Design and Build of an Anthropomorphic Active Vision System

By

Sahand Shaghaghi

BEng, Ryerson University, 2016

A Thesis

presented to Ryerson University

in partial fulfillment of the

requirements for the degree of

Master of Applied Science

in the program of

Civil Engineering

Toronto, Ontario, Canada, 2019

Author’s Declaration

Abstract

Design and Build of an Anthropomorphic Active Vision System

Sahand Shaghaghi

MASc, Civil Engineering, Ryerson University, 2019

The aim of this thesis project is to design an anthropomorphic active vision system which builds on biomimicry of the human visual system. The proposed system has some of the characteristics associated with such a visual system, such as degrees of freedom associated with movements of the eyes in addition to foveation and vergence capacities associated with human vision.

Through this thesis, novel approaches are proposed in regard to specific elements of system design and system testing. Novel approaches have been proposed regarding foveation and vergence relating to system design and use of Inertial Measurement Unit (IMU) devices incorporating Kalman filters relating to testing of the prototyped device.

For foveation, integration of fast Deep Convolutional Neural Networks (DCNN) has been proposed. For vergence, a variation of Cross-correlation has been used. This method has the benefit of being computationally inexpensive which is beneficial for real-time operations of the proposed system.

Acknowledgments

I would like to thank my supervisor Dr. Michael Chapman for his support of my studies throughout this master's degree program. He has given me the chance to truly pursue what I am passionate about and I am grateful to him because of this. He has supported me with my ideas regarding my thesis research, though ambitious in nature. Again, I am truly grateful for this.

I would like to thank Dr. Mai for her teachings regarding vision and Human Computer Interaction (HCI) which have been vital for the completion of this project.

I would like to thank Dr. Ferworn for his teachings regarding robotics and biomimetics which has also been vital for the completion of this project.

I would like to thank Jimmy Tran. His support at the Ryerson Library's Collaboratory has been instrumental towards learning the needed skills for the fabrication of the prototype needed for this thesis.

I would like to thank Liam Gregory for his comradery during the completion of this thesis project.

I would like to thank Hamdy Elsayed for his support for learning the needed knowledge regarding SLAM procedures and signal analysis needed for this project.

I would like to thank the organizers at Ryerson Library Collaboratory, where the majority of the physical build and experimentation for this project was completed.

I would like to thank all the professors I have taken courses with, within my time as a master's student, through departments of Civil Engineering, Electrical Engineering, and Computer Science. These courses have been instrumental in the completion of this thesis project.

Table of Contents

List of Tables

List of Figures

Appendix List

Chapter 1 . Introduction

1.1 Thesis Statement

Visual biomimicry has been a topic of interest for many decades now. Researchers have tried to mimic animate vision organisms to design vision systems which function seamlessly. The aim of this project has been to demonstrate that it is possible to design and build systems which could replicate the human visual system regarding foveation and vergence. Through this thesis, efforts were made to design an experimental apparatus which would then allow us to execute algorithms mimicking human foveation and vergence. So, in short, it could be said that: It is possible to create a system which replicates human’s visual system, and which is able to foveate on salient objects and for the said system to verge on a point of foveation, with both eyes.

1.2 Motivation

Vision capture devices are a staple part of many robotic devices at the present. For a long time, 2D vision systems were the standard in robotic device integration due to the unavailability of 3D vision capture systems and due to the limitations associated with the processing power needed for 3D vision systems. As of late, with the advancements in parallel processing and availability of better GPUs, this is not the case anymore. As such efforts have been made towards study and integration of 3D capture devices in robotics design. The majority of these capture devices have fixed camera positioning, which makes calibration and 3D image registration and reconstruction feasible.

In my opinion, the next steps in terms of advancements in 3D vision capture technology would be the design of anthropomorphic active vision systems which mimic human eye operations including human eye movements. This opinion is deeply rooted in biomimetics which will be reviewed in Chapter 2.

Alas, in this thesis efforts have been made to study the feasibility of this proposition, through design and build of a novel device which could possibly fulfill these objectives. To this end, efforts were first made to understand the major components of human vision and then to build a system based upon this organism.

Such a system, if designed and built successfully, could have major positive implications in different sectors such as research and development, biomedical industries, etc. Some of the possible contributions of such a technology would be “Depth-Precepting Object-Centric Visual Capture Devices”, humanoid robotics, etc. A possible area of contribution could be autonomous vehicles. Object centric capture devices could be of benefit to possibly better detect and analyze objects of interest.

Another area of contribution could be eye implants. This would be possible with advancements in signal processing and synthetic muscles. With advancements in anthropomorphic vision systems, this technology could hold future solutions for functional eye implants.

Though these future applications may appear farfetched, it should be kept in mind that the idea of a fully encompassing personal computer which would fit in one's pocket was a far-fetched idea only a few decades ago.

1.3 Objective

There are several objectives associated with this study. There has been a major emphasis on finding novel solutions regarding anthropomorphic active vision. These objectives can be classified into 5 main categories:

1. Design of a functional anthropomorphic vision system
2. Physical build of the designed system
3. Design of the algorithmic structure needed for system functionality
4. Design of novel foveation and vergence algorithm solutions
5. Design of benchmarking procedures for the proposed system

Below, a brief description of each of these categories is provided. A more detailed description of these categories is provided throughout this written thesis:

Since the proposed system is niche in nature, only a few research projects have made attempts at design of such systems and as such, no full plans of such system designs are available. Hence, there was a need to come up with original plans for the physical system design, taking into consideration available elements from previous designs. Due to the complexity of the envisioned system, the proposed design was broken into its making elements.

First, physical design was tackled since the presence of a physical prototype would allow experimentations with algorithm design. Throughout design efforts, the idea of iterative design was kept in mind: It is better to have a prototype than no prototype at all. Flaws and shortcomings of each prototype could be addressed through iterations of the design.

Next, the algorithm structure and software protocol were considered. In this step, the most suitable platforms were chosen to work in tandem for operations of the proposed system. In order to come up with the best solution for this, different platforms were experimented with.

A major emphasis of this thesis project is the creation of solutions for foveation and vergence. As such different algorithms were designed which could fulfill these tasks. Computer vision methodologies and machine learning methodologies were experimented with, resulting in some novel algorithm designs.

Finally, methods for benchmarking of the designed system were created. Due to the novel nature of the proposed system, there were no standard tests which could be used for evaluation of the performance of designed system. Furthermore, there is no consensus on test types to be used. Hence, inspiration from physiology and neuroscience was taken to design tests for benchmarking of the proposed system. Inertial measurement units and Kalman filters are used for this purpose.

1.4 Contribution

This work closely follows previous studies set forth in the area of anthropomorphic active vision. These works have been extensively reviewed in Chapter 2 of this thesis. Mentioned studies have tried to address the many challenges associated with the design of active vision systems. This includes the physical design of such systems and algorithm and logic design needed for functioning of such systems. This work builds on ideas proposed in these studies to address such challenges.

Even though there has been major progress made by previous studies, there are still many questions needing to be answered. Even the most advanced anthropomorphic vision systems built to-date are still lacking in capabilities and performance when compared to the human visual system. This is not only a matter of optimization but a matter of deeply rooted algorithmic and physical build upgrade needs.

To this end, in this study efforts were made to introduce unique approaches which could make operations of such systems more robust and closer in nature to the human visual system. These contributions are as below:

1. Physical system build
2. Foveation algorithm design
3. Vergence algorithm design
4. System benchmarking procedure design

These contributions are detailed in Chapter 3.

1.5 Thesis Outline

In this chapter, motivation, contributions, and objectives are outlined.

In the next chapter, a literature review of composing concepts of this thesis is presented. This includes topics such as human vision research in the areas of neuroscience, robotics and computer vision. Biomimicry is reviewed due to its relevance to this project. Then, a brief review of human vision physiology is provided. Previous efforts towards the build of anthropomorphic active vision systems are listed and then relevant computer vision concepts are detailed. This includes saliency, Viola Jones and cross-correlation, to name a few.

In the third chapter, procedures for different steps of design and build of the proposed system have been detailed. This is done in chronological order, so the reader can get a clear understanding of the design process.

In the fourth chapter, design of procedures for experimentation and benchmarking of the proposed system have been discussed and the results are detailed. Here the necessary background for the techniques and devices used in these procedures is provided.

Finally, in the fifth chapter, efforts are concluded and proposed future works have been laid out.

Chapter 2 . Background and Related Works

This project has great ties with biomimetics. As such, this section will provide an overview of biomimetics followed by a review of efforts towards design and build of anthropomorphic active vision systems as they relate to biomimetics.

In order for this thesis project to be graspable for readers, a brief review of eye characteristics as they relate to biomimetics and system design is included. This is followed by a review of visual computing concepts used in this thesis, this includes topics such as saliency, Viola Jones object detection framework, and convolutional neural networks.

2.1 Anthropomorphic Vision

2.1.1 Brief Review of the Human Visual System

Animate visual systems have long been of interest in academic circles. Human and animal vision have been studied in detail for centuries. These studies, which include philosophical hypotheses regarding functionality of these organisms, go back to the time of ancient Greece. These studies have branched out to various fields of study such as physiology through the 1800s and 1900s, followed by ophthalmology and neuroscience later on.

The coverage of human vision in this section is by no means extensive and is only focused on what is needed for the purpose of this thesis. It should be noted that human vision is extremely sophisticated in nature and it would take books upon books to cover what has been discovered regarding human vision up to this day. One might say, considering the sheer amount of discovery regarding human vision so far, that we have only scratched the surface. Today, there is a major focus on neuroscientific aspects of vision and relationship between animate visual systems and brain. Animate vision refers to a vision system with human like characteristics. This term was first coined by Dana Ballard. To review what is needed for this thesis project several studies relating to human vision spanning from the early 1900s to 2000s have been utilized.

The principles of human vision most relevant to this thesis project are eye physiology relating to its physical structure, muscles and eye movements, specifically: saccades, foveation, and vergence. Below the physical structure of the eye muscles could be seen illustrated:

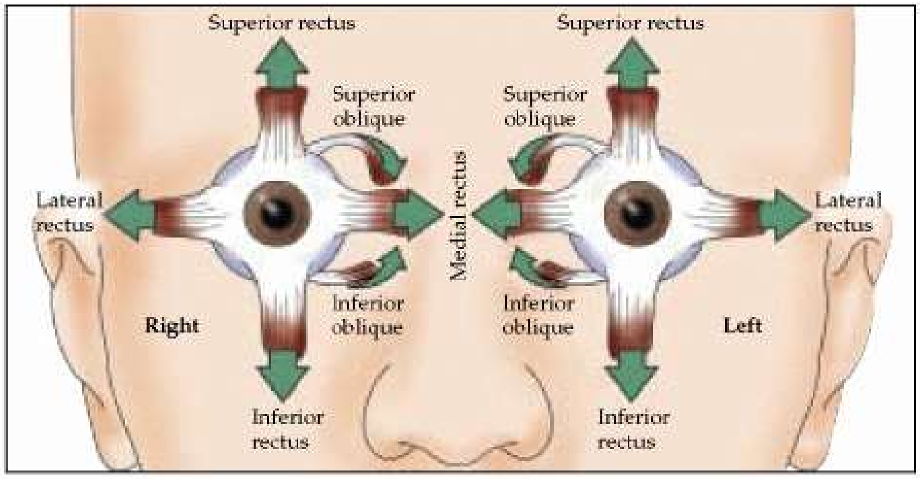


Figure 2.1 Eye muscles and their placing, Accessed at: [1]

As illustrated in Figure 2.1, each eye has 6 muscles responsible for the movements of the eye. Lateral rectus and medial rectus are responsible for pan movement of the eye. This is achieved by flexion and extension of these two muscles. As one is flexing, the other extends accordingly. Superior and Inferior rectus and Superior and Inferior oblique are jointly responsible for the tilt movement of the eye. This depends on the horizontal position of the eye [1]. When straight ahead all four muscles contribute by flexion and extension. When the eye is abducted (away from the nose), rectus muscles are the main contributors. And when the eye is adducted (towards the nose), oblique muscles are the primary contributors.

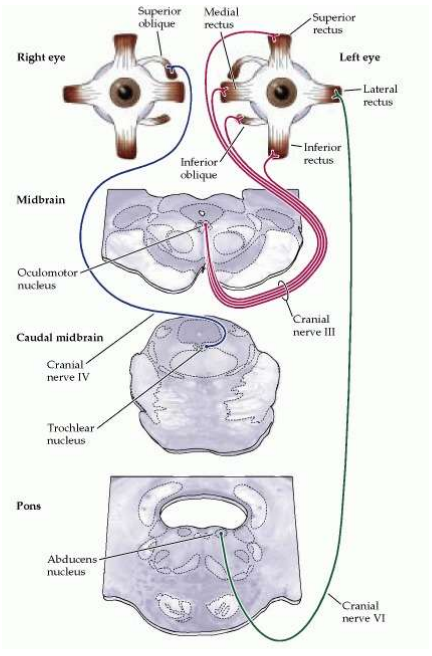


Figure 2.2 Eye muscle connections to brain regions, Accessed at: [1]

Above (Figure 2.2), nerve connections of mentioned muscles to their relevant brain regions can be seen. Though this is not of direct use in this project, it is beneficial to have some basic knowledge of neurological aspects of the human visual system. The oculomotor muscles are connected to lower motor neurons through cranial nerves [1]. As could be seen, the majority of the oculomotor muscles are connected to the oculomotor nuclease at the midbrain region. Superior oblique is connected to the trochlear nucleus and the lateral rectus to Abducens nucleus.

This thesis project focusses on the pan and tilt movement of the eye. The muscle pairs were replaced with servo motors in the proposed design. Ideally, to mimic these muscles, artificial muscles should be used. This could be a topic of future research.

Descartes first modeled eye movements in 1630 [2]. Following this, different types of eye movements were classified and explained by Dodge in the 1900s [3]. These include saccades, pursuit, vergence and Vestibulo-ocular movements. In this thesis project, only saccades and vergence are focused on. Saccades are rapid movements of the eye which abruptly change the point of fixation [1]. These are needed for foveation and discovery of salient points of interest. In this project, there has been an aim of recreating foveation. This project has not attempted to replicate saccades exactly as they happen for an animate organism. The mimicry of saccades in this thesis project is subjective at best. Pursuit movements are much slower tracking movements of the eye which are meant to keep the salient object of interest on the fovea [1]. As mentioned, this type of movement has been beyond the scope of this thesis project. Vergence movements align the two eyes to the same salient target point in the visual scene. This is needed in order for depth perception. This is done through convergence or divergence of two eyes.

Westheimer first studied and characterized saccadic movements of the eyes in the 1950s [4]. He came up with an apparatus design and setup for the recording of fine eye movements and was able to record velocity and acceleration of eye movements (Figure 2.3, 2.4, 2.5). He made a few interesting observation regarding the nature of saccadic eye movements one of which is “Overshoot and small Oscillation around the final position”[4]. It is interesting to note that this behavior is present in the present design as well. When conducting saccadic foveation this overshoot, and oscillation is present in the vicinity of the salient target. This raises the question of whether the human motor command is of stepwise nature as well? Another interesting observation made by Westheimer is the fact that velocity of saccade increases with the extent of angular distance of the target. This then partially explains the constant foveation time for targets of different distance, discussed in Chapter 4.

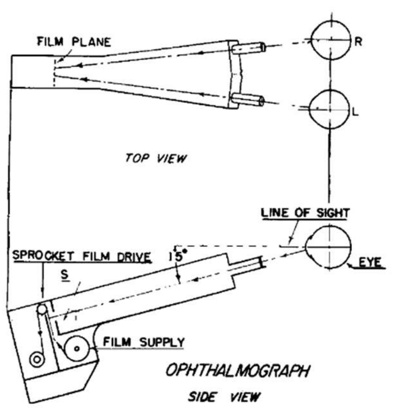


Figure 2.3 Ophthalmograph as used by Westheimer, Accessed at: [4]

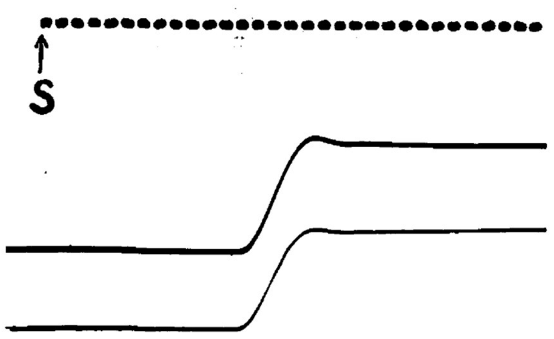


Figure 2.4 - 20-degree saccadic eye movement (angle vs time), Accessed at: [4]

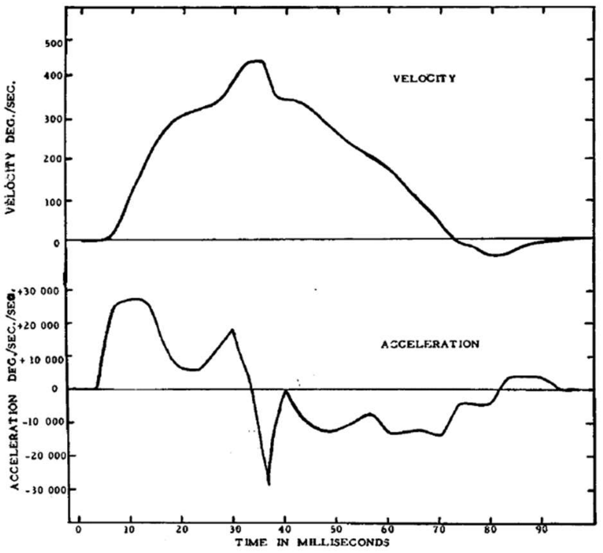


Figure 2.5 Acceleration and velocity as recorded by Westheimer, Accessed at: [4]

Further efforts were made by the likes of Thomas [5] and Robinson [6] in order to analyze and describe saccadic eye movements (Figure 2.6). Efforts were also made to understand the relationship between saccadic eye movements and smooth pursuit eye movements [7]. In the below diagram it could be seen that the times relating to vergence for all angular distances are approximately the same range (0 to 100ms), which confirms the findings by Westheimer.

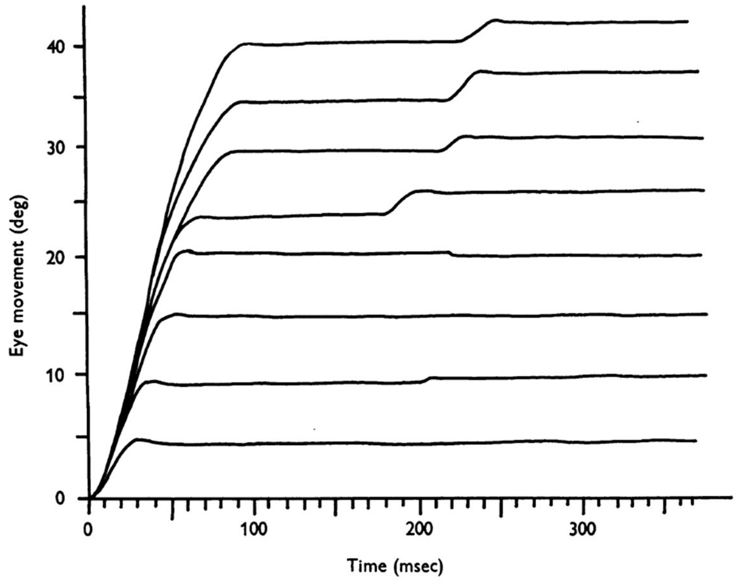


Figure 2.6 Saccadic recordings for steps of 5 degrees, Accessed at: [6]

Efforts are made by Yang et al. [8] to analyze saccade, vergence and combined eye movements latency in relationship to age and growth stages. Latency is defined as “time from the presentation of the target to the commencement of the saccade” [9]. This is a factor of relevance in specific system design related cases, for example, the robot ICUB, which is meant to explore biomimicry of a human infant. In such a case, said latency observations could be used in order to benchmark the system. Yang makes several observations through this study [8]: First, that there is a progressive decrease in mean latency with age increase. Second, that the convergence latency is longer than divergence latency in adults and most of children (this by itself could be a topic of exploration). And third, that saccade- and vergence- mechanisms are distinct and mature progressively with age. This last observation is of importance to this thesis project due to the fact that the proposed design is based on the fact that saccade and vergence are separate operational entities with their own self-reliant designs.

As it could be seen in Figures 2.4 and 2.6, it is common to employ the use of time vs eye movement magnitude combination for characterization of eye performance. This methodology will be used in Chapter 4 of this thesis project for benchmarking of the proposed system. Inspiration has been taken from these diagrams to characterize and benchmark the performance of the proposed system.

Innovation and unique system design approaches relating to human visual system date back at least few decades. Idea of eye implants have been explored as early as the 1950s [10]. Though, the implant suggested was solely aesthetic in nature (Figure 2.7), it still showcases an interest in building technologies which could be replacements for human visual system components. As such, the desire to replicate visual organism into technological creations can be traced back to the previous century.

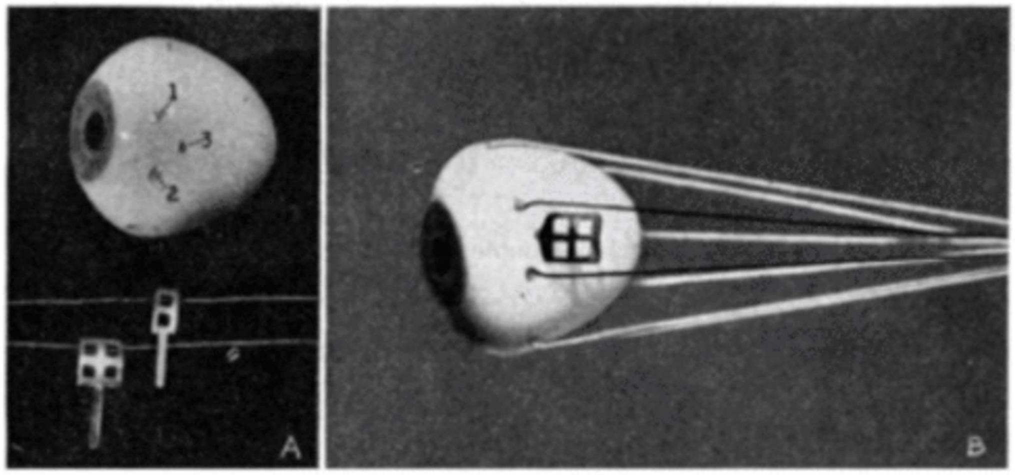


Figure 2.7 Eye implant structure proposed by Ruedemann, Accessed at: [10]

The aim of this thesis project has been to take notes from previously designed animate vision systems to build and design more innovative solutions. First, in order to be able to mimic the human visual organism, there is a need to understand the making components of this organism. Alas, there are two categories of study, the physical components of this organism, and the visual functioning of this organism. Physical aspects entail an understanding of the eye movements, muscle/ physicals components of relation to these movements. Visual aspects entail an understanding of topics such as saccades, foveation and vergence. In relation to physical components, it is decided for the time being to design a much simpler apparatus which is referred to in many of studies which will follow. This is due to the fact that human visual muscle morphology is sophisticated in nature and an attempt to replicate something of this caliber would take a much longer time to complete which is beyond the scope of this project. In relation to the visual aspects, two main characteristics are implemented, these are saccadic foveation and vergence logic.

2.1.2 Previous Attempts at Design of Similar Anthropomorphic Active Vision Systems

This section will cover some of the efforts put forward to mimic and replicate animate visual systems. One of the first champions of the anthropomorphic active visual systems design is Dana Ballard. He conducted majority of his work regarding anthropomorphic vision at University of Rochester in 1980s. Here some of the philosophies and design efforts put forward by him is covered:

One of the ideas covered by Ballard which is of importance to anthropomorphic vision system design is the concept of “Object Centric Vision” (Figure 2.8). One might ask, what is the benefit of an anthropomorphic active vision system when a fixed vision system could achieve the same area of coverage as an active vision system, with less complexity and less susceptibility to cumulative errors? There needs to be an answer to this question other than the simple statement of “For the sake of biomimicry”. Ballard has answered this question with great insight. The answer relays in the reduction of computational costs associated with switching from a camera centric frame to an object centric frame. The fixed stereo camera has a camera centric frame, but an anthropomorphic active camera system has an object centric frame. This has to do with the fact that such a camera system could foveate on a salient target of interest (which is generally an object) and then keep that target in the center of foveation using smooth pursuit and other eye movements explained by Dodge. This makes such a camera system object centric in nature.

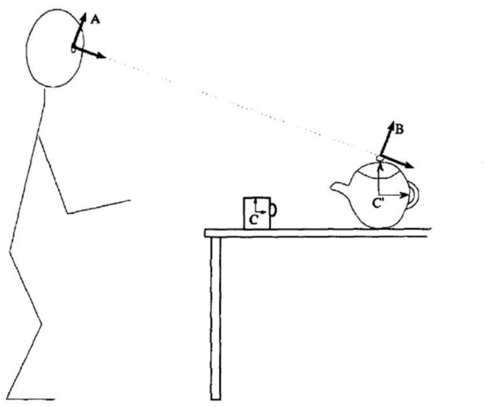


Figure 2.8 Device Centric vs. Object Centric reference systems, Accessed at: [11]

An object centric vision system reduces the computational cost associated with the visual field, since the high value pixels in such a system resides in the foveal center of the capture sensor, much like human visual system. Ballard coined the term “Animate Vision” which refers to animate visual systems which have anthropomorphic features such as binocularity, foveas and high-speed gaze control [12]. This terminology is used in various parts of this thesis paper. Ballard made attempts at design and build of anthropomorphic active vision systems at the University of Rochester in 1980s (Figure 2.9). Below is an example of one of these attempts.

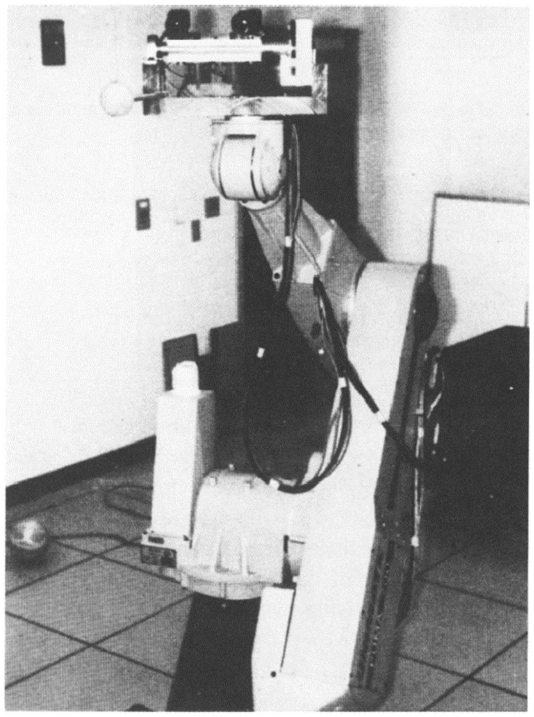


Figure 2.9 Vision system designed by Ballard, Accessed at: [12]

There are two projects coming from Rodney Brook’s lab at MIT in 1990s which are of interest in the area of animate vision: First “COG” as it relates to Brian Scassellati and second “Kismet” championed by Cynthia Breazeal.

Brian Scassellati was a student of Rodney Brook during his Ph.D. studies in the late 1990s. His dissertation was regarding “Foundations for a Theory of Mind for a Humanoid Robot” and as such he worked on various anthropomorphic vision projects [13], [14]. One of the main contributions he made has been his part in the design of visual system used in the COG humanoid robot. COG is an upper torso human robot [14] designed at MIT Artificial Intelligence Laboratory. 3D rendering of the vision system incorporated in this robot is shown below (Figure 2.10). This system has four cameras. Each eye is rendered through two cameras. One camera acts as the complete field of view camera, and the second camera, with a longer focal length lens, acts as the foveal area of the human visual organism.

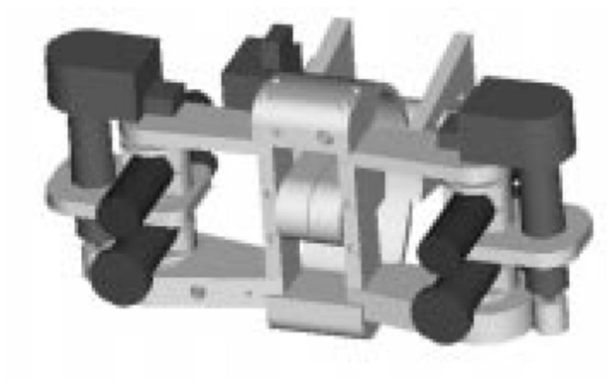


Figure 2.10 3D rendering of COG visual system design, Accessed at: [14]

One of the aspects of this system which has inspired the proposed design is the dimensions used in this system (Figure 2.11). Below are these dimensions:

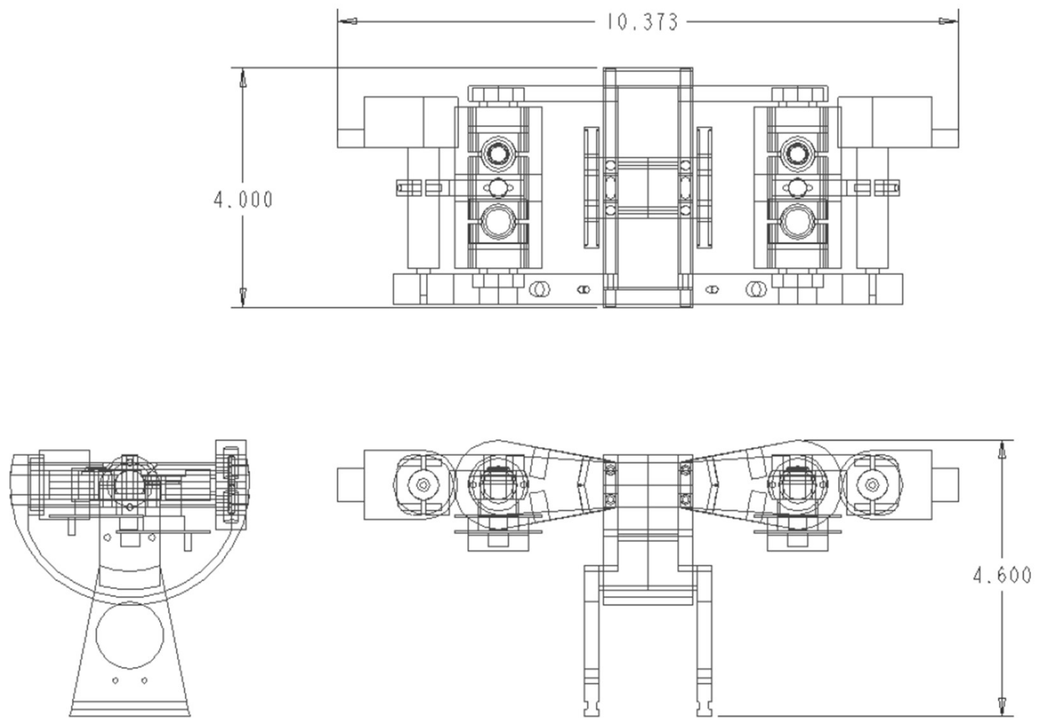


Figure 2.11 Orthographic drawings for the desktop apparatus above, Accessed at: [14]

Note that both cameras are tilted together (Tilt is fixed on the main shaft) since they sit on the main shaft, designed for this task. This is one of the steps taken by the likes of Scassellati for the simplifications of computations down the line. In order for this to work optimally, there is a need for precision in the fabrication of the shaft and landing pads, so the two cameras align perfectly in regard to their static angular tilt angel and vertical baseline displacement. This methodology was used in the proposed design as well. Some inaccuracies were faced, due to fabrication shortcomings, which are detailed later on.

Scassellati's work on COG (Figure 2.13) has led to natural progression which resulted in systems such as Kismet. Shortcomings of COG were tried to be addressed through the following systems designed at MIT Artificial Intelligence Laboratory. One of the issues which Scassellati points to it the fact that at the time, coordinate system conversions between wide angled cameras and foveal cameras for each eye was not known and hard to compute. Kismet tries to address this issue by relocation of the wide angled cameras to a fixed coordinate frame located where the nose is supposed to be.

Interestingly, Scassellati makes use of machine learning for estimation of saccade function [14]. Due to lens distortions and visual system uncertainties, motor commands would not linearly correspond to pixel value distances and as such, there is a need for a workaround (Figure 2.12). Scassellati uses online unsupervised learning to incrementally determine the saccade map.

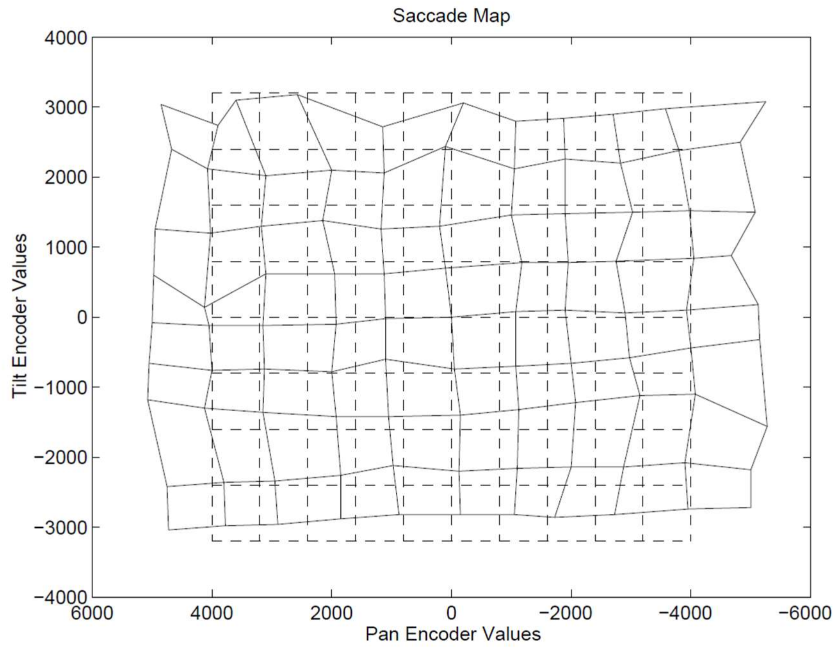


Figure 2.12 Saccade map after 0 (dashed line) and 2000 (solid line) learning trials, Accessed at:[14]

One of the topics highlighted in Scassellati's efforts is the use of parallel network architecture in their system design. At the time they made use of TIM-40 architecture which allows distributed computation and use of comports.

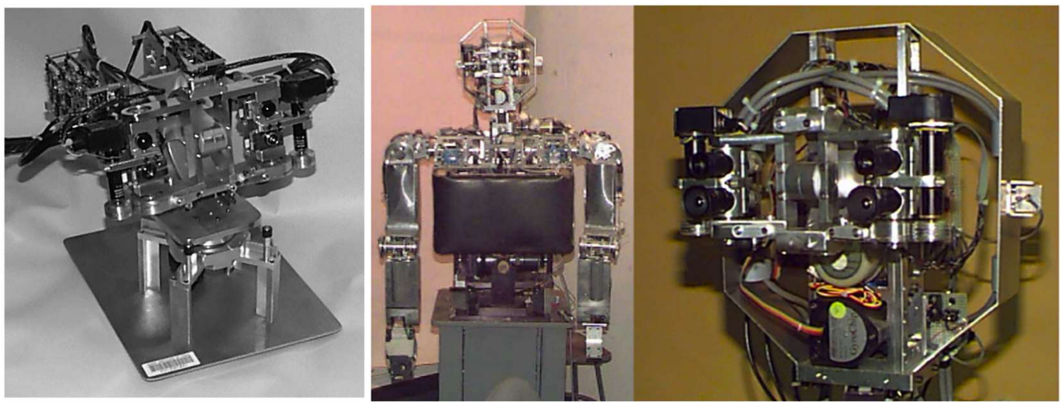


Figure 2.13 (Left) Tabletop vision system used in COG, (Centre) COG Robot, (Right) Closeup of COG's visual components, Accessed at: [14]

Scassellati adopted algorithms for face detection which could be used in the COG system's operations (Figure 2.14), in the pre Viola Jones time frame [13]. He mentions considerations for adoption, which to this day are of importance for design of similar machine vision algorithms [13], [15]: Algorithm's relative simplicity for real-time performance considerations, Algorithms acceptable performance in complex/ social visual settings and Algorithm's biological plausibility.

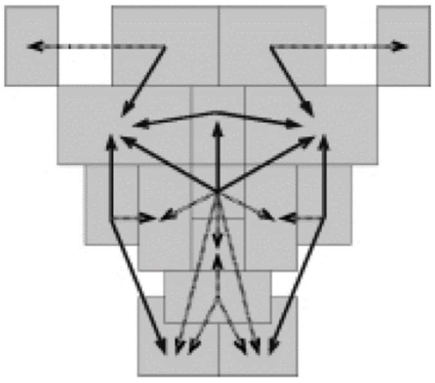


Figure 2.14 Scassellati's adopted face detection mask for COG operations (Pre Viola Jones), Accessed at: [13]

Another researcher of note who worked under the supervision of Rodney Brooks is Cynthia Breazeal. She was supervised for her Ph.D. degree by Brooks in the late 1990s. She developed a robotic system called Kismet (Figure 2.15). Kismet is a socially interactive robotic system which shares some visual systems functionalities with COG's visual system.

