Hand Gesture

4.2.3 Music

Within modern car stereo systems one can expect Bluetooth functionality, allowing the driver or passenger to sync their music device to the vehicles built-in console. Although this is now common practice, the way the user then controls their music is often awkward as they have to use the controls from the central console. More importantly, this could be potentially hazardous whilst driving as it requires visual and mental cognition. Although some of the music features can be controlled through haptic controls on the steering wheel, an off-device gesture alternative is worth exploring.

Considering the NHTSA's survey on this matter[8], it is especially important to determine if a series of small-scale gestures can be used to reduce the intensity on as driver's overall cognition.

As the gestures used for interacting with menus apply when choosing artists and songs, only music-specific gestures were tested.

4.2.3.1 Play/Pause

Consensus was strong amongst the participants that playing and pausing the current song should use the same gesture as selecting an option, in the same way that they would press play on their music device.

4.2.3.2 Increasing and Decreasing Volume

For altering the volume, opinion was divided somewhat as the best from of interaction.

As figure blah blah shows 69% of the participants felt that rotating clockwise and anti-clockwise to increase and decrease the volume respectively was the best approach to alter the volume. This can be attributed to their familiarity with rotational knobs for volume control and audio devices such as iPods, where rotations signify a change in volume when a song is currently playing.

The other 31% of participants felt that gesturing up and down were preferable, suggesting a rise and fall of volume. Both options are entirely valid.

4.2.3.3 Next and Previous Song

Skipping to the next and previous songs was a simple task for the participants. Every person that took place in the experiment carried out the same gestures – A swipe left to go to the previous song and a swipe right to the next song

This was primarily due to familiarity with audio controls to skip to the next song, the respective skip next button is always to the right, adjacent to the play/pause. Similarly, the haptic button for the previous song is always on the opposite side. As such, the choice for the participants was therefore an instantaneous reaction.

4.2.3.4 Suggested Gestures

Given how natural the gestures used throughout the music options were to users, I would suggest the following irrespective of technical limitations:

Task	Gesture	Classification
Play	”Screen Tap”	Finger Gesture
Pause	”Screen Tap”	Finger Gesture
Next Song	Swipe Right	Hand Gesture
Previous Song	Swipe Left	Hand Gesture
Increase Volume	Rotate Clockwise	Finger Gesture
Decrease Volume	Rotate Anti-Clockwise	Finger Gesture

Not only are these gestures familiar to users, they are also simple in execution and should hopefully provide a better alternative to haptic controls.

4.2.3.5 Gestures Defined for *In-Car Gestural Interaction*

For this project, all of the preferred gestures suggested by the participants are viable as they can be detected by the LeapMotion. Therefore the suggested gestures will be replicated in implementation.

4.2.4 GPS

4.2.4.1 Zoom In

4.2.4.2 Zoom Out

4.2.4.3 Place Marker

4.2.4.4 Scrolling

4.2.4.5 Suggested Gestures

4.2.4.6 Gestures Defined for *In-Car Gestural Interaction*

4.2.5 Contacts

4.2.5.1 Call Contact

4.2.5.2 End Phonecall

4.2.5.3 Reject Incoming Call

4.2.5.4 Suggested Gestures

4.2.5.5 Gestures Defined for *In-Car Gestural Interaction*

4.2.6 Air-Conditioning

4.2.6.1 Increasing and Decreasing Fan Strength

4.2.6.2 Increasing and Decreasing Temperature

4.2.6.3 Suggested Gestures

4.2.6.4 Gestures Defined for *In-Car Gestural Interaction*

4.2.7 Windows

4.2.7.1 Opening and Closing the Driver-side Window

4.2.7.2 Opening and Closing the Passenger-side Window

4.2.7.3 Opening and Closing the Both Windows Simultaneously

4.2.7.4 Suggested Gestures

4.2.7.5 Gestures Defined for *In-Car Gestural Interaction*

4.2.8 Menu Hot-keys

One of the main objectives of the experiment was to discover if a streamlining process could exist for the main technological features within a vehicle. Regardless of where and what the user was currently on, they could carry out a specific gesture to jump directly to their desired location. Conceptually this was presented to the participants as such:

This proved to be a challenging task and no common agreement existed between the participants. A number of methodologies were proposed by the users:

Myself: "Imagine a scenario where you are currently on your GPS, following a route and you wish to change the album you are listening to. Instead of going back through previous menus, is there a gesture you can think of that will take you directly to music?"

