Development Project:

Drive AwAI

CTEC 3451: Development Project

Nathan Lloyd

De Montfort University

Word Count:

Acknowledgements

Throughout this dissertation I have received an incredible amount of support, patience and assistance from friends, family and many of the CEM faculty at De Montfort University; without these qualities shown to me, my final year of undergraduate studies would not have been as enjoyable.

I would first like to thank my supervisor Dr. H. Malekmohamadi, who's guidance and support throughout my project has been invaluable.

Similarly, I would also like to thank Dr. A. Khuman for his support and advice through his curriculum and beyond.

I would also like to acknowledge all of the people I met and worked alongside whilst upon my placement at Loughborough Grammar School, with particular note to the Computer Science and Central Services teams for their great support and friendship during my post. Similarly, a big thank you to the great team I worked with at Raspberry Pi. I would particularly like to single our M. Calleja for being an incredible boss and being especially helpful towards my endeavours post role.

I would like to thank some of my past tutors: R. Bunn, L. Feeney, D. Holliday, S. Turner and H. Sohal, for great guidance early on in my education. The skills you built up have undoubtedly contributed to my success. I would also like to thank: A. Minshall, L. Redden and H. Thorpe, for any contributions they've made to this project; my design skills are heavily lacking.

Finally, I would like to thank my friends and family for any support or welcomed distractions they've provided throughout my studies. Specifically, my partner, 'editor and chief' K. Ingram, thank you for all your help; reminding me to add comma's and to use 'however' less. Thank you to my parents who have supported me more so in my final year, funding my ever-growing coffee and takeaway addiction.

Contents Page

Abstract

This report covers a prototyped semi-autonomous driving system utilising neural networks, developed using a Raspberry Pi 4 and a Coral USB accelerator for improved tensor computation. These artefacts will be developed in cohesion with a user interface to ensure control is optimised for the user; simplicity being a pivotal requirement of vehicle control. In addition, an important aspect of the paper is to explore how the domain has developed overtime whilst discovering the underpinning technologies. This research is aimed to enhance the final deliverable whilst gaining insight into the necessity of such technologies in the current social climate.

Literature Review

Nicolas-Joseph Cugnot created the first automobile capable of human transportation in 1769. The development from steam power to a combustion engine was just as significant, and in 1885 it led to the mass line production of the Benz; heralding Karl Benz the father of modern automobiles. Since then, a multitude of improvements on old and new systems have occurred; improving the safety, power and user experience. Arguably one of the most significant progression milestones transpired over half a century after the Benz, with the creation of cruise control; patented by Ralph Teetor (Teetor, 1950). This system now widespread in modern vehicles, is stated to be the progenitor technology to the systems which focus on simplifying and automating aspects of automobiles, expressly to improve the safety for civilian vehicles (Simões et al., 2016). Past developments have somewhat focused on materialistic improvements or improving on current systems, perhaps providing them with extra functionality. Typically, these artefacts created from new developments tend to be focused on a generalized type of problem, such as accessibility, comfort and power; which themselves are dramatically enhanced by the improvement of the utilities and technologies in the underlying processes (Myers and Venable, 2014). The rapidly increasing power of computation is definitive in the improvement in many other processes, Moore's Law seminal paper discusses this heavily, noting that transistors within an integrated circuit double almost every two years (Moore, 1965). Therefore, it is evident to see why machine learning algorithms and artificial intelligence are only now becoming common place in domestic technologies. It is the recent improvements in the availability of data and low cost and more efficient computation that allows for such powerful technologies to be used as an embedded device within a vehicle (Jordan and Mitchell, 2015).

Embedded computers such as the Raspberry Pi are gaining traction within both professional and hobbyist environments; particularly for their flexibility (García-Valls, Ampuero-Calleja, Ferreira, 2017). Embedded computers, or specifically System on Chip devices (SoC) provide considerable computation for a low cost. Moreover, the capability of interfacing with electronic components allows for a wide variety of prototyping; which are frequently applied as IoT solutions. In 2014, it was noted that the wide support and rapid improvements of computer vision on embedded devices, could lead to all automobiles being outfitted with miniature eyes for almost no cost (Senthilkumar, Gopalakrishnan and Sathish Kumar, 2014).Whilst there are a variety of automation projects that use the Pi as a development board, the release of the Pi 4 in 2019 has meant an overhaul in many of its specifications; being pronounced a massive leap forward (Heath, 2019).

Although the specifications have been vastly improved, machine learning tasks are highly demanding for ATX systems; a problem amplified on embedded devices. Nonetheless, due to the prevalence and growing popularity of data dependent services, there are various ways to mitigate the intensive computation specifically for embedded or mobile devices (Zhang et al., 2018). Google's TensorFlow Lite library is another example, it allows for models developed using the standard library to be condensed into a more manageable structure; which is optimal for mobile and embedded devices. As well as this, Google Coral have applied the edge computing paradigm to speed up data access over their new purpose-built, application-specific integrated circuit (ASIC) chip. This chipset is accessible as a development board; in addition to a coprocessors format such as the m.2 and USB 3.0. Aptly named the USB accelerator, Google's Coral TPU has improved tensor computation within a local AI platform (Coral.ai, 2019), a device now optimized with the Pi4; USB improvements providing a higher transfer speed (Spector, L., 2014). With increased computational power over compact devices, it is clear to see how these AI breakthroughs are being applied specifically within the commercial market. A focus of these technologies can be located within automotive industry, as they too have experienced a transformation towards more powerful driver assistance systems (Rodriquez-Hervas, Maile, Flores, 2014).

Despite modern companies such as Tesla dominating the market with their Autopilot feature (Tesla.com, 2019), they were not the first to tackle autonomous vehicles. From as early as the 1920's many endeavours were taken to tackle automated vehicles, including: the famous *Futurama* in 1939 and the Japanese video-based vehicle in 1977 (Gringer, 2018). Since then, Tesla, Waymo and other modern autonomous vehicle producers have achieved cumulative success due to the access of additional training data. New data enables the neural network computer vision systems to better encapsulate rare objects; becoming more effective when on the road. This access to non-simulated data from the vehicle improves the effectiveness of device and the underpinning systems; better data beats more data (Halevy, Norvig and Pereira, 2009). Consequently, the availability of new data has enabled the neural networks that are implemented within said systems to operate more productively. The artificial neural networks implemented within these systems are derived from the biological neural networks that exist within the brain; which are the foundations to our own intelligence. This emulation of a human cognition allows for a similar, albeit less complicated learning process to a human brain; using non-linear processes to encapsulate patterns.

It is estimated that the human brain has 86 billion neurons, each neuron holding 10 thousand connections to adjacent neurons (Azevedo et al. 2009). This has led scientists to declare that the brain is the most complicated object known in the universe (Kaku, 2015). Evidentially, the biological processes in which the brain uses to compute information, are an entirely different way to modern digital computers. Alternatively, artificial neural networks are employed to imitate human biology as method of gaining intelligence. Frank Rosenblatt, a renowned psychologist, is often credited to be the forefather of Artificial neural networks as it was his perceptron (Rosenblatt, 1958) based upon the McCulloch-Pitts neuron (McCulloch and Pitts, 1943), that provided clarity on the complex decisions and logic pathways of the brain.