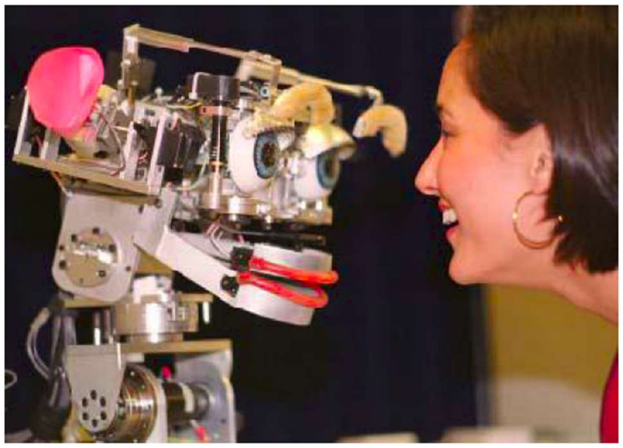


Figure 2.15 Kismet seen interacting with its creator, Accessed at:[16]

Even though this system not unlike COG has 4 cameras, the placing of these cameras is different in this system comparing to the placing of the cameras in COG. This could be seen in the Figure 2.16. In Kismet, the wide-angle cameras are set on center vertical axis independent of motor commands (fixed) which then allows for easier stereo calibration/ registration/ reconstruction of the scene and the foveal narrow field of view cameras are located at the center of the animate eye positions. Breazeal discusses the functionality of Kismet in a series of published papers [16]–[20].

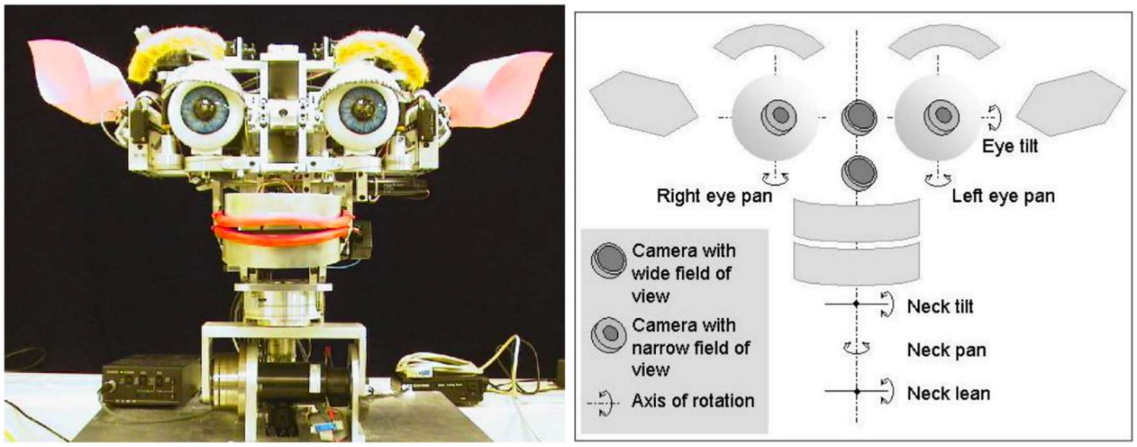


Figure 2.16 (Left) Kismet, (Right) Kismet's camera placing and visual component degrees of freedom, Accessed at: [18]

As mentioned previously, Kismet is a social robotic platform. This means that Kismet is designed so that it interacts with people in a social manner. This social nature of Kismet allows this robot to interact with people in an “adult-infant” context [16]. Kismet accomplishes this by gaze [17]. Gaze direction gives this robot a human-like demeanor. Eye contact especially, attracts full engagement of the human interacting with the robot. Kismet’s visual behavior is intricate in nature with 4 conceptual levels [18]: Social level, behavior level, skills level and primitives level. This level-oriented structure has led to complex interaction capabilities of Kismet. A point of interest is the fact that kismet makes use of color thresholding for visual feature detection. A similar approach has been used in this thesis project as well.

A more recent project of interest is the ICUB robot (Figure 2.17). This robot is developed by the team at the Italian Institute of Technology (IIT) [21]. This robot is a prime example of integration of anthropomorphic active vision systems into a bigger wholesome system design (A fully operating humanoid robot). This humanoid robot has been in making since 2005. This robot has a versatile Biomorphic vision system. It is interesting to note that the structure for visual motor system is very similar to COG and Kismet in nature. There is a single motor which controls the tilt of both cameras and individual motors which control pan of each camera.

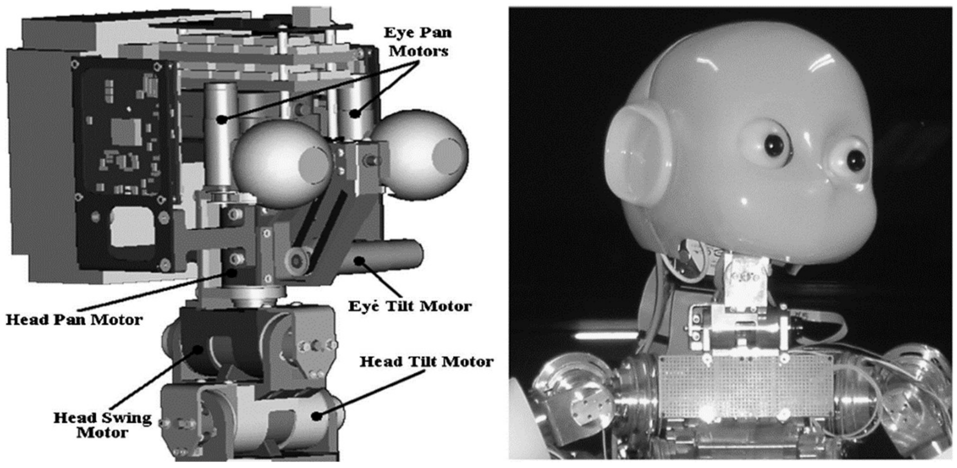


Figure 2.17 (Left) 3D rendering of ICUB's head model, (Right) ICUB's upper section, Accessed at: [21]

Vision research for ICUB has been ongoing and as such, there has been different implementations of visual system to this date. In one implementation blob detection is used, not unlike one of the implementations in this project [22] (Similarity on concept, implementation here is much more sophisticated).

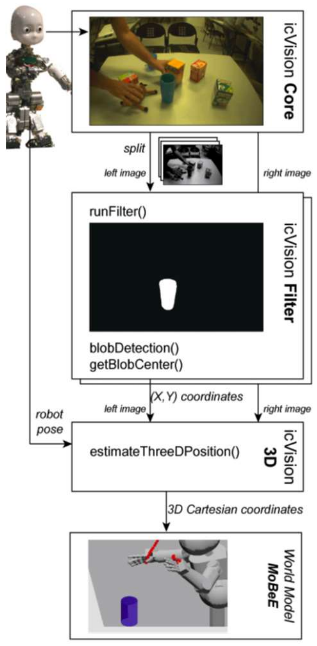


Figure 2.18 icVision's system diagram, Accessed at: [22]

In newer Implementations of vision for ICUB, Dynamic Vision Sensors (DVS) are used [23]. These vision sensors are unique in the sense that they process data relating to variation and discontinuities in the visual field while discarding redundancies. This is mimicked after primitive nervous systems and is beneficial in system design due to the optimization it delivers. Majority of the static visual data does not need to be processed by incorporation of these sensors, making way for faster process of vital visual info (e.g. Movements in visual fields, living subjects, etc.).

Efforts are made to create visual datasets using the ICUB's visual system [24]. This dataset is called "iCubWorld" which is an object recognition dataset. This dataset is unique in the sense that CNNs trained using such a dataset would be prime candidates for incorporation into humanoid robotic/ anthropomorphic vision systems. This is due to the fact that this dataset is recorded using an apparatus of same characteristics which would result in better mimic of controls of such systems.

Another project of interest is the anthropomorphic head designed by KIM et al. at the University of California San Diego (Figure 2.19). This project is unique in the sense that unlike previous projects it makes use of off-the-shelve components to create the intended apparatus. Here analog servo motors are used in the creation of the apparatus. Of-the-shelve components use through this project has been an influence in the present thesis project.

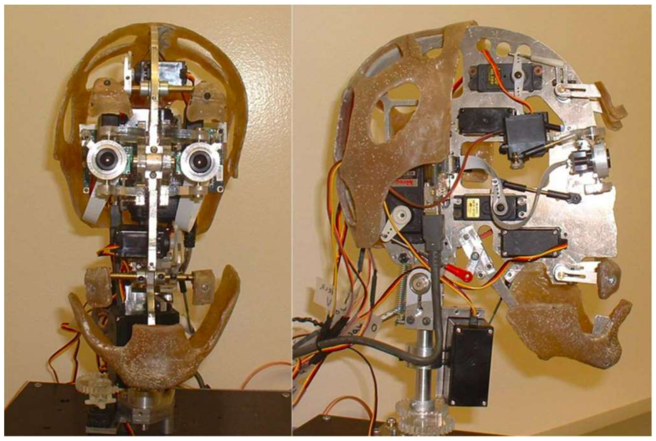


Figure 2.19 Anthropomorphic robot head designed at UCSD, Accessed at:[25]

Some of the more interesting recent research in this field comes from the Technical University of Munich [26]. In this research project, Villgrattner explores the recreation of a motor system for a single camera which is more similar to the muscle structure of the eyes (Figure 2.20). This is a move in the right direction comparing to previous attempts which make use of pan and tilt motors instead which are not biomimetically accurate.

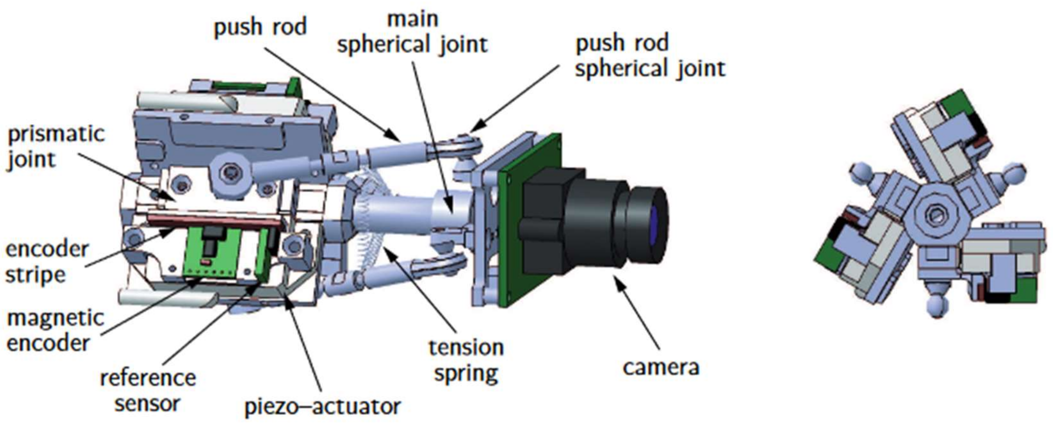


Figure 2.20 front (Right) and profile (Left) view of the system designed at TUM, Accessed at: [26]

2.2 Computer Vision Matters

2.2.1 Review of Saliency

Saliency prediction is a complex problem of importance, especially when dealing with animate vision systems. It has real practical applications such as predicting where someone might look in a picture or to decide where a machine vision system should focus on. For this matter, different deep convolutional neural networks, in addition to different datasets available for saliency detection were explored. Focus was on usability, functionality and (primarily) familiarizing with procedures associated with usage of neural networks for determination of saliency maps in images and gaze detection.

This section explores the concepts and practices used to determine human gaze using convolutional neural networks. Saliency has been an area of interest for quite a long time now and there has been an effort made within the computer vision community to design methods to determine saliency, post and prior to the rise of neural networks in this field. As such, there have been numerous models created for saliency determination throughout the past couple of decades. These include handcrafted and convolutional neural networks (CNN) based models. For ease of understanding, saliency determination could be divided into two main categories: Semantic segmentation and gaze detection. For the purpose of this study, the focus will be primarily on the approaches related to gaze detection. Since gaze acts as an agent of visual attention which then determines the area of interest and focus, gaze detection is of key importance. This has many applications such as determination of areas of interest in webpages and published materials, possible focusing areas for cameras, or simply as an extension to object detection as a concept. For the purpose of this study, the possible applications of saliency in the area of stereo vision, and its uses for stereo calibration and depth perception are of foremost interest.

Semantic segmentation deals with segmentation of the image into salient / non-salient regions (Figure 2.21). On the other hand, gaze detection deals with prediction of point of attention in an image, based on where the points of attention have been in previous images. In order to achieve this goal, generally there is a need for an algorithm and annotated dataset which helps to train the algorithm. In such instance, the algorithm falls under the machine learning umbrella.

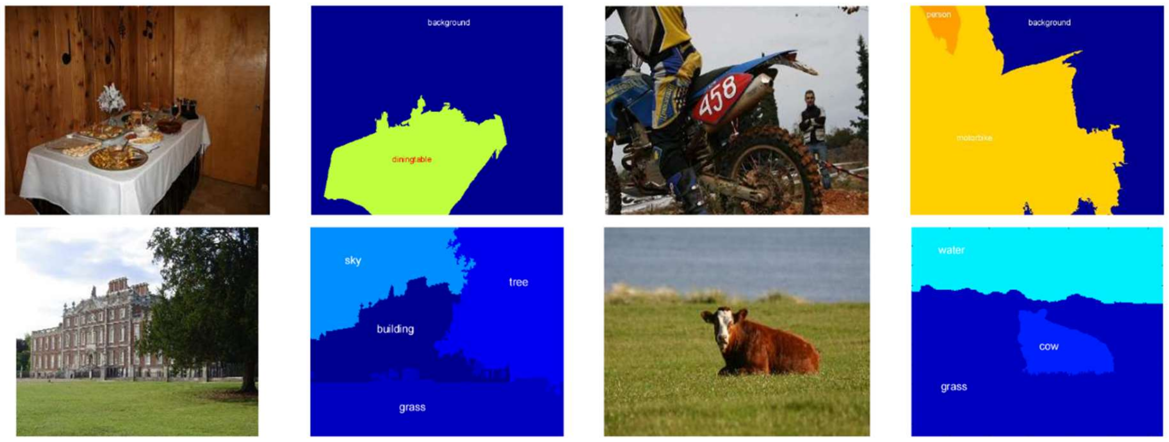


Figure 2.21 Example of semantic segmentation results, Accessed at: [27]

It should be noted that the annotated dataset used for the learning (in the context of saliency), is just as important as the algorithms used for saliency prediction. To date, there have been two main approaches for the population of these datasets. One approach is the use of special glasses for the purpose of tracking eye movement while looking at images. In this approach, the observer is shown an image for a specific amount of time (e.g. 20 seconds) and the eye movement is recorded during this period, marking the points of attention on the shown image. This method is costly to implement, and time consuming to process. One of the major standardized datasets used in the saliency field (Pascal with the gaze annotations) was compiled using this method. Due to the cumbersome nature of this approach, datasets produced tend to be limited in the number of observations included (Pascal contains roughly 300 images). The second method uses a common mouse tracker to record gaze. This method is discussed in detail in later sections.

2.2.1.1 Early Works in Saliency

Studies in saliency have a strong connection to vision research in experimental psychology. One of the pioneers in this field is Alfred Lukyanovich Yarbus [28]. His studies in the field of eye movement have been a source of inspiration in vision research for decades. Some of the most interesting research done by Yarbus related to saliency. His eye movement path studies (Figure 2.22) are discussed at length in the last Chapter of his book "Eye Movements and Vision"[29]. In this work he records the movement path of subject's eyes, using a mechanically designed apparatus while looking at the painting "Unexpected Visitor". This study produced very valuable observations:

First, he concludes that different observer's points of attention tend to be different while observing the same scene. This is a major point of contention in approaches taken today for saliency detection. The majority of approaches don't fully incorporate variations in observer's point of view in their model. Bruce notes this discrepancy in his work, as well [30].

Second, he concludes that there are different stages of gaze focus. In this sense, the nature of gaze is hierarchical. In the first 25 milliseconds of focus on an image, there tends to be primary/ low levels of gaze focus which is conducted. In this stage, there tends to be an immediate gaze attention on faces. But then, in the remaining time of focus on an image, the gaze seems to be refined and gaze points seem to shift. Again, this is something which many current models are unable to register as they are only designed to detect one stage of gaze. Some models accommodate for this by discarding the initial gaze data. This is discussed by Judd in "Learning to Predict Where Humans Look" [31] when discussing her approach for data set preparation. This is not a full solution to this matter since there is ultimately detection of saliency for only one stage of attention.

Third, he concludes that there seem to be revisits to attention points mainly in the areas of eyes and mouth by the observers, in the later stages of focus. This seems to help subjects with cognitive understanding.

The most influential part of Yarbus's work could be argued to be his eye movement tracking diagrams. The diagrams, as explained previously, show the eye movement/ gaze patterns in subjects.

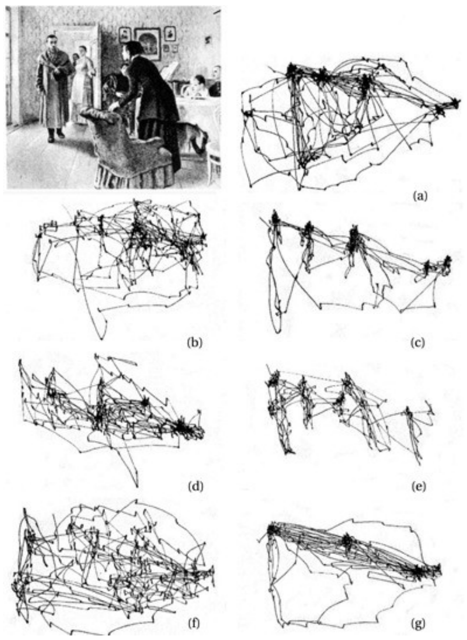


Figure 2.22 Unexpected Visitors, and gaze patterns for the same observer looking at the painting with different instructions, Accessed at: [28]

2.2.1.2 Cognitive Saliency

As explained at the beginning of this section, the focus of the proposed system ultimately is gaze detection (Figure 2.23). The final goal is to design an algorithm which could predict areas of human gaze in a video feed. First, it should be noted that as mentioned by Bruce [30], gaze is subjective in nature and no two persons would look at the exact same points in an image. One solution for this matter could be the use of various observers for eye fixation dataset preparation. Salicon executes this methodology, however this by itself though does not solve the issue.

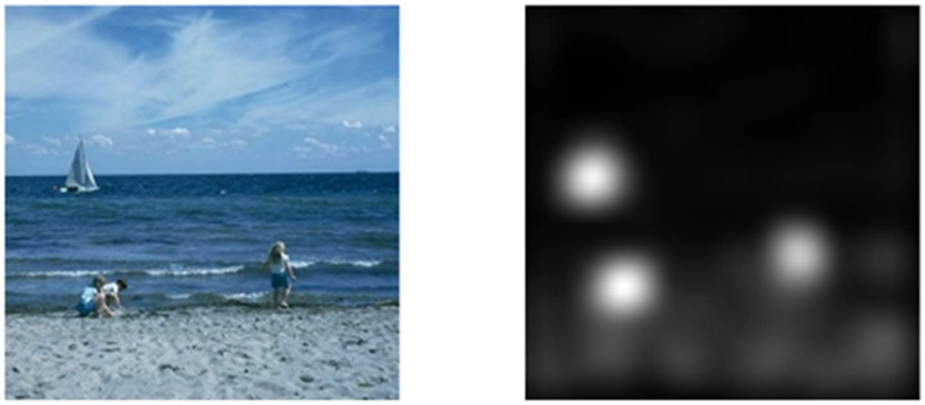


Figure 2.23 Example of cognitive saliency prediction results. Image (Left) and saliency map for it (Right), Accessed at: [32]

Saliency detection can be categorized into two categories: Static and dynamic detection. These differ in that observers don't observe static images as they do observe dynamic images [33]. The duration of viewing differs, and dynamic cases have motion associated with them as a defining factor. For the purpose of this paper, the focus is on the exploration of cognitive saliency analysis in static images.

As mentioned previously, methods designed for saliency prediction fall into two categories: Hand-crafted and CNN based. Borji in his paper [33] lists some of the most influential handcrafted and non-hand-crafted methods. Some of the hand-crafted methods of influence are: AIM, GBVS, SUN, SR, AWS, BMS, JUDD, OBDL, AWS-D, PQFT, and RUDY. JUDD is likely the most influential of these models. Shortly after JUDD, CNN based models for saliency detection began emerging. The most prominent advantage of CNN in terms of implementation is transfer learning: Networks which were initially trained for object detection can now be used for gaze detection. Many of the CNN saliency architectures have VGG, RESNET or Alex Net as their main building blocks. All three of these networks were trained for object detection. These networks are used in saliency detection with their trained weights. The reason for this is twofold: First, training a network of such magnitude takes a large amount of data, which would be more than what is available even through Salicon. Second, the low/ mid-level features detected both in the context of object detection and saliency are the same in nature.

Following, is some of the more robust deep models (CNN models) as of 2018: On top of the charts (AUC-J) is Deep Gaze II and EML-NET. Deep Gaze uses Alex Net as its base. One of the newer models introduced (2018) is SAM. In this study, this model was chosen as the base for at hand experimentations. This was done due to several reasons including: Novelty of this methodology, and consistent implementation of the code and clear coding style. Further, the author was available for questions and consultations. And finally, due to the fact that the methodology was designed to work with the SALICON dataset.

2.2.1.3 Influential Works in Cognitive Saliency

It is worth noting some of the more influential works in this field. In "Learning to Predict Where Humans Look" [31] Judd sets forth a protocol for the creation of annotated saliency datasets. This includes procedures for gathering of images, viewing sessions, calibrations and so forth. This methodology is used as inspiration for the creation of the datasets such as Pascal-s and SALICON later on. It is due to these datasets that major progress in the field of saliency has been made.

Additionally, Judd was the first to make use of training procedures and then testing, as used in the machine learning discipline. Training on the data in her proposed system is done on the low-level, mid-level and high-level predefined features. This is a methodology which in some ways resembles learning for image-based convolutional neural networks.

It is interesting that the method designed by Judd still performs relatively well compared to simpler CNN based saliency models (Figure 2.24).

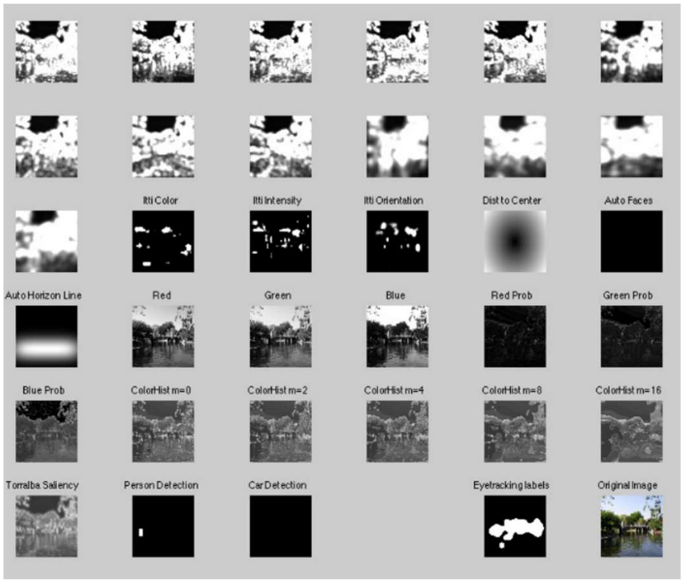


Figure 2.24 Schematic representation of the learning windows Judd uses for training of an example image, Accessed at: [31]

The other paper of note in this field is “Saliency Based on Information Maximization”[34]. In this paper, Bruce suggests a framework for a bottom-up approach based upon self-information of the local image content [34]. In the model proposed by Bruce (Figure 2.25), the low-level patches selected for algorithm processing purposes are ones similar to what is present in low level visual learning in humans. These patches are selected from a pool of predefined patches using Infomax ICA and then these patches are used in a convolutional neural network setting. The end goal of the CNN in this context is to maximize the self-information. Bruce makes a clear distinction between entropy and self-information in this paper.

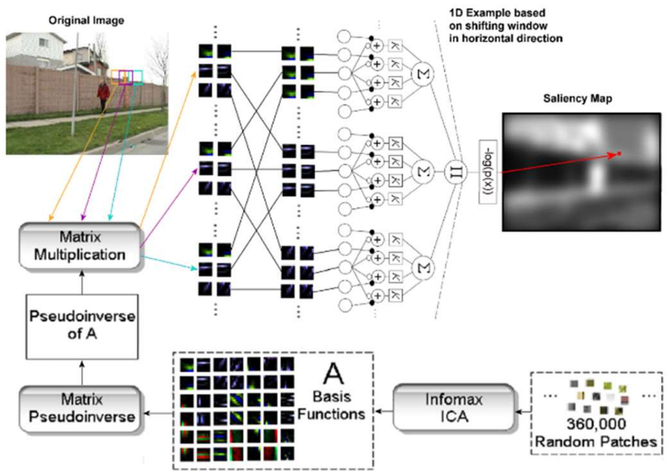


Figure 2.25 Schematic model of the methodology proposed by Bruce, Accessed at: [34]

2.2.1.4 An Introduction to Deep Neural Networks

As mentioned previously, all the top performing saliency models are deep models. This makes it important for us to have a familiarity with deep models, their components and their functionality. Following will be a brief review of these models and their architecture. Since this project makes use of the VGG16 later in the study, this model will be used for explanation purposes.

VGG16 is a deep convolutional model designed by Oxford’s Geometry Group in 2014 (Figure 2.26). This model was originally designed for image localization and image classification. VGG uses convolution operations for learning of operational windows in low-level, medium-level and high-level stages. With the use of layers of convolution and pooling, the output of the network is eventually reduced to a 1 by 1000 array. This array has the output form of 1000 class classification which VGG16 was designed for.

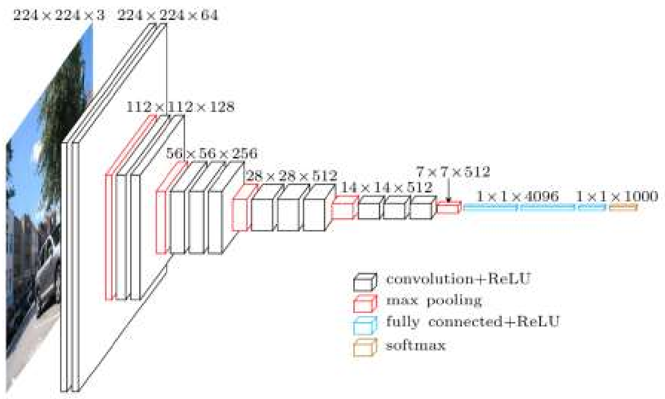


Figure 2.26 Architecture of VGG16, Accessed at: [35]

In the case of saliency, the last few components of VGG16 are dropped since what is need is a saliency model and not a classification model (Figure 2.27).

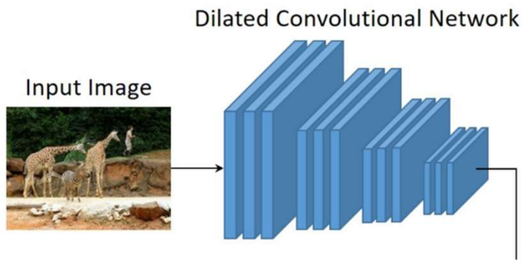


Figure 2.27 Representation of VGG16 modified (truncated) model used in SAM-VGG, Accessed at: [36]

2.2.1.5 Models Explored For this Study

A variety of deep models were explored for the purpose of adoption in this study. This included Deep Gaze, SalNet and SAM. The goal was to find Keras implementations of these models, which would then be easier for a novice learner to grasp on and analyze. It was possible to find Keras implementations for SalNet but not Deep Gaze. The issue with this implementation was its vague definition of the dataset used in the code and lack of comments in the code. Code for Deep Gaze was fully based on Tensorflow and complicated in nature. Code implemented for SAM has components of Keras, is clearly written and has cohesive comments and has an active support forum on GitHub. Additionally, the main designer of the algorithm (Cornia) was very responsive to emails with questions in regard to the code. For the reasons listed above, SAM was initially chosen for experimentations in this project.

2.2.1.6 Backgrounder on KERAS and THEANO

There are various libraries in the python environment which are used for the purposes of building neural networks. Tensorflow, Torch, Theano, Keras, to name a few. Many of the newer saliency models tend to adopt Theano as their main work frame. Theano was developed at the University of Montreal at Montreal Institute for Learning Algorithms. Theano utilizes Numpy. One of the reasons for this is the fact that Theano is more mature compared to Tensorflow. Many researchers in the field of machine learning have adopted Theano from the early days. The trend though is migrating to TensorFlow due to the creation of TPUs which are specifically designed for Tensorflow processing.

Keras works as an intermediary to platforms such as Tensorflow and Theano to simplify the process of creating neural networks. This is especially helpful for people new to Neural Networks and Deep Learning.

2.2.1.7 SAM Architecture

SAM (Saliency Attentive Model) [36], designed by Cornia et al is an LSTM saliency prediction model [36]. This model has two implementations: One using Res Net as the base and the other using VGG16 as the base. For these experimentations, the implementation with VGG16 was used as the base due to it being less sophisticated compared to the Res Net implementation, and thus less computationally demanding.

One of the novelties associated with this model is the use of LSTM for the improvement of gaze prediction. Even though LSTM is majorly used in the time domain [36] SAM modifies this technique so it could be used iteratively to refine the results of saliency prediction.

LSTM stands for Long Short-Term Memory. This method is covered in detail in “Deep Learning”[37] by Goodfellow. This method is a more advanced form of recurrent neural networks. In a Recurrent Neural Network (RNN), data from a node is passed through that node again in the next run (Figure 2.28). This could then help with the use of previous data in future calculations.

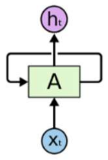


Figure 2.28 Schematics for a simple recurrent neural network, Accessed at: [38]

LSTM builds upon the idea of recurrent neural networks. In LSTM unlike RNN there is only focus on the short term past info. This is the case since only the most recent info in the neural network processes would help with the refinement of the results. In order to achieve such a task, recurrence happens in a more sophisticated manner with more complex mathematical calculations. In SAM (Figure 2.29) results from VGG16 calculations are fed to LSTM sub-model (Figure 2.30) to update the stream H_i.

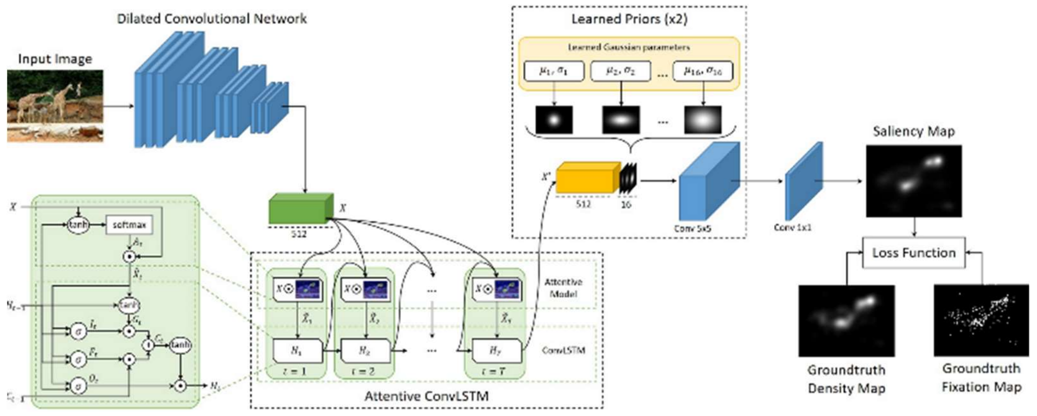


Figure 2.29 Complete SAM model schematics, Accessed at: [36]

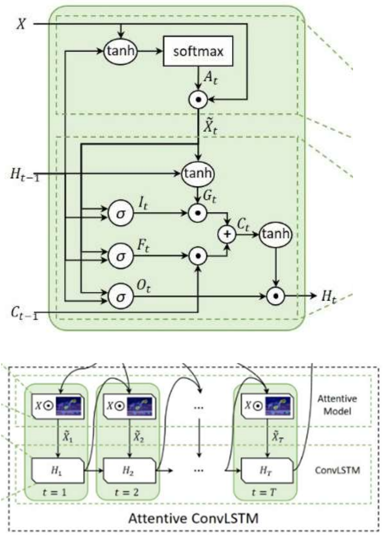


Figure 2.30 Schematics of LSTM design used in SAM. The internals for updates for each H_i (Up) and LSTM model as a whole (Down), Accessed at: [36]

Learned Gaussian parameters are the other form of classical computer vision techniques incorporated in this model (Figure 2.31). SAM has procedures in place to learn 16 Gaussian parameters on 2 iterations. These priors are then applied to the results retrieved from LSTM operations to add centre bias to the saliency predictions. This centre bias is much needed in saliency detection since human gaze tends to focus on the middle of an image. Previously, only one or a few Gaussian predefined parameters were applied in similar models. As such, one of the advantages of SAM is the fact that instead of using predefined Gaussian parameters for adding of a centre weight, it uses a learning process in order to drive Gaussian priors which could then be applied for addition of centre bias.

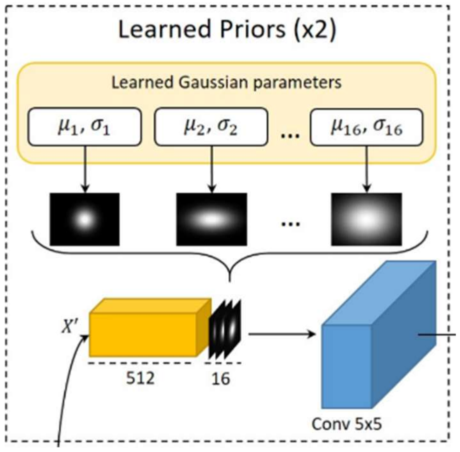
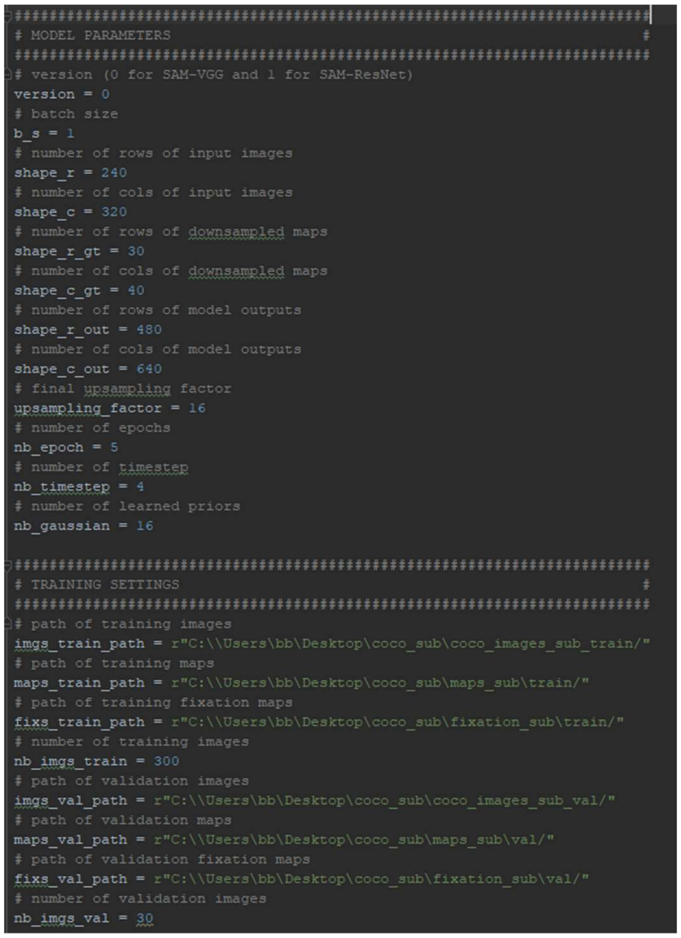


Figure 2.31 Schematics for learned priors in SAM, Accessed at: [36]

KL-Divergence is the loss parameter used in these experiments. Kullback-Leibler divergence is a measure of information loss when choosing an approximation for a distribution [39]. In a nutshell, what KL-Divergence gives us is the entropy of the initial distribution minus entropy of the approximation. This sums up to be the information loss between two epochs in this case. It should be noted that the aim is for a Kl-Divergence closest to zero (minimal information loss). Here, p is the initial distribution and q is the approximated distribution.

$$D_{KL}(p\|q) = \sum_{i=1}^{N} p(x_i) \cdot (\log p(x_i) - \log q(x_i)) \qquad (2.1)$$

SAM-Net comes in a package of multiple python files. This includes: “main.py”, “models.py”, “config.py”, “heatmap.py”, “attentive_convlstm.py”, “gaussian_prior.py”, “utilities.py”, “dnc_resnet.py”, “dnc.vgg.py”. “main.py” is the core of this code. “dcn_vgg.py” is what holds the VGG16 parts of the code. “models.py” is what holds the entirety of this saliency model. “config.py” is what contains directory paths and model parameters included in the code such as epoch numbers and batch size (Figure 2.32). The first section of this file contains model parameters and the second section contains directories.



```
#########################################################################
# MODEL PARAMETERS                                                      #
#########################################################################
# version (0 for SAM-VGG and 1 for SAM-ResNet)
version = 0
# batch size
b_s = 1
# number of rows of input images
shape_r = 240
# number of cols of input images
shape_c = 320
# number of rows of downsampled maps
shape_r_gt = 30
# number of cols of downsampled maps
shape_c_gt = 40
# number of rows of model outputs
shape_r_out = 480
# number of cols of model outputs
shape_c_out = 640
# final upsampling factor
upsampling_factor = 16
# number of epochs
nb_epoch = 5
# number of timestep
nb_timestep = 4
# number of learned priors
nb_gaussian = 16

#########################################################################
# TRAINING SETTINGS                                                     #
#########################################################################
# path of training images
imgs_train_path = r"C:\\Users\bb\Desktop\coco_sub\coco_images_sub_train/"
# path of training maps
maps_train_path = r"C:\\Users\bb\Desktop\coco_sub\maps_sub\train/"
# path of training fixation maps
fixs_train_path = r"C:\\Users\bb\Desktop\coco_sub\fixation_sub\train/"
# number of training images
nb_imgs_train = 300
# path of validation images
imgs_val_path = r"C:\\Users\bb\Desktop\coco_sub\coco_images_sub_val/"
# path of validation maps
maps_val_path = r"C:\\Users\bb\Desktop\coco_sub\maps_sub\val/"
# path of validation fixation maps
fixs_val_path = r"C:\\Users\bb\Desktop\coco_sub\fixation_sub\val/"
# number of validation images
nb_imgs_val = 30
```

Figure 2.32 Content of “config.py”, the model parameters are changes for each experimental run, as needed

This model is designed so it could be called (trained and tested) from bash, or, in this case ANACONDA command prompt since Windows operating system was used in these initial tests. For training of the model, the command “python main.py train” is executed from the command prompt. For testing of the model, the following command is used: “python main.py test” from the command prompt. It should be noted that in the original code, location of the test files would be called with the test command in bash but due to complications with Windows, these locations were hardcoded into the python code (in "main.py").