1. **Letter Recognition** – Two people (15%) suggested writing a letter associated with each feature within the vehicle. Although a novel idea, issues could arise if features share the same key letter or if the letter chosen for the feature does not resonate with the user in such cases, the user may select the wrong option or forget the gesture completely, leading to frustration.

2. **Pre-determined Areas** – Another two participants (23%) constructed the idea of having a pre-defined area in the vehicle for each feature. Another interesting concept but has flaws, which can be best surmised in the following conversation with a user during the experiment:

   User: ”Say I have four areas, or however many areas are required in the top left region, I have music. In the top right, I have GPS. And so on. Thats what I would do.”

   Me: ”Is that more of a free-motion gesture then? Will that not be quite a distraction as your hand is away from the wheel and gearstick for a period of time?” User: ”It doesnt need to be a long period of time, just put that hand in that area quickly and itll switch to whatever feature.”

   Me: ”But say you are about to make a swipe gesture and instead it recognises your hand in that region and triggers that instead.”

   User: ”Yeah...its difficult then, but that is what I would suggest.”

   Furthermore, such gestures within the spatial area over the steering wheel could be misinterpreted by other drivers as signs to pull out or overtake, leading to accidents.

3. **Finger Representation** – Four users (31%) proposed the idea of defining each area to a number, in a similar manner to hot-keys on a mobile phone. With the features assigned to a number, they can then be accessed by holding the corresponding number of fingers. Although this method requires recall - the driver needs to remember which number represents each finger this appears to be an elegant solution. The issue of recall can be resolved through the UI design shown on the central console; this could either be a numbered grid or a list of options for example.

4. **Mimic Gestures** – five participants (38%) suggested using the gesture that is the most synonymous for each task. For example, for music, due to the charade gesture for ”sounds like”, these participants felt this was the best way to directly access music.

   Disregarding the limitations of hardware, I would suggest the ”pulling earlobe” to be the standardised gesture for music as it is characteristic of the drivers desire (to access music) and is an identifiable gesture for hearing.

   This, however, is not feasible using the LeapMotion due to its limited range. The alternative methods for gestures that are unattainable using the LeapMotion will be explored in Section blah blah.

4.2.8.1 Suggested Gestures

Disregarding the limitations of hardware, I would suggest the ”pulling earlobe” to be the standardised gesture for music as it is characteristic of the drivers desire (to access music) and is an identifiable gesture for hearing.

This, however, is not feasible using the LeapMotion due to its limited range. The alternative methods for gestures that are unattainable using the LeapMotion will be explored in Section blah blah.

4.2.8.2 Gestures Defined for *In-Car Gestural Interaction*

As it is difficult to replicate mimicking gestures with the LeapMotion as it only tracks hand movements and not your full body, an alternative was required.

Given that the majority of participants did not feel that there was a recognised symbol for every menu option, for the ***In-Car Gestural Interaction Project***, the most reliable way to allow users to switch between menus quickly was to determine their number of fingers detected over a set period of time.

Given that four participants suggested this and the limitations of the LeapMotion, I felt that this was a feasible option. It may require a slight learning curve with users as it is not particularly intuitive, but once this is achieved, it should allow users to access the menu they desire, quickly and efficiently.

Furthermore, the driver will only have to remove their hand from the steering wheel for a brief period of time and as finger-counting is a small-scale gesture, the driver can quickly revert their hands to the wheel as required.

4.2.9 Post Experiment Questionnaire

4.3 Designing the User Interface

4.3.1 Paper Prototype

4.3.2 Storyboard

Chapter 5

Implementation

5.1 Creating the User Interface

5.2 Finger Counting

5.3 Gesture Recognition

Chapter 6

Motivation

The problem is a byproduct of attempting to classify partial linear spaces that can be produced during the execution of an extension of Stinson's hillwalking algorithm for block designs with block size 4 (see [2]). First we need some definitions.