The single perceptron is limited, and so, layers upon layers of perceptron's are stacked to form basic neural networks, or a rudimentary multilayer perceptron. Single and multilayer perceptron's utilize a feedforward algorithm to ensure data moves from input through hidden layers towards the output. This is the simplest type of an artificial neural network. For more complex applications however, especially that of image or speech recognition, the neural networks require more depth and fine tuning which can be achieved with the addition of backpropagation. Continuous backpropagation as a concept was discussed in the early 60's (Kelley, 1960) (Bryson, 1961), but it wasn't until the 1980's where this concept became a standard implementation within neural networks as a way of modifying the weights between each input (Hopfield, 1982) (Rumelhart, Hinton, Williams, 1986). This optimization algorithm remembers the previous output, enabling it to learn over each epoch by feeding back data to automatically balance the weights; thereby, updating the estimated parameters. Unlike simple feedforward neural networks, the implementation of backpropagation and gradient descent is more often utilized over artificial neural networks, as it automatically finds the optimal combinations of inputs and weights them accordingly; a technique now pivotal to successful computer vision.

Human vision is arguably one of the most important senses, but for the longest time has been one of the most difficult problems to solve (Vasilev et al., 2019). Computer vision has roots within the 60's, one of the first projects was Larry Roberts' Blocks World (1965), which aimed to tackle the computer vision problem by recognition and reconstruction of simple geometric shapes. This research was developed further in an MIT summer project specifically focused on the development of a visual system (Papert, 1966); research now known to be a prime mover of the field. Similar to the key advancements with backpropagation, many of the fundamental advancements came through research and development in the 80's. David Marr's abstraction of the visual system is one of the most notable, using simple structures such as edges to mimic how the brain processes visual data (1982). This research has paved forward many contemporary advancements to the computer vision problem and simple geometric shapes or lines and edges are frequently used within contemporary solutions (Canny, 1986) (Lowe, 1987). At this time the lack of accessible data held back substantial development for focused solutions such as object recognition and face detection, but through widespread utilisation of the internet throughout the mid 90's this problem would begin to decrease (Roser, Ritchie and Ortiz-Ospina, 2019).

The culminated advancements through the availability of data, paired with the rapidly improving algorithms throughout the decade (LeCun et al., 1998), led to these focused and specific visual problems being easier to tackle. Undoubtedly, these developments paved forward development of focused automation systems such as automatic parking (Gorinevsky, Kapitanovsky, Goldenberg, 1996), yet no general solutions to vehicle control had currently been created. Although data was more accessible than ever, a general algorithm for driving was heavily bound by the catalytic type effect that training data has upon the success of models; data trumps everything (Kasparov, 2017). With collectives such as ImageNet building large scale image databases (Deng and Dong, 2009), advancements in all applications of computer vision was sure to follow. ImageNet and deep convolutional neural networks have both successfully been implemented in research focused specifically upon the issue of self-driving vehicles; notably the YOLO project (Nugraha and Su, 2017). Although the collation of data and combination of neural networks have allowed for development of systems, the amount of road traffic accidents and total users of vehicles have also increased (Department for Transport, 2019).

It is estimated by The World Health Organisation that by 2030 road traffic collisions will be the fifth leading cause of death (Who.int, 2019, with supporting research suggesting that human error is the leading cause (Singh, 2015). It is therefore apparent that the proposed technologies to improve the safety and simplicity of driving is key. Although self-driving vehicles are gaining traction with consumers, the adoption of this technology is currently stifled by psychological roadblocks with 78% of Americans fearing riding in an autonomous vehicle (Stepp, 2017). Perceptions are currently bound by a lack of trust and transparency with the underlying processes (Shariff, Bonnefon and Rahwan, 2017), perhaps it is this lack of understanding that alienates consumers. Nonetheless, this shift from the minimization of post-crash injury to collision prevention has the potential to improve road safety, reducing injuries and fatalities.

As artificial neural networks are designed to mimic human neurology, developing such a system would first require focusing upon the basic stimuli humans use to judge road safety, more often this is as simple as green means go, red means stop (Rodriquez-Hervas, 2019). Whilst this first step is comparatively easy, computer vision and neural networks can be applied to many problems on the road, most notably, lane keeping (Chen and Huang 2017) and obstacle avoidance (Bills, Chen and Saxena, 2011). Many systems like the latter have been incorporated into unmanned vehicles, often miniature for the development of a prototype but still returning high levels of success. There has been previous research into the utilisation of a Raspberry Pi for a self-driving car (Bechtel et al., 2018), however, the system did not display the ability for manual control. Furthermore, with the Google Coral and increased data transfer now available through the Pi 4 there are evident opportunities for the proposed system to be much more efficient than the latter. Evidentially the field of intelligent transportation systems is vast and is now widely adopting neural networks as a central solution to many of its problems. As the levels of technology supporting the domain have rapidly developed, so too has the demand for specialist devices, consequently leaving a gap within the literature.

System Design

UI Design

The UI was designed with simplicity in mind. Human Car Interaction (HCaI) is an important consideration of an advanced car system as to not distract the driver from the road (Nakrani, 2015); clearly by displaying only the necessary items this issue would be diverted. This includes directional buttons, a speed slider and speed setter button, access to additional settings via a menu and most importantly, a visual interface to stream the devices onboard camera. As the design of the system was iterative in nature it was important to start with the most important features, evolving the system consistently to be more pertinent. A primitive but simple design exhibiting the general layout of the application was created; figure 1.

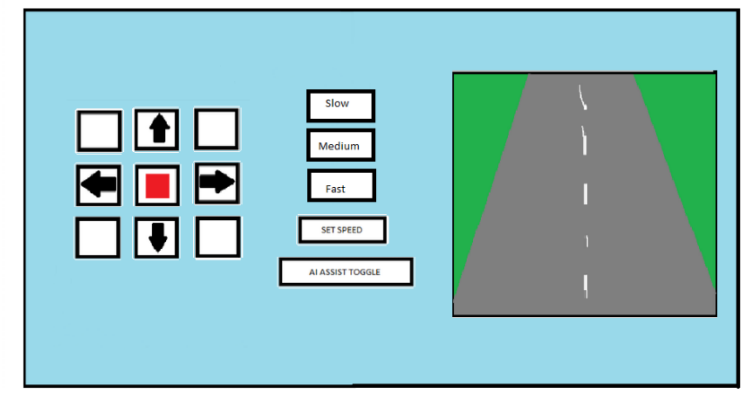


Figure 1: Basic UI Design

From this simple diagram it was clear to see that the proportions of the three originally proposed panels within the user interface were not acceptable, the panels had taken up much of the screen and so they needed to be revised to simplify and declutter. Viewing the road ahead of the vehicle is pivotal in this application, thereby permitting more area for the video component was clearly necessary. Improving the visual elements incorporated within the system was also a requirement, the text used on the buttons within the original design were not optimal and perhaps presented accessibility issues; non English speakers etc.. Moving away from rudimentary “slow, medium and fast” speed selections was also an important factor as this is not realistic to real vehicle control. In the context of an application, it was also necessary to host features that could have appropriate settings modified and so a menu was an integral design consideration, these were taken forward into the next design; figure 2.