2.2.1.8 Deep Learning Challenges and Solutions

As would later be explained in Chapter 3, ultimately it is decided to utilize a different CNN for the proposed system's functionality named "DeepLab". This CNN is designed for semantic segmentation. Even though semantic segmentation is different in terms of the final outcome from saliency map determination, it is of a relevant nature to saliency determination in the sense that semantic segmentation attempts to define the bearings of the visual scene and to determine the various components of the visual scene. As such this type of CNN could be used as an alternative to a saliency CNN, for the least amount of architectural variance in CNN design.

One of the main benefits of the DeepLabv3 (Figure 2.33) is that it is built on Tensorflow Architecture. Tensorflow products are updated continually, which are optimized for Nvidia GPU integration. This is a major departure from Likes of SAM-VGG which is built on a Theano backend and which is very slow on forward passes. Deeplabv3 specifically has implementations designed for mobile devices. These implementations have been utilized in this thesis project.

There are published pre-trained weights available for DeepLabv3, one of which is used in the proposed design implementation. Below a schematic view of DeepLabv3 architecture could be seen:

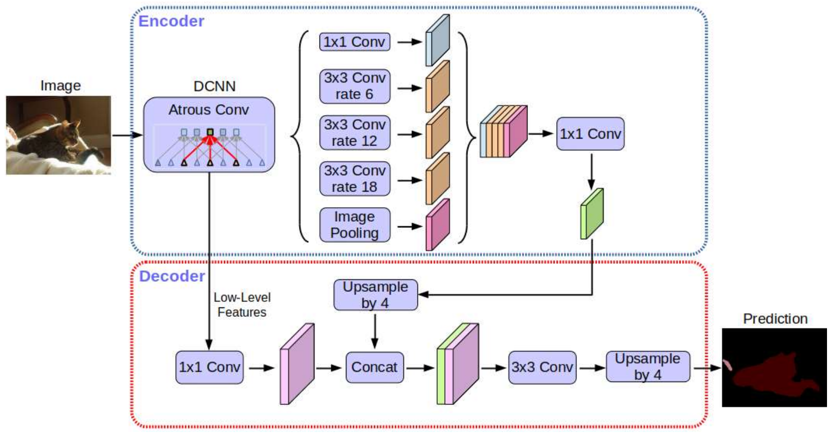


Figure 2.33 DeepLab-v3+ model architecture, Accessed at: [40]

As could be seen DeepLabv3 makes use of Atrous (Trous meaning hole in French) Convolution (Figure 2.34). In this type of convolution, gaps (holes) are added when sampling the target pixels.

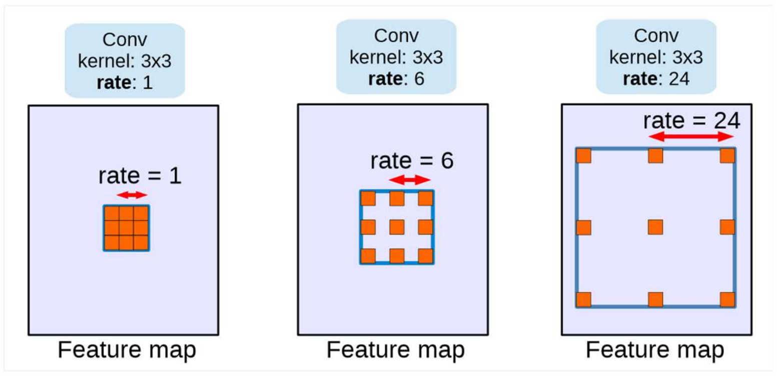


Figure 2.34 Visualization of Atrous Convolution, Accessed at: [41]

This type of convolution could be formulated as below:

$$\boldsymbol{y}[\boldsymbol{i}] = \sum_{\boldsymbol{k}} \boldsymbol{x}[\boldsymbol{i} + r \cdot \boldsymbol{k}]\boldsymbol{w}[\boldsymbol{k}] \qquad (2.2)$$

“r” being the stride rate. This formulation is similar to conventional convolution with the only difference being the addition of “r” factor into formulation.

Atrous convolution is used due to the fact that its use could mitigate the spatial information loss which segmentation depends upon. This can be seen in the schematics below (Figure 2.35).

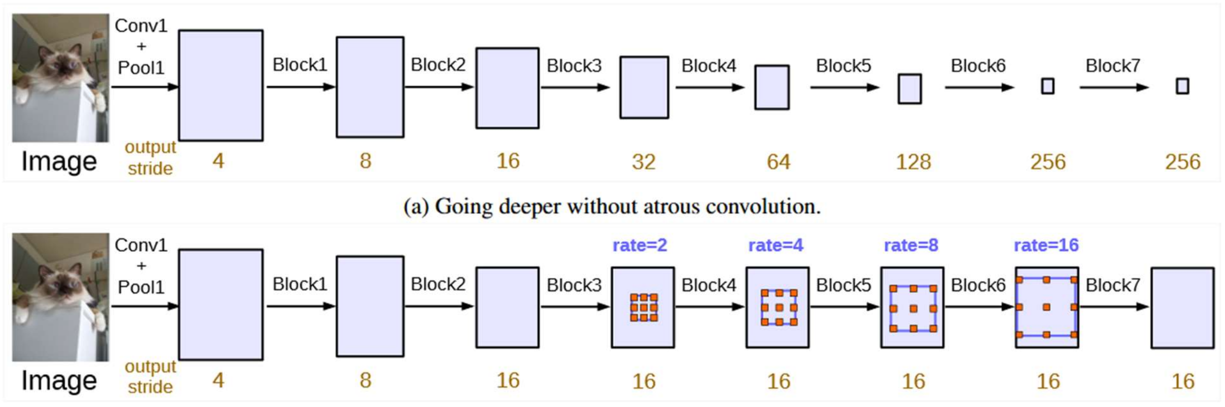


Figure 2.35 (Top) Conventional convolution stride, (Bottom) Atrous convolution stride, Accessed at: [41]

Finally, this model makes use of Atrous Spatial Pyramid Pooling (ASPP) to upkeep the integrity of produced weight values in presence of Atrous convolution.

2.2.2 Review of Viola Jones/ OpenCV

One of the methods used in this study to determine the location of foveation point is Viola Jones. This method has been widely used in computer vision related projects since its creation. Viola Jones is great due to its relatively low computational cost and low false positive detection. Below is a brief review of this method:

Viola Jones is an algorithm proposed by Paul Viola and Michael Jones in 2001 in their seminal paper "Rapid Object Detection Using a Boosted Cascade of Simple Features" [42]. This algorithm is used to this day in many devices which have face detection abilities, due to its low computational costs.

Viola Jones is unique due to various reasons: First, for its uses of ensemble learning in deciding on face-ness of specific blocks of an image. Second, for its use of integral image. And third, for its use of simple features in its classification procedures (Figure 2.36).

Feature (filter) windows combined with convolution operations is used in many of methods relating to computer vision. These windows are convoluted over the target image for identification of areas of interest.

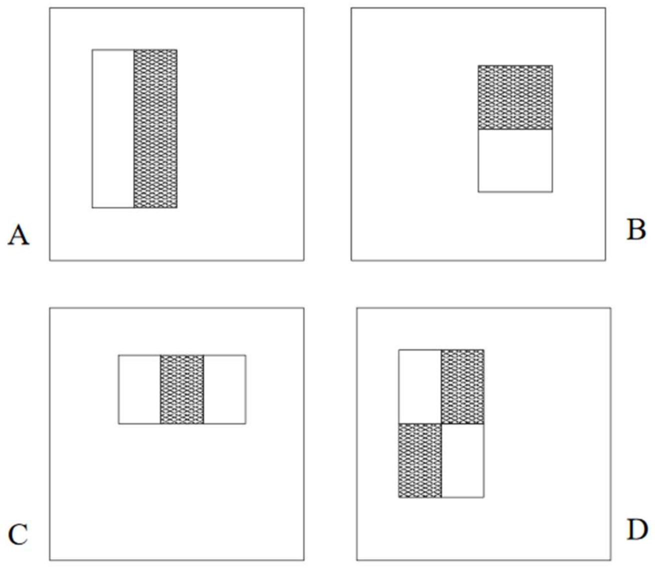


Figure 2.36 Example of features used in Viola Jones, Accessed at: [42]

Simple windows used in Viola Jones method are not effective separately. But when the results of these filter convolutions are combined, the results are of high accuracy. This combination of results is done through ensemble learning.

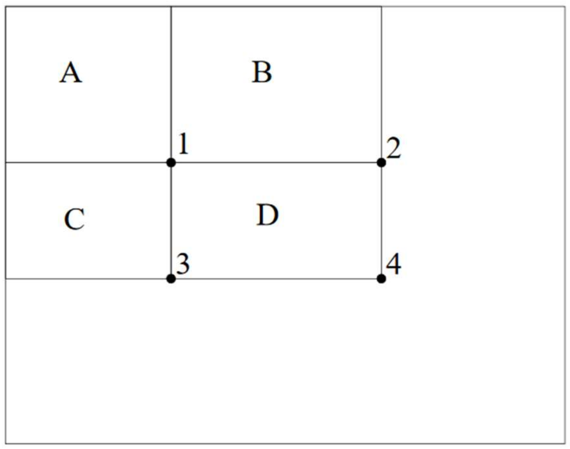


Figure 2.37 Integral Image, Accessed at [42]

In order for Viola Jones to be able to conduct filter convolution calculations with low computational cost, Integral Image is used (Figure 2.37). To explain integral image, refer to the figure above is made. Integral image represents the sum of the pixels up to that pixel location at any given pixel location. As an example, in the image above, the results of integral image at point 1 would entail the sum of all pixel values at region A.

OpenCV Library is used in a big portion of the code needed for the visual functioning of the proposed system. OpenCV is an open source computer vision library which is compatible with C++ and Python. This Library has been around since the early 2000s. In one of the foveation tasks, only OpenCV functions were used for the completion of task. This designed method is later used for benchmarking and experimentation with the proposed system. Performance of the system was measured through these experimentations. Functions used for this purpose are detailed in Chapter 3.

2.2.3 Review of Gaze Vergence and Cross-Correlation

In the past few decades, there has been various attempts in the area of active vision systems for creation of vergence solutions. These methods could be divided into three categories [43]: The first category makes use of feature matching for vergence solutions. This is done using methods such as Zero Disparity Filter. The second category makes use of phase difference for vergence solutions. In this methodology, images are transformed to frequencies domain using Fourier transform which is then used to calculate phase shift for motor commands. The third category makes use of local correlation for vergence solutions (e.g. Cross-correlation). The third category is what is used in this thesis project.

The backbone of the vergence algorithm in this thesis project is cross-correlation. As mentioned previously, various methods have been proposed for the determination of vergence point and ultimately execution of vergence, some of which were reviewed for this thesis project [43]–[50]. Some of these studies make use of disparity and phase difference for vergence but majority of the studies reviewed have a cross-correlation backbone. Cross-correlation methods covered use either cartesian space or log polar space for operation purposes. In the past few years, cross-correlation method variations using cartesian space have had a resurrection [43]. This can likely be attributed, in part, to advancements in processing power which make it possible to run cartesian space cross-correlation, in real time.

Log Polar transform (Figure 2.38) is used to convert images from cartesian space to log polar space. Log polar space is beneficial due to its resemblance of retinal image structure. This then gives a higher weight to features in the center of log polar space comparing to the features in the periphery. In this system, ξ and η are the coordinate in log-polar space, while ρ and θ are the coordinates of the same point in Cartesian space (refer to the diagram below). k is the decay factor.

$$\begin{cases} \xi = \log_k(\rho) \\ \eta = \theta \end{cases} \tag{2.3}$$

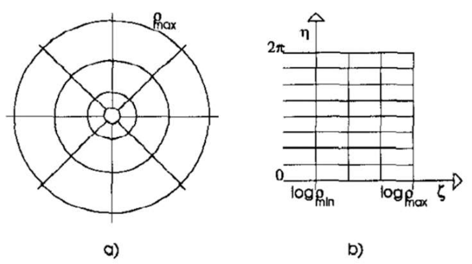


Figure 2.38 Cartesian (a) to log-polar(b) space transformation, Accessed at [47]

As mentioned, there are runtime advantages to use of log polar space while dealing with vergence. This thesis project makes use of cartesian space in order to keep the complexity of processes to a minimum. One of the studies of interest [49] makes use of Hering's law in the modeling of vergence. Hearing states that left and right eye muscles are innately connected such that "conjugacy of movement of our two eyes is basic and innate" [51]. As such left and right eye move somewhat in parallel. In terms of motor commands for a vision system, this would manifest itself as similar motor commands for follower camera as the master camera during the initial stage of foveation followed by the vergence motor commands for ultimate vergence. This methodology would be worth considering for future works.

It would be beneficial to review cross-correlation due to the fact that it makes the backbone of the proposed vergence methodology. Cross-correlation is a classic computer vision process. In this process, a reference patch is correlated against the area of interest in target image, in order to determine peak areas of likelihood (Figure 2.39). Here, the formulation for cross-correlation is briefly reviewed. Use was made from the lectures by Cyrill Stachniss at the University of Bonn:

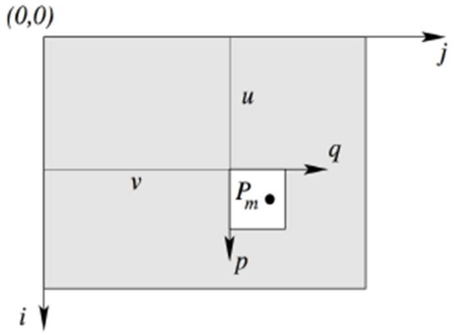


Figure 2.39 Cross-Correlation refrence patch overlaying on the target image

Above statement could be formulated as below, as a geometric transformation (refer to Figure 2.39). $\begin{bmatrix} u \\ v \end{bmatrix}$ is the unknown desired value set in cross-correlation. $\begin{bmatrix} i \\ j \end{bmatrix}_m$ represents target image. $\begin{bmatrix} p \\ q \end{bmatrix}_m$ represents reference patch.

$$T_G\colon \begin{bmatrix} p \\ q \end{bmatrix}_m = \begin{bmatrix} i \\ j \end{bmatrix}_m - \begin{bmatrix} u \\ v \end{bmatrix} \quad m = 1, \dots, M \tag{2.4}$$

When conducting cross-correlation, the aim is to calculate a similarity map for the area of correlation. Different methods could be employed for this purpose (here g represent intensity values for pixels):

$$SSD = \sum\nolimits_m (g_2(m) - g_1(m))^2 \tag{2.5}$$

Which stands for "Sum of Squared Differences"

$$SAD = \sum\nolimits_m |g_2(m) - g_1(m)| \tag{2.6}$$

Which stands for "Sum of Absolute Differences"

$$\text{Max} = \max_m |g_2(m) - g_1(m)| \tag{2.7}$$

Which stands for "Maximum of differences"

The issue with the above similarity measures is that they are not invariant to brightness and contrast changes and as such are not robust.

Due to this reason cross-correlation is use as the similarity measure of choice.

$$[\hat{u}, \hat{v}] = argmax_{u,v} \rho_{12}(u, v) \tag{2.8}$$

In which cross-correlation is defined as:

$$\rho_{12}(u, v) = \frac{\sigma_{g_1 g_2}(u,v)}{\sigma_{g_1}(u,v)\sigma_{g_2}} \tag{2.9}$$

Below, the components of this operation are detailed:

$$\sigma_{g_2}^2 = \frac{1}{M-1}\left[\sum\nolimits_{m=1}^{M} g_2^2(i_m, j_m) - \frac{1}{M}\left(\sum\nolimits_{m=1}^{M} g_2(i_m, j_m)\right)^2\right] \tag{2.10}$$

In which $\sigma_{g_2}^2$ is the standard deviation of intensity values in the reference patch.

$$\sigma_{g_1}^2(u, v) = \frac{1}{M-1}\left[\sum\nolimits_{m=1}^{M} g_1^2(i_m - u, j_m - v) - \frac{1}{M}\left(\sum\nolimits_{m=1}^{M} g_1(i_m - u, j_m - v)\right)^2\right] \tag{2.11}$$

In which $\sigma_{g_1}^2(u, v)$ is the standard deviation of intensity values in the target image at location (u, v).

$$\sigma_{g_1 g_2}(u,v) = \frac{1}{M-1}\left[\sum_{m=1}^{M} g_1(i_m-u,j_m-v)g_2(i_m,j_m) - \frac{1}{M}\sum_{m=1}^{M} g_1(i_m-u,j_m-v)\sum_{m=1}^{M} g_2(i_m,j_m)\right] \tag{2.12}$$

Which is the covariance of the two image intensity values at location (u, v).

Cross-correlation produces a value between 0 and 1. This value then could be filtered when dealing with visual targets to only drive the most suitable matches. Below, a schematic presentation of cross-correlation is shown (Figure 2.40): A reference patch (left) is correlated against a target image (right).

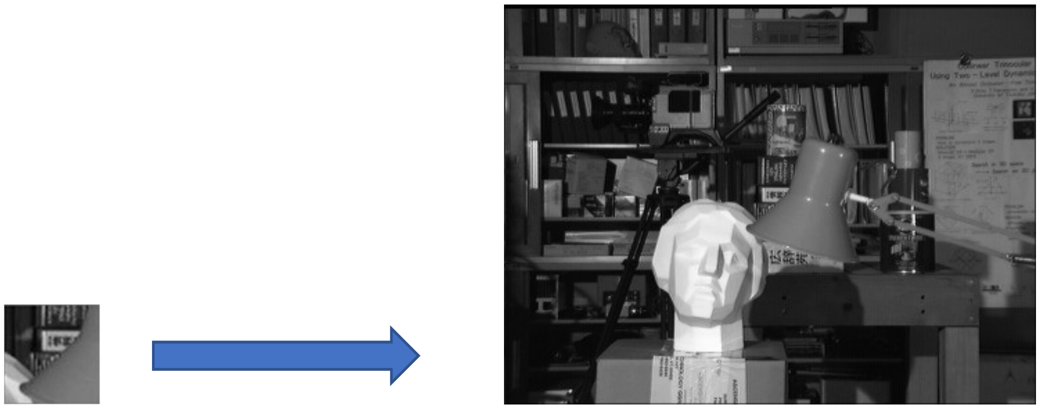


Figure 2.40 Schematics of Cross-Correlation

2.3 Summary

Visual Biomimicry has been a topic of interest for many decades now. Researchers have tried to mimic animate vision organisms to design vision systems which function seamlessly. Some of the efforts put forward in this arena has been reviewed in the sections above.

In this Chapter, making components of this thesis project were reviewed. First, Biomimicry was reviewed. This thesis project is based on biomimicry in the sense that it builds on the concepts coined by this discipline. Here the goal is to mimic nature – in this case, the human visual system.

After, specific cases of visual biomimicry in anthropomorphic vision systems design are detailed. To set the tone for this a brief background on the human visual system was provided. The discussed cases helped to shape the design and planning of the proposed system.

Then, computer vision relevant topics have been discussed in detail. This includes topics such as saliency, cross-correlation, Viola Jones, etc. Computer vision is an integral component of this thesis project and as such a detailed review of computer vision related topics are needed.

In this thesis project the aim has been to design an anthropomorphic vision apparatus which would allow us to experiment with various foveation and vergence algorithms. Inspiration has been taken from the studies mentioned in this chapter for design of this apparatus. The reason for choosing of vergence and foveation for experimentation is the fact that these two eye movements are the main movements which enable animate vision systems to perceive the external world and which enable such systems to be able to perceive depth. As such the first step in design of depth perceiving anthropomorphic vision systems would be to establish robust foveation and vergence methodologies.

Chapter 3 . Methodology

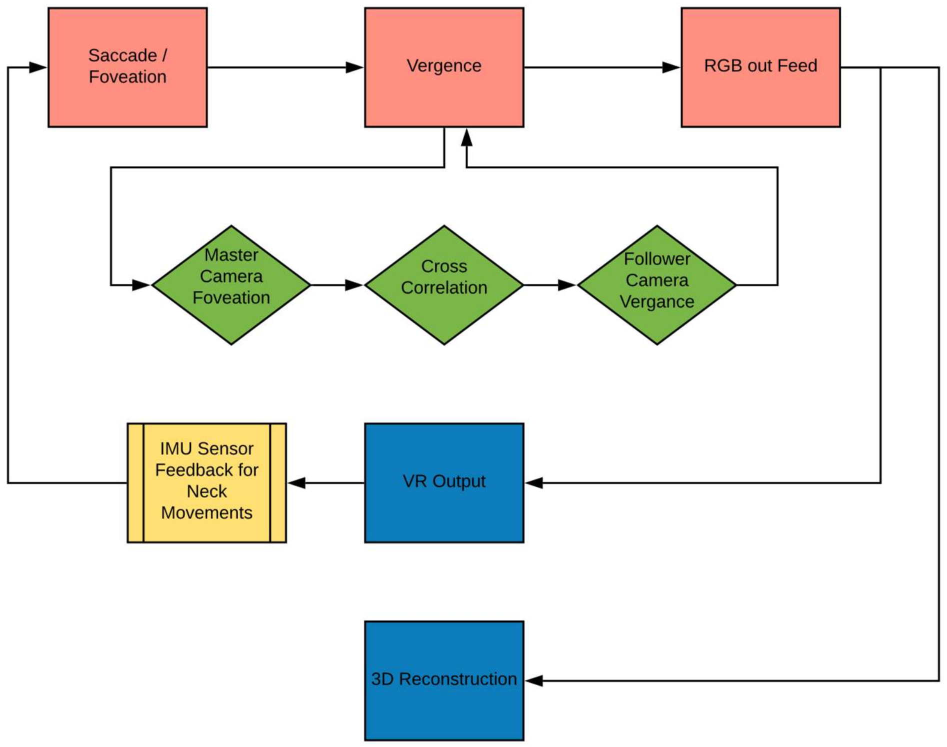


Figure 3.1 System Diagram

This section will cover the steps involved in the design and build of the proposed system. This coverage will be done in a logical (and chronological) sequence. The design steps involved are shown in Figure 3.2. A simplified system diagram for the proposed system could be seen in Figure 3.1.

As explained in Section 2.3, one of the main objectives of this thesis study is to design robust foveation and vergence algorithms for anthropomorphic vision systems. It should be noted that saccades (which is used for foveation) and vergence are not the only eye movements present in human visual system. There are other types of movements such as smooth pursuit and vestibulo-ocular movements which have been beyond the scope of this study. Saccades and vergence were chosen since they were determined to be the most critical eye movements which could lead to plausible depth perception in a controlled setting.

In order to achieve above objectives and establish the logical steps involved (Figure 3.2), several fundamental questions were posed which then made the choice for next logical steps clear. These research questions are:

1. The master camera's choice of most salient point as it relates to foveation.
2. The follower's cameras vergence towards the master camera's point of focus.
3. Feasible low-cost design for the apparatus.

After taking these questions into consideration, it became clear that first physical designed needed to be tackled followed by vision algorithm design. In relation to the physical design, first system prototyping was tackled followed by motor and control systems design. System prototyping revolved around camera's physical characteristics. Visual algorithm design was divided into foveation and vergence algorithmic design as separate entities.

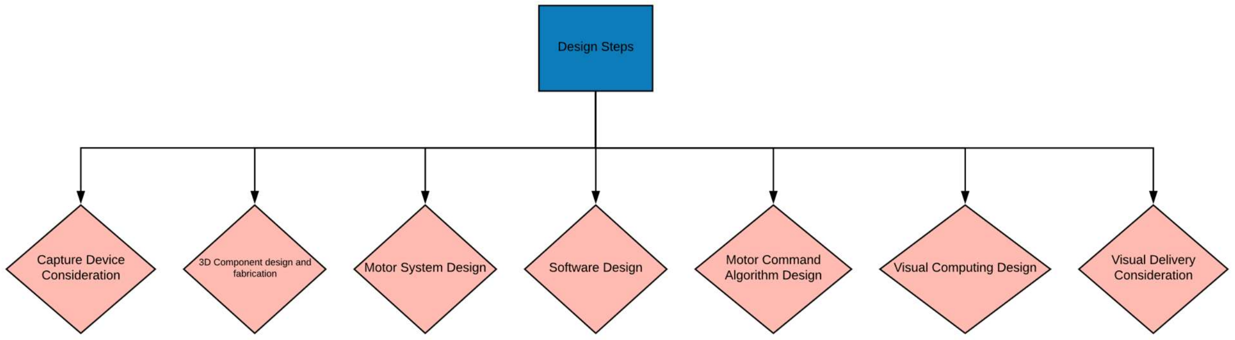

Design Steps
Capture Device Consideration
3D Component design and fabrication
Motor System Design
Software Design
Motor Command Algorithm Design
Visual Computing Design
Visual Delivery Consideration

Figure 3.2

Below is an attempt at build of a prototype of the proposed system:

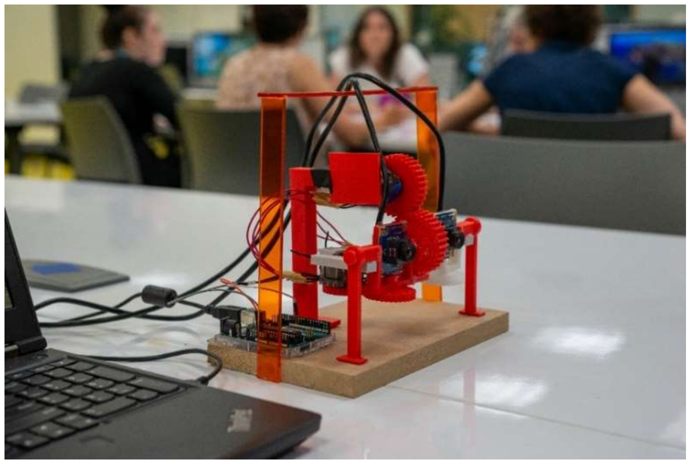


Figure 3.3 Apparatus seen from the left side

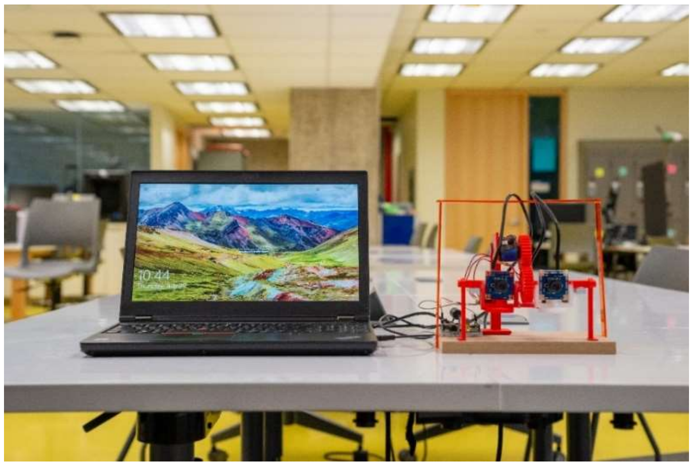


Figure 3.4 Apparatus seen from front

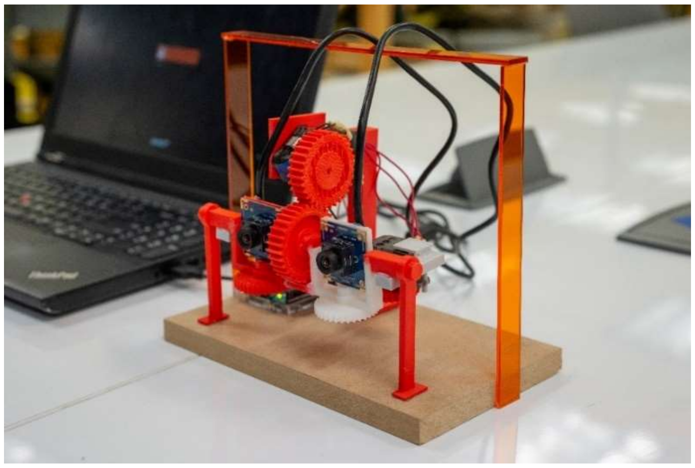


Figure 3.5 Apparatus seen from the right side

3.1 Visual Capture Device Consideration

Different camera devices were considered for the development of the proposed system. The main factors taken into consideration when deciding on the camera were: image characteristics, lens considerations (distortion, etc.) shape factor, and cost (Figure 3.6).

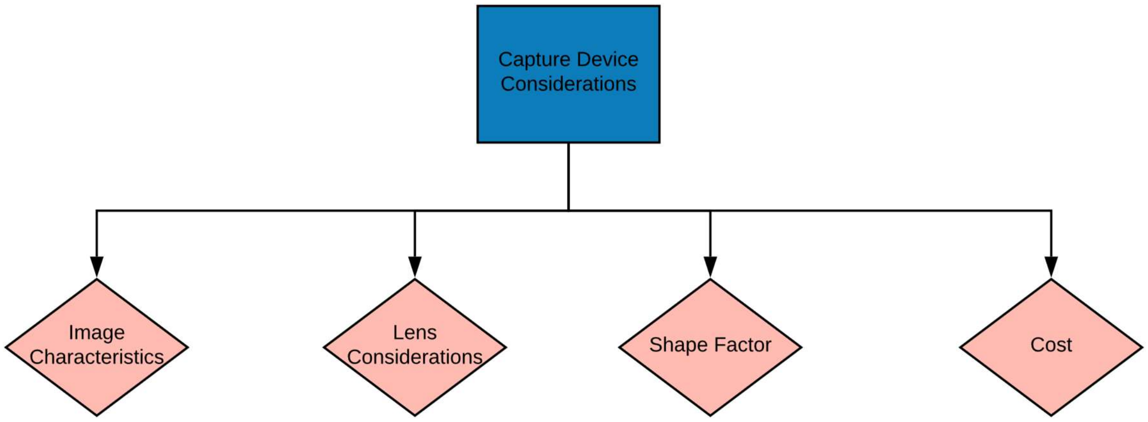


Figure 3.6

Here, a brief backgrounder on lens distortion is provided:

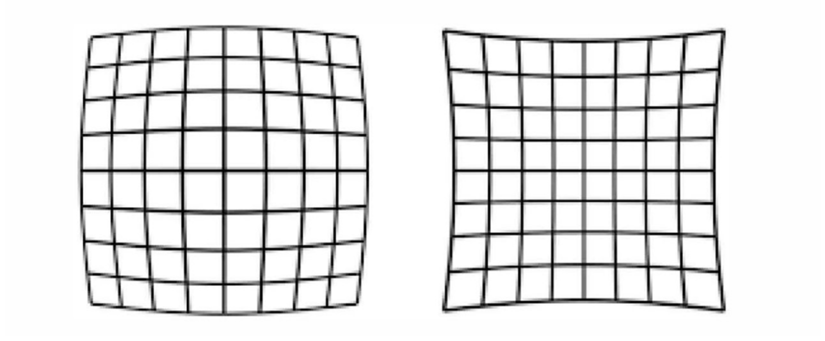


Figure 3.7 Left: Barrel distortion (Positive radial distortion), Right: Pincushion Distortion (Negative radial distortion), Accessed at: [52]

As pictured above (Figure 3.7), different lens types cause different types of distortion. These distortions could be classified into barrel distortion and pincushion distortion. When dealing with wide lenses which provide a large field of view, barrel distortion is present. As such distortion correction is needed to get a proper representation of the image. Distortion is formulated in terms of radial and tangential vectors. Radial distortions could be compensated for by:

$$x_{\text{corrected}} = x(1 + k_1r^2 + k_2r^4 + k_3r^6) \quad (3.1)$$

$$y_{\text{corrected}} = y(1 + k_1r^2 + k_2r^4 + k_3r^6) \quad (3.2)$$

Tangential distortions could be compensated for by:

$$x_{\text{corrected}} = x + [2p_1xy + p_2(r^2 + 2x^2)] \quad (3.3)$$

$$y_{\text{corrected}} = y + [p_1(r^2 + 2y^2) + 2p_2xy] \quad (3.4)$$

And where:

$$Distortion_{Coefficien} = (\mathrm{k}_1 \quad \mathrm{k}_2 \quad p_1 \quad p_2 \quad k_3) \quad (3.5)$$

$$r^2 = x^2 + y^2 \quad (3.6)$$

For this thesis project, these rectifications were not employed but these should be included in future works.

Camera solutions from manufacturers such as Point Grey were considered. Point Grey is a Canadian machine vision device manufacturer located in lower mainland British Colombia which was recently acquired by FLIR. Researchers in the field of machine vision have used visual capture devices designed by Point Grey for the past four decades. This is due to the innovation and precision involved in Point Grey’s designs. Though considered, the majority of product by this manufacturer were either unavailable to the public or expensive in price point. One of the major considerations in this project was to fabricate devices and algorithms which could be readily refabricated by others. This is in line with the open source/ DIY movement which has made a big portion of recent technological advancements possible. As such, the visual capture device selected for this project is ELP OV5640 (Figure3.8) equipped with a 100 degree autofocus capable lens.

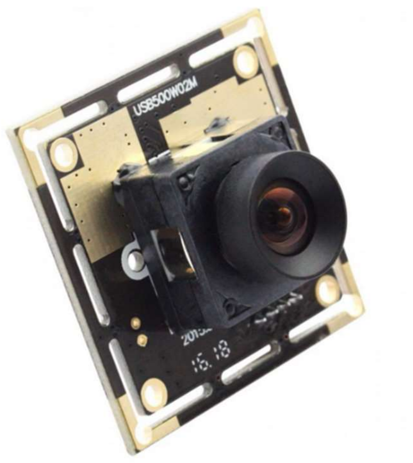
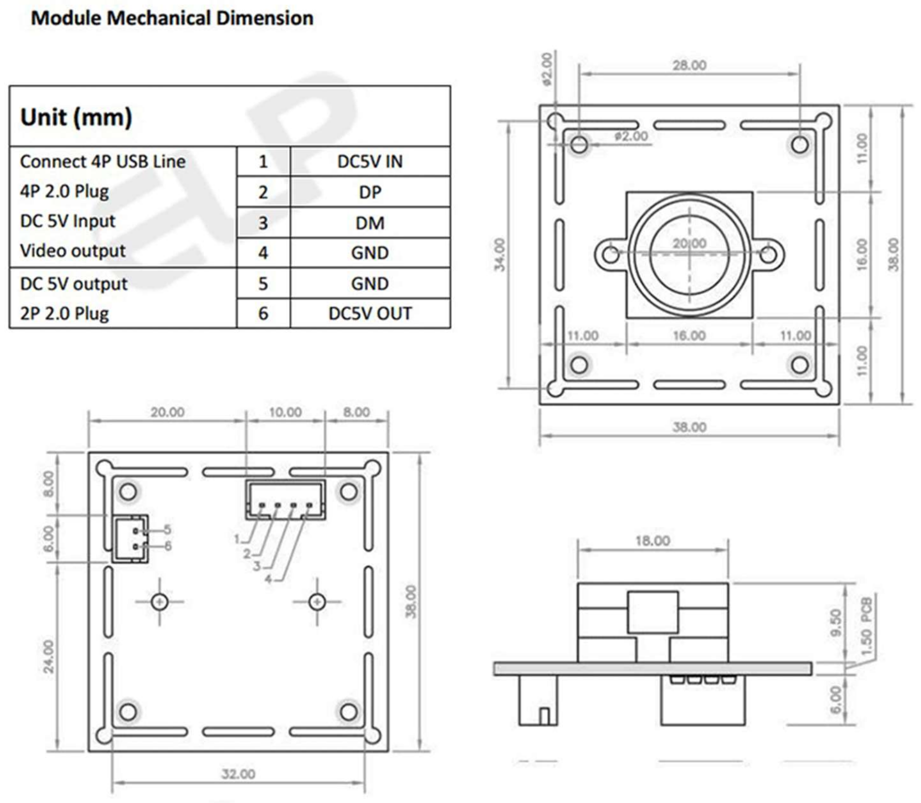


Figure 3.8 ELP OV5640 Sensor (Taken from the manufacturer's online store)



Unit (mm)		
Connect 4P USB Line 4P 2.0 Plug DC 5V Input Video output	1	DC5V IN
	2	DP
	3	DM
	4	GND
DC 5V output 2P 2.0 Plug	5	GND
	6	DC5V OUT

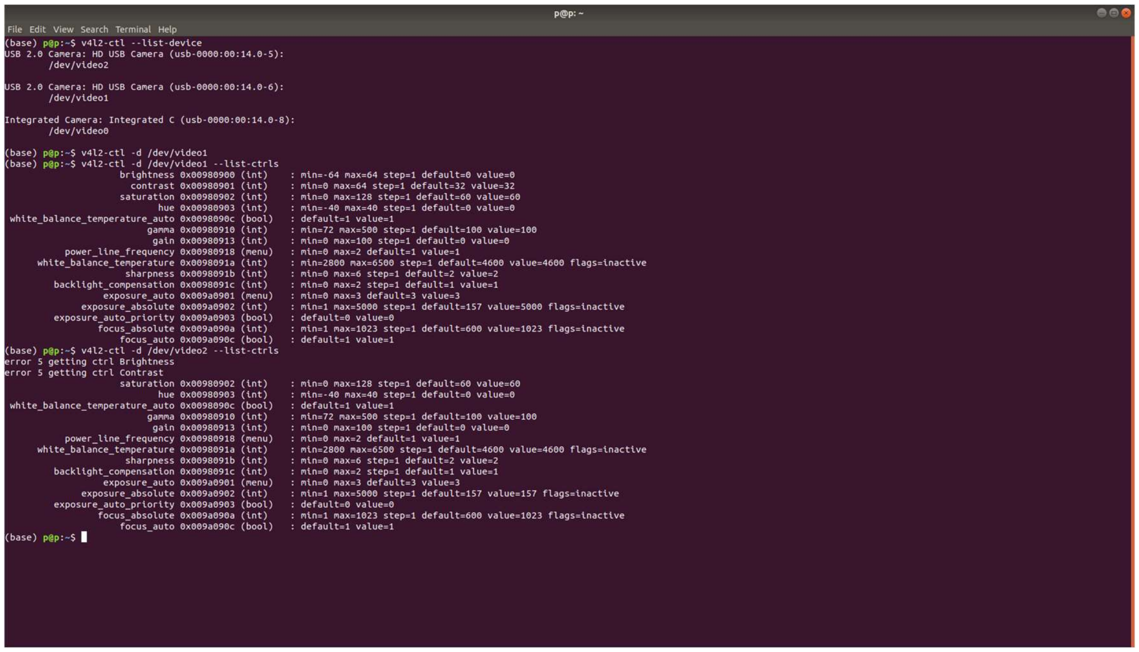
Figure 3.9 Module Mechanical Dimensions for the selected camera (Taken from the manufacturer's online store)

This camera was purchased from Aliexpress.com for the price of approximately 50 dollars. Below are the specifications of this camera taken from the seller's website (Table 3.1). This camera does not have a datasheet associated with it. A datasheet for a very similar camera was located and used for reference.