Definition 1. *A Balanced Incomplete Block Design (BIBD) is a pair (V, B) where V is a set of n points and B a collection of subsets of V (blocks) such that each element of V is contained in exactly r blocks and every 2-subset of V is contained in exactly λ blocks.*

Variations on $BIBD$s include *Pairwise Balanced Designs* (PBDs) in which blocks can have different sizes, and *linear spaces* which are PBDs in which every block has size at least 2. It is usual to refer to the blocks of a linear space as a *line*. A partial linear space is a set of lines in which every pair appears in *at most* λ blocks. Here we refer to a BIBD with $\lambda = 1$ as a *block design* and to a partial linear space with $\lambda = 1$, having s_i lines of size i, where $i \geq 3$ and $s_i > 0$ as a $3^{s_3}4^{s_4}\ldots$ structure. For example, a block design on 7 points with block size 3 is given by the following set of blocks:

$$\{(1,2,3),(1,6,7),(1,4,5),(2,5,6),(3,4,6),(3,5,7),(2,4,7)\}$$

and a $3^4 4^1$ structure on 8 points by the following set

$$\{(1,2,3,4),(1,5,6),(1,7,8),(2,5,7),(2,6,8)\}$$

Note that in the latter case we do not list the lines of size 2. Block designs with block size 3 are known as Steiner Triple Systems (STSs). These exist for all n for which $n \equiv 1,3 \pmod 6$ [6]. For example the block design given above is the unique STS of order 7 (STS(7)). Similarly block designs with block size 4 always exist whenever $n \equiv 1,4 \pmod{12}$.

Algorithm 1 allows us to generate an STS for any n and is due to Stinson [7]. This algorithm always works, i.e. it never fails to terminate due to reaching a point where the STS is not created and there are no suitable pairs (x, y) and (y, z).

A natural extension to this algorithm, for the case where block size is 4, is proposed in Algorithm 2. Note that the triples in set $WeightedTriples$ are all initially assigned weight 0 (line 2). Triples can only be selected to make a new block if they have weight zero. If S is a set of triples and X a set of points then the alogrithms **IncreaseWeight**(X, S) and **DecreaseWeight**(X, S) (lines 7 and 10) increment (decrement) the weight of every element of S that contains π, for all pairs π of distinct points from X.

This algorithm does not always work. It is possible for a situation to be reached from which one pair of triples is constantly swapped with another, in which case the algorithm fails to terminate. It is also possible for

Algorithm 1: Generate a block design, block size 3, n points

Stinson(n)
begin
  $LivePairs \leftarrow \{(i,j) : 1 \leq i < j \leq n\}$
  $Blocks \leftarrow \emptyset$
  while $LivePairs \neq \emptyset$ **do**
    choose pairs (x,y) and (y,z) from $LivePairs$
    $LivePairs \leftarrow LivePairs \setminus \{(x,y)\}$
    $LivePairs \leftarrow LivePairs \setminus \{(y,z)\}$
    if $(x,z) \in LivePairs$ **then**
      $LivePairs \leftarrow LivePairs \setminus \{(x,z)\}$
    else
      $Blocks \leftarrow Blocks \setminus \{(w,x,z) : (w,x,z) \in Blocks\}$
      $LivePairs \leftarrow LivePairs \cup \{(w,x)\}$
      $LivePairs \leftarrow LivePairs \cup \{(w,z)\}$
    $Blocks \leftarrow Blocks \cup \{(x,y,z)\}$
  return $Blocks$

Algorithm 2: Generate a block design, block size 4, n points

Stinson4(n)
begin
  $WeightedTriples \leftarrow \{\langle (i,j,k), 0\rangle : 1 \leq i < j < k \leq n\}$
  $Blocks \leftarrow \emptyset$
  while $\langle (w,x,y), 0\rangle \in WeightedTriples \ \wedge \ \langle (x,y,z), 0\rangle \in WeightedTriples$ **do**
    choose $\langle (w,x,y), 0\rangle$ and $\langle (x,y,z), 0\rangle$ from $WeightedTriples$
    for $(u,v,w,z) \in Blocks$ **do**
      DecreaseWeight$(\{u,v,w,z\}, WeightedTriples)$
      $Blocks \leftarrow Blocks \setminus \{(u,w,x,z)\}$
    $Blocks \leftarrow Blocks \cup \{(w,x,y,z)\}$
    IncreaseWeight$(\{w,x,y,z\}, WeightedTriples)$
  return $Blocks$

the algorithm to terminate but fail to create a block design due to reaching a point at which WeightedTriples contains elements of weight zero but does not contain suitable triples (w, x, y) and (x, y, z) with weight zero. In this case the algorithm produces a 4^{s_4} structure (where s_4 is less than the number of blocks in the corresponding block design) for which the complement has no pair of triples (w, x, y), (x, y, z), with weight zero. I.e. the complement graph is diamond-free. When $n = 13$ the algorithm either produces a block design or a 4^8 structure whose complement graph consists of 4 non-intersecting triangles.