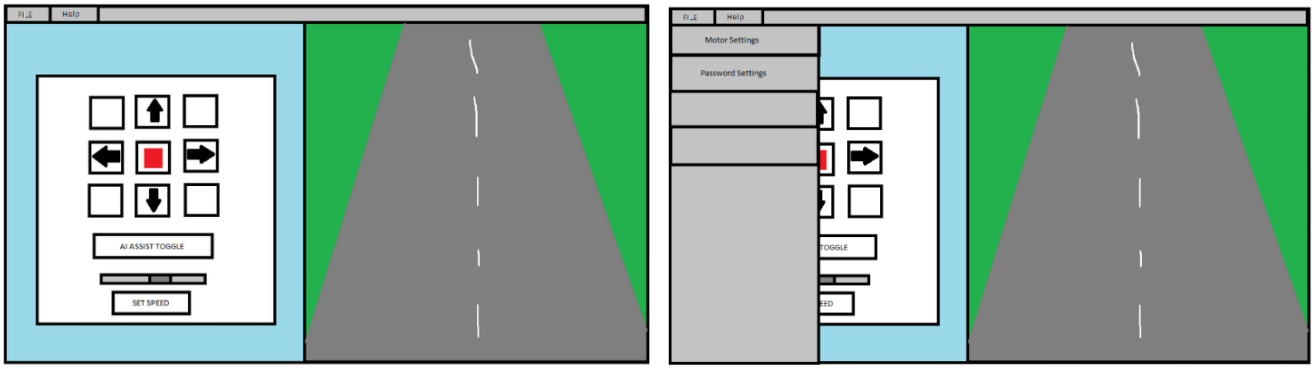


Figure 2: Simplified UI Design with Menu Utilities

The revised design is evidentially simplified and permitting of a larger proportion of the screen to the video, this core design was deemed fit for purpose. Further time was then spent to finalise the main design from a simple sketch-up into a final more familiar and appealing user interface, a paramount commodity for the driver experience both in ease and aesthetic (Schmidt et al., 2010); figure 3.

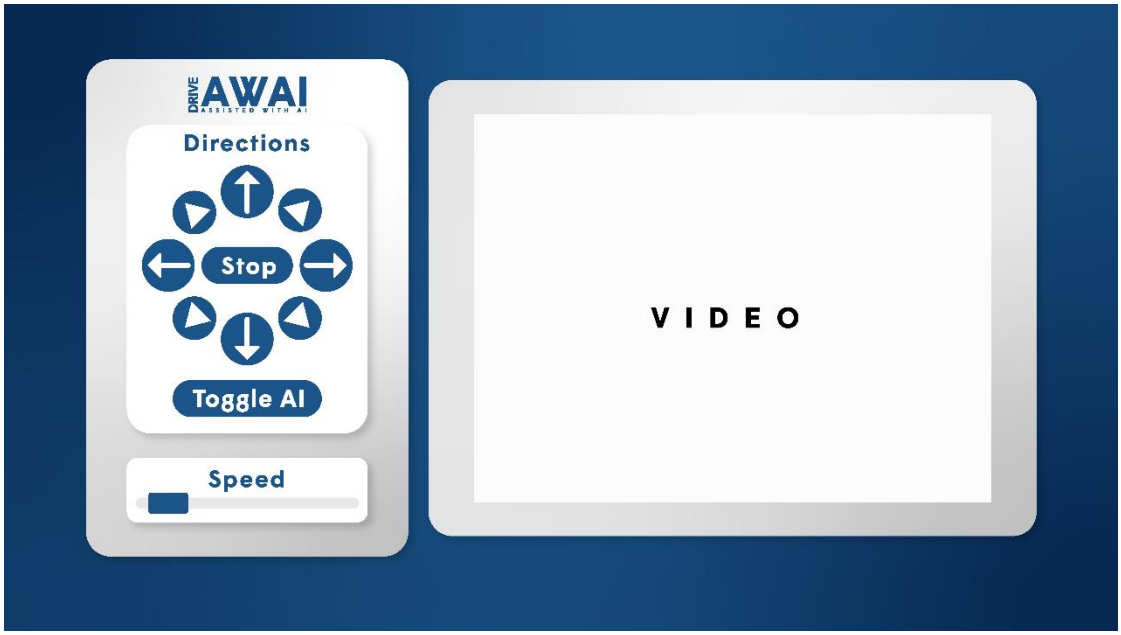


Figure 3: Stylised UI

Vehicle Design & Electronics

Away from the human computer interaction, another core element of the vehicle is the electronical components utilised to achieve mobility. As this project is built using a Raspberry Pi there are various methods of achieving motor control, and so, during research into suitable components the Explorer-Hat was found; appendix 1. The Explorer-Hat has two H-Bridge motor controllers which are suitable for motors either side of the vehicle, the Explorer-Hat also has a multitude of other features built into the device; providing potential for upgrades whilst maintaining simplicity (Shop.pimoroni.com, 2015). Early in development it was clear to see that the expansion this device offers could provide a PIN system, enabling security for the vehicle via the implementation of haptic sensors. Figure 4 highlights the hardware's design and implementation.