Model	ELP-USB500W02M-AF100
Sensor	OV5640
Lens Size	1/4 inches
Pixel Size	1.4µm x1.4µm
image area	3673.6µmx2738.4µm
Max. Resolution	2592(H) X 1944(V)
Compression format	MJPEG / YUV2（YUYV）
Resolution & frame	2592x1944 MJPEG 15fps YUY2 3fps 2048x 1536 MJPEG 15ps YUY2 3fps 1600x 1200 MJPEG 15fps YUY2 3fps 1920x 1080 pixels MJPEG 15fps YUY2 3fps 1280x 1024 MJPEG 15fps YUY2 7.5fps 1280x 720 MJPEG 30fps YUY2 7.5fps 1024x 768 MJPEG 30fps YUY2 15fps 800x 600 MJPEG 30fps YUY2 30fps 640x 480 MJPEG 30fps YUY2 30fps
Camera Board Assembly technique	SMT (ROSH)
Focus	Auto
Resolution	600LW/PH (Center)
S/N Ratio	36dB
Dynamic Range	68dB
Sensitivity	600 mV/Lux-sec
Shutter Type	Electronic rolling shutter / Frame exposure
Connecting Port type	USB2.0 High Speed
Free Drive Protocol	USB Video Class (UVC）
AEC	Support
AEB	Support
AGC	Support
Adjustable parameters	Brightness, Contrast, Saturation, Hue, Sharpness, Gamma, Gain, White balance, Backlight Contrast, Exposure
Lens Parameter	Size: 1/4, Iris: F2.8, FOV:100Degree Relative Illumination (Sensor): 70% IR Filter: 650±10nm
LED board power connector	Support 2P-2.0mm socket
Power supply	USB BUS POWER 4P-2.0mm socket
Operating Voltage	DC5V
Power consumption	150mW (VGA); 200mW (QSXGA)
Working current	active: 140 mA standby: 20 µA
Working temperature	-20～70℃
Storage temperature	0～65℃
Board size /Weight	38X38mm（Compatible 32X32mm）/ about 30g
Cable	Standard 1M / 2M,3M,5M optional
Operating system request	Win XP/Vista/Win7/Win8 / Linux with UVC (above linux-2.6.26） MAC-OS X 10.4.8 or later/ Android 4.0 or above with UVC
Certificate	CE&FCC

Table 3.1 Specifications for the camera used in this thesis project (Taken from manufacturer's online store)

This camera has autofocus capabilities. Initially, autofocus was incorporated as a variable element in the design but due to the low performance quality of the autofocus circuitry design in the chosen camera system, focus was set manually in following efforts. This was done using the command line in Linux, executing v4l2-clt module (Figure 3.10).



```
(base) p@p:~$ v4l2-ctl --list-device
USB 2.0 Camera: HD USB Camera (usb-0000:00:14.0-5):
        /dev/video2

USB 2.0 Camera: HD USB Camera (usb-0000:00:14.0-6):
        /dev/video1

Integrated Camera: Integrated C (usb-0000:00:14.0-8):
        /dev/video0

(base) p@p:~$ v4l2-ctl -d /dev/video1
(base) p@p:~$ v4l2-ctl -d /dev/video1 --list-ctrls
                     brightness 0x00980900 (int)    : min=-64 max=64 step=1 default=0 value=0
                       contrast 0x00980901 (int)    : min=0 max=64 step=1 default=32 value=32
                     saturation 0x00980902 (int)    : min=0 max=128 step=1 default=60 value=60
                            hue 0x00980903 (int)    : min=-40 max=40 step=1 default=0 value=0
 white_balance_temperature_auto 0x0098090c (bool)   : default=1 value=1
                          gamma 0x00980910 (int)    : min=72 max=500 step=1 default=100 value=100
                           gain 0x00980913 (int)    : min=0 max=100 step=1 default=0 value=0
           power_line_frequency 0x00980918 (menu)   : min=0 max=2 default=1 value=1
      white_balance_temperature 0x0098091a (int)    : min=2800 max=6500 step=1 default=4600 value=4600 flags=inactive
                      sharpness 0x0098091b (int)    : min=0 max=6 step=1 default=2 value=2
         backlight_compensation 0x0098091c (int)    : min=0 max=2 step=1 default=1 value=1
                  exposure_auto 0x009a0901 (menu)   : min=0 max=3 default=3 value=3
              exposure_absolute 0x009a0902 (int)    : min=1 max=5000 step=1 default=157 value=5000 flags=inactive
         exposure_auto_priority 0x009a0903 (bool)   : default=0 value=0
                 focus_absolute 0x009a090a (int)    : min=1 max=1023 step=1 default=600 value=1023 flags=inactive
                     focus_auto 0x009a090c (bool)   : default=1 value=1
(base) p@p:~$ v4l2-ctl -d /dev/video2 --list-ctrls
error 5 getting ctrl Brightness
error 5 getting ctrl Contrast
                     saturation 0x00980902 (int)    : min=0 max=128 step=1 default=60 value=60
                            hue 0x00980903 (int)    : min=-40 max=40 step=1 default=0 value=0
 white_balance_temperature_auto 0x0098090c (bool)   : default=1 value=1
                          gamma 0x00980910 (int)    : min=72 max=500 step=1 default=100 value=100
                           gain 0x00980913 (int)    : min=0 max=100 step=1 default=0 value=0
           power_line_frequency 0x00980918 (menu)   : min=0 max=2 default=1 value=1
      white_balance_temperature 0x0098091a (int)    : min=2800 max=6500 step=1 default=4600 value=4600 flags=inactive
                      sharpness 0x0098091b (int)    : min=0 max=6 step=1 default=2 value=2
         backlight_compensation 0x0098091c (int)    : min=0 max=2 step=1 default=1 value=1
                  exposure_auto 0x009a0901 (menu)   : min=0 max=3 default=3 value=3
              exposure_absolute 0x009a0902 (int)    : min=1 max=5000 step=1 default=157 value=157 flags=inactive
         exposure_auto_priority 0x009a0903 (bool)   : default=0 value=0
                 focus_absolute 0x009a090a (int)    : min=1 max=1023 step=1 default=600 value=1023 flags=inactive
                     focus_auto 0x009a090c (bool)   : default=1 value=1
(base) p@p:~$ 
```

Figure 3.10 Linux command line used for setting camera parameters for each camera

Below (Figure 3.11), the internals for the autofocus system can be seen (note that this schematic is taken from spec sheet for a similar camera design due to lack of spec sheet for used camera):

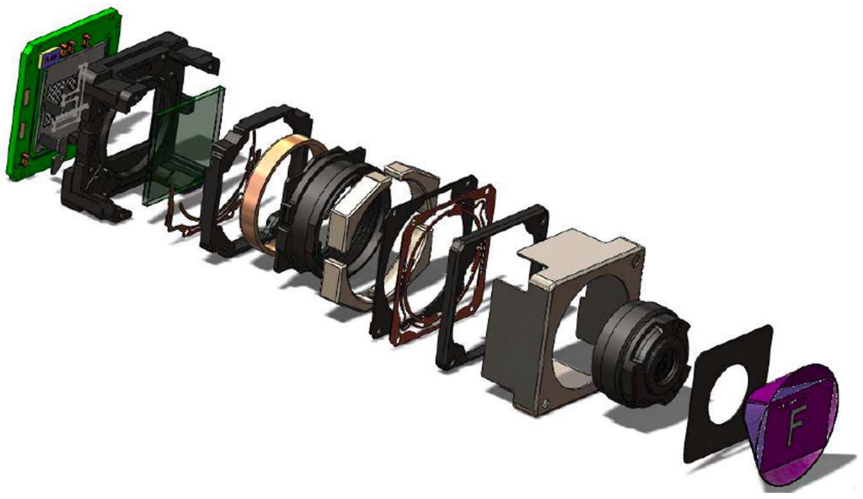


Figure 3.11 Internal components of an Autofocus enabled relevant camera, Accessed at: [53]

The capture sensor sits on a square circuit board with dimensions of 38mm by 38mm. In the following section design of the landing pad for the camera board and other components is detailed.

3.2 Component Design and Fabrication

The second step involved in this thesis project was the physical design of the apparatus (Figure 3.12). The physical design of the apparatus was conducted piecewise by first designing the landing platform for the capture device.

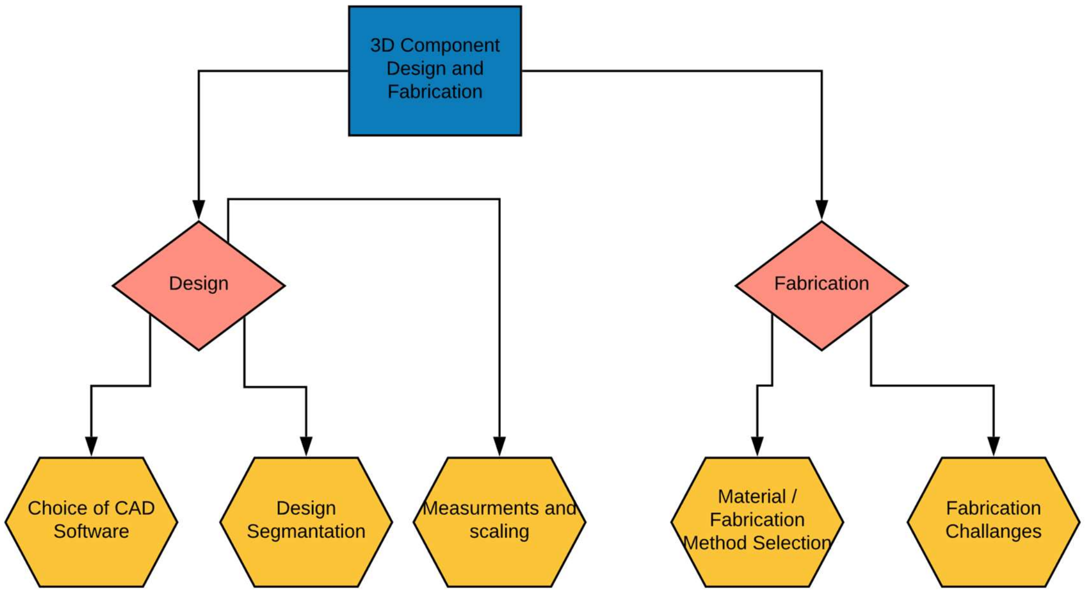


Figure 3.12

This landing platform was designed taking into consideration mechanical design measurements and characteristics for capture device circuit board shown in Figure 3.9. The first attempt at this design had a few issues which made a second design attempt necessary. Below is CAD renders for the second attempt at this design (Figure 3.13). Note the bar on the bottom used to stabilizes the pan movements of the platform, which is vital for proper performance of the design.

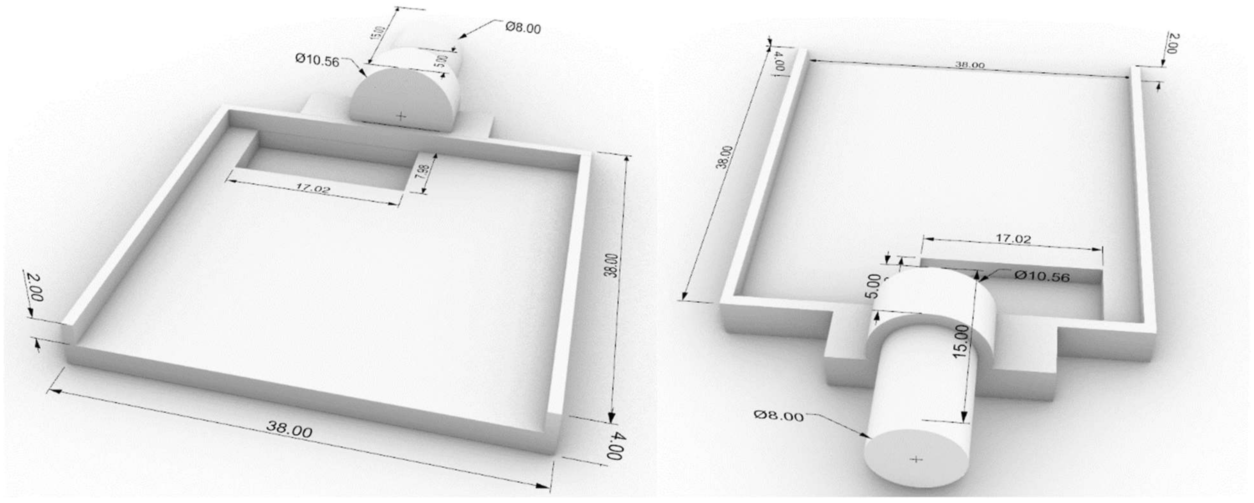
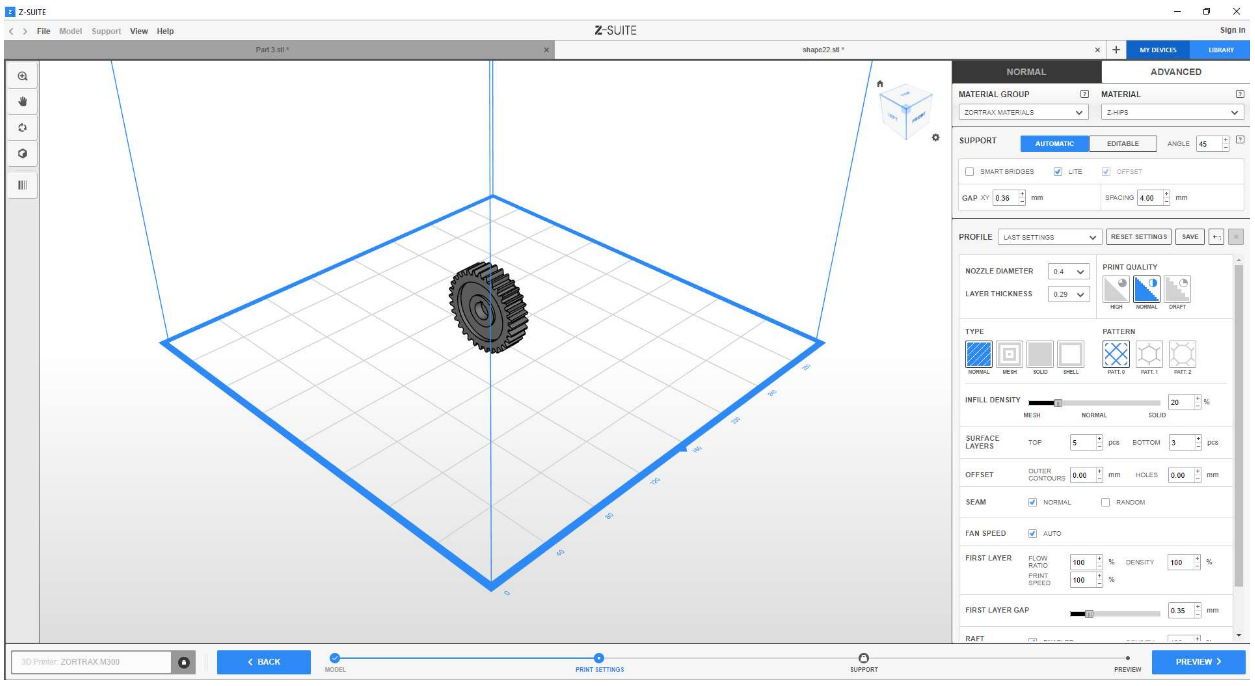


Figure 3.13 Camera Landing Platform

It should be noted that the choice of CAD software for this project was Rhino. Rhino is used mostly in design and architecture. Various CAD software was considered for design purposes such as AutoCAD, Fusion 360, Inventor and Solid Works. Solid Works is considered to be the standard for such designs but at the time of design this software was not available for use. The intuitive design nature of Rhino over AutoCAD and the rest of the software mentioned, made it the logical choice for this project. One of the great benefits of Rhino is its intuitive approach towards the use of blocks for creation of complex designs and its simplified interface.

After the successful rendering of the landing platform, 3D printing of this designed piece was commenced. For this purpose, STL exports of the CAD files were used which were then converted into Zcode (Gcode) for the printer use. This conversion to Zcode allows for control of the characteristics of the print such as density and fineness of the print (Figure 3.14). Print density was a major consideration in this project due to the fact that denser prints have lower stress and bending characteristics under shear and torsion forces and are better for this specific system design.



MATERIAL GROUP: ZORTRAX MATERIALS
MATERIAL: Z-HIPS
SUPPORT: [✓ AUTOMATIC] EDITABLE, ANGLE 45
☐ SMART BRIDGES ☑ LITE ☑ OFFSET
GAP XY 0.36 mm SPACING 4.00 mm
PROFILE: LAST SETTINGS
NOZZLE DIAMETER 0.4
LAYER THICKNESS 0.29
PRINT QUALITY: HIGH [✓ NORMAL] DRAFT
TYPE: [✓ NORMAL] MESH SOLID SHELL
PATTERN: [✓ PATT. 0] PATT. 1 PATT. 2
INFILL DENSITY 20 %
SURFACE LAYERS TOP 5 pcs BOTTOM 3 pcs
OFFSET OUTER CONTOURS 0.00 mm HOLES 0.00 mm
SEAM ☑ NORMAL ☐ RANDOM
FAN SPEED ☑ AUTO
FIRST LAYER FLOW RATIO 100 % DENSITY 100 % PRINT SPEED 100 %
FIRST LAYER GAP 0.35 mm

Figure 3.14 GUI for Z Suite and the controls available

3D printing for this project was done mainly using Zortex M300 printer using Z-Hips material. This material is special in the sense that it has better shear and torsion resistance characteristics as compared to conventional 3D printing materials, and as such is more suitable for fabrication of practical components. Additionally, this material has lower shrinkage and warping characteristics post printing, which makes the fabrication of dimension precise components more feasible. It should be noted that limited shrinkage is still present in the prints.

One of the main challenges associated with 3D printing was shrinkage, as mentioned previously. This made it necessary to make multiple prints (with increased and decreased overall size by a small margin) so that final print would be of the desired size. This was the case with the print of the main shaft which was commenced after the print of the landing component.

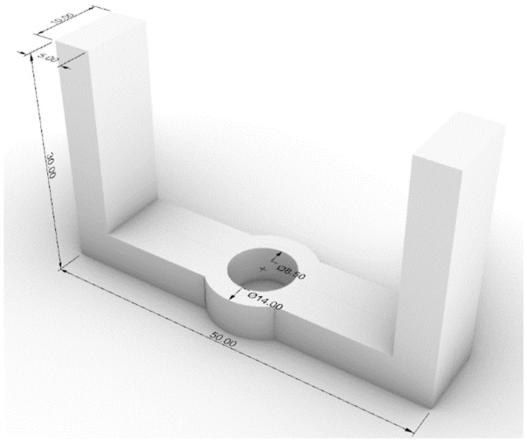


Figure 3.15 Section of Main Shaft

Since the cylinder piece in the landing component needed to fit in the cylinder groove in the shaft component, such that the two components act as a bushing joint, there was a need for several print attempts with alterations in the overall sizing of the components for fit, as pointed to previously (Figure 3.15).

The other challenging aspect of the 3D printing is presence of the print supports and non-precise edge components at the final layers of the print (Figure 3.16). Due to 3D support considerations, shaft design needed to be broken into various making components so that print supports could be placed at a location which would not compromise functional integrity of the build. This was especially important in print of the gear component and end piece components involved in the shaft design.

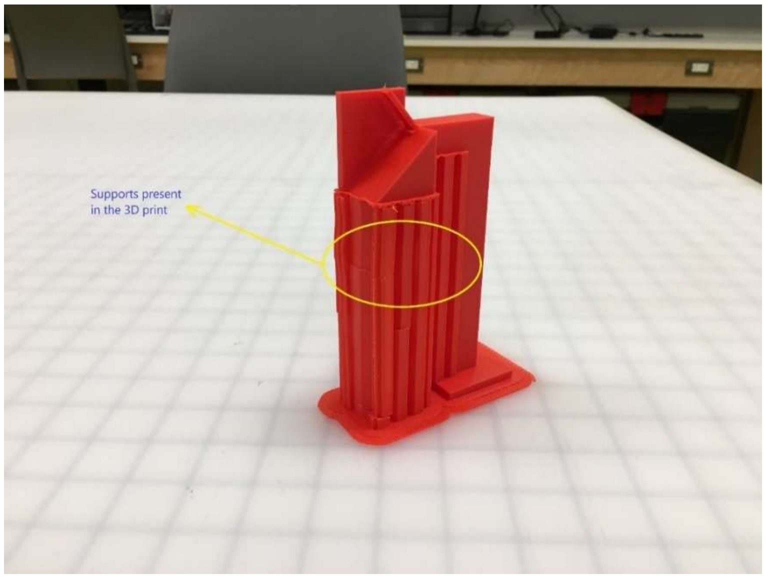


Figure 3.16 example of supports present in the 3D print

It was especially important to plan for the prints so there would be no / minimal supports present around the printed functional joints (such as gears and cylinders). Even though supports could be removed there would usually be a need to sand and to melt fine remains of the supports which then would lead to inaccuracies in functionality of the components.

The next component considered was the gear component present in the main shaft design (Figure 3.17). This gear acts as an intermediary to transfer rotational tilt movement from the servo motor responsible for the tilt of both cameras (note that both cameras are tilted together). This is one of the steps taken by the likes of Scassellati et al. for simplification of computations down the line. In order for this to work optimally there is a need for precision in the fabrication of the shaft and landing pads, so the two cameras align perfectly in regard to their static angular tilt angle and vertical baseline displacement.

This is one of the reasons that the ultimate prototype fabrication of such an apparatus would be designed out of a metal such as aluminum, which would bring the physical design accuracy of functional components into more desired thresholds. Below CAD renderings for this gear could be seen:

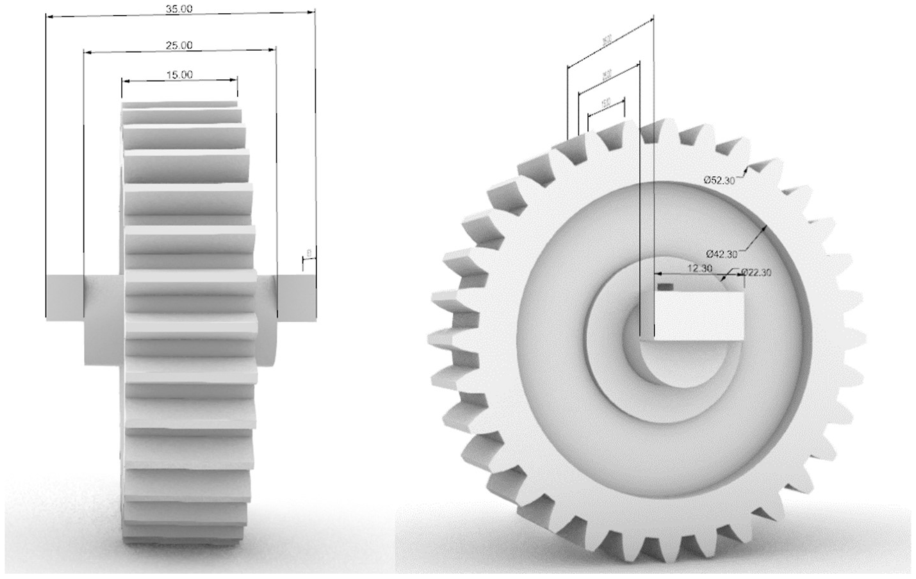


Figure 3.17 Main Shaft Gear Design

The next component designed was the end components for the shaft. Again, these needed to be printed separately due to the presence of cylinders in the design. After the completion of fabrication of the components involved in the main shaft, design and fabrication of ancillary components and motor supports was tackled. The pillars needed for main shaft support and placing were first modeled and fabricated.

Next was designs needed for motor support. Motors used in the proposed apparatus is Dynamixel XL-320 (Figure 3.18). Choice of these servos is discussed in detail in following sections. Below are the physical component drawings of this motor with the needed dimensions:

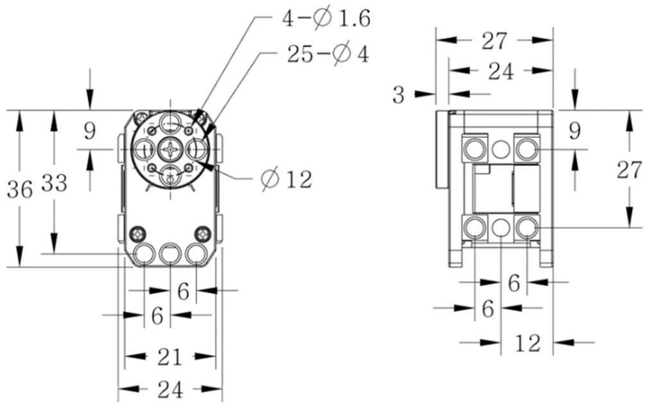


Figure 3.18 Xl-320 physical dimensions (Taken from manufacturer's website)

Designs were made for three types of gears supporting the motors:

1. Gear support for the landing pad
2. Gear support for the pan motors
3. Gear support for the tilt motor

Landing pad and pan motor gears are similar in dimensions with minor adjustments. This was followed by support design for tilt motor (Figure 3.19) as the last step of the physical build. Below support structure for the tilt motor could be seen:

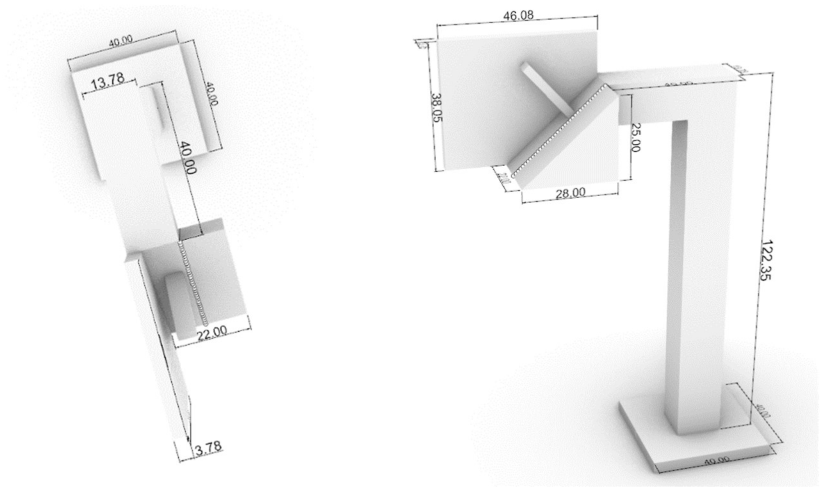


Figure 3.19 Tilt Motor platform

3.3 Motor System Design Considerations

As mentioned previously the motors used for this thesis research are the Dynamixel XL-320 digital servo motors (Figure 3.21). This choice of motors was decided upon after experimenting with a few different motor types. The motor types tested were: Step motors, analog servos and digital servos. Servo motors were chosen for several reasons: servo motors have substantially faster response times compared to stepper motors and are widely used in the area of robotics. Additionally, many of the similar attempts at systems similar to what is being designed here, use servo motors.

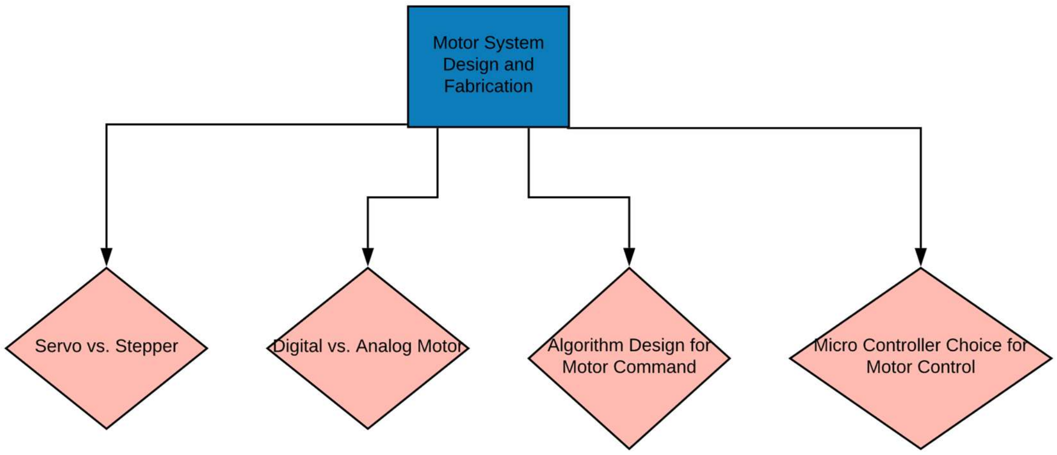

Motor System Design and Fabrication
Servo vs. Stepper
Digital vs. Analog Motor
Algorithm Design for Motor Command
Micro Controller Choice for Motor Control

Figure 3.20

Initially, experiments with analog servo motors were conducted. As a result, it was decided to use digital servo motors for additional benefits associated with these motors, which could be put in use in the future works for this thesis project. One of these benefits is closed loop feedback from the digital servo motors and the fact that these motors could be set up in a serial formation with each servo's unique identification number. These digital servo motors are widely used in the field of robotics and are commonly used in design and build of humanoid robots.

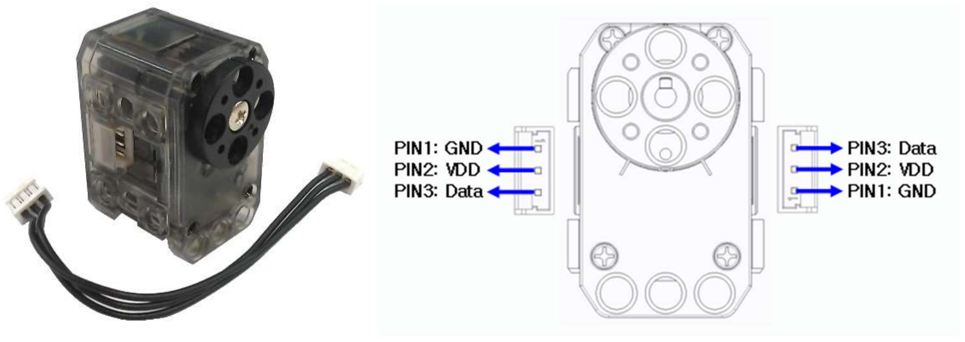


Figure 3.21 XL-320 and it's wiring diagram (Taken from the manufacturer's website)

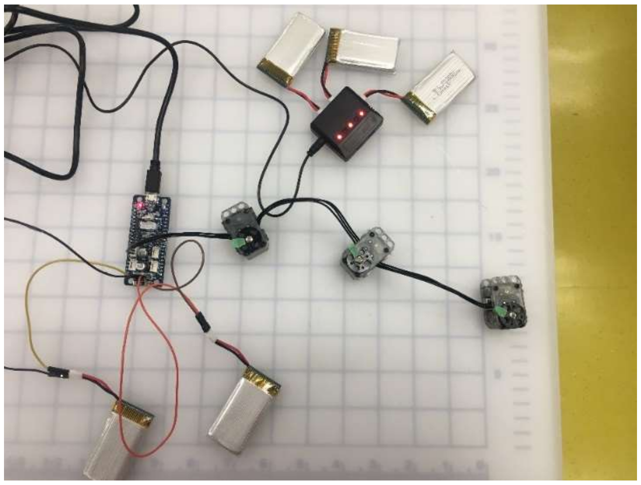


Figure 3.22 Dymanixel XL-320 motors connected to OpenCM board in series

There is a specifically designed microcontroller board designed by Robotis which is used for the control of XL-320 motors (OpenCM board). This board was used initially for experimentations (Figure 3.22). One of the challenges with this board is the fact that it does not use the conventional Arduino IDE for its operations. The IDE used for this board is a variation of Arduino IDE which does not support many of the commands, protocols and libraries used by the Arduino IDE. There were attempts made to use this board with Arduino IDE using workarounds with no luck. As such, attempts were made to use XL-320 motors with an Arduino Uno board (Figure 3.23, 3.28).

Implemented motor setup for this project is composed of 3 digital servo motors wired in serial. These are assigned IDs from 1 to 3. Servo 3 (Pan servo) and 1 (Tilt servo) are designated for the control of pan on left and right cameras accordingly and servo2 (Verge servo) is designated for control of tilt for the main shaft and alas left and right camera simultaneously.

Figure 3.23 Arduino UNO (Taken from manufacturer's website)

In order to be able to use these motors with an Arduino board, there was a need to find libraries making control of these motors through Arduino IDE feasible. A library for this purpose was retrieved at https://github.com/hackerspace-adelaide/XL320. This library gives users the ability to broadcast commands to the motors through software serial. A Baud rate of 1000000 bits is used for this matter. This then allowed us to control the three motors involved in the proposed design independently, in a serial setup. It should be noted that there is speculation that the use of soft-serial might cause delay in relay of motor commands.

3.4 Software Design

One of the most important aspects of the proposed system is the collection of algorithms which allow the system to operate as intended to. These sets of algorithms need to work in tandem to make operations feasible. To this end, considerations were made for the programming language choices and whether if all coding should be done in a single language (Figure 3.24).

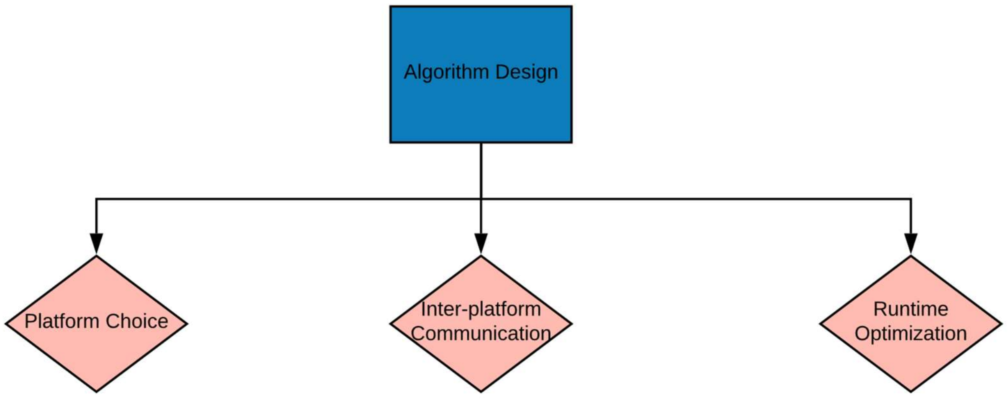


Figure 3.24

Initially, MATLAB was considered to keep all the coding on one platform. This idea was then discarded due to the fact that many of the deep learning algorithms wanting to be used have implementations in python and due to the comparingly poor runtime performance of MATLAB.

Since the majority of the deep learning algorithms needed for visual computing had implementations in Python, it was decided to use Python for visual computing (Python supports OpenCV as well) and to use Arduino IDE for motor commands.

Python communications with Arduino IDE was conducted through the serial port using a physical connection (Figure 3.25).

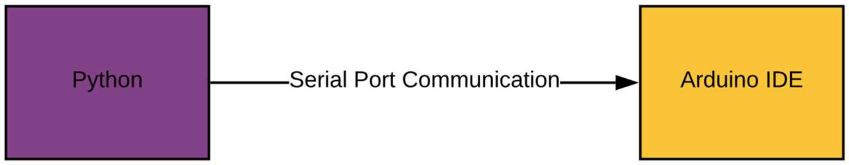


Figure 3.25

3.5 Motor Command Algorithm Design

As explained previously, choices of Arduino Uno for microcontroller and Arduino IDE for motor control were decided upon for proposed system design. One of the benefits of the Arduino IDE is its easy integration for data parsing over the serial port. Based upon visual computing conducted in Python, Servo horn positions for servos 1,2 and 3 are updated which is then parsed and communicated to Arduino IDE through the serial port. Motor horn positions are updated constantly since the algorithms consist of a main loop. In the next iterations of the algorithm, this might be altered for consistent gaze on an object and smooth pursuit purposes. Updating of servo horn's position is of stepwise nature (one step per each loop iteration, with a pre-defined step value). The update could be either step to right, step to left or no step at all. Ultimately, this implementation would need to be updated to a Proportional Integral Derivation (PID) feedback control approach.