The next open problem therefore is for $n = 16$. If the algorithm terminates but does not produce a block design, what is the nature of the structure it does produce? To do this, we need to classify the 4^{r_4} structures whose complement graph is diamond-free.

The cases for which the 4^{s_4} structure has at least 2 points that are in the maximum number of blocks (5) are fairly straightforward. (There are fewer cases as this number increases.). However if the number of such points is 0 or 1, there is a large number of sub-cases to consider. The problem is simplified if we can dismiss potential 4^{s_4} structures because the degree sequences of their complements can not be associated with a diamond-free graph. This leads us to the problem outlined in this report: to classify the degree sequences of diamond-free graphs of order 15 and 16. Note that each point that is not in 5 blocks is either in no blocks or is in blocks with some number of points, where that number is divisible by 3. Thus for every point there is a vertex in the complement graph whose degree is also divisible by 3. In addition, since the number of pairs in both a block design on 16 points and a 4^{s_4} structure are divisible by 6, the number of edges in the complement graph must be divisible by 6. We can immediately eliminate some cases via the following lemmas. In all cases G is a diamond-free graph with n vertices for which every vertex has degree greater than 0 and divisible by 3.

Lemma 1. *If $n = 16$ then no vertex has degree 15.*

Proof Suppose that u be a vertex that has degree 15. Then all other vertices are adjacent to u. Let v be such a vertex. Since v has degree at least 3, there are two vertices, w and x that are adjacent to both u and v. There is a diamond on vertices u, v, w and x. This is a contradiction.

Lemma 2. *If $n = 15$ and $\delta(1) = 12$ then the degree sequence is either*

1. $(12, 12, 12, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3)$*, or*
2. $(12, 6, 6, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3)$.

Proof Let u be a vertex that has degree 12 and $N(u)$ the set of vertices that are in the neighbourhood of (i.e. adjacent to) u. Then there are two vertices, v and w that are not u and are not in $N(u)$. No element of $N(u)$ can have degree greater than 3, for then it would have degree at least 6 and must be adjacent to at least two other elements of $N(u)$, and we would have a diamond. Let $\delta(v)$ and $\delta(w)$ be the degrees of v and v respectively. Without loss of generality we can assume that $\delta(v) \geq \delta(w)$. Then G has degree sequence $(12, \delta(v), \delta(w), 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3)$, where $\delta(v) + \delta(w)$ is divisible by 12. Hence $(\delta(v), \delta(w))$ is $(12, 12)$, $(9, 3)$ or $(6, 6)$.

If $(\delta(v), \delta(w)) = (12, 12)$ there is a solution. In this case every element of $N(u)$ is adjacent to both v and w.

If $(\delta(v), \delta(w)) = (9, 3)$, then suppose that v and w are adjacent. None of the 8 vertices in $N(u)$ that are adjacent to v can be adjacent to w (or we have a diamond), so they must all be adjacent to one of the 4 remaining vertices in $N(u)$. Hence some vertices in $N(u)$ are adjacent to more than one other vertex in $N(u)$, and there is a diamond. A similar argument holds if v and w are not adjacent.

If $(\delta(v), \delta(w)) = (6, 6)$ there is a solution and it can be constructed as follows. Divide the 12 vertices in $N(u)$ into two disjoint sets of equal cardinality $\alpha = \{a_1 \ldots a_6\}$ and $\beta = \{b_1 \ldots b_6\}$. Connect vertex a_i to b_i for

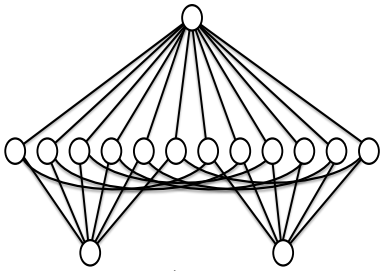


Figure 6.1: Diamond-free $(12, 6, 6, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3)$.