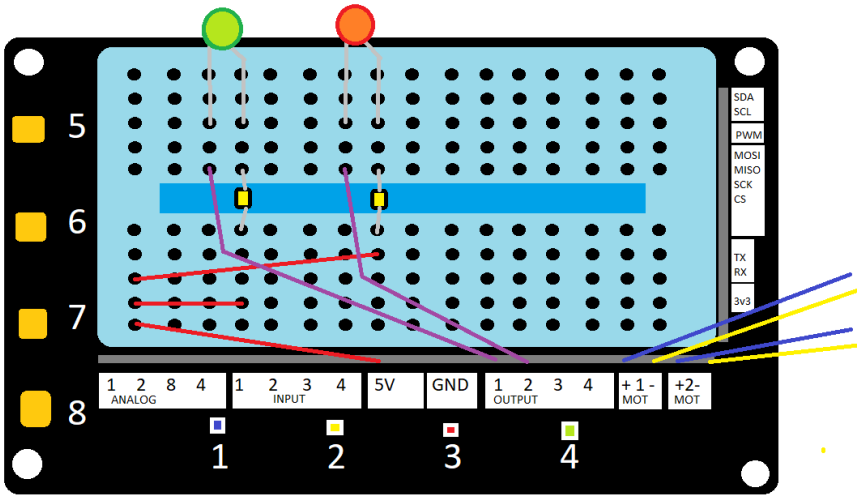


Figure 4: Explorer-Hat Electronical Layout

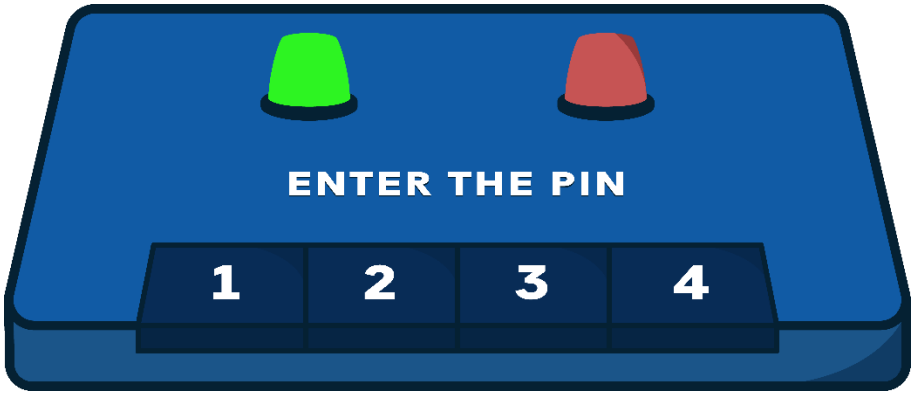


Figure 5: Explorer-Hat Pin System

Whilst the Explorer-Hat fulfils the core components by enabling control of the motors, the additional features it has the possibility to provide are also worth highlighting. Its implementation enables a needed layer of security upon the prototyped vehicle, with unauthorised access to the vehicle reducing safety; emulating security upon a full-size vehicle. The extra hardware used for this project does provide beneficial features for the final vehicle, however, both the core and extra hardware require space within the design of the vehicle.

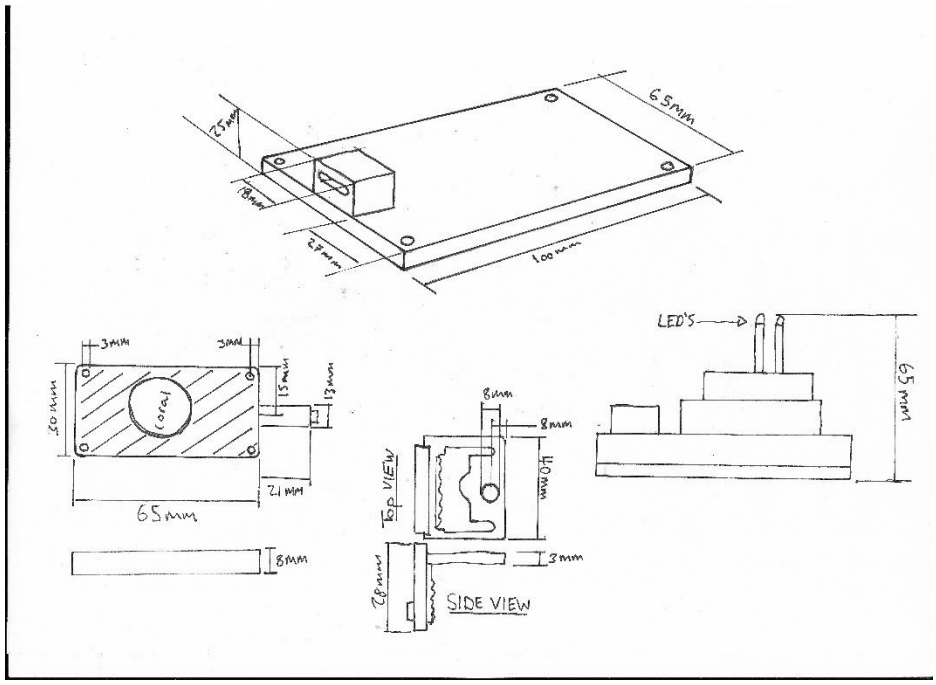


Figure 6: Explorer-Hat Pin System

The vehicles design is of utmost importance for the cohesion of the interlocking parts. One of the first issues encountered when booting the Raspberry Pi 4 was its idle temperature (Bate, 2019), this side effect could cause throttling of the CPU and the gradual depletion of power. Similarly, the dedicated tensor processor is known to heat up causing discomfort to this skin after operating for an extensive period (Coral.ai, 2019); a CPU fan cooler and proper placement of the TPU were implemented to mitigate any heat issue; see figure appendix 2. The final physical constraint was the positioning and angle of the Raspberry Pi camera. Being the only source of input for the neural networks, attaining clear and actionable data was of paramount importance within the design, to ensure the successful stream of video data a case was fitted to the camera which enabled fitting points; as seen bottom centre of figure 6.

UML

The final level of design was visualizing the independent class systems within a UML, allowing for the system model to be well documented. As noted within prior design documents, early development as focused specifically on the fundamentals: UI, electronics and physical design. This focus has inadvertently caused minimal development upon the neural network system that supports the computer vision system; thereby requiring it to be treated as a black box within the UML.