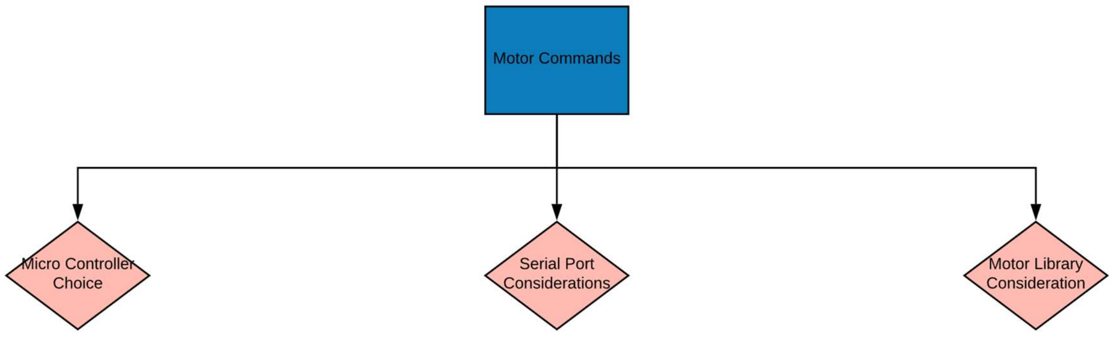


Figure 3.26

As mentioned previously, digital servos used are not compatible for use out of the box, with Arduino IDE. As such, there was a need for use of an Arduino library which would make the use of these servos through Arduino IDE possible. As such, a library provided at https://github.com/hackerspace-adelaide/XL320 was used. This library makes communications between the Arduino board and Dynamixel digital servos possible through software serial.

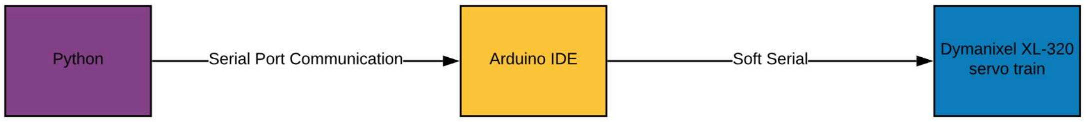


Figure 3.27

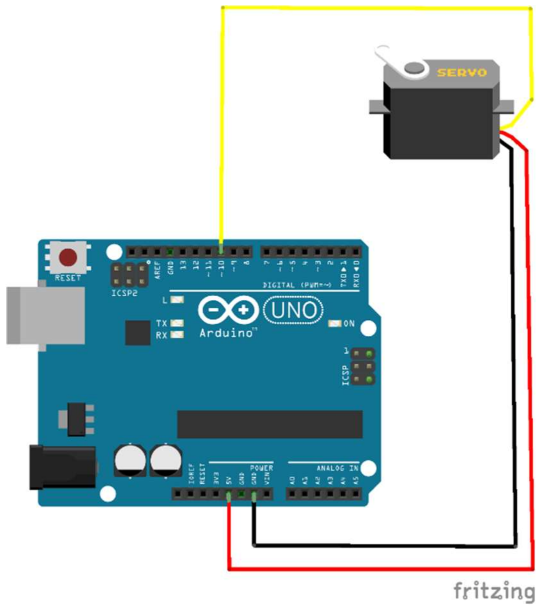


Figure 3.28 Single XL-320 wired to Arduino UNO

This library allows us to set the speed, torque and initial position of the three servo motors through Arduino IDE. The motor command algorithm in Arduino IDE is designed so it receives data from the serial port and parses the received data which is then converted into motor commands which is communicated with the servos through soft serial (Figure 3.27). Servo speeds are set to max.

3.6 Visual Computation Design

The next step for consideration is visual computation (Figure 3.29). It was decided that the visual computation needed should be solely conducted in Python. It would have been beneficial to be able to configure the visual computing in C/C++ for optimization of runtime, but cumbersome dealings with the code for CNNs involved in said languages, made this task non-feasible for the present thesis project.

Visual computation algorithm design is consistent of three major components:

1. Vergence calculation and command design
2. Saccadic foveation command design
3. Motor position update command design

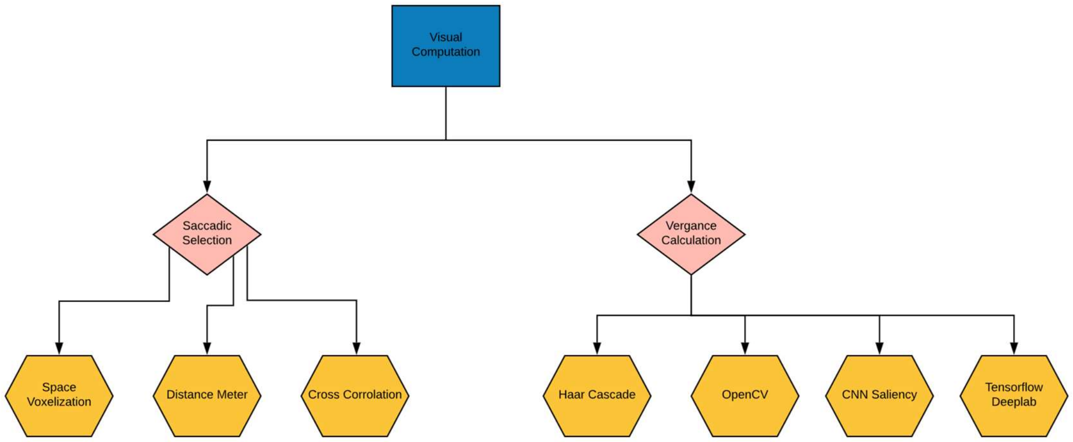


Figure 3.29

3.6.1 Vergence Command Design

Planning for this task took a long time due to consideration of different solutions, but finally the simplest solution was chosen from the bunch. This is a testament to the fact that best solution is not necessarily the most sophisticated solution. The solution chosen for this purpose is a variation of cross-correlation. One of the benefits of cross-correlation is its relatively low computational cost. Though easy to implement and low in computational cost, cross-correlation has drawbacks which are discussed later on.

OpenCV was used for implementation of cross-correlation. Initially, a standalone code was designed, which was tested on a static stereo image pair. The image pair chosen is “head and lamp" stereo scene released by University of Tsukuba in 1997, which is taken from Middlebury Stereo Imagery Library which is a go-to for stereo evaluation purposes. In this case, the far right and far left images from this set was chosen (Figure 3.30). As could be seen, location of the cameras capturing these images are separated by a baseline which is comparable in length with left and right eye baseline length in the proposed system setup. As such, if the designed vergence algorithm works on this dataset, it then should technically work on the real time video stream set up as well. Note the emphasis on technically, since this might not very well be the case and the performance might not be on par. This being said, it is a good idea to limit the variables in testing for design at this stage, so one variable at the time could be added to the proposed design in every successive design iteration.

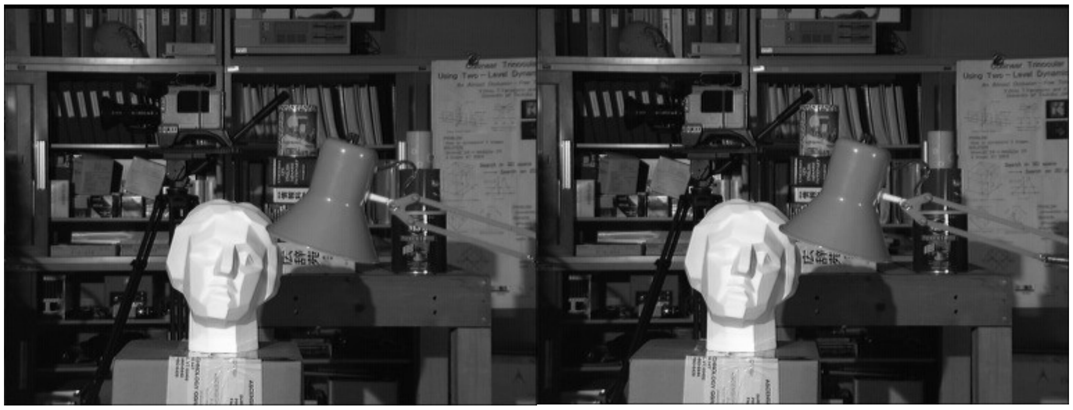


Figure 3.30 “head and lamp" stereo scene released by the University of Tsukuba in 1997, Accessed at [54]

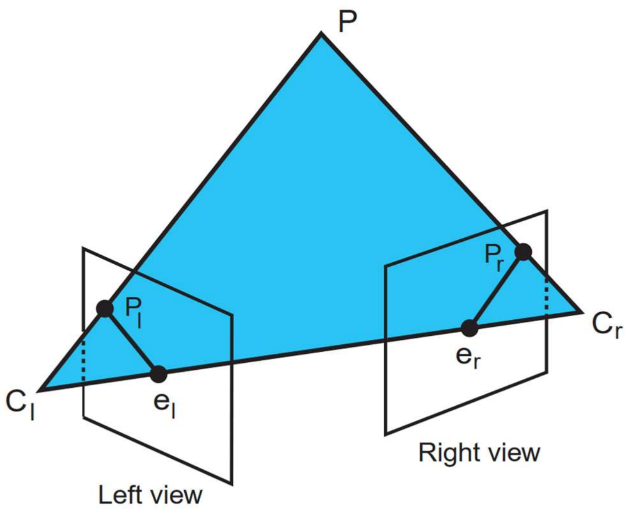


Figure 3.31 A schematic representation of Epipolar Geometry, Accessed at: [55]

The center of cameras in the left and right images in the Tsukuba set are C_l and C_r (Figure 3.31). As such the baseline is calculated as b = C_l - C_r . Ideally, for the matter of cross-correlation using a reference patch (which is the approach here), it would be lowest in cost to perform cross-correlation convolution in the target image across the baseline only. This, however, would be complicated in the present case since the baselines alter due to the constant movement of the cameras. As such, cross-correlation convolution operation was conducted throughout the entirety of the target image. This is an improved implementation which could be considered for future works. Other additional improvements which could possibly make the cross-correlation results more reliable are: solving for image distortion and image rectification.

In order to conduct cross-correlation, first a reference patch from the right image (Figure 3.32) is chosen (for the real-time system this reference patch would be chosen from the vicinity of the image center, since the vergence point needs to be center of the image).

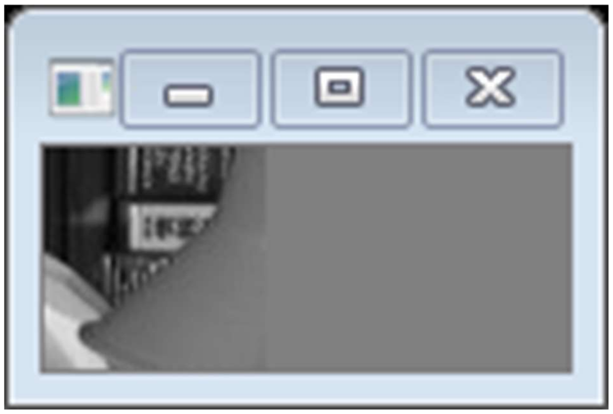


Figure 3.32 Selected refrence patch

This filter window was then convolved against the target image (left image) for peak correlation values. Threshold for matching could be set in the algorithm. This threshold correlation value is between 0 and 1. If the value is too low then there would be many matches, which is not ideal for this project's purposes. As such, an appropriate value should be chosen so neither to get many matches nor to get no matches at all.

Cross Correlation results (Figure 3.33) with threshold set to 0.5 (the value of cross correlation which is between 0 and 1, discussed previously):

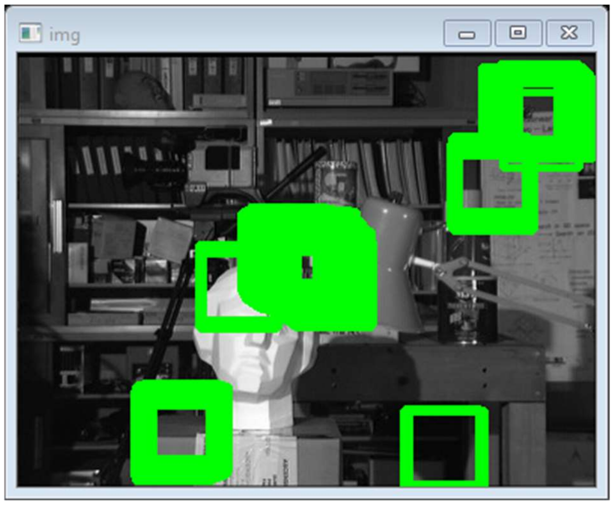


Figure 3.33 Cross-correlation results for 0.5 threshold

Cross Correlation results (Figure 3.34) with threshold set to 0.8 (the value of cross correlation which is between 0 and 1, discussed previously):

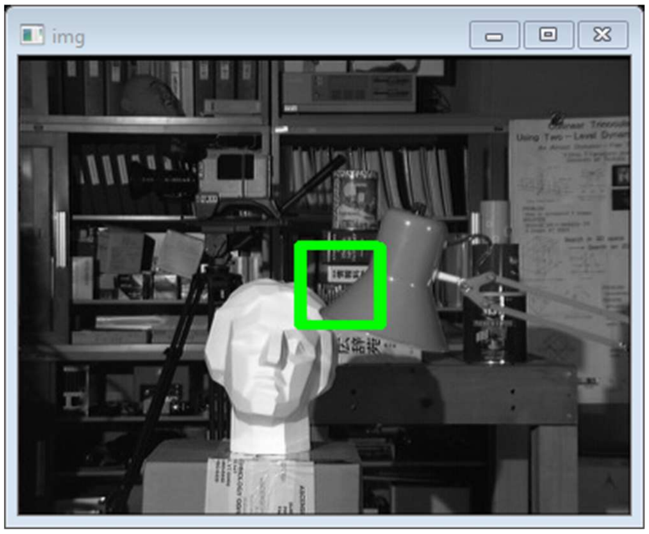


Figure 3.34 Cross-correlation results for 0.8 threshold

As could be seen (Figure 3.34), when the threshold is set to 0.8, there is only few matches which all are located in the vicinity of the area of interest. From these matches, only one needed to be selected since only one command should be sent to the relating motor for horn position update at each algorithm loop iteration. For this purpose, additions to the algorithm were made so the median of matched locations could be calculated so a single match location could be chosen for motor command purposes.

This algorithm was then put to test in the real time system for vergence purposes. The size of the filter windows was experimented with for optimization of vergence performance. One of the draw backs of using cross-correlation is that this method does not perform well when dealing with low/ no texture environments and environments with no visually distinctive attributes.

3.6.2 Saccadic Foveation Command Design

Foveation was one of the major focuses of this project. Various approaches were considered, one of which is novel in nature:

1. First use of classic computer vision methods such as Viola Jones was considered.
2. Uses of OpenCV functions for foveation on specific objects was considered, which was then used for benchmarking of the designed system.
3. Uses of machine learning methods for purposes of foveation was explored. This was started with uses of saliency neural networks. (SAM-VGG)
4. Due to the slow speed of these networks, uses of faster CNNs was explored. This was done using the Deeplab model implemented in Tensorflow.

3.6.2.1 Viola Jones

The first method used for foveation is Viola Jones. Functionality of this method has previously been explained in Chapter 2. In the proposed system, the left camera is the master camera and the right camera is the follower camera (Figure 3.35). As such, the left camera conducts foveation and right camera conducts vergence operations.

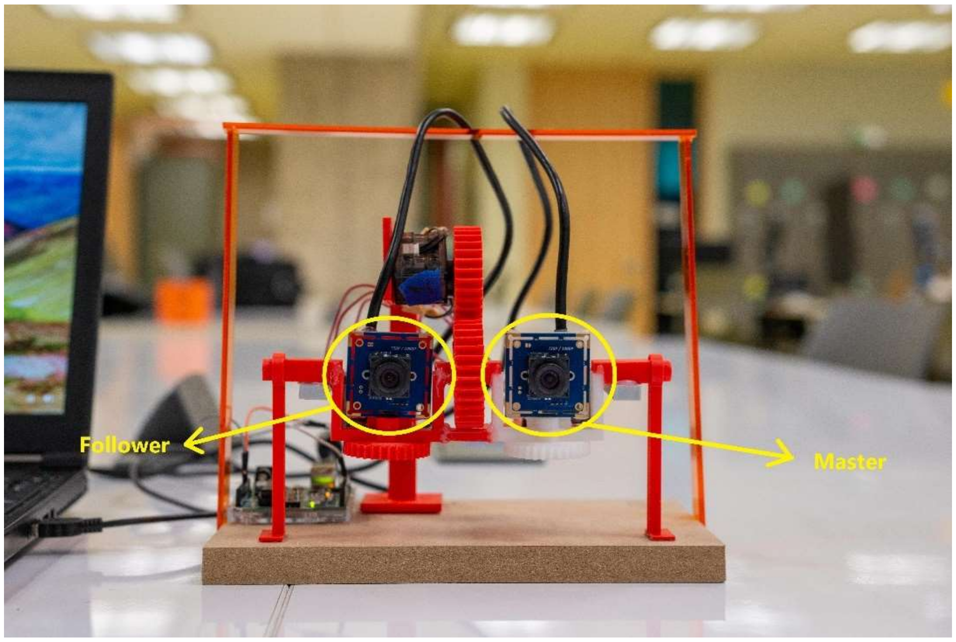


Figure 3.35 Master and Follower camera

Viola Jones execution (Figure 3.36) in OpenCV is conducted using the function “cv.CascadeClassifier” and “detectMultiScale”. First CascadeClassifier is used to load the features from an XML file. Then, detectMultiScale is used for detection of the desired targets. In this case, faces are desired targets and as such the filters in the XML provided are optimized for face targets.

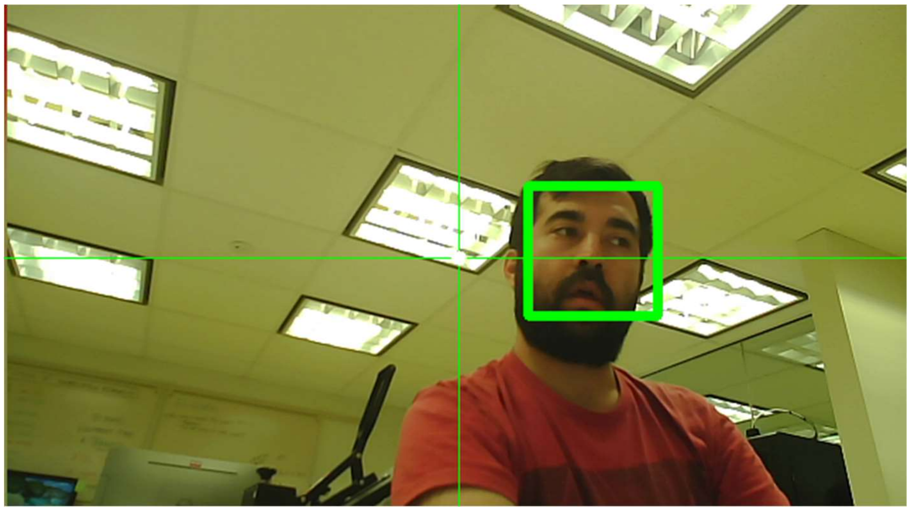


Figure 3.36 Viola Jones face detection

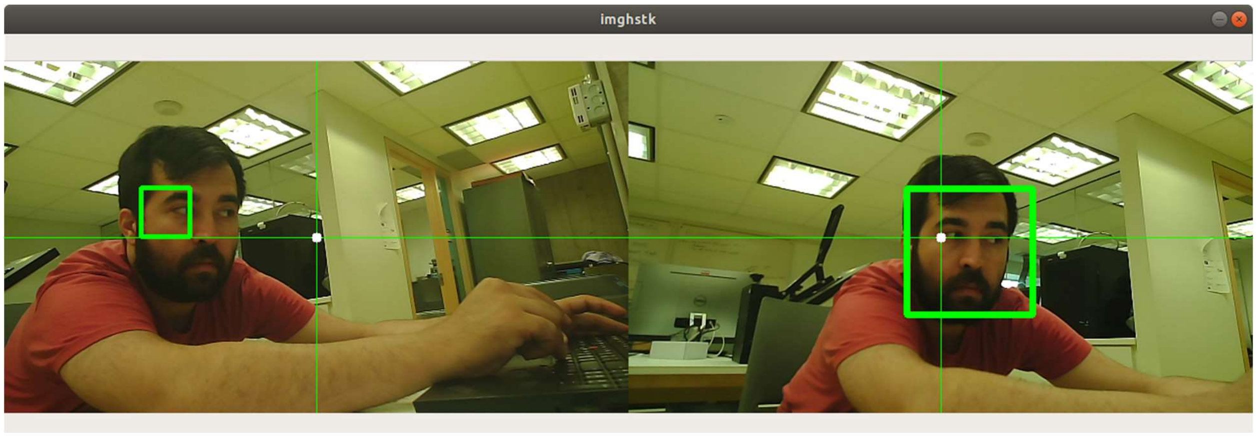


Figure 3.37 (Left) Cross-Correlation windows (Right) Viola Jones face detection

There are drawbacks to implementation of Viola Jones used in the proposed system (Figure 3.37). One of the main drawbacks is the fact that the detecting algorithm loses lock on the detected objects as the master camera moves using motor commands. Even though this target loss is not major, it is common enough to call for improvements.

3.6.2.2 OpenCV Target Detection Algorithm

Functions from OpenCV were used utilizing master camera in a foveation context to be able to detect objects of specific shape and color. The shape chosen is circle and color chosen was adjusted so it would match to the target color. The specific colour threshold adjustment values used for benchmarking in this project are:

```
lower = {'orange':(0, 50, 80)}

upper = {'orange':(24,255,255)}
```

First, image is blurred using Gaussian blur cv2.GaussianBlur(). Then cv2.inRange(, lower[key], upper[key]) is used to find the desired colour range. cv2.morphologyEx() then is used to clean missing spots and noise in the detected blob. After cv2.findContours() is used to find the contour of the detected blob. And finally, cv2.minEnclosingCircle() is used to find the fitting circle to the detected contour and ultimately circle’s center and radius (Figure 3.38).

Results for the circular object detection are seen below:

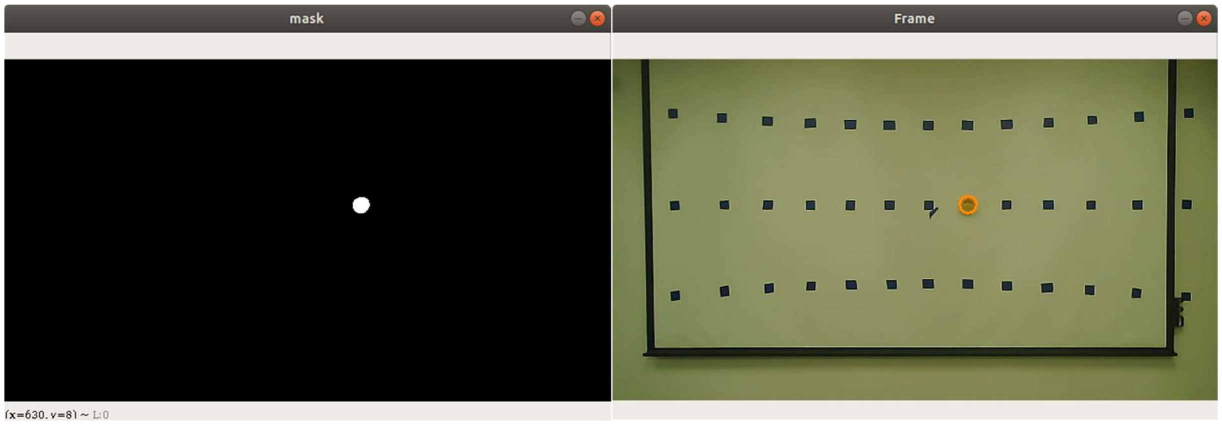


Figure 3.38 (Left) Traget color thresold output (Right) detected target output

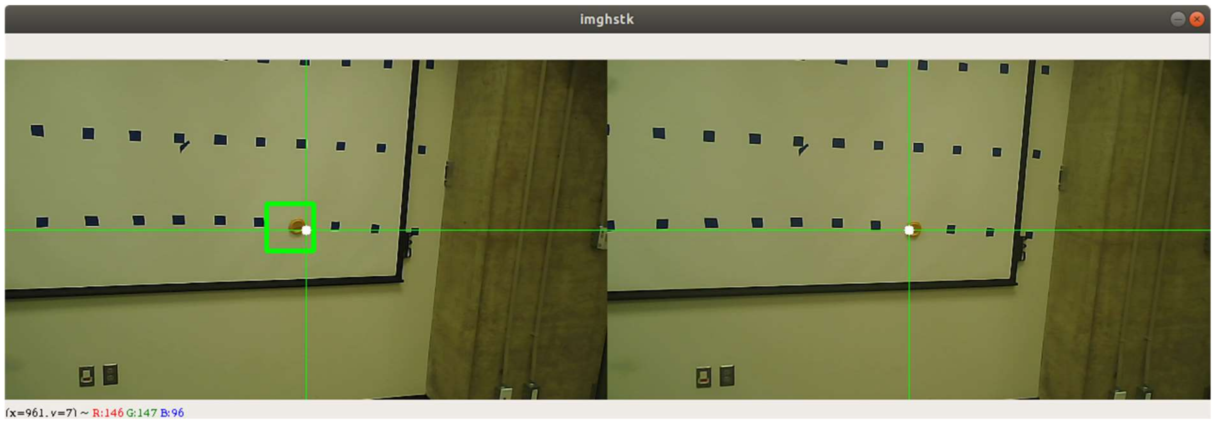


Figure 3.39 Locked final target on the Master (right) and Follower (left) camera

3.6.2.3 Machine Learning / DeepLab

The main interest in terms of foveation functionality in this project is the development of algorithms which make use of machine learning and specifically deep learning methodologies. It has been a major motivation to make use of CNNs designed for saliency detection in visual settings. To this end, experimentations with SAM-VGG were conducted which are detailed below.

3.6.2.3.1 SAM-VGG Experimentations

In Chapter 2, Saliency deep learning possibilities through SAM-VGG were detailed. Here, setup and experimentations with this network is detailed, in order to determine if this CNN could be possibly used in the proposed system design.

3.6.2.3.1.1 Experimental Setup and Challenges

The main challenges associated with the setup of SAM-VGG were due to the need for code compliance with Windows environment (Note: This was before it was decided to migrate fully to Ubuntu for all intents and purposes). Since it was attempted to run this code on a Windows machine, various alterations to the code were needed. One of the main challenges in a Windows setting is how directory paths are named.

The other source of complication was due to the fact that the designed code would compile only using specific versions of needed libraries in Python. As an example, this code would run only on Keras 1.0.7 and not a newer version of Keras. Additionally, a Cuda and Pylib installations were necessary to be able to run the code on the older version of Keras (1.0.7).

There was additionally a need to alter some of the internals of Keras for it to work with Windows. The "generic_utils.py" file under the Keras installation folder had to be located in order to change all ".decode('raw_unicode_escape')" to ".decode("windows-1252")".

3.6.2.3.1.2 Dataset Setup for SAM

Initially, there was an intention to use the Pascal-s dataset for the purposes of testing. Pascal-s makes use of Pascal dataset. Pascal dataset is a visual object recognition dataset, first provided by Oxford in 2005. This dataset has about 500,000 images with bounding boxes and labeling included. Even though this dataset was originally designed for classification purposes, it was eventually adopted for saliency detection purposes. In order to prepare the dataset for this purpose, a small portion of the dataset (850 images) was chosen and through use of eye tracking devices on various subjects, the smaller selected dataset was annotated with eye fixation index maps (Figure 3.40). It should be noted that this form of eye fixation mapping is costly in nature, as mentioned previously.

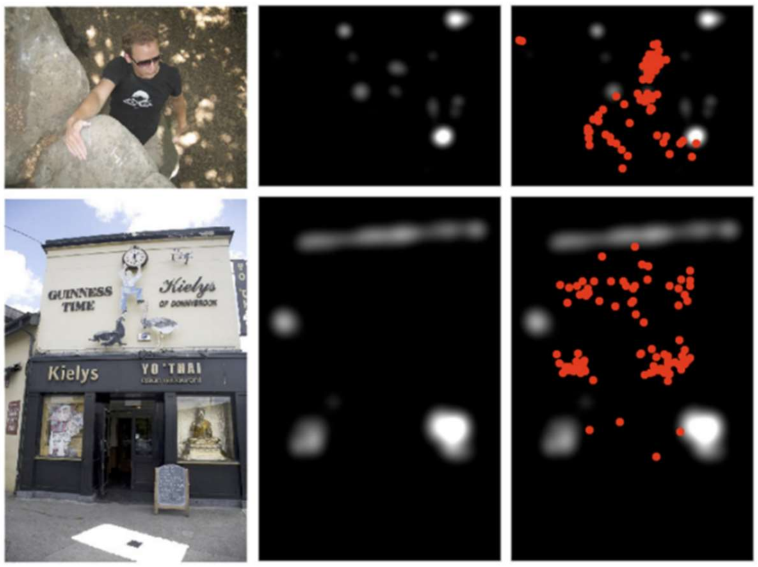


Figure 3.40 Example of saliency maps vs. eye tracking data. Middle column shows the saliency maps. Column to the right shows the fixation points (red dots), Accessed at: [31]

Another method for recording eye fixations is by using a mouse tracker. Many of the newer eye fixation datasets make use of this method in some form. One of these datasets is SALICON. This dataset uses a subset of the images present in a dataset named COCO. COCO is an image database managed by Microsoft. This database has been widely used in the previous years, in the field of image classification, using deep nets. Again, this dataset was not designed for saliency prediction and hence the need for creation of SALICON.

As stated, SALICON (10,000 training images) makes use of mouse-tracking (Figure 3.41) movements for fixation annotation in the images.

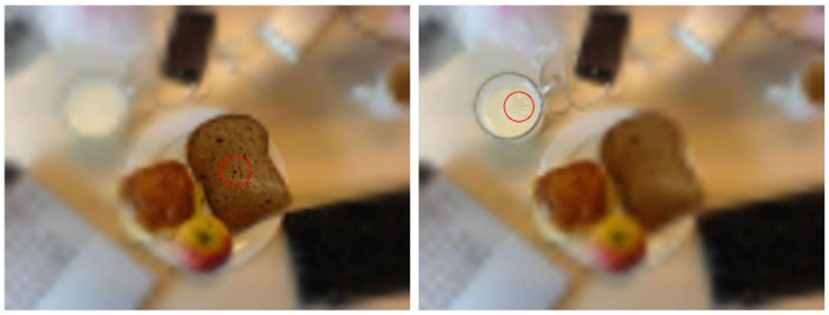
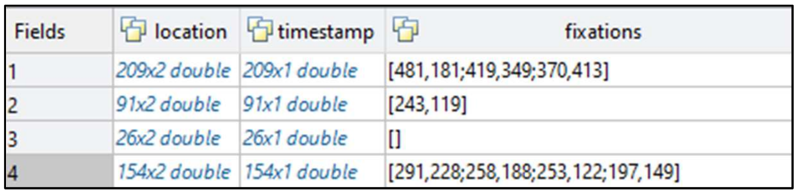


Figure 3.41 Demonstration of the method used in SALICON dataset for recording of eye fixations, Accessed at: [56]

SALICON makes use of a method called "Mouse-Contingent Stimuli" for tracking of eye fixations. Images present to the viewers are blurred using an equivalent of Gaussian blur, minus the immediate area under the pointer, controlled by the mouse, which is in focus. Hence the viewer needs to move the mouse to be able to clearly see areas of interest in an image. These areas tend to be the areas of immediate eye fixation.

This data is then recorded in a ".mat" file for each image. This is basically a hyper matrix of N by 3. N represents the number of viewers for each image and the 3 columns are "location", "time stamp" and "fixation". "Location" indicates the location of mouse cursor pause. "Time stamp" is the time stamp for the said location. "Fixation" column which is the column of interest for us, indicates where the fixations in the location hypermatrix are for the said viewer (Figure 3.42). These tend to be points with more than one "location" visit.



Fields	location	timestamp	fixations
1	209x2 double	209x1 double	[481,181;419,349;370,413]
2	91x2 double	91x1 double	[243,119]
3	26x2 double	26x1 double	[]
4	154x2 double	154x1 double	[291,228;258,188;253,122;197,149]

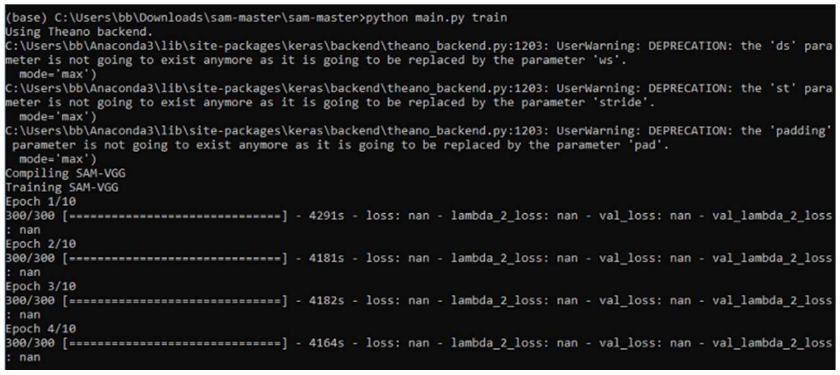
Figure 3.42 Example of a few rows of "fixation.mat" for a training image in SALICON dataset

One of the advantages of SALICON dataset (COCO to be exact) is the fact that the images are all standardized and have the same dimensions (480 by 640). This makes the ingestion into the convolutional model much easier to handle, as compared to Pascal which has images of various sizes. Due to this fact, it was decided to use the SALICON dataset for these experimentations.

A subset of SALICON dataset was chosen as the designated dataset for experimentations. This dataset has 300 training images and 10 validation images (10% of training portion). The reason to choose such a small dataset was the limited processing power available to us at the time of this study. Use of Google Colaboratory was not a possibility due to complicated setup involved for execution of the utilized model.

3.6.2.3.1.3 Experimentations with SAM

After setup of the designated dataset was complete, experimentations with the configured SAM model were undertaken. There was a need to alter a portion of SAM source code for at hand experimentations, so it would run on Windows operating system. First, training of the model using the unaltered SAM architecture (VGG16 variation) using 300 training and 10 validation images, utilizing 10 epochs for the training process was initiated. SAM's source code provides the option of using three different loss measures. These are: Kl-Divergence, Normalized Scanpath Saliency and Linear Correlation Coefficient. The algorithm is designed to provide all three of these loss metrics on each training run but, in these experiments, when all three of the metrics were used, loss values would converge to NAN on all occasions (Figure 3.43). On a suggestion from Cornia (author of the code/ SAM) training was done including only one loss metric. Using this setup, conclusive loss metrics were achieved. It was decided to go with KL-Divergence since this metric is the most relevant of the loss metrics regarding saliency prediction.



```
(base) C:\Users\bb\Downloads\sam-master\sam-master>python main.py train
Using Theano backend.
C:\Users\bb\Anaconda3\lib\site-packages\keras\backend\theano_backend.py:1203: UserWarning: DEPRECATION: the 'ds' para
meter is not going to exist anymore as it is going to be replaced by the parameter 'ws'.
  mode='max')
C:\Users\bb\Anaconda3\lib\site-packages\keras\backend\theano_backend.py:1203: UserWarning: DEPRECATION: the 'st' para
meter is not going to exist anymore as it is going to be replaced by the parameter 'stride'.
  mode='max')
C:\Users\bb\Anaconda3\lib\site-packages\keras\backend\theano_backend.py:1203: UserWarning: DEPRECATION: the 'padding'
 parameter is not going to exist anymore as it is going to be replaced by the parameter 'pad'.
  mode='max')
Compiling SAM-VGG
Training SAM-VGG
Epoch 1/10
300/300 [==============================] - 4291s - loss: nan - lambda_2_loss: nan - val_loss: nan - val_lambda_2_loss
: nan
Epoch 2/10
300/300 [==============================] - 4181s - loss: nan - lambda_2_loss: nan - val_loss: nan - val_lambda_2_loss
: nan
Epoch 3/10
300/300 [==============================] - 4182s - loss: nan - lambda_2_loss: nan - val_loss: nan - val_lambda_2_loss
: nan
Epoch 4/10
300/300 [==============================] - 4164s - loss: nan - lambda_2_loss: nan - val_loss: nan - val_lambda_2_loss
: nan
```

Figure 3.43 SAM-VGG, 10 epochs, batch size 1, all error metrics included, "nan" error

The mentioned training process, on a Thinkpad Workstation equipped with Intel Xeon E3-1505M processor and Nvidia Quadro M2200 graphics card, took upwards of 5 hours, with no conclusive final weights (The training stopped after four epochs due to no conclusive weights. This is a hard-coded condition in the code). For this training, the batch size was set to 1. Later, it was realized that this is not something which is recommended as bigger batch sizes are preferred.