$1 \leq i \leq 6$. Now connect vertex v to all vertices in α and connect w to all vertices in β. Such a graph is shown in Figure 6.1.

Lemma 3. *If* $n = 16$ *and* $\delta(1) = 12$ *then the degree sequence is either*

1. $(12, 12, 9, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3)$*, or*
2. $(12, 12, 6, 6, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3)$*, or*
3. $(12, 9, 9, 6, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3)$*, or*
4. $(12, 6, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3)$.

Proof Let u and $N(u)$ be defined as above, and let v, w and x be vertices that are not u and are not in $N(u)$, with corresponding degrees $\delta(v) \geq \delta(w) \geq \delta(x)$. By an argument similar to the above, G has degree sequence

$$(12, \delta(v), \delta(w), \delta(x), 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3)$$

where $\delta(v) + \delta(w) + \delta(x)$ is divisible by 12. Then we must have $(\delta(v), \delta(w), \delta(x)) = (12, 12, 12)$, $(12, 9, 3)$, $(12, 6, 6)$ or $(9, 9, 6)$ or $(6, 3, 3)$. If $(\delta(v), \delta(w), \delta(x)) = (12, 12, 12)$ then there are at least 3 vertices in $N(u)$ that are adjacent to all of u, v, w and x, which is impossible since, as before, vertices in $N(u)$ must have degree 3. In all other cases there are solutions (which we do not include here).

Chapter 7

Constraint Programming Models

We present two constraint models for the diamond-free degree sequence problem. The first model we call model A, the second model B. In many respects the two models are very similar but what is different is how we solve them. In the subsequent descriptions we assume that we have as input the integer n, where $|V| = n$ and vertex $i \in V$. All the constraint models were implemented using the choco toolkit [3]. Further we assume that a variable x has a domain of values $dom(x)$.

7.1 Model A

Model A is based on the adjacency matrix model of a graph. We have a 0/1 constrained integer variable A_{ij} for each potential edge in the graph such that $A_{ij} = 1 \iff \{i, j\} \in E$. In addition we have constrained integer variables deg_1 to deg_n corresponding to the degrees of each vertex, such that

$$\forall_{i \in [1..n]}\ dom(deg_i) = [3 \,..\, n-1] \tag{7.1}$$

We then have constraints to ensure that the graph is simple:

$$\forall_{i \in [1..n]} \forall_{j \in [i..n]}\ A_{i,j} = A_{j,i} \tag{7.2}$$

$$\forall_{i \in [1..n]}\ A_{i,i} = 0 \tag{7.3}$$

Constraints are then required to ensure that the graph is diamond-free:

$$\forall_{\{i,j,k,l\} \in V}\ [A_{i,j} + A_{i,k} + A_{i,l} + A_{j,k} + A_{j,l} + A_{k,l} \leq 4] \tag{7.4}$$

Finally we have constraints on the degree sequence:

$$\forall_{i \in [1\ ..\ n]}\ deg_i = \sum_{j=1}^{j=n} A_{i,j} \tag{7.5}$$

$$\forall_{i \in [1\ ..\ n-1]}\ deg_i \geq deg_{i+1} \tag{7.6}$$

$$\forall_{i \in [1\ ..\ n]}\ deg_i \ \mathbf{mod}\ 3 = 0 \tag{7.7}$$

$$\left(\sum_{i=1}^{i=n} deg_i \right) \ \mathbf{mod}\ 12 = 0 \tag{7.8}$$

The vertex degree variables deg_1 to deg_n are the decision variables. The constraint model uses $O(n^2)$ constrained integer variables and $O(n^4)$ constraints.

7.2 Model B

Model B is essentially model A broken into three parts, each part solved separately. The first part is to produce a degree sequence that meets the arithmetic constraints. The second part tests if that degree sequence is graphical and if it is the third part determines if there exists a diamond-free graph with that degree sequence. Therefore solving proceeds as follows.

1. Generate the next degree sequence $\pi = d_1, d_2, ..., d_n$ that meets the arithmetical constraints. If no more degree sequences exist then terminate the process.
2. If the degree sequence π is not graphical return to step 1.
3. Determine if there is a diamond-free graph with the degree sequence π.
4. Return to step 1.