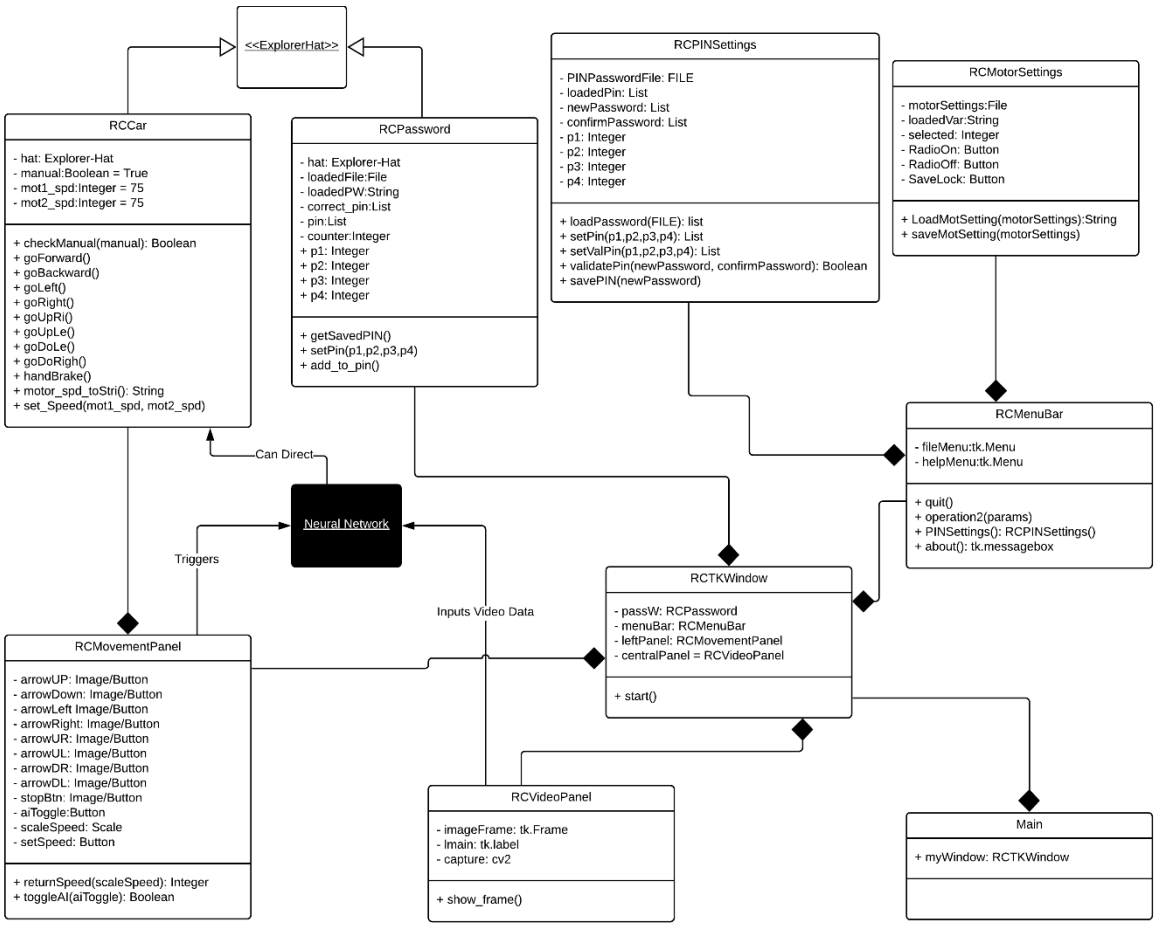


Figure 7: UML

As figure 7 presents the Explorer-Hat is the parent of both the RCCar class and RCPassword class, this is a logical design as they inherit functionalities which enable the interaction between application and hardware components; this abstraction allowed for the application of the DRY principle (Don't Repeat Yourself). This principle was carried forward throughout the development to ensure simplicity within the code. The UML also displays the relationships between all of the classes, for example a menu bar is composed of the menu items. Whilst both the technical and design aspects have been covered, deeper design of the functionality must also be explored to ensure the device is used as it should be by the end user.

Functional Requirements

Further designs were created to explore the functionality of the device and its utilization by the end user, implementing modelling concepts such as use case diagrams allow for the logical flow of the system to be presented formally. These functional requirements have been presented within a use case diagram, use case specification and a physical-software schema.

Use Case Diagram

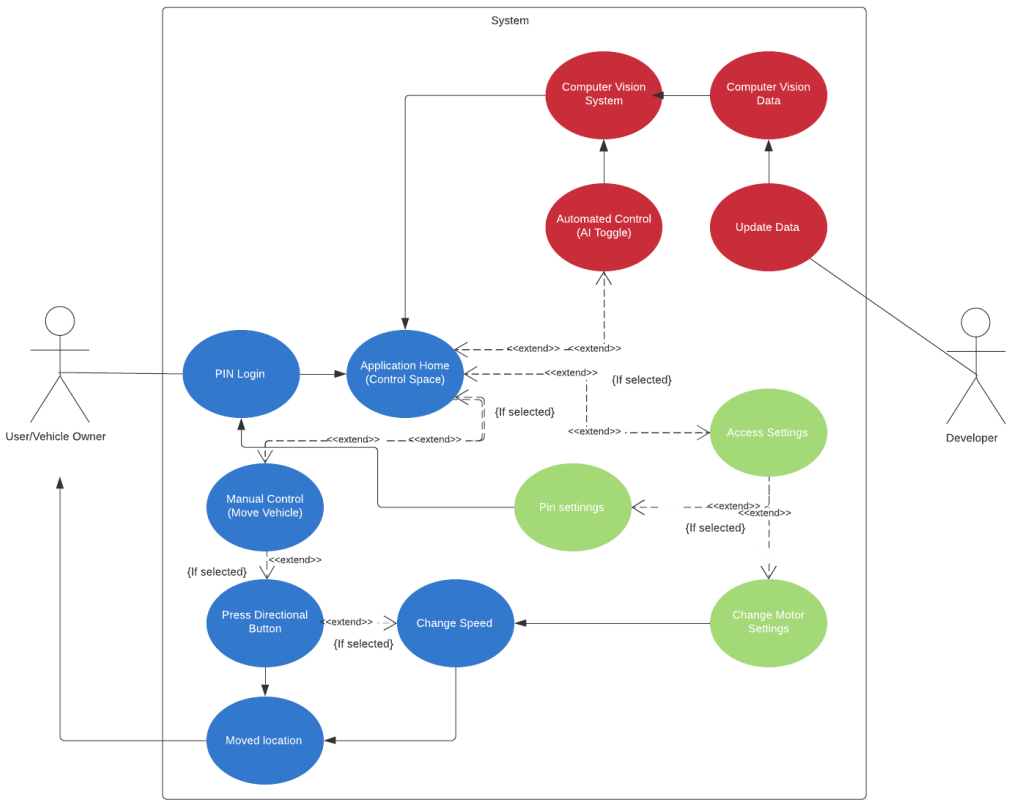


Figure 8: Use Case Diagram [Blue: Manual control] [Red: Automated] [Green: Settings]

The use case diagram enables perspective upon the behavior of a user within the designed system. This perspective has been used to display the different sub-systems used and the path the 'actor' in this scenario can take. In figure 8 both the end user and developers' roles are explored, whilst the end user has already been explored in depth, the developer role has not. The 'developer' or data manager has the ability to provide more data to improve the computer vision system.

Use Case Specification: Manual Control

Overview	
Title	Manual Control on Road Example 1
Description	Green light requires driver to drive forward, 1 minute later vehicle hits another red light. Wait for green light, drive forward and then pull over to the left.
Actors & Interfaces	Car Owner/Driver
Initial Status and Preconditions	Car is stopped, Manual control active, speed is lowest setting
Basic Flow	
Step 1: Observe green light on video stream Step 2: Press forward directional button, continue video observation Step 3: Increase speed on slider, continue video observation Step 4: Observe amber light Step 5: Decrease speed on slider, continue video observation Step 6: Use stop button before red light, continue video observation Step 7: Wait, continue video observation Step 8: Observe green light on video stream Step 9: Press forward directional button, continue video observation Step 10: Press forward left directional button, continue video observation Step 11: Press forward directional button, continue video observation Step 12: Use stop button	
Post Condition	
Any directional button Change starting speed for next route Toggle AI Interact with settings quit	

Table 1: Use Case Specification for Manual Control Road Example 1

Similarly, a use case specification provides insight into the interlinking processes to complete an action. Table 1 presents one way how the system is used to fulfill a specific goal as displayed within the title and description. The basic flow represents a timeline of events or a method to complete said goal whilst post condition displays what options are available to the user whilst in that state. For example, once the user has stopped the vehicle in step 12 they may either interact with the device further or simply quit.