Below, is the image used for testing purposes (Figure 3.44):

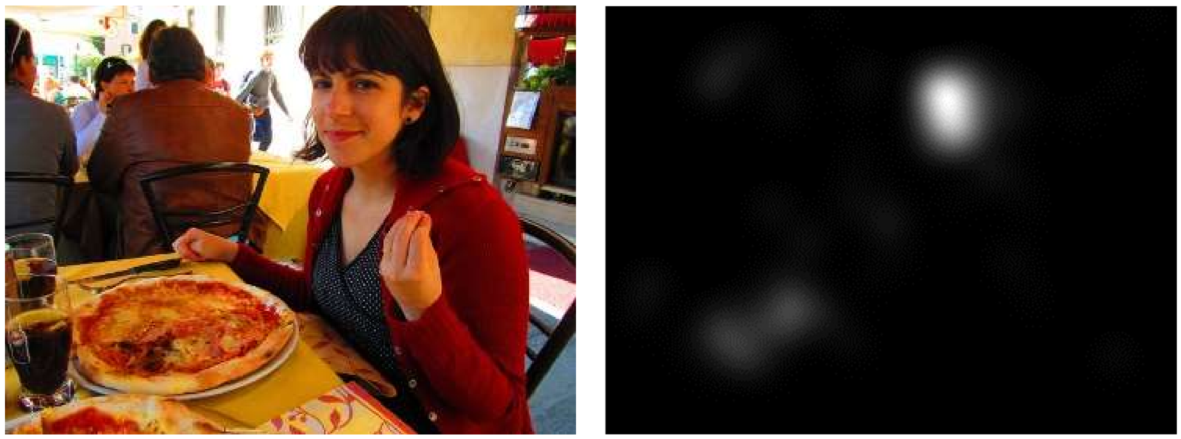


Figure 3.44 Test image used for the evaluation of performance of the trained model. ("test.jpg") (Right) Actual image. (Left) Saliency ground truth map for the image. (benchmark for further calculated saliency maps)

Finalized weights usable on the CNN model are calculated through training of the model. These weights are stored in a ".pk1" file. SAM-VGG model is provided with the weights for training on the full SALICON dataset, with the optimal number of epochs, batch size, etc. (published weights) Initial testing was done using these weights. The weights file used for testing purposes is declared in line 105 of "main.py". Next, the specified test image was processed through the CNN using this provided weight set (Figure 3.45):

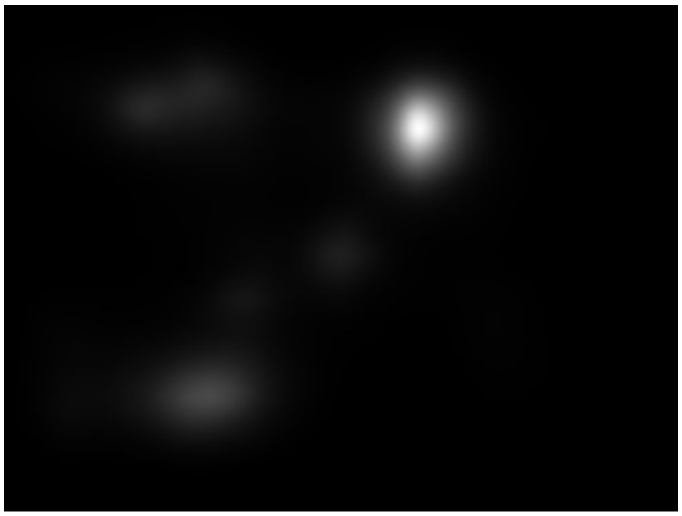


Figure 3.45 Saliency prediction map for "test.jpg" using published weights for SAM-VGG

In order to be able to observe the differences between estimated saliency map and the ground truth subtract function was used in CV2 (written in "difference.py"). Using this, the difference between ground truth and the previous prediction were calculated (Figure 3.46).

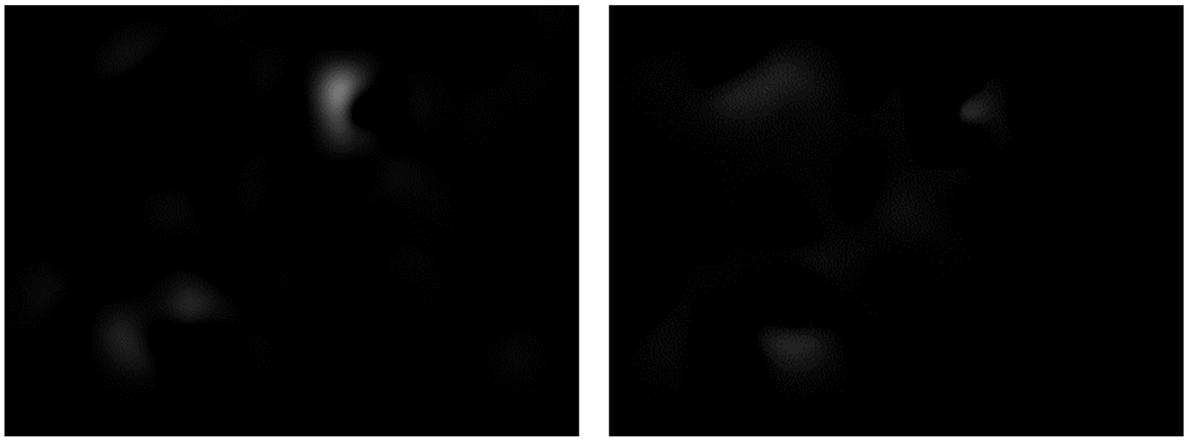


Figure 3.46 Subtraction of predicted saliency map ("test.jpg") from ground truth using published weights for SAM-VGG. (Right: ground truth-prediction, Left: prediction- ground truth)

As it could be seen above, adopted model using said weights performs adequately. There is a region to the left (left map) of the face which is not detected fully and detection on the food could be slightly more pronounced. The over detection (right map) is quite minimal.

Next, it was decided to proceed with one epoch only using a batch size of one, keeping the architecture of the adopted model intact, using only KL-Divergence (Figure 3.47). This was done in order to get familiar with possible challenges/ characteristics of a one epoch run. Below, are the snapshots of progression of loss reduction value in this one epoch run.

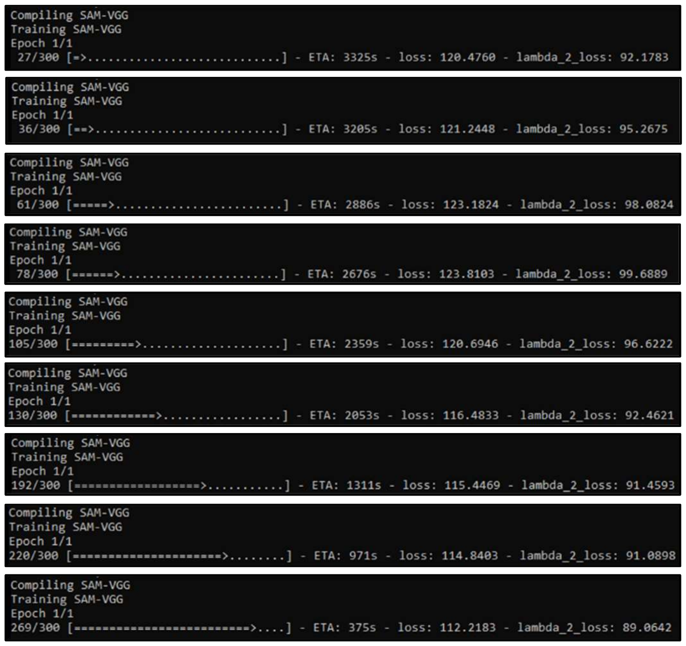

```
Compiling SAM-VGG
Training SAM-VGG
Epoch 1/1
 27/300 [=>............................] - ETA: 3325s - loss: 120.4760 - lambda_2_loss: 92.1783
```
```
Compiling SAM-VGG
Training SAM-VGG
Epoch 1/1
 36/300 [==>...........................] - ETA: 3205s - loss: 121.2448 - lambda_2_loss: 95.2675
```
```
Compiling SAM-VGG
Training SAM-VGG
Epoch 1/1
 61/300 [=====>........................] - ETA: 2886s - loss: 123.1824 - lambda_2_loss: 98.0824
```
```
Compiling SAM-VGG
Training SAM-VGG
Epoch 1/1
 78/300 [======>.......................] - ETA: 2676s - loss: 123.8103 - lambda_2_loss: 99.6889
```
```
Compiling SAM-VGG
Training SAM-VGG
Epoch 1/1
105/300 [=========>....................] - ETA: 2359s - loss: 120.6946 - lambda_2_loss: 96.6222
```
```
Compiling SAM-VGG
Training SAM-VGG
Epoch 1/1
130/300 [============>.................] - ETA: 2053s - loss: 116.4833 - lambda_2_loss: 92.4621
```
```
Compiling SAM-VGG
Training SAM-VGG
Epoch 1/1
192/300 [==================>...........] - ETA: 1311s - loss: 115.4469 - lambda_2_loss: 91.4593
```
```
Compiling SAM-VGG
Training SAM-VGG
Epoch 1/1
220/300 [=====================>........] - ETA: 971s - loss: 114.8403 - lambda_2_loss: 91.0898
```
```
Compiling SAM-VGG
Training SAM-VGG
Epoch 1/1
269/300 [=========================>....] - ETA: 375s - loss: 112.2183 - lambda_2_loss: 89.0642
```

Figure 3.47 SAM-VGG, 1 epoch, batch size 1, looking at the trends in KL-Divergence

As it could be seen, KL-Divergence is steadily decreasing through the progression of training. It should be noted though that the error rate is ultimately very high (approximately 112 by the end) and this mainly due to the chosen dataset being very small (300 images). What is ultimately desired is a KL-Divergence with a value close to zero.

Then, this trained model was tested, using the weights calculated in this training process, on the chosen test image. Below is the saliency map produced (Figure 3.48).

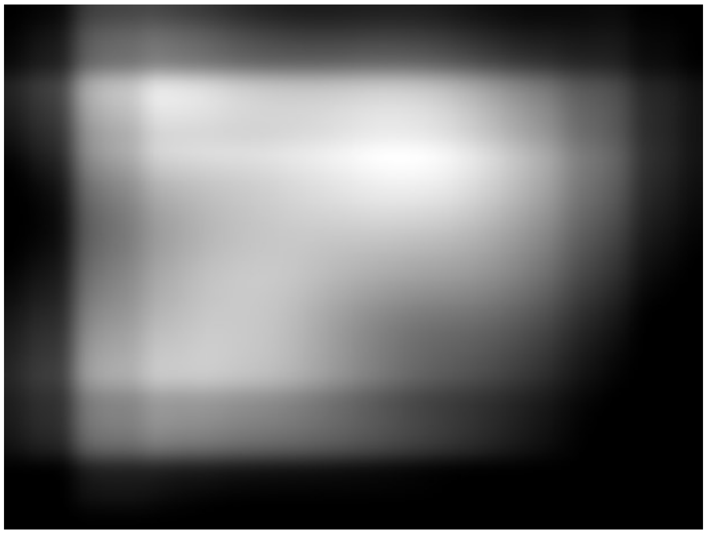
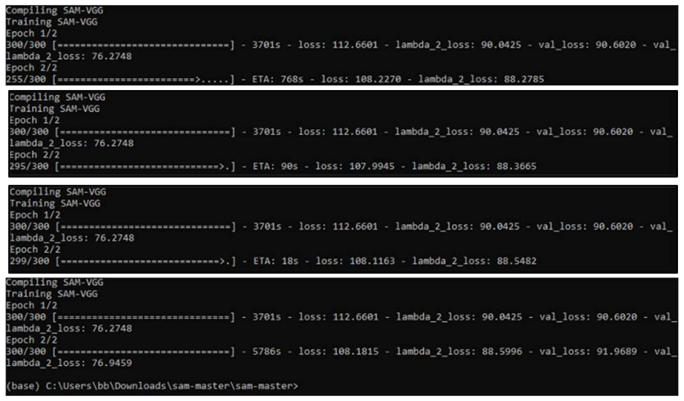


Figure 3.48 SAM-VGG, 1 epoch, batch size 1, saliency map prediction for test image

As it could be seen above, this training instance does not produce acceptable results and as such the difference map calculations for this predicted saliency map is not going to be of relevance.

Next, it was decided to increase the number of epochs to two. So, for this training instance, two epochs were used with a batch size of 1 and the original architecture. Below is the Kl-Divergence for this training (Figure 3.49).



```
Compiling SAM-VGG
Training SAM-VGG
Epoch 1/2
300/300 [==============================] - 3701s - loss: 112.6601 - lambda_2_loss: 90.0425 - val_loss: 90.6020 - val_
lambda_2_loss: 76.2748
Epoch 2/2
255/300 [=========================>.....] - ETA: 768s - loss: 108.2270 - lambda_2_loss: 88.2785
```

```
Compiling SAM-VGG
Training SAM-VGG
Epoch 1/2
300/300 [==============================] - 3701s - loss: 112.6601 - lambda_2_loss: 90.0425 - val_loss: 90.6020 - val_
lambda_2_loss: 76.2748
Epoch 2/2
295/300 [============================>.] - ETA: 90s - loss: 107.9945 - lambda_2_loss: 88.3665
```

```
Compiling SAM-VGG
Training SAM-VGG
Epoch 1/2
300/300 [==============================] - 3701s - loss: 112.6601 - lambda_2_loss: 90.0425 - val_loss: 90.6020 - val_
lambda_2_loss: 76.2748
Epoch 2/2
299/300 [============================>.] - ETA: 18s - loss: 108.1163 - lambda_2_loss: 88.5482
```

```
Compiling SAM-VGG
Training SAM-VGG
Epoch 1/2
300/300 [==============================] - 3701s - loss: 112.6601 - lambda_2_loss: 90.0425 - val_loss: 90.6020 - val_
lambda_2_loss: 76.2748
Epoch 2/2
300/300 [==============================] - 5786s - loss: 108.1815 - lambda_2_loss: 88.5996 - val_loss: 91.9689 - val_
lambda_2_loss: 76.9459

(base) C:\Users\bb\Downloads\sam-master\sam-master>
```

Figure 3.49 SAM-VGG, 2 epochs, batch size 1, looking at the trends in KL-Divergence

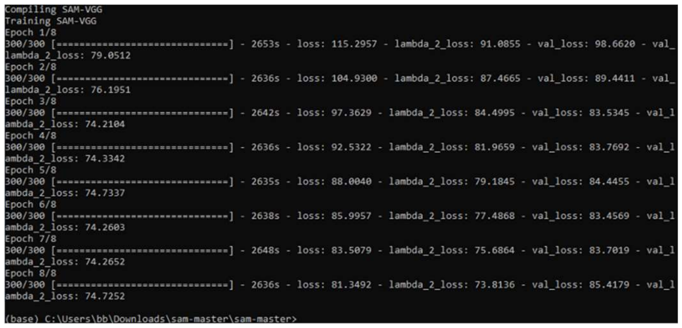
As seen above, there is an improvement (reduction) in the KL-Divergence value on the second epoch. It should be noted though that again, the dataset size is small, losses are very high and for us to have descent results, many more epochs are needed.

Then, the test image was run though the CNN using these trained weights. Below is the predicted saliency map (Figure 3.50):

Figure 3.50 SAM-VGG, 2 epochs, batch size 1, saliency map prediction for test image

As pictured above, the results are more refined compared to 1 epoch training. There is refinement around body area and food, (pizza) in addition to the face in the background. Again, the difference map calculation for this saliency map is not of relevance since results are not refined enough.

Next, it was decided to train the adopted model with 8 epochs and a batch size of 10 (as suggested) to try and get better results. Below are the KL-Divergence results for this training instant (Figure 3.51).



```
Compiling SAM-VGG
Training SAM-VGG
Epoch 1/8
300/300 [==============================] - 2653s - loss: 115.2957 - lambda_2_loss: 91.0855 - val_loss: 98.6620 - val_
lambda_2_loss: 79.0512
Epoch 2/8
300/300 [==============================] - 2636s - loss: 104.9300 - lambda_2_loss: 87.4665 - val_loss: 89.4411 - val_
lambda_2_loss: 76.1951
Epoch 3/8
300/300 [==============================] - 2642s - loss: 97.3629 - lambda_2_loss: 84.4995 - val_loss: 83.5345 - val_l
ambda_2_loss: 74.2104
Epoch 4/8
300/300 [==============================] - 2636s - loss: 92.5322 - lambda_2_loss: 81.9659 - val_loss: 83.7692 - val_l
ambda_2_loss: 74.3342
Epoch 5/8
300/300 [==============================] - 2635s - loss: 88.0040 - lambda_2_loss: 79.1845 - val_loss: 84.4455 - val_l
ambda_2_loss: 74.7337
Epoch 6/8
300/300 [==============================] - 2638s - loss: 85.9957 - lambda_2_loss: 77.4868 - val_loss: 83.4569 - val_l
ambda_2_loss: 74.2603
Epoch 7/8
300/300 [==============================] - 2648s - loss: 83.5079 - lambda_2_loss: 75.6864 - val_loss: 83.7019 - val_l
ambda_2_loss: 74.2652
Epoch 8/8
300/300 [==============================] - 2636s - loss: 81.3492 - lambda_2_loss: 73.8136 - val_loss: 85.4179 - val_l
ambda_2_loss: 74.7252

(base) C:\Users\bb\Downloads\sam-master\sam-master>
```

Figure 3.51 SAM-VGG, 8 epochs, batch size 10, KL-Divergence values

One thing to note is the fact that increase in batch size dramatically reduces the processing time. Time spent on each epoch has gone down from approximately 4500 seconds to approximately 2500 seconds. Additionally, as it can be seen, the loss values steadily drop with each epoch, starting at approximately 115 going down to approximately 81.

Then, the chosen test image was run through the adopted CNN, using the produced weights. Below is the predicted saliency map (Figure 3.52):

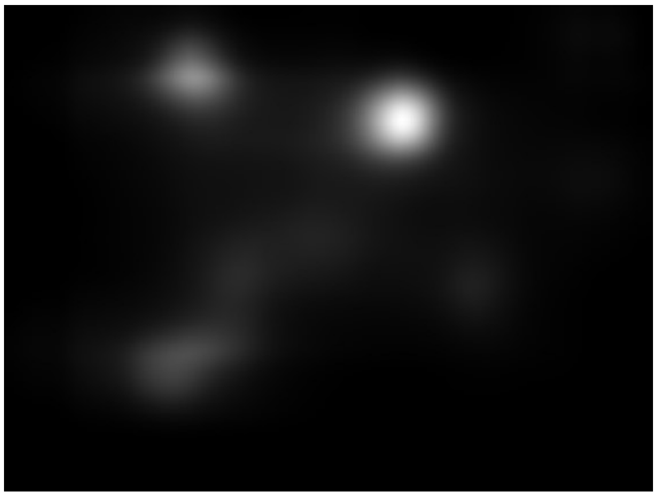


Figure 3.52 SAM-VGG, 8 epochs, batch size 10, saliency map prediction for test image

As pictured, these results are drastically better than the previous try. Now, there is a focus on the areas of interest (main subject's face, pizza and the face in the background). Since the results are much improved, the difference operation for this saliency map prediction was conducted (Figure 3.53):

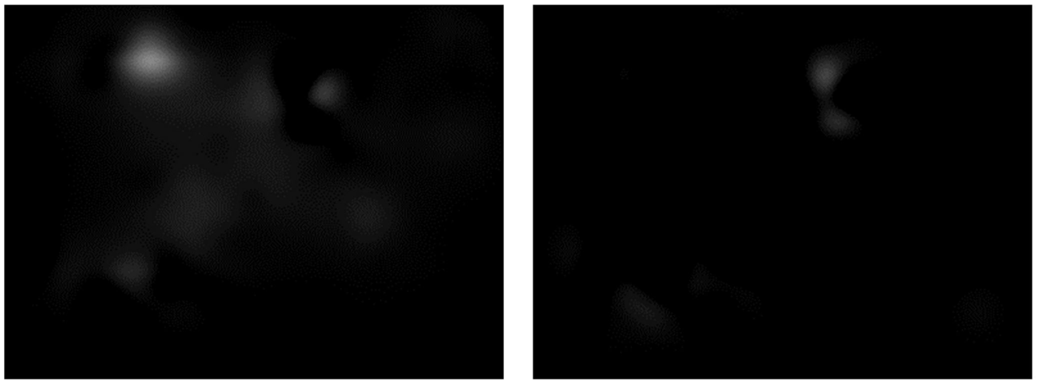


Figure 3.53 SAM-VGG, 8 epochs, batch size 10, prediction vs benchmark saliency. (right: ground truth-prediction, left: prediction-ground truth)

As it could be seen above (left), the adopted model using this set of weights, over predicts on the face in the background mostly. It could be said that the model is overperforming on food and surroundings. Additionally (right) the model is not fully detecting all the area associated with subject's face (left hand side of the face).

Then, it was decided to start experimenting with the architecture of the model itself. Here, the aim was simplification of the system model. In order to do this, the available options were omitting either LSTM or learned priors. It was determined (through trial and error) that LSTM was one of the major components contributing to computational cost/time in the training process and as such, it was decided to omit LSTM. In order to do this, changes were made in "model.py". Lines 105, 106 and 107 were disabled and in line 111 "att_convlstm" was replaced with "dcn.output". this was done so the learned prior model would ingest the output from the modified VGG16 procedure, so LSTM could be bypassed altogether.

After doing this, training of the modified model was initiated. First, the model was trained with 1 epoch, batch size of 10. The chosen test image was run through the CNN, using the weights from this training. Below is the predicted saliency model (Figure 3.54):

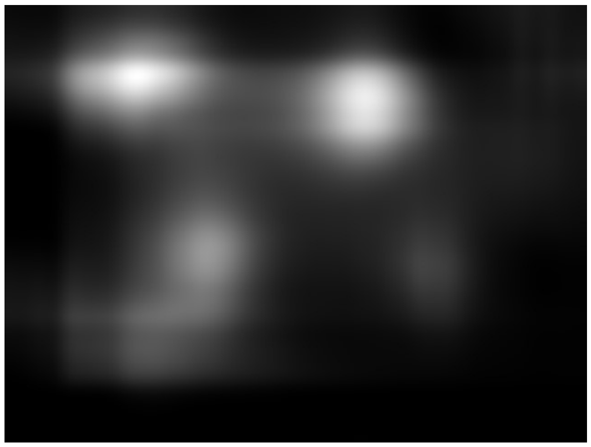
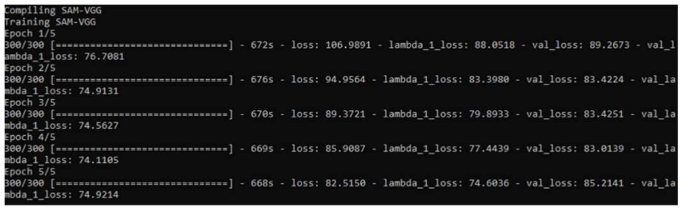


Figure 3.54 SAM-VGG¬-noLTSM, 1 epoch, batch size 10, saliency map prediction for test image

Again, the prediction is not very good. But it is an improvement on "Original architecture, 1 epoch, batch size 1". This could be due either to the batch size or the fact that LSTM was omitted from the design. It should be noted though that LSTM, in theory, is meant to refine the results and hence, in this case, to refine the saliency area around subjects of interest and as such omitting of LSTM should not theoretically improve the results. Here the difference map wasn't implemented since the results need much improvement.

Next, it was decided to repeat this training on the altered architecture, with 5 epochs and a batch size of 10. Below is the KL-Divergence for this training (Figure 3.55):



```
Compiling SAM-VGG
Training SAM-VGG
Epoch 1/5
300/300 [==============================] - 672s - loss: 106.9891 - lambda_1_loss: 88.0518 - val_loss: 89.2673 - val_l
ambda_1_loss: 76.7081
Epoch 2/5
300/300 [==============================] - 676s - loss: 94.9564 - lambda_1_loss: 83.3980 - val_loss: 83.4224 - val_la
mbda_1_loss: 74.9131
Epoch 3/5
300/300 [==============================] - 670s - loss: 89.3721 - lambda_1_loss: 79.8933 - val_loss: 83.4251 - val_la
mbda_1_loss: 74.5627
Epoch 4/5
300/300 [==============================] - 669s - loss: 85.9087 - lambda_1_loss: 77.4439 - val_loss: 83.0139 - val_la
mbda_1_loss: 74.1105
Epoch 5/5
300/300 [==============================] - 668s - loss: 82.5150 - lambda_1_loss: 74.6036 - val_loss: 85.2141 - val_la
mbda_1_loss: 74.9214
```

Figure 3.55 SAM-VGG-noLTSM, 5 epochs, batch size 10, KL-Divergence values

As it could be seen, KL-Divergence loss value is decreasing steadily through each epoch, starting at approximately 106 decreasing to approximately 82. The other observation of interest is the estimated time for each epoch. This time has decreased to approximately 650 seconds for each epoch now. This is a great improvement if training time is of importance in the operations and if the omittance of LSTM does not substantially degrade the performance (computational cost vs. quality).

Next, the chosen test image was run though the CNN using the calculated weights. Below is the saliency prediction for the chosen test image (Figure 3.56):

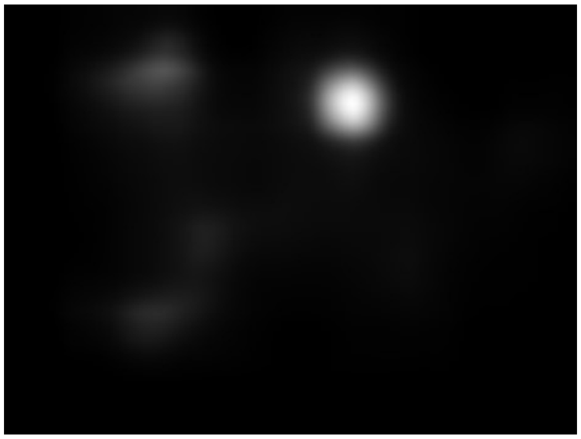


Figure 3.56 SAM-VGG¬-noLTSM, 8 epochs, batch size 10, saliency map prediction for test image

As could be seen, this saliency prediction again is closer to optimal results. There is a focus on the face in the foreground, plus some saliency focus on pizza and the face in the background. Below is the difference visualization for this prediction (Figure 3.57).

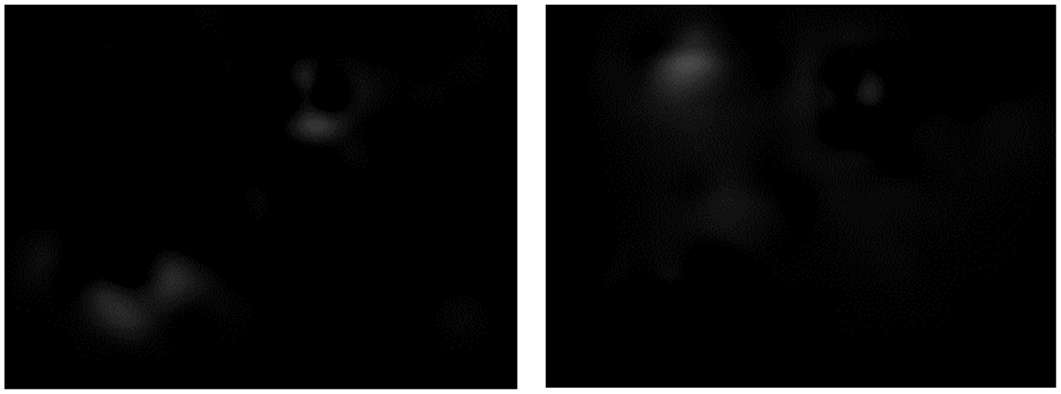


Figure 3.57 SAM-VGG-noLTSM, 5 epochs, batch size 10, prediction vs benchmark saliency. (Right: ground truth-prediction, left: prediction-ground truth)

As could be seen above, the results are close to the benchmark both on left and right. It may be said that these results, even on a 5-epoch training are better in nature than an 8-epoch LSTM included training. It should be noted though that this is only one test image and a personal opinion and that to truly quantify the performance better measures/ more test images are needed.

Additionally, it was decided to conduct some empirical testing with the published weights. Two paintings - one from Jackson Pollock - were chosen for training purposes with the original CNN structure and published weights (Figure 3.58). Testing this image through the adopted model, the saliency prediction map below is the outcome:

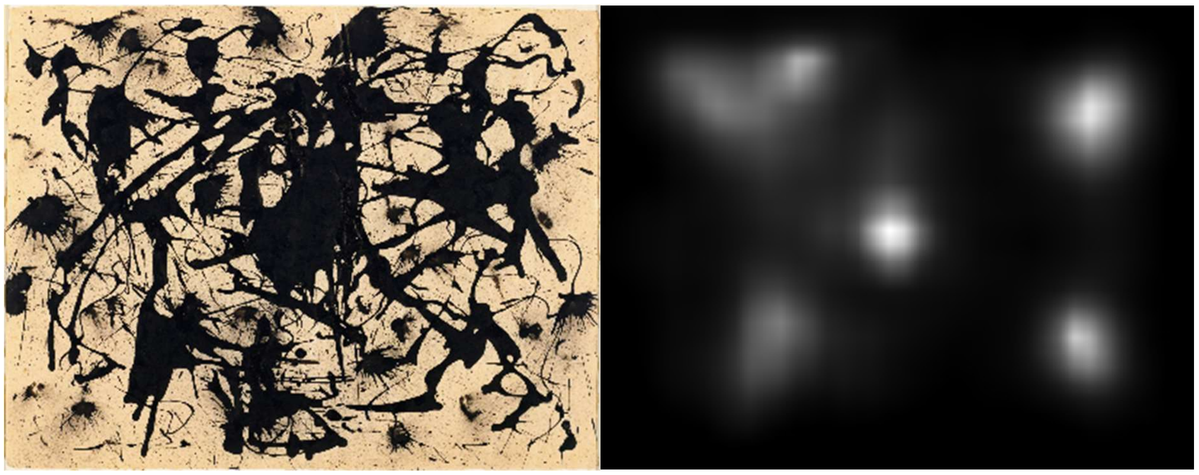


Figure 3.58 (Left) Painting by Jackson Pollock, (Right) Saliency map for this painting, Painting Accessed at: [57]

Even though this is an abstract painting, it could be seen that the adopted model finds the most distinct blobs in the image as features which could cause saliency. These are the big prominent black brush strokes in the painting. Now, if this painting is replaced with a more abstract painting (Figure 3.59) with no apparent patterns (almost noise-like) it could be see what the adopted model would act like when encountering noise. For this purpose, another painting inspired by Jackson Pollock was chosen:

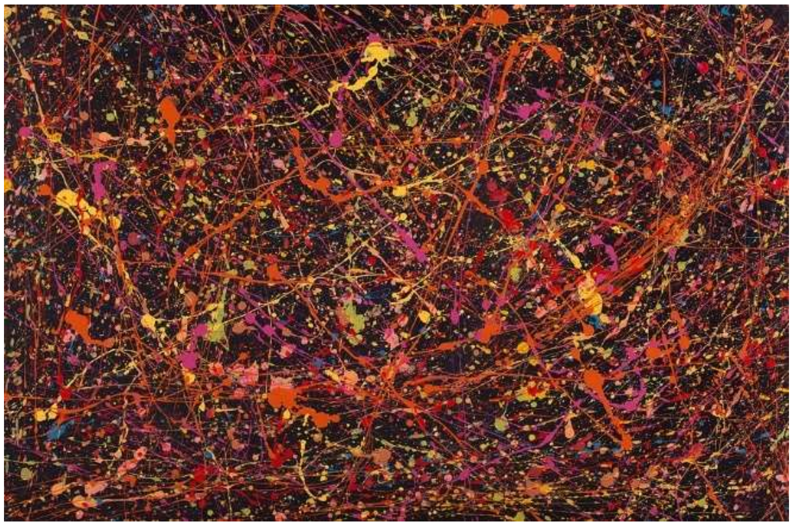


Figure 3.59 Painting inspired by Jackson Pollock, Accessed at: [58]

When tested through the adopted model, result is (Figure 3.60):

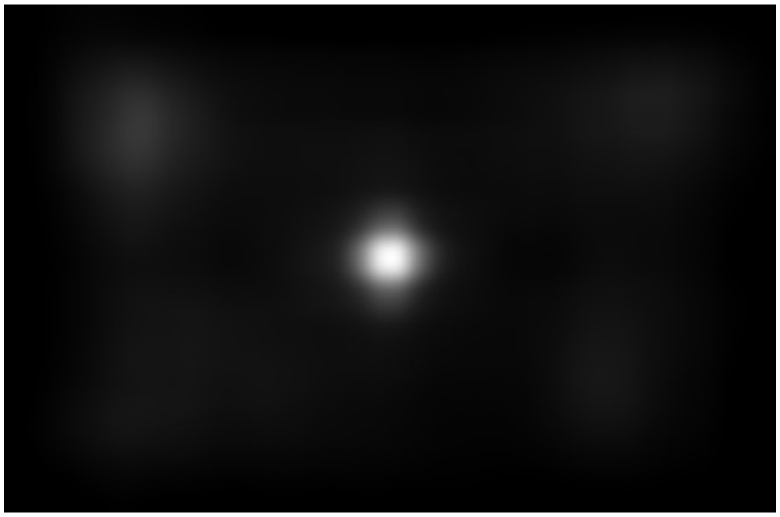


Figure 3.60 Saliency map for painting above (Figure 3.59)

This is very interesting since it shows the substantial centre bias in the adopted model. This centre bias is due to the learned priors included in the adopted model, using Gaussian parameters.

Finally, SAM-VGG was tested on Tsukuba set image (right image) to test its performance and to assess the characteristics of its output (Figure 3.61).

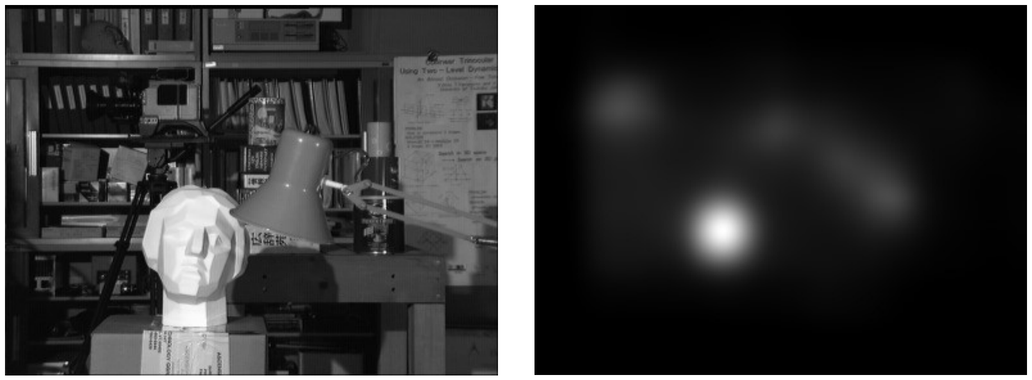


Figure 3.61 (Left) Tasukuba steroe set right image (Right) Saliency map for this image

As could be seen above, the main saliency region is determined properly, with the head being the center of interest. There are a few less salient spots selected as well, which relate to the lamp and the tapes on the shelves. This leads to a few points of concern:

1. Various regions of saliency were detected, but only one region of interest is desired for the motor commands
2. How would only one point be chosen from the region of interest for motor command functionality?
3. SAM-VGG runtime is not fast enough for the proposed system purposes. It takes upwards of 40 seconds to run one forward pass on an image. The max time considered would be 1 second and as such, this algorithm with the present architecture is not usable in the proposed system.

Due to the third reason, it was decided to experiment with other possible CNNs which would perform faster than SAM-VGG. The model of choice is Deeplab implemented through Tensorflow. Due to reasons explained in Chapter 2, this model performs much better than SAM-VGG in terms of runtime. In this instance, “mobilenetv2” backbone was selected for the implementation since the main objective was real time operation of the algorithm (.vs accuracy).

3.6.2.3.2 DeepLabv3 Integration

DeepLabv3 produces a semantic segmentation output which then needs to be further processed for motor command purposes. Below is a screenshot of what DeepLabv3 produces (Figure 3.62) when implemented on the previously proposed system design:

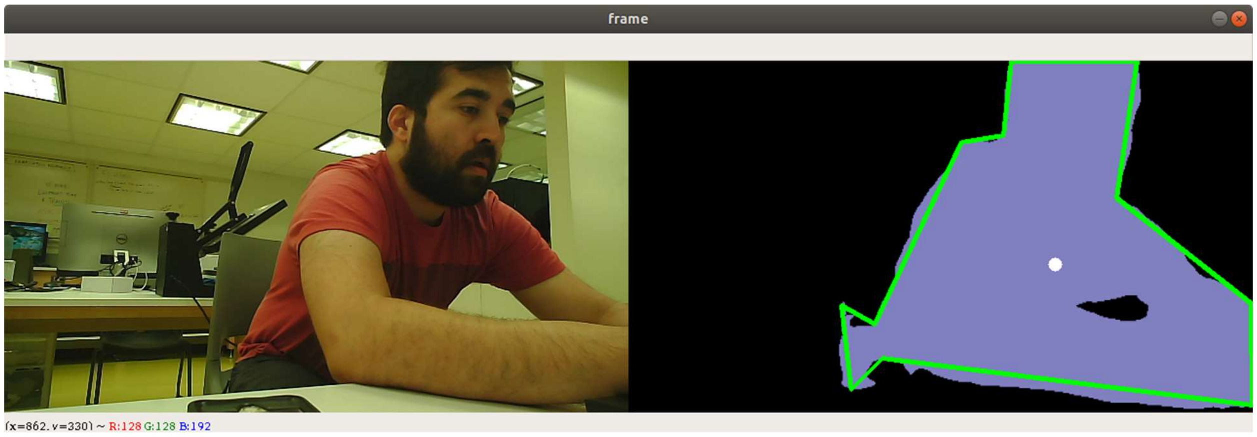
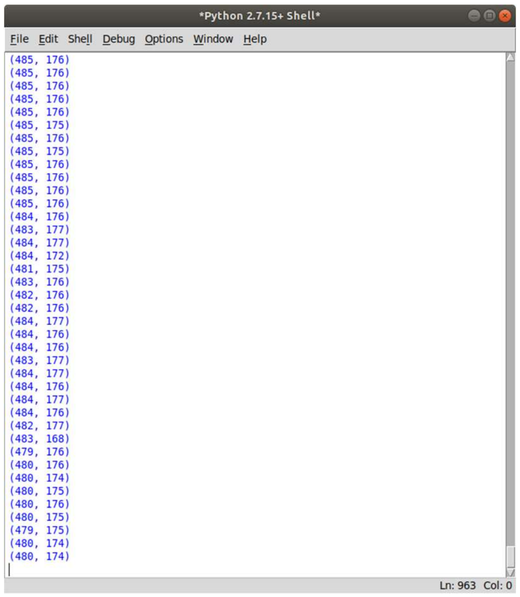


Figure 3.62 (Left) incoming frame (Right) DeepLab output + segmentation outline + Moments

It should be noted that the specific weights provided are for training on PASCALVOC which is composed of everyday images lots of which contain human figures. As such, weights provided by training done on this dataset is robust for detection of humans.



```
(485, 176)
(485, 176)
(485, 176)
(485, 176)
(485, 176)
(485, 175)
(485, 176)
(485, 175)
(485, 176)
(485, 176)
(485, 176)
(485, 176)
(484, 176)
(483, 177)
(484, 177)
(484, 172)
(481, 175)
(483, 176)
(482, 176)
(482, 176)
(484, 177)
(484, 176)
(484, 176)
(483, 177)
(484, 177)
(484, 176)
(484, 177)
(484, 176)
(482, 177)
(483, 168)
(479, 176)
(480, 176)
(480, 174)
(480, 175)
(480, 176)
(480, 175)
(479, 175)
(480, 174)
(480, 174)
```

Figure 3.63 Centeroid values for Moments commands

As could be seen above, Weights assigned to DeepLabv3, Identify human figure segmentations in the experimental setup. The human figure outlines (Figure 3.62) then need to be determined in order to be able to identify a center like location for motor commands. To this end, OpenCV functions are used. First segmentation contours are determined using OpenCV functions.