The first part of model B is then as follows. Integer variables deg_1 to deg_n correspond to the degrees of each vertex and we satisfy constraints (1), (6), (7) and (8) to generate a degree sequence.

Each valid degree sequence produced is then tested to determine if it is graphical (step 2 above) using the Havel-Hakimi algorithm. We have used the $O(n^2)$ algorithm [4] although the linear algorithm of [5] could equally well be used and would have been more efficient.

If the degree sequence is graphical (step 3) we create an adjacency matrix with properties (2) and (3) and post the constraints (4) and (5) (diamond free with given degree sequence) where the variables deg_1 to deg_n have already been instantiated (in step 1). Finally we are in a position to post static symmetry breaking constraints. If we are producing a graph and $deg_i = deg_j$ then these two vertices are interchangeable. Consequently we can insist that row i in the adjacency matrix is lexicographically less than or equal to row j. Therefore we post the symmetry breaking constraints:

$$\forall_{i \in [1\ ..\ n-1]} [deg_i = deg_{i+1} \Rightarrow A_i \preceq A_{i+1}] \tag{7.9}$$

where $\preceq$ means lexicographically less than or equal. In this second stage of solving the variables $A_{1,1}$ to $A_{n,n}$ are the decision variables.

Chapter 8

Solutions

Our results are tabulated in Table 8.1 for $8 \leq n \leq 16$. All our results are produced using model B run on a machine with 8 Intel Zeon E5420 processors running at 2.50 GHz, 32Gb of RAM, with version 5.2 of linux. The longest run time was for $n = 16$ taking about 5 minutes cpu time. Included in Table 8.1 is the cpu time in seconds to generate all degree sequences for a given value of n.

All our results were verified. For each degree sequence the corresponding adjacency matrix was saved to file and verified to correspond to a simple diamond-free graph that matched the degree sequence and satisfied the arithmetic constraints and this is an $O(n^4)$ process. The verification software did not use any of the constraint programming code.

n	time	degree sequence
8	0.1	3 3 3 3 3 3 3 3
9	0.1	6 6 6 3 3 3 3 3 3
10	0.5	6 6 3 3 3 3 3 3 3 3
11	0.8	6 3 3 3 3 3 3 3 3 3 3
12	1.4	3 3 3 3 3 3 3 3 3 3 3 3 6 6 6 6 3 3 3 3 3 3 3 3 6 6 6 6 6 6 6 6 6 6 6 6 9 6 6 3 3 3 3 3 3 3 3 3
13	3.7	6 6 6 3 3 3 3 3 3 3 3 3 3 6 6 6 6 6 6 6 3 3 3 3 3 3 6 6 6 6 6 6 6 6 6 6 3 3 9 6 3 3 3 3 3 3 3 3 3 3 3
14	14.0	6 6 3 3 3 3 3 3 3 3 3 3 3 3 6 6 6 6 6 6 3 3 3 3 3 3 3 3 6 6 6 6 6 6 6 6 6 6 3 3 3 3 6 6 6 6 6 6 6 6 6 6 6 6 6 6 9 3 3 3 3 3 3 3 3 3 3 3 3 3 9 6 6 6 3 3 3 3 3 3 3 3 3 9 9 6 6 3 3 3 3 3 3 3 3 3 9 9 9 3 3 3 3 3 3 3 3 3 3
15	107.7	6 3 3 3 3 3 3 3 3 3 3 3 3 3 3 6 6 6 6 6 3 3 3 3 3 3 3 3 3 3 6 6 6 6 6 6 6 6 6 3 3 3 3 3 3 6 6 6 6 6 6 6 6 6 6 6 6 6 3 3 9 6 6 6 3 3 3 3 3 3 3 3 3 3 3 9 6 6 6 6 6 6 6 3 3 3 3 3 3 3 9 6 6 6 6 6 6 6 6 6 6 3 3 3 9 9 6 3 3 3 3 3 3 3 3 3 3 3 9 9 6 6 6 6 6 3 3 3 3 3 3 3 9 9 6 6 6 6 6 6 6 6 3 3 3 3 9 9 9 6 6 6 3 3 3 3 3 3 3 3 9 9 9 9 9 9 6 6 6 6 6 6 6 6 12 6 6 3 3 3 3 3 3 3 3 3 3 3 12 12 12 3 3 3 3 3 3 3 3 3 3 3 3
16	339.8	3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 6 6 6 6 3 3 3 3 3 3 3 3 3 3 3 3 6 6 6 6 6 6 6 6 3 3 3 3 3 3 3 3 6 6 6 6 6 6 6 6 6 6 6 6 3 3 3 3 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 9 6 6 3 3 3 3 3 3 3 3 3 3 3 3 3 9 6 6 6 6 6 6 3 3 3 3 3 3 3 3 3 9 6 6 6 6 6 6 6 6 6 6 3 3 3 3 3 9 9 3 3 3 3 3 3 3 3 3 3 3 3 3 3 9 9 6 6 6 6 3 3 3 3 3 3 3 3 3 3 9 9 6 6 6 6 6 6 6 6 3 3 3 3 3 9 9 6 6 6 6 6 6 6 6 6 6 6 6 3 3 9 9 9 6 3 3 3 3 3 3 3 3 3 3 3 9 9 9 6 6 6 6 6 6 6 6 3 3 3 3 9 9 9 9 6 6 6 6 6 6 6 6 3 3 3 9 9 9 9 6 6 6 6 6 6 6 6 6 6 6 9 9 9 9 9 6 6 6 6 6 6 6 6 3 9 9 9 9 9 9 6 6 6 6 6 6 6 3 3 12 6 3 3 3 3 3 3 3 3 3 3 3 3 3 3 12 9 9 6 3 3 3 3 3 3 3 3 3 3 3 12 12 6 6 3 3 3 3 3 3 3 3 3 3 3 3 12 12 9 3 3 3 3 3 3 3 3 3 3 3 3