Physical – Software Schema

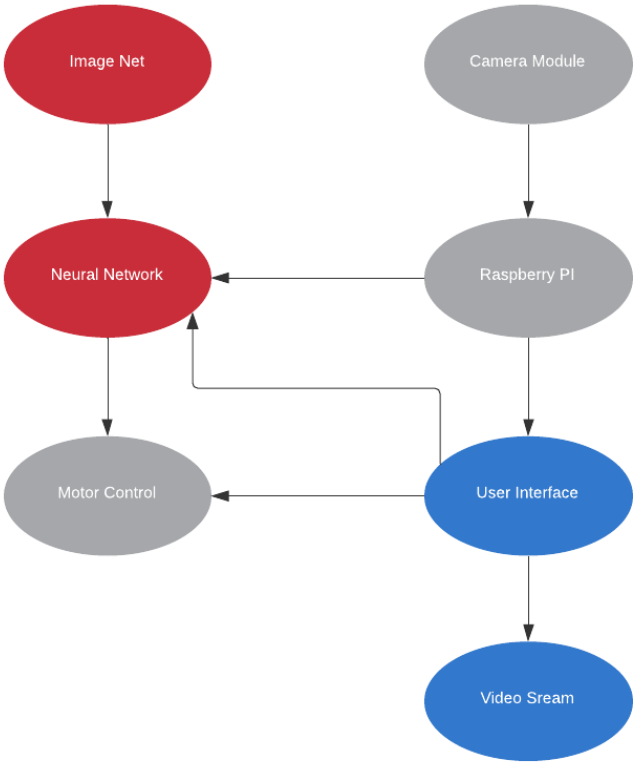


Figure 9: Schema for hardware – software interaction [Blue: UI] [Red: Automated] [Grey: Hardware]

Figure 9 decomposes the underlying systems once more, displaying the relationships between features and a working path throughout the system for the end user. Alternatively, from the other models, this schema presents three different systems: hardware, UI and neural networks, and their interactions with one another; particularly focusing upon the levels of physical control.

Test Plan

As the project is loosely following a dynamic methodology with a prototype style software development lifecycle, testing will be conducted iteratively upon the addition or completion of features. This method of implementation is to ensure that the features added do not negatively interfere with previously added features; working as expected when added to the build. This method of design and testing allows for the evolution of the system, refining aspects as and when it is required. The core scope of the project is unlikely to change and only supporting features to be added, and so, it was believed this approach would return the most useful elements for the final device. Prototyping is typically used in projects where scope is likely to change, but it is known to be a very successful framework to follow when creating user interfaces. Prototyping the UI is optimal as feedback can be received instantly after implementation and testing; allowing for a flexible design. Whilst this flexibility could be exhaustive with potentially rolling back unused features, the general scope of the project well understood therefore this was unlikely. Consequently, the prototyped style approach allowed for a multitude of testing throughout the development of the UI and the underpinning features, allowing for errors to be detected much earlier in development. A lot of the features that are implemented require physical interaction or provide a physical output, ergo, these interactions cannot be shown within an image format alone, requiring testing to be also conducted via video recordings. These recordings which document the testing process and results can be accessed within the shared OneDrive folder; videos italicised.

Testing Motor GUI Control

Although the primary feature of this device is the automatic driving, a vehicles ability to change direction and speed manually is clearly vital. Manual control of the device has been achieved through multiple iterations of development, after soldering the wires (appendix 3) a simple test for forward and backward control was undertaken; see *MotorControl_T1* video. This initial testing proved that the motors could be controlled upon command using basic inputs, requiring the next stage of testing to implement controls via graphical through Tkinter. Shown in figure 10.

Figure 10: Displaying the GUI control

To test the successful implementation of GUI controls upon the motor another round of testing was conducted. Shown through *MotorControl_T2*, it is clear to see that changing the direction from forward to backwards as well as stopping worked as expected. However, the motor was not attached to a chassis during testing, and so, directional turning was not yet implemented; instead returning text within the terminal. The orientation of the motors upon installation would define the direction, therefore, directional features would be added later when added to the chassis.

Testing PIN System

Although not a main feature, security is a very important consideration particularly on a modern vehicle that could be stolen due to the technology inside. To mitigate theft or tampering of the vehicle a PIN system was added. Through the Explorer-hat motor controller implemented four haptic controllers were accessible; this presented an ideal opportunity to add a feature at no extra cost. See appendix 1.

After testing this feature, it was apparent that there was no limit for the PIN system which would allow users to brute force into the system, to solve this issue a three-attempt check was added. With unlimited attempts removed, the ability to change a password was clearly an important feature. Changing a password was naturally the next step of development and whilst a save system was not initially included at the time, the application did load in a PIN from a file. A default password could be entered manually outside of the app. See figure 11.

File and code snippet

Figure 11: Password File Loading

From test video *Brute_T* the addition of 3 attempts check clearly disrupted any potential of mischief by exiting the application altogether. Whilst not discussed within the test video it is visible that when the application is launched, no access to control or camera view is given; further dispelling any potential misuse. To ensure that the PIN system did not interfere with the rest of the application another test was conducted using the same conditions, instead for correct password entry. Witness in *CorrectPW_T* upon the correct password being input, full access is granted.

Implementation Report

The current prototype has focused heavily upon the implementation of the front-end features. As a user upon starting the application a visible boot sequence using LED's can be seen, this is to notify the user that a password is ready to be entered. The password is entered using the haptic sensors upon the explorer-hat, again notifications of correct or incorrect responses can be viewed with LED's as well as through the terminal if needed. The password entered is compared to a password that is loaded from a file, this provides the user the ability to change the password through settings once logged on; the PIN system only allows three attempts before quitting. Once the correct password is entered the main hub is displayed, on the left-hand side are controls and to the right a live video stream from the Pi Camera.

Control of the vehicle is clearly a vital feature of the device and it can be done so by interacting with the directional buttons on screen. Currently not all directional buttons are connected as the fixed orientation of the motor will decide the code that prompts directional changes. The AI toggle feature does not yet work due to no computer vision existing within the device. The speed slider can be used to change the speed of the vehicle. Furthermore, a menu has been included that hosts a few menu items, firstly, there is an about setting under the help menu which activates a messagebox. Under the file menu there are two more items: password settings and motor settings. Password settings enables the user to change the password required to be entered at the log-in PIN phase of the application. Motor Settings is incomplete but is designed to enable a speed changing to be turned on or off.

Bibliography

Adaptive Cruise Control Stage. 2019. [Image] Available at: https://www.bosch-mobility-solutions.com/en/products-and-services/passenger-cars-and-light-commercial-vehicles/driver-assistance-systems/adaptive-cruise-control/

Azevedo, F.A., Carvalho, L.R., Grinberg, L.T., Farfel, J.M., Ferretti, R.E., Leite, R.E., Filho, W.J., Lent, R. and Herculano-Houzel, S., 2009. Equal numbers of neuronal and nonneuronal cells make the human brain an isometrically scaled-up primate brain. Journal of Comparative Neurology, 513(5), pp.532-541.