After the contours for human segmentations were determined, the centroid of the human figure segmentation was found using the Moments function in OpenCV (cv2.moments(c)). This function calculates the arithmetic mean of all the pixel locations in the segmentation shape.

$$c = \frac{1}{n}\sum_{i=1}^{n} x_i \qquad (3.7)$$

This moments point was then used as the center of foveation, for motor commands (Figure 3.64).

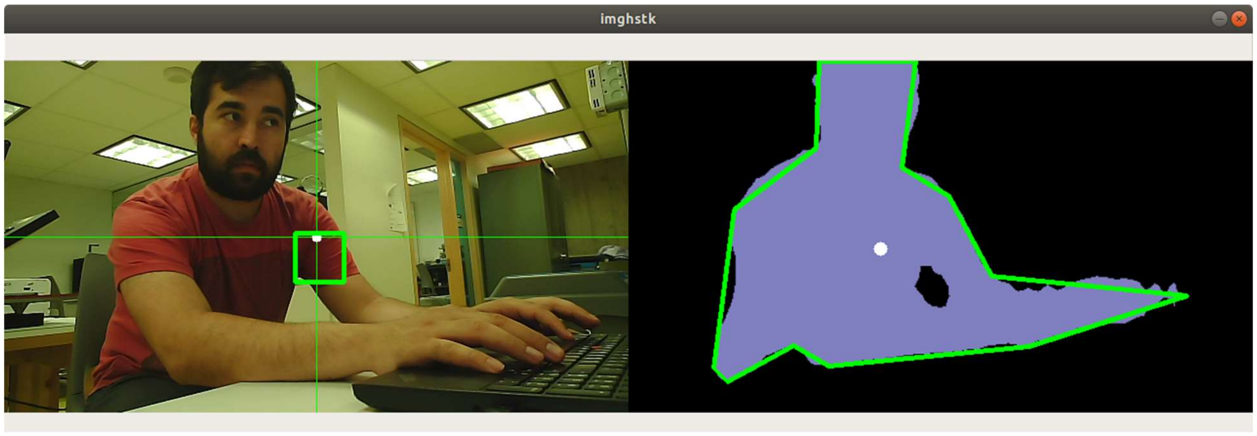


Figure 3.64 Apparatus output (Left) follower eye verged (right) master eye foveated

3.6.3 Motor Positioning Update Design

This brings us to the topic of driving motor commands from calculated target locations. In the previous section, possible solutions used in this study to drive target locations are discussed. These target locations now need to be transformed into motor commands which then would physically move master camera's image center to the calculated target location.

Initially, attempts were made to design an algorithm which could reorient the camera in a singles-iteration command. This was unsuccessful due to the fact that image feed gets updated at a fast rate and big sum updates in motor location result in drastic overshoot.

Next, an iterative approach was taken (Figure 3.65). In this approach, a step size was chosen. Considering the location of the target in the visual frame and its relation to the present center on the image location, a motor command was executed adding or subtracting a step per iteration to Pan and Tilt servo horn position. Servo 3 (Pan servo) and 1 (Tilt servo) are used for the foveation process. Servo2 (Verge servo) is used for the vergence process. Fine tuning of the step size was conducted in order to get optimal results.

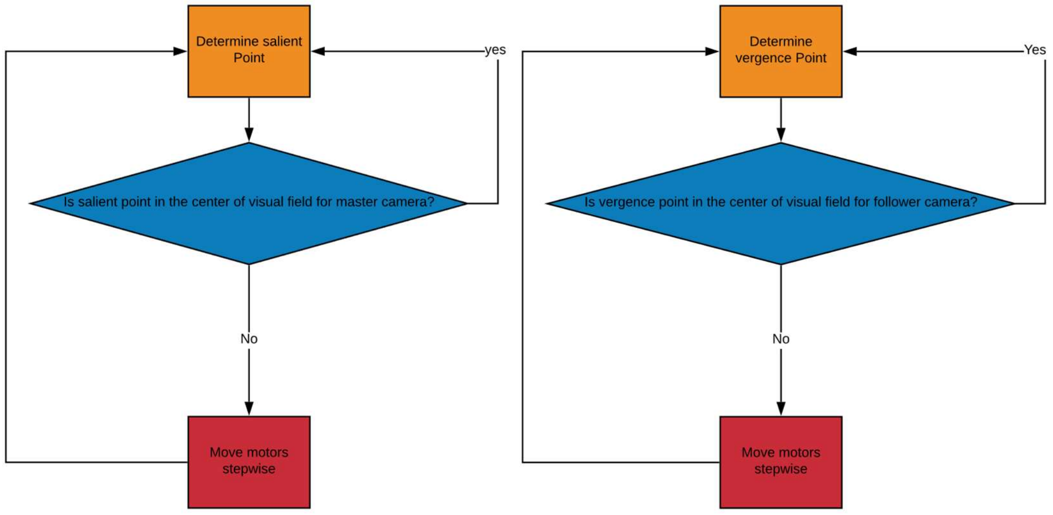

Determine salient Point
yes
Is salient point in the center of visual field for master camera?
No
Move motors stepwise
Determine vergence Point
Yes
Is vergence point in the center of visual field for follower camera?
No
Move motors stepwise

Figure 3.65 Foveation and Vergence process

3.7 Visual Delivery Design

Though visual delivery has been beyond the scope of this thesis project, it is useful to consider some possibilities, for future work. The following is a brief discussion of thoughts on visual delivery.

First, implementations of delivery for Virtual Reality (VR) environments is a possibility, which could be used for teleoperation. This implementation would be more feasible than the 3D point cloud reconstruction and should be attempted first. Since the visual calculations are conducted in Python, a Python library named “ZMQ Python” could be used to transfer the final verged right eye and left eye video streams to the VR host computer over the network for VR delivery and teleoperation purposes.

The ZMQ library was created by Jeff Bass, which is an efficient way of parsing and sending data through the network. This library is available at: https://github.com/jeffbass/imagezmq. One of the issues faced here would be dealing with latency. The latency needs to be minimized for any sort of closed loop teleoperation design to be of use.

After the two video streams are transferred to the host computer, a few different methods could be used to broadcast these two streams to a VR environment. Either “open VR” which is a Python library could be used or Steam VR which is a dedicated VR platform could be used.

The second objective in terms of delivery would be depth map reconstruction. This is challenging due to the fact that the two cameras are in constant movement. Even though challenging, depth map recreation is a possibility since angular orientations of the two cameras are known due to known angular orientations of servo motors responsible for setting the pan and tilt of two cameras. This has been one of the reasons for the selection of digital servo motors due to their capability for closed loop position feedback. In this way, an accurate position location for each servo is obtained.

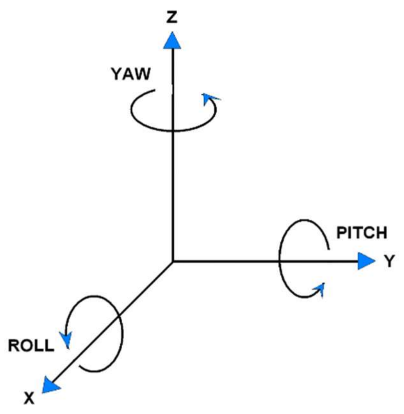


Figure 3.66 Yaw or pan / Pitch or tilt

Knowing the pan and tilt (refer to Figure 3.66 for pan and tilt definition) for left and right cameras, it is possible to process depth map registration and reconstruction, if extrinsic and intrinsic parameters are known. Intrinsic parameters have to do with focal length (f_x, f_y) and optical center (c_x, c_y). The intrinsic parameters have to do with individual camera factors such as distortion. Extrinsic parameters have to do with translation and rotation between the two cameras. The Intrinsic parameters give us the camera matrix:

$$camera\ matrix = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \tag{3.8}$$

Extrinsic parameters help us with determining the Essential matrix. Motor orientations give us the elements of R matrix and baseline provide us with the translation matrix elements ($\mathrm{T_0}$, $\mathrm{T_1}$, $\mathrm{T_2}$). Essential matrix then gives us the ability for 3D scene reconstruction.

$$\mathrm{E} = \begin{bmatrix} 0 & -\mathrm{T_2} & \mathrm{T_1} \\ \mathrm{T_2} & 0 & -\mathrm{T_0} \\ -\mathrm{T_1} & \mathrm{T_0} & 0 \end{bmatrix} * \mathrm{R} \tag{3.9}$$

Where $\mathrm{T_0}$, $\mathrm{T_1}$ and $\mathrm{T_2}$ are the components of the translation matrix (T) between the two cameras and R the rotations matrix between the two cameras.

$$R = \begin{bmatrix} \mathrm{r_{11}} & \mathrm{r_{12}} & \mathrm{r_{13}} \\ \mathrm{r_{21}} & \mathrm{r_{22}} & \mathrm{r_{23}} \\ \mathrm{r_{31}} & \mathrm{r_{32}} & \mathrm{r_{33}} \end{bmatrix} \tag{3.10}$$

Another possible approach worth exploring would be the use of StereoNet [59]. This is a recently developed end-to-end stereo vision CNN. Further investigation is needed to determine if StereoNet could be of use for this specific instance.

Chapter 4 . Experiments and Results

After the design of the prototype was completed, steps were taken to test and benchmark performance of the designed system. Initially, creation of a validation dataset using human vision with the Eyelink 1000 device was considered (Figure 4.1). This device is used in the field of experimental psychology for eye related measurements and monitoring.

Figure 4.1 Eyelink 1000 device, Accessed at: [60]

The dataset would have included saccade latencies plus saccade execution times for the human visual system for saccadic foveation on objects in different locations in the field of view. After some research and literature review, this idea was discarded due to the realization that saccadic latency remains relatively constant regardless of the angular distance of the target location for foveation [9] plus that saccade execution times are clearly identified for various angular displacements (which are between 0 to 100ms / refer to figure 2.6).

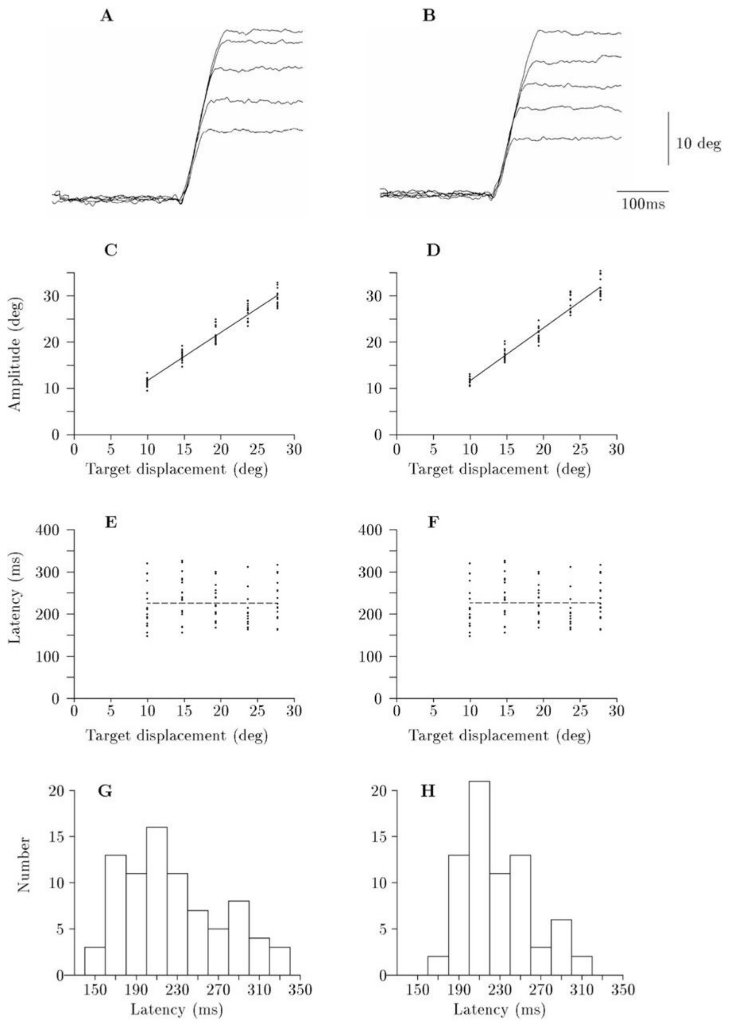


Figure 4.2 Comparison of saccades in response to a stationary target of normal intensity (left hand column) and of a reduced intensity red target (right hand column) for direct viewing in the same subject, Accessed at: [9]

As pictured above (Figure 4.2), as the angular distance increases the amplitude of the saccade increases accordingly, but the latency stays the same. The logic behind this is explained in detail by Darrian el al [9]. As it can be seen the latencies for various angular locations are someplace around 200ms. Saccade execution adds 0 to 100ms to latency bringing the sum to someplace between 200 to 300ms.

Knowing this fact, efforts were made to record the timing for foveation for various angular displacements. In order for a precise setup of these angles and set up of targets on said angle locations a total station is used (Figure 4.3). The total station used is Leica TS12. This instrument is a commonly used instrument in the land surveying industry. One of the advantages of total stations is their accuracy in terms of measuring angles and distances. For this project, only angles were measured. This instrument has a

5"degrees angular precision. The instrument used by us most likely does not have such an accuracy since it hasn't been calibrated by the manufacturer in a long time.

Targets were set for angular displacement of 10, 20 and 30 degrees in the calibrated environment in both directions on the horizontal axis. As could be seen below, every 5 degrees in the horizontal axis was marked and the 3 rows are 10 degrees apart in the vertical direction. After the target setting was concluded, the Total Station was removed from the tripod and was replaced by the designed apparatus, so that the master camera occupies the same focal centre as the Total Station. In order to achieve this centering, measurements with a tape measure in conjunction with markings on the wooden platform (Figure 4.4) were utilized to ensure the centering of the camera in x, y, z direction. It should be noted that this method of centering is accurate to a certain degree and for a more accurate centering use of a second total station set on an independent control point is recommended.

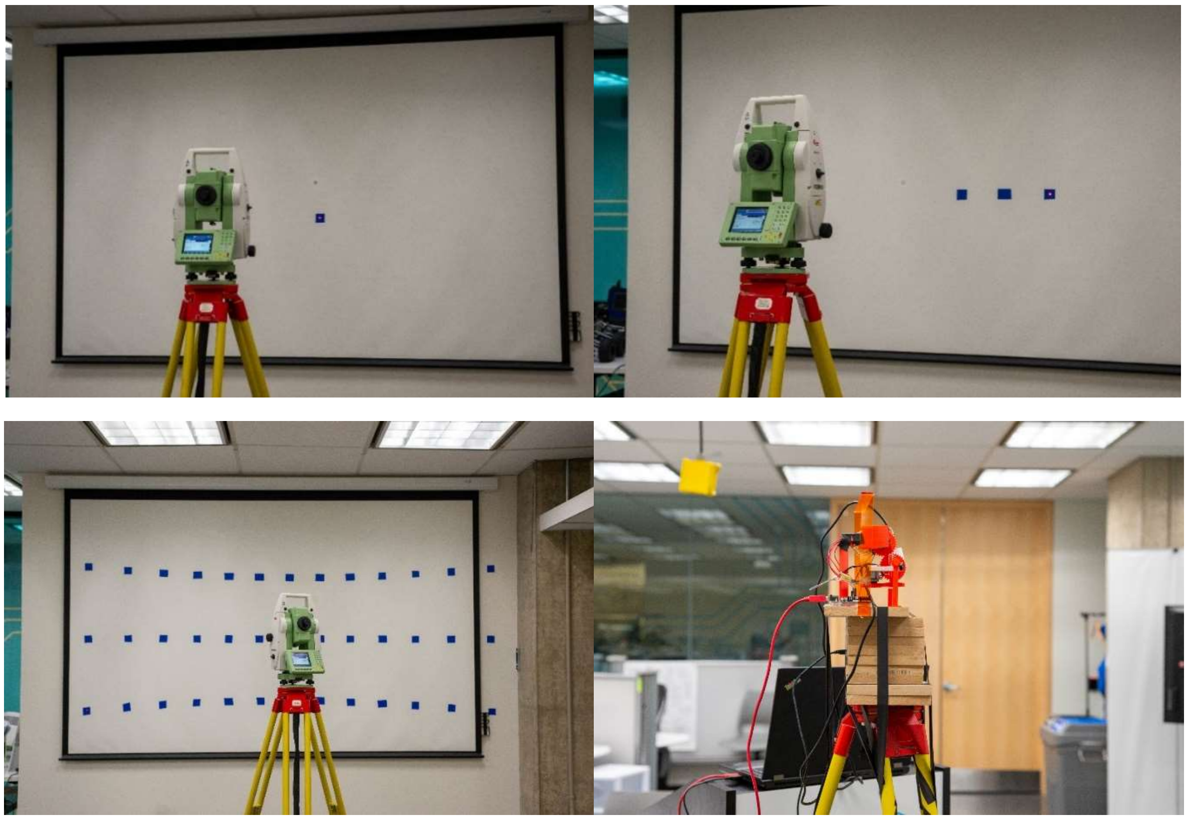


Figure 4.3 Test setup seen in sequence, targets set in sequence starting from the center target, lower righthand picture shows swapping of the Total station with the designed appratus

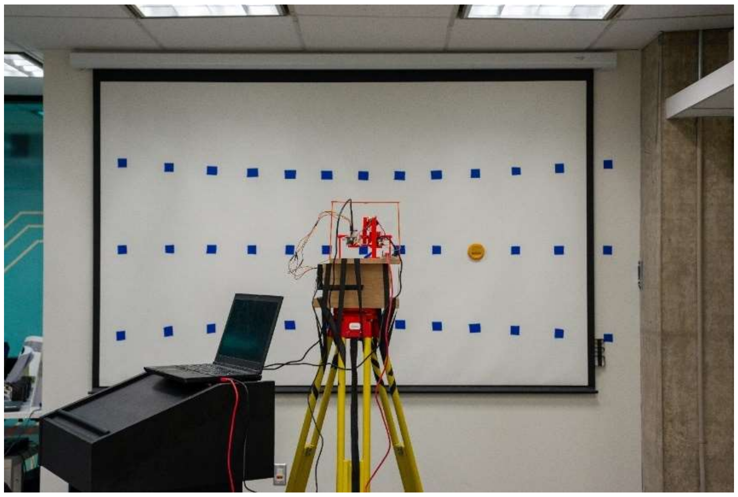


Figure 4.4 Designed appratus observing a test target in the test setting

4.1 Experimental Setup Design

A novel method of benchmarking was considered in this thesis project. It was speculated that performance of the designed system would not be on par with the human visual system and as such the measurement of saccade times on their own would not be of much value. However, recordings of trends in the saccadic camera displacements vs time would be of value in assessing and comparing the performance of the proposed system to the human visual system. Produced graphs would resemble graphs created in the field of physiology, seen in Chapter 2.

4.2 Experimental Apparatus Design

Inertial Measurement Units (IMU) are used for benchmarking. IMU microprocessors have the ability to provide us with the orientation of a given object using a set of data recorded by the IMU. The IMU unit used in this instance is MPU9250. MPU9250 is a microchip designed to work with microcontrollers such as Arduino and Raspberry Pi. This microchip is designed by Invensense. The chip is a 9 axis motion processing unit [61]. This unit is used in cellphones and tablets. This chip incorporates an accelerometer, gyroscope and compass on board.

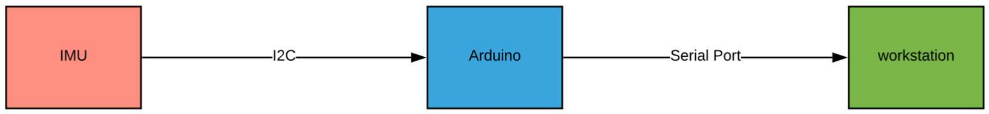


Figure 4.5

Benchmarking setup has consisted of a total station and the mentioned IMU chip, Arduino Uno and Thinkpad P51 Workstation (Figure 4.5). IMU circuitry is connected to Arduino Uno. Raw data from MPU9250 is broadcasted to Arduino. Arduino then communicates with Workstation running Ubuntu 18.04 over a serial port. Serial plotter (CoolWire) is used to capture and record the raw data from IMU.

In this case, only the data from accelerometer and gyroscope was used in calculations (6DOF). Ultimately, the data from magnetometer needs to be used as well for the best orientation results. In this case, a special type of Arduino board called “Stem Tera” is used. This board combines a breadboard and an Arduino all at one place and as such simplifies the physical build.

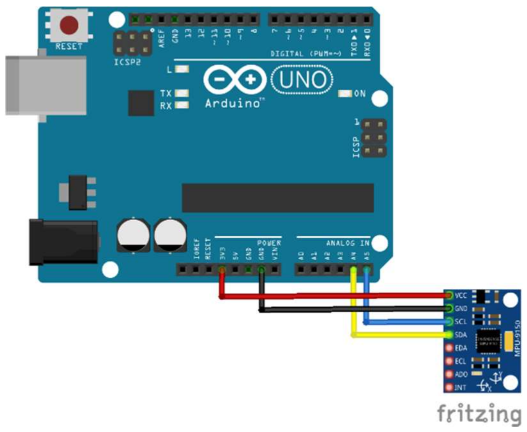
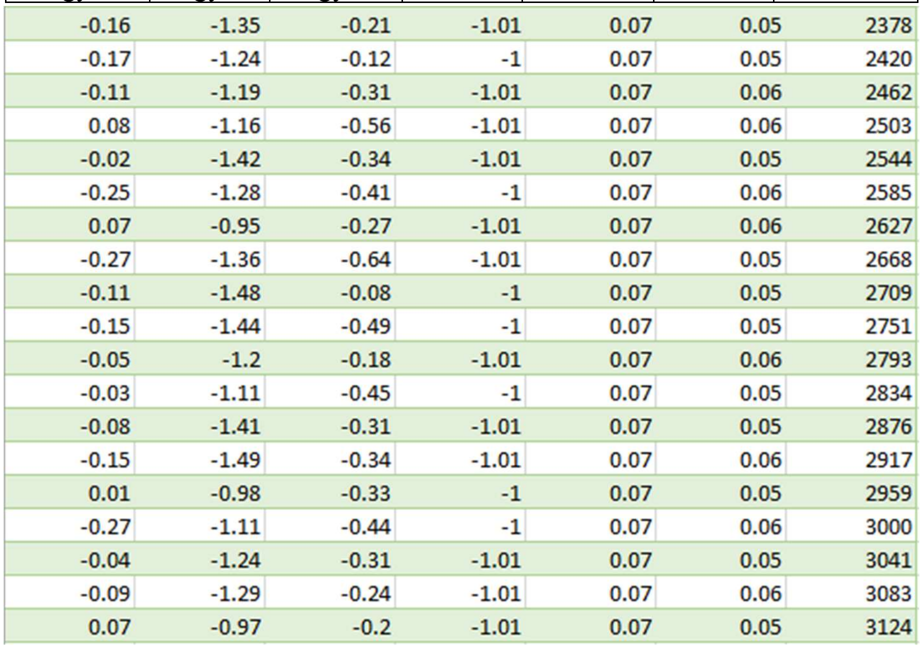


Figure 4.6 MPU9250 wired to Arduino UNO

Communications with the IMU circuit were conducted using the I^2C protocol through the SDA and SCL ports (Figure 4.6). I^2C(inter-integrated circuit) is a communication protocol designed by Philips Semiconductors in 1982. This protocol acts on SCL and SDA pins on the circuit board. Clock data is transferred through SDA connection and raw data (accelerometer, gyroscope and compass) is transferred through SCL.

The data transfer interval from the Arduino Board to Workstation is approximately each 40 milliseconds. This is derived from the raw data received from the IMU circuit board seen in below (Table 4.1). The first three columns are the values of angular velocity from the gyroscope in radians and the second three columns are the acceleration values in gravity units. Note that these sample values are taken at the resting position of the IMU board.



Xgyro	Ygyro	Zgyro	Xacce	Yacce	Zacce	Time
-0.16	-1.35	-0.21	-1.01	0.07	0.05	2378
-0.17	-1.24	-0.12	-1	0.07	0.05	2420
-0.11	-1.19	-0.31	-1.01	0.07	0.06	2462
0.08	-1.16	-0.56	-1.01	0.07	0.06	2503
-0.02	-1.42	-0.34	-1.01	0.07	0.05	2544
-0.25	-1.28	-0.41	-1	0.07	0.06	2585
0.07	-0.95	-0.27	-1.01	0.07	0.06	2627
-0.27	-1.36	-0.64	-1.01	0.07	0.05	2668
-0.11	-1.48	-0.08	-1	0.07	0.05	2709
-0.15	-1.44	-0.49	-1	0.07	0.05	2751
-0.05	-1.2	-0.18	-1.01	0.07	0.06	2793
-0.03	-1.11	-0.45	-1	0.07	0.05	2834
-0.08	-1.41	-0.31	-1.01	0.07	0.05	2876
-0.15	-1.49	-0.34	-1.01	0.07	0.06	2917
0.01	-0.98	-0.33	-1	0.07	0.05	2959
-0.27	-1.11	-0.44	-1	0.07	0.06	3000
-0.04	-1.24	-0.31	-1.01	0.07	0.05	3041
-0.09	-1.29	-0.24	-1.01	0.07	0.06	3083
0.07	-0.97	-0.2	-1.01	0.07	0.05	3124

Table 4.1 Raw readings from MPU9250

4.3 Kalman Filter Review

Data fusion is experimented with, in order to render better orientation results using the raw data from the IMU sensor. To this end, Kalman filter is used for minimization of cumulative error in the observed values from the IMU unit. This ultimately rendered undesirable values in this instance, however. Below a brief description of the formulation used is given:

Accelerometer data (gravitational acceleration with 3DOF) is noisy on its own. Gyroscope data (angular rate of change with 3 DOF) is prone to drift due to cumulative integration of errors in the time domain. Hence, a model that combines these two data streams for a better future pose estimation would be optimal. Steps involved are:

1. using raw accelerometer data (A_x, A_y, A_z) below calculations is obtained:

$$\widehat{\phi_{Acc}} = \arctan\left(\frac{A_y}{\sqrt{A_x^2+A_z^2}}\right) \cdot \frac{180}{\pi} \tag{4.1}$$

$$\widehat{\theta_{Acc}} = \arctan\left(\frac{A_x}{\sqrt{A_y^2+A_z^2}}\right) \cdot \frac{180}{\pi} \tag{4.2}$$

2. using raw gyroscope data (G_x, G_y, G_z) $\dot{\phi}$, $\dot{\theta}$ and $\dot{\psi}$ are calculated. The variables p, q, r are the same as G_x, G_y, G_z. θ and ϕ are grabbed from the calculations in Step 1.

$$\begin{pmatrix}\dot{\phi}\\ \dot{\theta}\\ \dot{\psi}\end{pmatrix} = \begin{pmatrix}1 & \sin\phi\tan\theta & \cos\phi\tan\theta\\ 0 & \cos\phi & -\sin\phi\\ 0 & \sin\phi\sec\theta & \cos\phi\sec\theta\end{pmatrix}\begin{pmatrix}p\\ q\\ r\end{pmatrix} \tag{4.3}$$

3. Once above values are obtained, fusion could be done using Complimentary filter:

$$\widehat{\phi_{t+1}} = (1-\alpha)\cdot\left(\widehat{\phi_t} + \dot{\phi_G}\cdot\Delta t\right) + \alpha\cdot\widehat{\phi_{Acc}} \tag{4.4}$$

4. Or Kalman filter could be utilized for possibly better results, Where $\overrightarrow{x_t}$ is the system's state vector at time t and $\overrightarrow{u_t}$ is the input vector at time t and **A**, **B**, and **C** are the system matrices: **A** relates the current states to the next states, **B** relates inputs to the next states, and **C** relates the system states to the measured states (y is output vector, v and w are respectively process and measurement noise).

$$\overrightarrow{x_{t+1}} = \boldsymbol{A}\cdot\overrightarrow{x_t} + \boldsymbol{B}\cdot\overrightarrow{u_t} + \overrightarrow{w_t} \tag{4.5}$$

$$\overrightarrow{y_{t+1}} = \boldsymbol{C}\cdot\overrightarrow{x_{t+1}} + \overrightarrow{v_{t+1}} \tag{4.6}$$

5. The below matrices would be used in above and following equations. b is gyro bias, hat indicates estimate:

$$\overrightarrow{x_t} = \begin{bmatrix} \widehat{\phi_t} \\ b_{\widehat{\phi_t}} \\ \widehat{\theta_t} \\ b_{\widehat{\theta_t}} \end{bmatrix} \tag{4.7}$$

$$\overrightarrow{u_t} = \begin{bmatrix} \dot{\phi}_{G_t} \\ \dot{\theta}_{Gt} \end{bmatrix} \tag{4.8}$$

$$\overrightarrow{z_t} = \begin{bmatrix} \widehat{\phi_{Acct}} \\ \widehat{\theta_{Acc_t}} \end{bmatrix} \tag{4.9}$$

6. A, B and C in equations of step 4 for this specific instance (Gyrometer, Accelerometer and no magnetometer) are defined as:

$$\overrightarrow{x_{t+1}} = \begin{bmatrix} 1 & -\Delta T & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & -\Delta T \\ 0 & 0 & 0 & 1 \end{bmatrix} \overrightarrow{x_t} + \begin{bmatrix} \Delta T & 0 \\ 0 & 0 \\ 0 & \Delta T \\ 0 & 0 \end{bmatrix} \overrightarrow{u_t} \tag{4.10}$$

$$\overrightarrow{y_{t+1}} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \overrightarrow{x_{t+1}} + \overrightarrow{v_t} \tag{4.11}$$

7. Using the above defined parameters, now the desired recursive Kalman model could be set up as below (P is covariance matrix of errors which is predefined depending on system errors, Q and R process and measurement noise which are set respectively, K is Kalman gain):

$$\overrightarrow{x_{t+1}} = \boldsymbol{A} \cdot \overrightarrow{x_t} + \boldsymbol{B} \cdot \overrightarrow{u_t} \qquad (4.12)$$

$$\boldsymbol{P} = \boldsymbol{A} \cdot \boldsymbol{P} \cdot \boldsymbol{A}^{\boldsymbol{T}} + \boldsymbol{Q} \qquad (4.13)$$

$$\widetilde{y_{t+1}} = \overrightarrow{z_{t+1}} - \boldsymbol{C} \cdot \overrightarrow{x_{t+1}} \qquad (4.14)$$

$$\boldsymbol{S} = \boldsymbol{C} \cdot \boldsymbol{P} \cdot \boldsymbol{C}^{\boldsymbol{T}} + \boldsymbol{R} \qquad (4.15)$$

$$\boldsymbol{K} = \boldsymbol{P} \cdot \boldsymbol{C}^{\boldsymbol{T}} \cdot \boldsymbol{S}^{-\boldsymbol{1}} \qquad (4.16)$$

$$\overrightarrow{x_{t+1}} = \overrightarrow{x_{t+1}} + \boldsymbol{K} \cdot \widetilde{y_{t+1}} \qquad (4.17)$$

$$\boldsymbol{P} = (\boldsymbol{I} - \boldsymbol{K} \cdot \boldsymbol{C}) \cdot \boldsymbol{P} \qquad (4.18)$$

8. Keep in mind that estate estimate (x) should be initiated (Initial value needed, e.g. 0,0,0) and then it gets updated (desired values) using the Kalman model above present in the relevant code.

4.4 Results

In order to capture the needed data, an IMU unit was mounted in a few different positions on the master camera platform to determine which position is most suitable for readings (Figure 4.7, 4.8). This was done in order to capture data which could be processed into something meaningful.

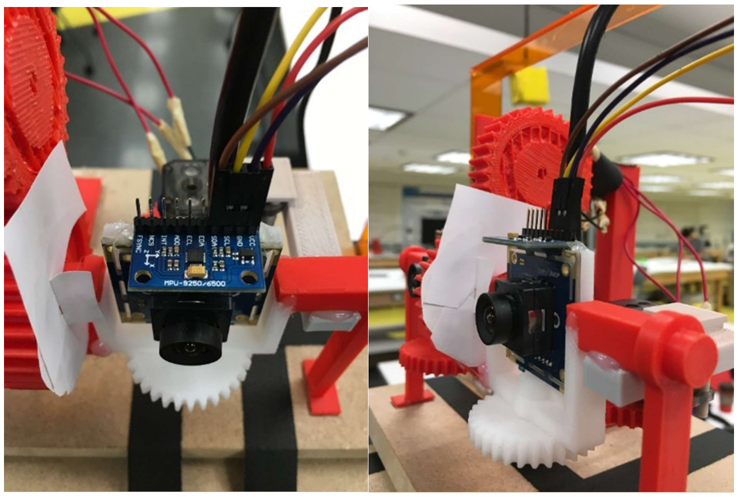


Figure 4.7 First Setup for IMU

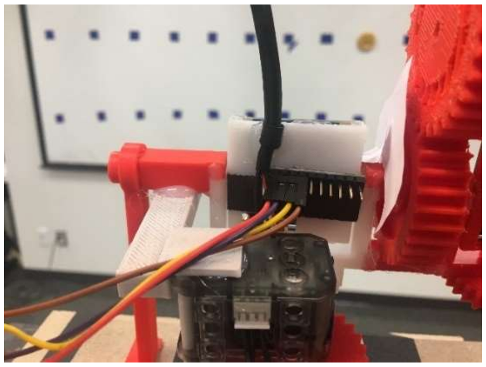
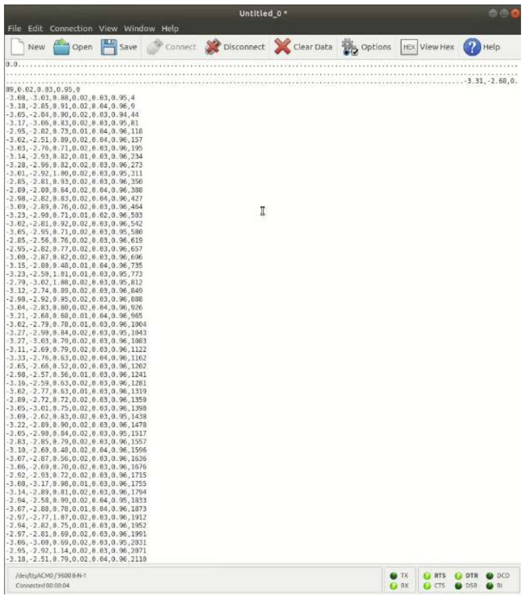


Figure 4.8 Third Setup for IMU

In order to record the data, first recording on Cool Wire was commenced (Figure 4.9) and then the operation of the apparatus was commenced. This is the reason for initial null value readings in the following graphs. Before this step though, for every single reading, positioning of the camera centre was adjusted to the centre point on the testing screen.



```
0.0.....................................................................
........................................................................
.................................................................-3.31,-2.68,0.
89,0.02,0.03,0.95,0
-3.00,-3.03,0.88,0.02,0.03,0.95,4
-3.18,-2.85,0.91,0.02,0.04,0.96,9
-3.05,-2.84,0.90,0.02,0.03,0.94,44
-3.17,-3.06,0.83,0.02,0.03,0.95,81
-2.95,-2.82,0.73,0.01,0.04,0.96,118
-3.02,-2.51,0.89,0.02,0.04,0.96,157
-3.03,-2.76,0.71,0.02,0.03,0.96,195
-3.14,-2.93,0.82,0.01,0.03,0.96,234
-3.28,-2.96,0.82,0.02,0.03,0.96,273
-3.01,-2.92,1.00,0.02,0.03,0.95,311
-2.85,-2.81,0.93,0.02,0.03,0.96,350
-2.89,-2.80,0.64,0.02,0.04,0.96,388
-2.98,-2.82,0.83,0.02,0.04,0.96,427
-3.09,-2.89,0.76,0.02,0.03,0.96,464
-3.23,-2.98,0.71,0.01,0.02,0.96,503
-3.02,-2.81,0.92,0.02,0.03,0.96,542
-3.05,-2.95,0.71,0.02,0.03,0.95,580
-2.85,-2.56,0.76,0.02,0.03,0.96,619
-2.95,-2.82,0.77,0.02,0.03,0.96,657
-3.00,-2.87,0.82,0.02,0.03,0.96,696
-3.15,-2.80,0.48,0.01,0.04,0.96,735
-3.23,-2.50,1.01,0.01,0.03,0.95,773
-2.79,-3.02,1.08,0.02,0.03,0.95,812
-3.12,-2.74,0.89,0.02,0.03,0.96,849
-2.98,-2.92,0.95,0.02,0.03,0.96,888
-3.04,-2.83,0.80,0.02,0.04,0.96,926
-3.21,-2.68,0.68,0.01,0.04,0.96,965
-3.02,-2.79,0.78,0.01,0.03,0.96,1004
-3.27,-2.90,0.84,0.02,0.03,0.95,1043
-3.27,-3.03,0.79,0.02,0.03,0.96,1083
-3.11,-2.69,0.79,0.02,0.03,0.96,1122
-3.33,-2.76,0.63,0.02,0.04,0.96,1162
-2.65,-2.66,0.52,0.02,0.03,0.96,1202
-2.98,-2.57,0.56,0.01,0.03,0.96,1241
-3.16,-2.59,0.63,0.02,0.03,0.96,1281
-3.02,-2.77,0.63,0.01,0.03,0.96,1319
-2.89,-2.72,0.72,0.02,0.03,0.96,1359
-3.05,-3.01,0.75,0.02,0.03,0.96,1398
-3.09,-2.62,0.83,0.02,0.03,0.95,1438
-3.22,-2.89,0.90,0.02,0.03,0.96,1478
-3.05,-2.98,0.84,0.02,0.03,0.95,1517
-2.83,-2.85,0.79,0.02,0.03,0.96,1557
-3.10,-2.60,0.48,0.02,0.04,0.96,1596
-3.07,-2.87,0.56,0.02,0.03,0.96,1636
-3.06,-2.69,0.70,0.02,0.03,0.96,1676
-2.92,-2.93,0.72,0.02,0.03,0.96,1715
-3.08,-3.17,0.98,0.01,0.03,0.96,1755
-3.14,-2.89,0.81,0.02,0.03,0.96,1794
-2.94,-2.58,0.99,0.02,0.04,0.95,1833
-3.07,-2.88,0.78,0.01,0.04,0.96,1873
-2.97,-2.77,1.07,0.02,0.03,0.96,1912
-2.94,-2.82,0.75,0.01,0.03,0.96,1952
-2.97,-2.81,0.69,0.02,0.03,0.96,1991
-3.06,-3.08,0.69,0.02,0.03,0.95,2031
-2.95,-2.92,1.14,0.02,0.03,0.96,2071
-3.18,-2.51,0.79,0.02,0.04,0.96,2110
```

```
/dev/ttyACM0 / 9600 8-N-1
Connected 00:00:04
```

Figure 4.9 Cool Wire Ubuntu Interface

In Cool Wire, raw data is recorded to a .txt file. This data is then imported into MATLAB for processing in order to drive angular values and to then to fuse the data through Kalman filter. Code used in MATLAB for this purpose is courtesy of Peter Salmony available at: https://github.com/pms67/Attitude-Estimation.