Table 8.1: Degree sequences, of length n, that meet the arithmetic constraints and have a simple diamond-free graph. Tabulated is n, cpu time in seconds to generate all sequences of length n and those sequences.

Chapter 9

Conclusion

We have presented a new problem, the generation of all degree sequences for diamond free graphs subject to arithmetic constraints. Two models have been presented, A and B. Model A is impractical whereas model B is two stage and allows static symmetry breaking.

There are two possible improvements. The first is to model A. We might add the lexicographical constraints, as used in model B, conditionally during search. The second improvement worthy of investigation is to employ a mixed integer programming solver for the second stage of model B.

We are currently using the lists of feasible degree sequences for diamond-free graphs with 15 or 16 vertices to simplify our proofs for the classification of 4^{s4} structures with diamond-free complements, when the number of points in the maximum number of blocks is 1 or 0 respectively. The degree sequence results for a smaller number of points will also help to simplify our existing proofs for cases where more points are in the maximum number of blocks. Ultimately we would like to use our classification to modify the extension of Stinson's algorithm for block size 4 to ensure that a block design is always produced.

In the more distant future, we would like to analyse the structures produced using our algorithm when $n > 16$. The next case is $n = 25$ and the corresponding diamond-free graphs would have up to 25 vertices.

Acknowledgments

We would like to thank Ian Miguel for his help in setting up the original CSPLib problem [1], and Mike Codish and Brendan McKay for pointing out errors in an earlier draft of this paper.

Bibliography

[1] cspLib. http://www.csplib.org.

[2] C. Colbourn and J. Dinitz, editors. *Handbook of Combinatorial Designs*. CRC Press, New York, USA, 1996.

[3] CHOCO development. CHOCO Solver. http://www.emn.fr/x-info/choco-solver/.

[4] S.L. Hakimi. On the realizability of a set of integers as degrees of the vertices of a simple graph. *J. SIAM Appl. Math.*, 10:496–506, 1962.

[5] A. Iványi, L. Lucz, T. F. Móri, and P. Sótér. On the Erdós-Gallai and Havel-Hakimi algorithms. *Acta Univ. Sapientiae Inform.*, 3:230–268, 2011.

[6] T.P. Kirkman. On a problem in combinatorics. *Cambridge and Dublin Mathematics Journal*, 2:191–204, 1987.

[7] D. Kreher and D. Stinson. *Combinatorial Algorithms*. The CRC Press Series on Discrete Mathematics and its Applications. CRC Press, Boca Raton, Florida, USA, 1999.

[8] Dawn Royal. National survey of distracted and drowsy driving attitudes and behavior, 2003.