Bate, A. (2019). *Thermal testing Raspberry Pi 4 - Raspberry Pi.* [online] Raspberry Pi. Available at: https://www.raspberrypi.org/blog/thermal-testing-raspberry-pi-4/ [Accessed 8 Jan. 2020].

Bechtel, M.G., McEllhiney, E., Kim, M. and Yun, H., 2018, August. Deeppicar: A low-cost deep neural network-based autonomous car. In *2018 IEEE 24th International Conference on Embedded and Real-Time Computing Systems and Applications (RTCSA)* (pp. 11-21). IEEE.

Bills, C., Chen, J. and Saxena, A., 2011, May. Autonomous MAV flight in indoor environments using single image perspective cues. In *2011 IEEE International Conference on Robotics and Automation* (pp. 5776-5783). IEEE.

Bryson, A.E., 1961, April. A gradient method for optimizing multi-stage allocation processes. In *Proc. Harvard Univ. Symposium on digital computers and their applications* (Vol. 72).

Canny, J., 1986. A computational approach to edge detection. *IEEE Transactions on pattern analysis and machine intelligence*, (6), pp.679-698.

Chen, Z. and Huang, X., 2017, June. End-to-end learning for lane keeping of self-driving cars. In *2017 IEEE Intelligent Vehicles Symposium (IV)* (pp. 1856-1860). IEEE.

Coral.ai. (2019). [online] Available at: https://coral.ai/ [Accessed 20 Nov. 2019].

Coral.ai. (2019). [online] Available at: https://coral.ai/docs/accelerator/datasheet/ [Accessed 8 Jan. 2020].

Deng, J., Dong, W., Socher, R., Li, L.J., Li, K. and Fei-Fei, L., 2009, June. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition* (pp. 248-255). Ieee.

Department for Transport (2019). Vehicle Licensing Statistics: 2019 Quarter 2 (Apr - Jun). [online] Department For Transport. Available at: https://assets.publishing.service.gov.uk/government/uploads/system/uploads/attachment_data/file/830795/vehicle-licensing-statistics-april-to-june-2019.pdf [Accessed 25 Nov. 2019].

Ferrara, A. and Vecchio, C., 2006, October. Cruise control with collision avoidance for cars via sliding modes. In *2006 IEEE Conference on Computer Aided Control System Design, 2006 IEEE International Conference on Control Applications, 2006 IEEE International Symposium on Intelligent Control* (pp. 2808-2813). IEEE.

García-Valls, M., Ampuero-Calleja, J. and Ferreira, L.L., 2017, May. Integration of data distribution service and raspberry pi. In International Conference on Green, Pervasive, and Cloud Computing (pp. 490-504). Springer, Cham.

Gorinevsky, D., Kapitanovsky, A. and Goldenberg, A., 1996. Neural network architecture for trajectory generation and control of automated car parking. *IEEE Transactions on Control Systems Technology*, *4*(1), pp.50-56.

Gringer, B. (2018). *History of the Autonomous Car*. [online] TitleMax. Available at: https://www.titlemax.com/resources/history-of-the-autonomous-car/ [Accessed 5 Jan. 2020].

Halevy, A., Norvig, P. and Pereira, F., 2009. The unreasonable effectiveness of data.

Heath, N. (2019). Raspberry Pi 4 Model B review: This board really can replace your PC. [online] TechRepublic. Available at: https://www.techrepublic.com/article/raspberry-pi-4-model-b-review-this-board-really-can-replace-your-pc/ [Accessed 20 Nov. 2019].

Hopfield, J.J., 1982. Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the national academy of sciences*, *79*(8), pp.2554-2558.

Jordan, M.I. and Mitchell, T.M., 2015. Machine learning: Trends, perspectives, and prospects. Science, 349(6245), pp.255-260.

Kaku, M., 2015. The future of the mind: The scientific quest to understand, enhance, and empower the mind. Anchor Books.

Kasparov, G., 2017. *Deep thinking: where machine intelligence ends and human creativity begins*. PublicAffairs.

Kelley, H.J., 1960. Gradient theory of optimal flight paths. *Ars Journal*, *30*(10), pp.947-954.

LeCun, Y., Bottou, L., Bengio, Y. and Haffner, P., 1998. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, *86*(11), pp.2278-2324.

Lowe, D.G., 1987. Three-dimensional object recognition from single two-dimensional images. *Artificial intelligence*, *31*(3), pp.355-395.

Marr, D. and Vision, A., 1982. A computational investigation into the human representation and processing of visual information.

McCulloch, W.S. and Pitts, W., 1943. A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*, *5*(4), pp.115-133.

Myers, M.D. and Venable, J.R., 2014. A set of ethical principles for design science research in information systems. Information & Management, 51(6), pp.801-809.

Nakrani, P.K., 2015. *Smart car technologies: A comprehensive study of the state of the art with analysis and trends*. Arizona State University.

Nugraha, B.T. and Su, S.F., 2017, October. Towards self-driving car using convolutional neural network and road lane detector. In *2017 2nd International Conference on Automation, Cognitive Science, Optics, Micro Electro-Mechanical System, and Information Technology (ICACOMIT)* (pp. 65-69). IEEE.

Papert, S.A., 1966. The summer vision project.

Roberts, L.: Machine perception of 3-d solids. In: PhD. Thesis (1965)

Rodriquez-Hervas, B. (2019). *DRIVE Labs: Classifying Traffic Signs and Traffic Lights with SignNet and LightNet DNNs – NVIDIA Developer News Center*. [online] News.developer.nvidia.com. Available at: https://news.developer.nvidia.com/drive-labs-signnet-and-lighnet-dnns/ [Accessed 16 Nov. 2019].

Rodriquez-Hervas, B., Maile, M. and Flores, B.C., 2014, May. Comparative of signal processing techniques for micro-Doppler signature extraction with automotive radar systems. In Radar Sensor Technology XVIII (Vol. 9077, p. 90771A). International Society for Optics and Photonics.

Rosenblatt, F. (1958). The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, 65(6), pp.386-408.

Roser, M., Ritchie, H. and Ortiz-Ospina, E. (2019). *Internet*. [online] Our World in Data. Available at: https://ourworldindata.org/internet [Accessed 20 Dec. 2019].

Rumelhart, D.E., Hinton, G.E. and Williams, R.J., 1986. Learning representations by back-propagating errors. *nature*, *323*(6088), pp.533-536.

Schmidt, A., Dey, A.K., Kun, A.L. and Spiessl, W., 2010, April. Automotive user interfaces: human computer interaction in the car. In *CHI'10 Extended Abstracts on Human Factors in Computing Systems* (pp. 3177-3180). ACM.

Senthilkumar, G., Gopalakrishnan, K. and Kumar, V.S., 2014. Embedded image capturing system using raspberry pi system. International Journal of Emerging Trends & Technology in Computer Science, 3(2), pp.213-215.

Shariff, A., Bonnefon, J.F. and Rahwan, I., 2017. Psychological roadblocks to the adoption of self-driving vehicles. *Nature Human Behaviour*, *1*(10), p.694.

Shop.pimoroni.com. (2015). *Explorer HAT Pro - Prototyping Board for Raspberry Pi Projects – Pimoroni Store*. [online] Available at: https://shop.pimoroni.com/products/explorer-hat [Accessed 5 Jan. 2020].

Simões, J., Jaco, D., Gomes, R., Araujo, P., Fernandes, A. and Seabra, E., 2016. Evolution of the cruise control. In International Conference on Regional Triple Helix Dynamics (HELIX2016). Instituto Politécnico de Castelo Branco (IPCB).

Singh, S., 2015. Critical Reasons for Crashes Investigated in the National Motor Vehicle Crash Causation Survey. DOT HS 812 115. *National Highway Traffic Safety Administration, US Department of Transportation*.

Spector, L., 2014. USB 3.0 speed: real and imagined. PC World, 26.

Stepp, E., 2017. Americans Feel Unsafe Sharing the Road with Fully Self-Driving Cars. *AAA NewsRoom*.

Teetor, R.R., 1950. Speed control device for resisting operation of the accelerator. U.S. Patent 2,519,859.

Tesla.com. (2019). Autopilot. [online] Available at: https://www.tesla.com/en_GB/autopilot [Accessed 30 Dec. 2019].

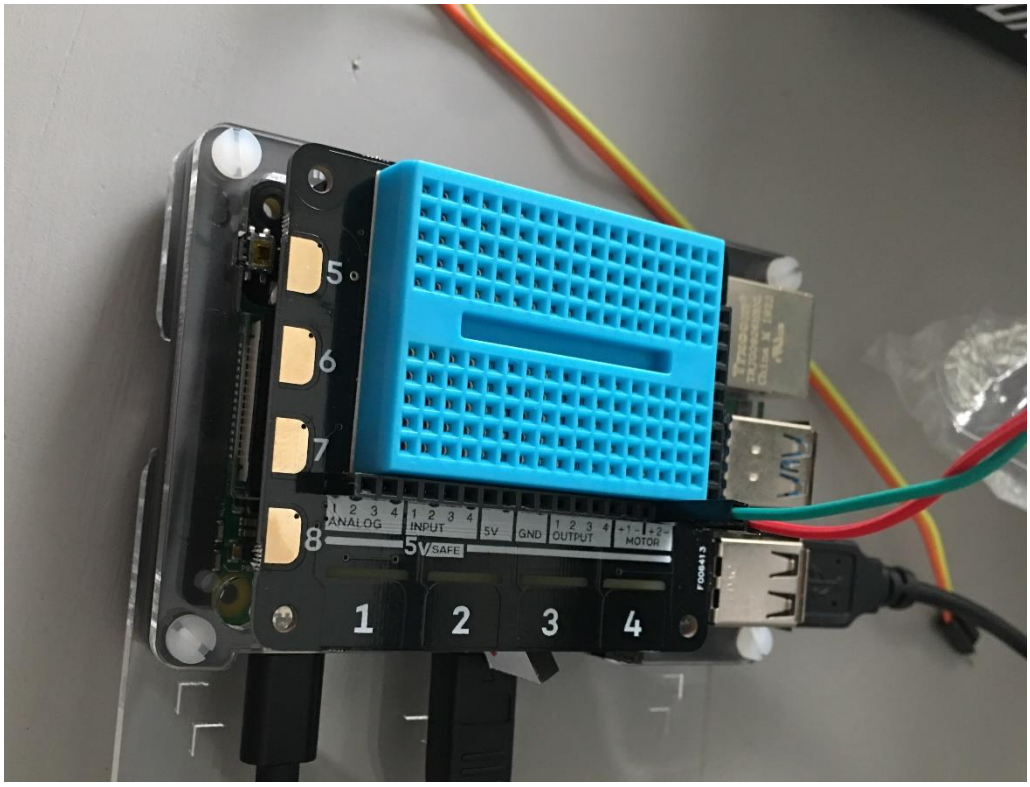
Who.int. (2019). WHO | World Health Organization. [online] Available at: https://www.who.int/ith/other_health_risks/injuries_violence/en/ [Accessed 05 Jan. 2020].

Vasilev, I., Slater, D., Spacagna, G., Roelants, P., Zocca, V., 2019. Python deep learning: exploring deep learning techniques and neural network architectures with PyTorch, Keras, and TensorFlow.

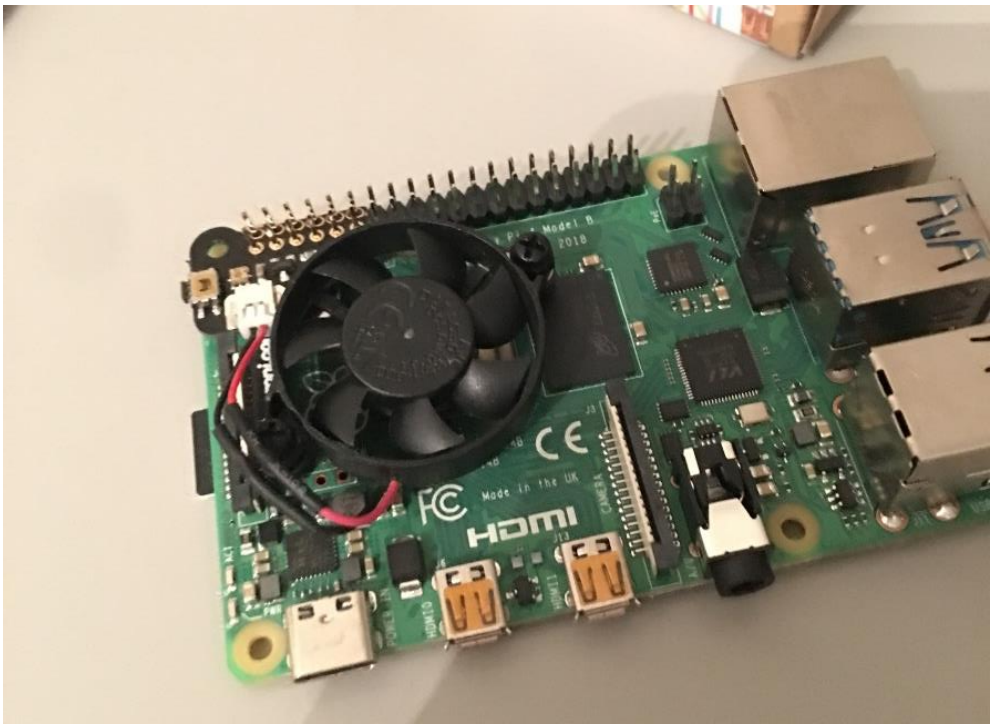
Zhang, X., Zhou, X., Lin, M. and Sun, J., 2018. Shufflenet: An extremely efficient convolutional neural network for mobile devices. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (pp. 6848-6856).

Appendices

Appendix 1 – Explorer-Hat



Appendix 2 – CPU Fan



Appendix 3 – Solder