It was observed that data recorded from the accelerometer is too noisy to have any meaningful value. This is due to the fact that in the proposed setup there is no meaningful gravitational acceleration readings since majority of movement takes place around the gravitational force axis [62], and due to the fact that only 6 DOF were used in the Kalman formulation instead of 9 DOF, incorporating the magnetometer. Incorporation of the magnetometer could make a difference. However, data recorded from the gyroscope produced meaningful values.

Since accelerometer produced data is null, accordingly data produced by Complimentary filter and Kalman filter is null as well. Regardless, outputs from Complimentary and Kalman filters are shown in graphs for reference. Accelerometer data is not shown since it's aesthetically too distracting.

In order to acquire the measurements needed with the system, a circular shape with a specific color was used. Previously designed circle finding foveation algorithm was used for this purpose. The circular shape was set at angular distance location desired. Foveation algorithm was initiated by initiation of system's operations and the system behavior was recorded using the IMU. Foveation movement patterns and time is then measured using the outputs of this algorithm, processed through MATLAB.

In the example below center and target are shown as seen through the system's master camera. In this instance, a target is set at 15 degrees horizontal and 10 degrees vertical upwards (Figure 4.10).

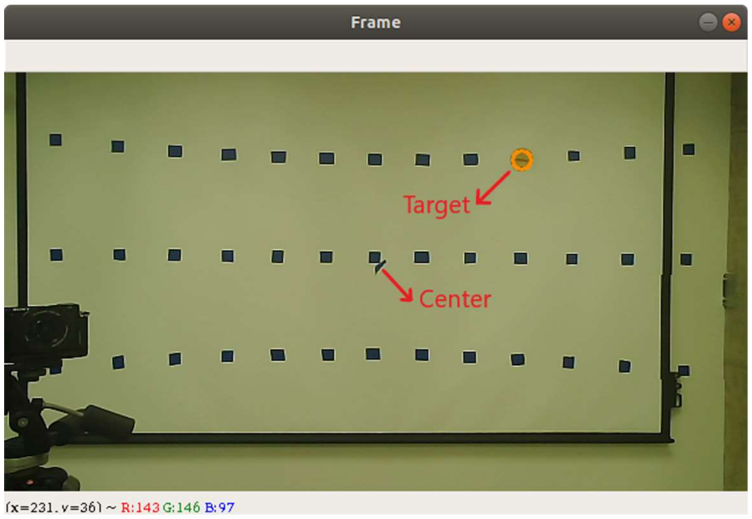


Figure 4.10 Experimental setup as seen from system's master camera

Using the methodology above, IMU readings were made for various locations. In the first set of experiments, readings were made for five-degree horizontal steps and relating vertical steps. In the later sets of experiments, readings were made for every 10 degrees horizontal steps and relating vertical steps. 10 degrees upwards and 10 degrees downward at the centre were recorded as well. Below, mask output for the target is shown (Figure 4.11). Colour range had to be experimented with in order to get the best results for target location (As explained in Chapter 3).

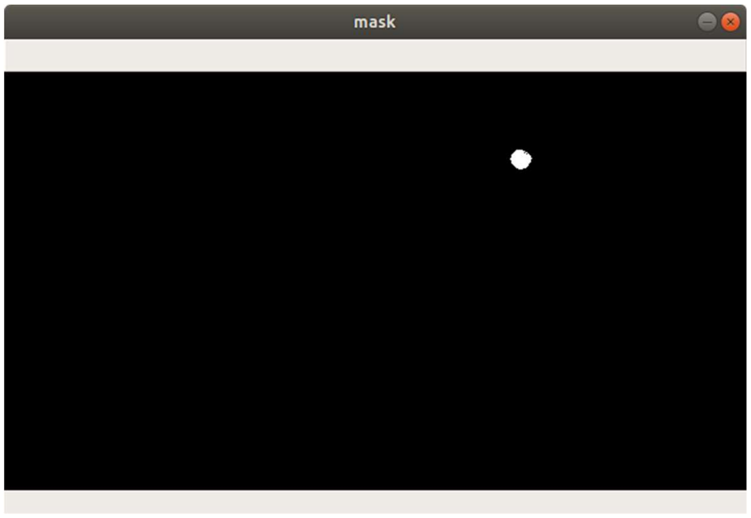


Figure 4.11 colour threshold output for the above target location (15 deg horizontal, 10 degrees upwards vertical)

Below graphs for system movement patterns for different target locations are provided. Target locations are provided in the figure captions:

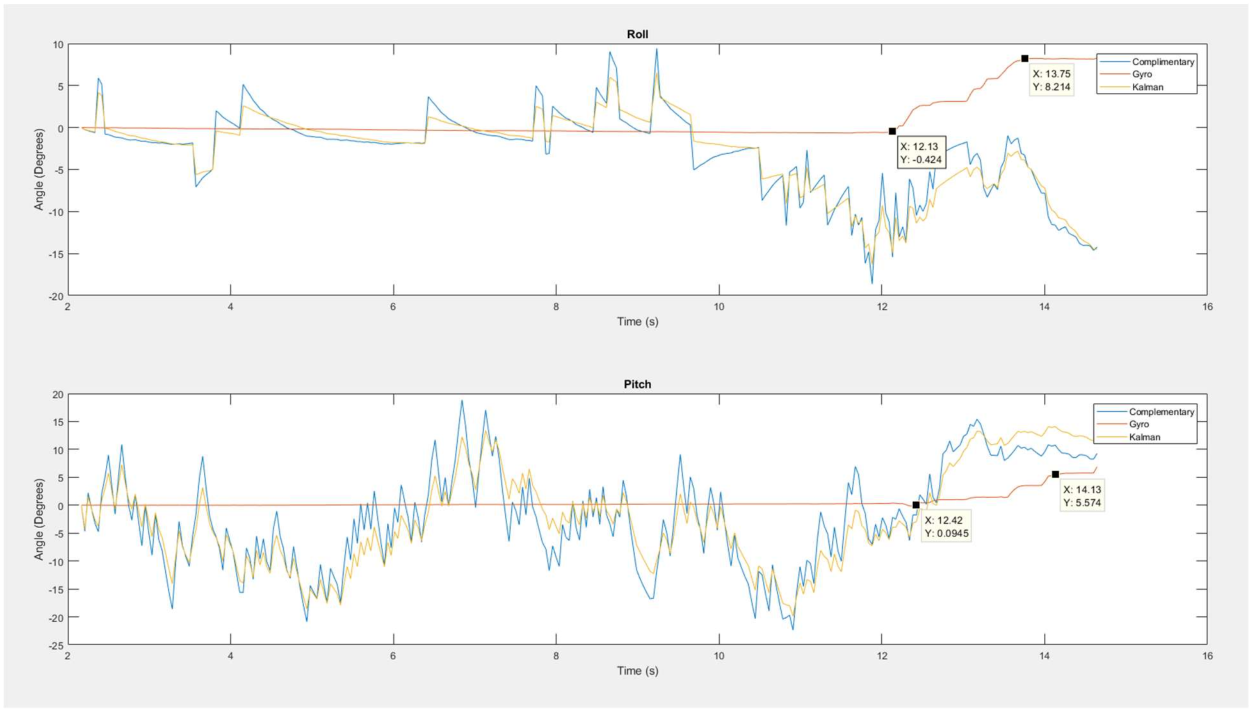


Figure 4.12 Processed IMU data (10 degrees horizontal / 10 degrees downward vertical), no accelerometer data shown

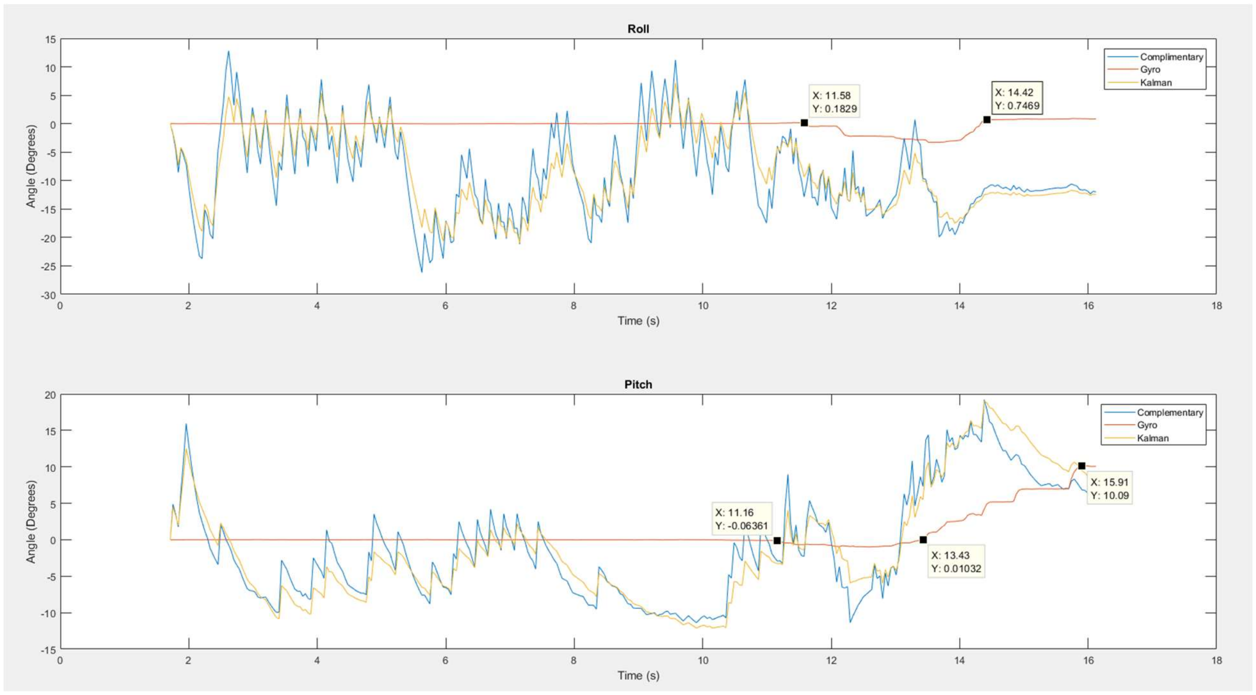


Figure 4.13 Processed IMU data (0 degrees horizontal / 10 degrees downward vertical)

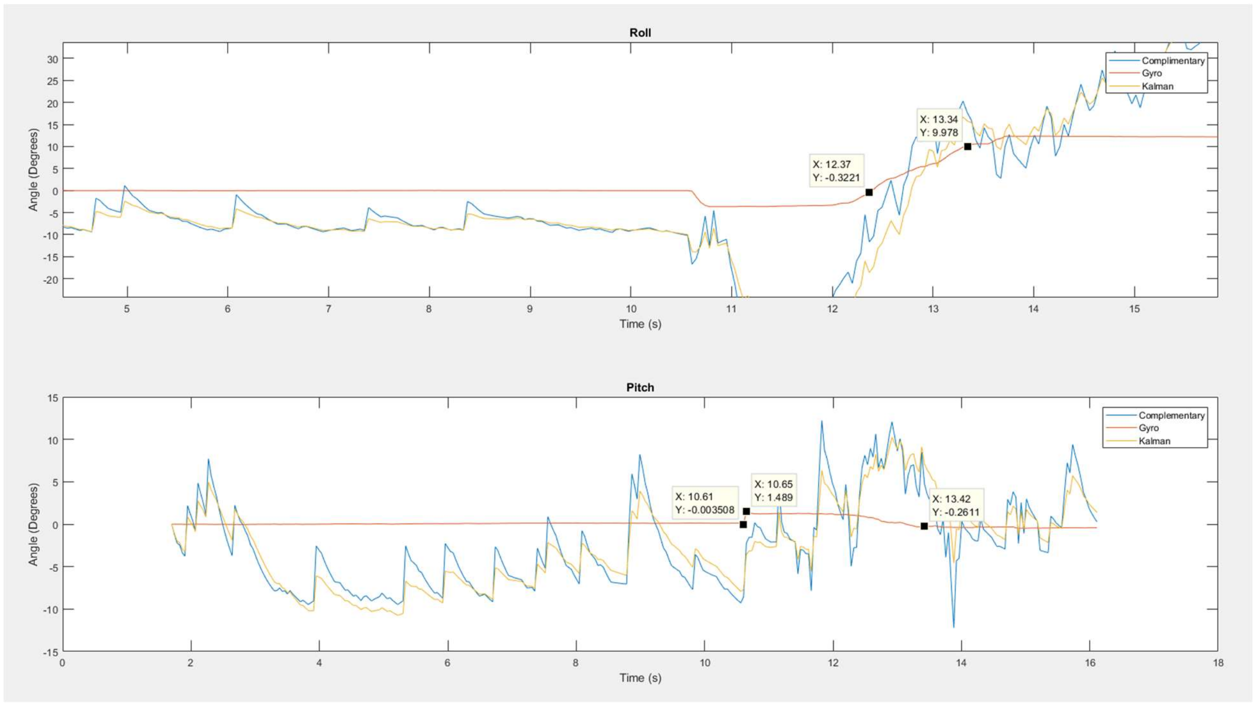


Figure 4.14 Processed IMU data (10 degrees horizontal / 0 degrees vertical)

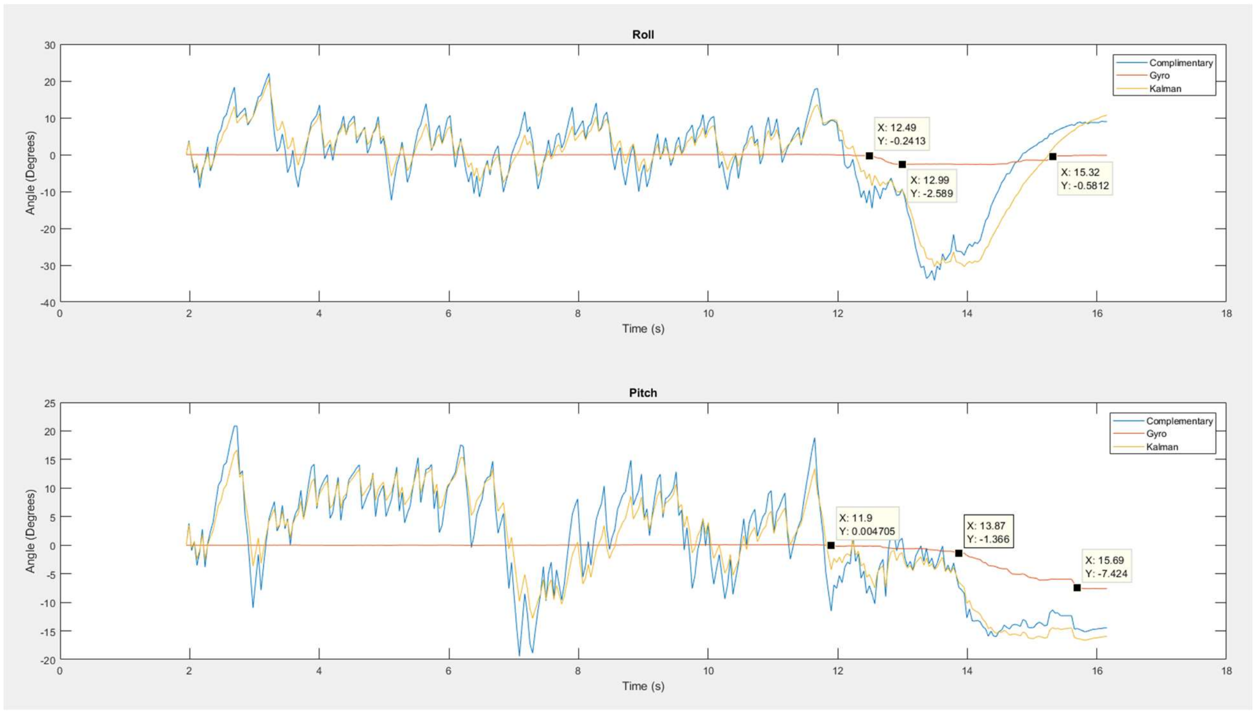


Figure 4.15 Processed IMU data (0 degrees horizontal / 10 degrees upward vertical)

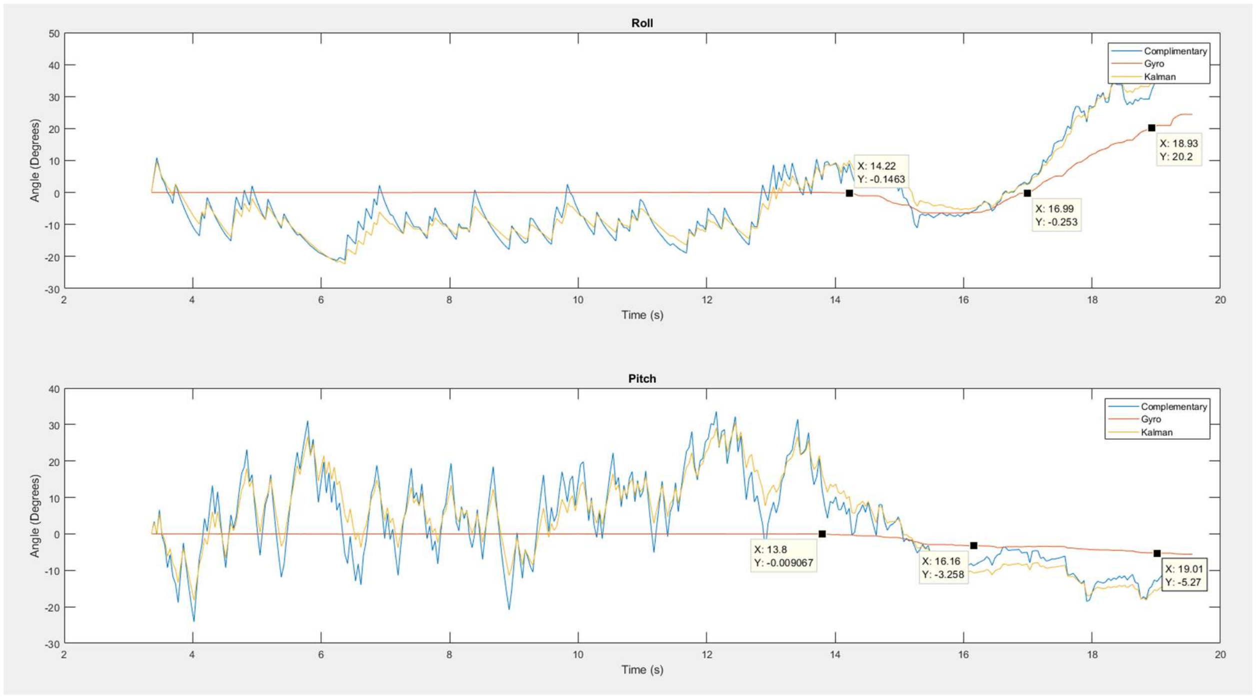


Figure 4.16 Processed IMU data (20 degrees horizontal / 0 degrees vertical)

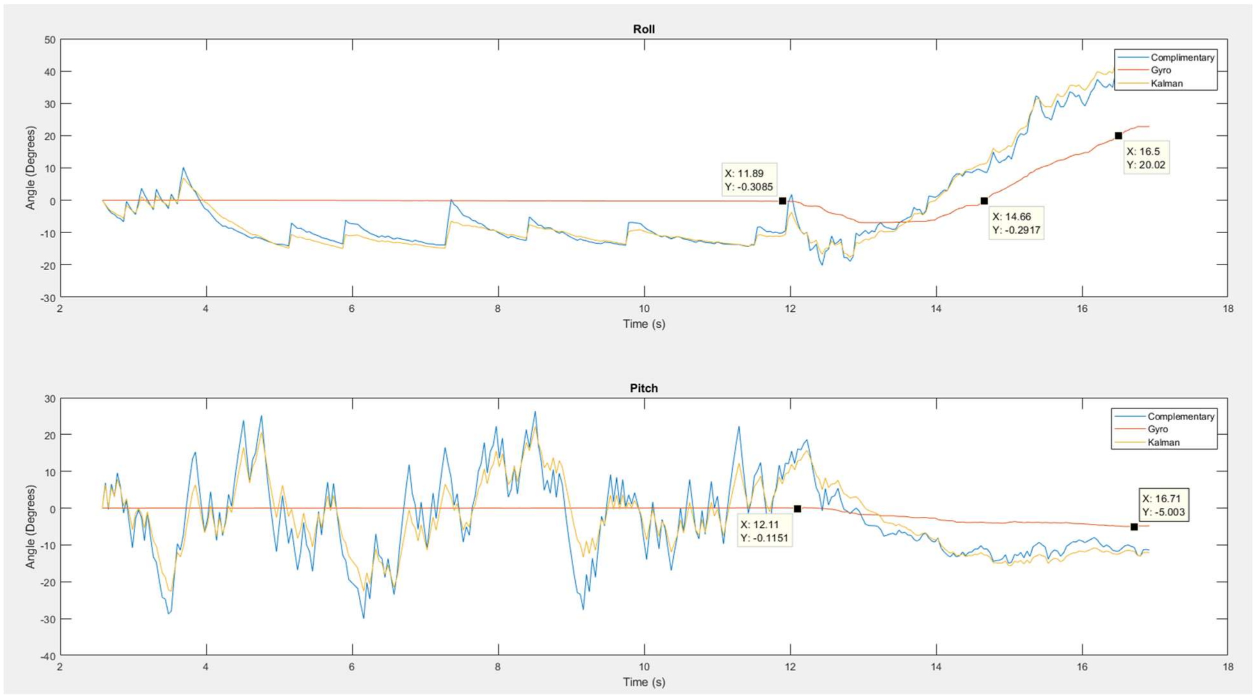


Figure 4.17 Processed IMU data (20 degrees horizontal / 0 degrees vertical), Second try

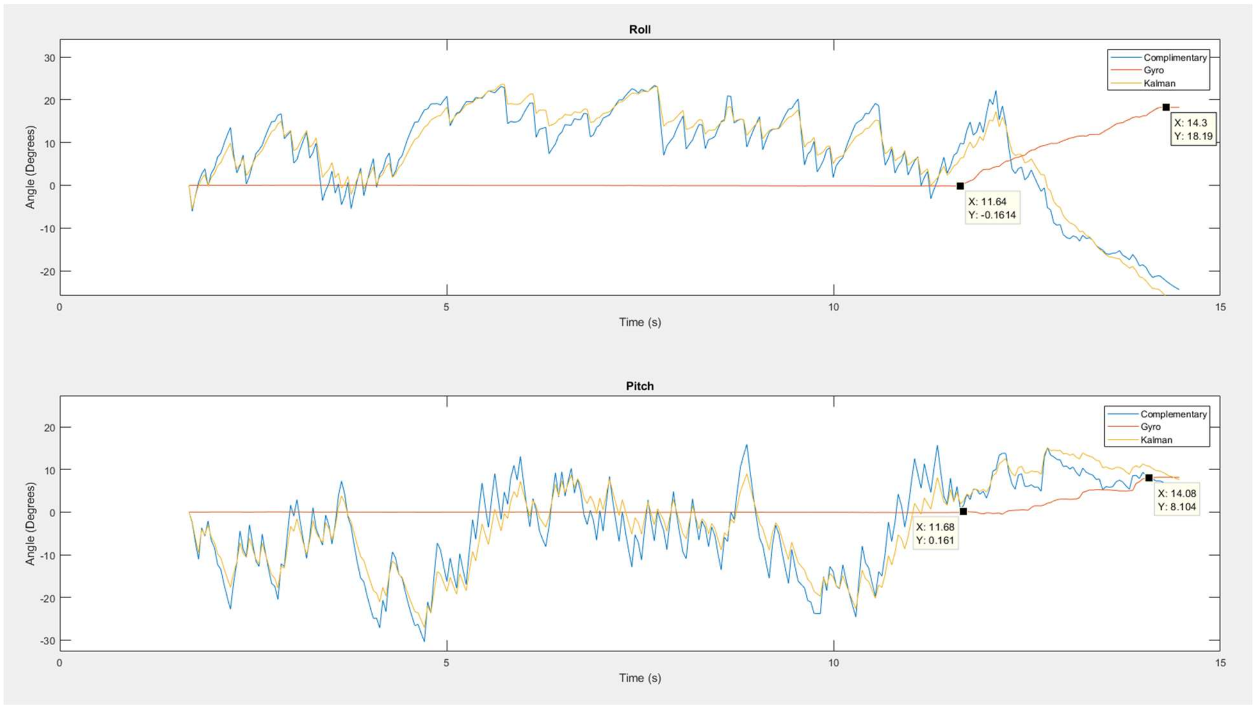


Figure 4.18 Processed IMU data (20 degrees horizontal / 10 degrees downward vertical)

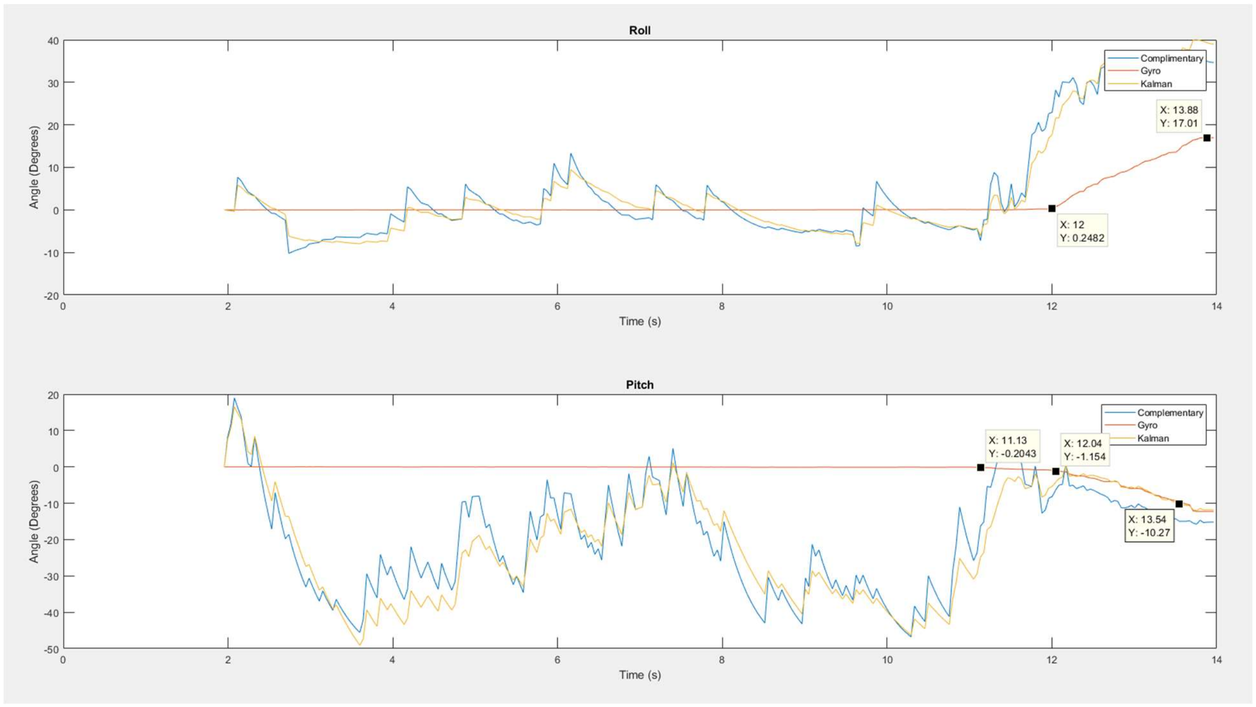


Figure 4.19 Processed IMU data (20 degrees horizontal / 10 degrees upward vertical)

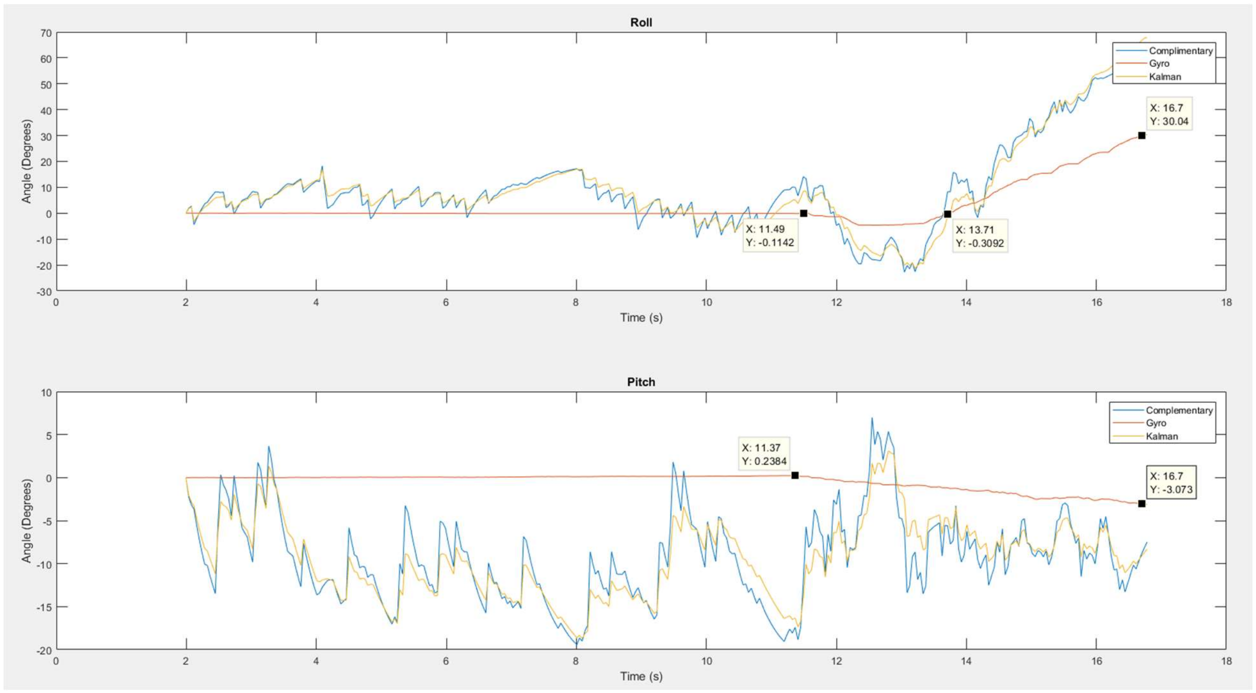


Figure 4.20 Processed IMU data (30 degrees horizontal / 0 degrees vertical)

Through analysis of these graphs, several trends are observed:

1. Time of the saccade increases somewhat linearly in relation to the angular distance:
   10deg: 0.97s, 20deg: 1.84s, 30deg: 2.99s
   More experiments are needed to determine the specific nature of these movements.
2. Time of the saccade is much higher than the human eye saccade time. This needs to be improved upon, further down the line.
3. Initial random movements in the motors due to the play in the gears. Motor doesn't always engage as soon as the operations are commenced.
4. Bias removal for gyroscope data degrades the true orientation value derived from gyroscope data.
5. When compared to Figure 2.6, the movement patterns of the built system are not of completely the same nature as the human visual system. There is a need to mathematically model human eye movement and attempt to improve upon the present motor algorithm, so it mimics mathematical modeling of the human visual system.
6. It could be seen that physical system build is not fully aligned (not fully straight) due to the movements in the axis where there is meant to be no movements (Figure 4.20 Lower graph).

More than anything else, these observations and trends are helpful for the design and build of the next iterations of the proposed system. As it could be concluded, saccade times could be improved upon. For improvements relating to above observations, some possible solutions could be incorporation of PID and use of more suitable motors. The other improvement would have to do with update of gear design so that the play in the gears could be eliminated. Other implementation which again relates to the physical build, is the need for more precise joint design. For this matter, proper joint design and updated physical design need to be incorporated. It would be ideal to design a one-piece main shaft which would then minimize errors regarding the ultimate horizontal orientation of two cameras in relation to each other.

Chapter 5 . Conclusion and Future Work

In this Chapter, a conclusion of the work conducted through this thesis project is provided. this includes observed advantages and disadvantages of the proposed system and the future work needed for the project.

5.1 Advantages

The prototyped system, while not perfect, has provided us with various advantages. First and foremost, the prototyped system has been a platform to investigate the plausibility of various algorithms for functioning of proposed active vision systems. If not for this prototyped system, the majority of the algorithms discussed would have remained theoretic in nature.

The second advantage provided by the efforts made here is the creation of a foveation algorithm using CNNs (DeepLab). This algorithm is optimal due to the fact that detection of desired features is consistent in the time domain. One of the issues faced with approaches such as Viola Jones is the dropped detection as the master camera moves. With the incorporation of DeepLab problem of dropped feature detection is mainly solved.

Another advantage of the prototyped system is its low cost and off the shelve components. The main frame and supporting pieces were all produced using 3D printers. And the components were all sourced from robotics suppliers. These were all popular components used in robotics design. This approach makes the replication of such a system by other researchers feasible, without the need to resort to sophisticated solutions.

5.2 Limitations

It should be noted that the present manifestation of the prototype is not ideal in nature. There are design elements which could be improved upon, for a better performing system. some of these improvements have been referred to in the previous sections.

One of the main limitations of the present design is motor performance. As seen previously, saccade times in proposed implementation are subpar comparing to recorded times form a human visual organism. Even though saccade patterns somewhat replicate human visual saccade patterns, times could be improved upon. This could be done by the use of more suitable motors and improved motor command algorithms.

The second limitation of note is the performance of the follower camera. Vergence of the follower camera, though functional, is slow. Hering's law could be incorporated in the algorithm design strategy to address this issue. This then would manifest itself in the form of initial saccade movement (replication of master camera's movement) on the follower camera followed by a loop of vergence command for full vergence. This, in turn, would translate into version followed vergence for the follower camera.

Another limitation mentioned previously is vergence algorithm's susceptibility to cross-correlation pitfalls. This includes cross-correlation's poor performance in featureless and texture-less areas. Lack of dedicated tracking algorithms which would act as smooth pursuit is the other limitation which could be improved upon. Also, saccade CNN algorithm (DeepLab) is non-optimal due to the runtime lag associated with it. Even though this lag does not impede the operations of the system, it is something which could be improved upon.

5.3 Future Works

This study, through its hands-on in nature, has provided a valuable insight into needed future steps for design of a wholesome anthropomorphic vision system. Through reflections on the present system, a list of possible future additions and implementations is composed (in order of necessity):

1. Addition of PID for motor controls
2. Addition of neural networks which specifically utilize saliency
3. Improvements of vergence solution
4. Depth map registration
5. Depth map reconstruction
6. Addition of a neck component
7. Addition of VR delivery methods
8. Addition of IMU unit on the VR rig for neck control
9. Addition of eye tracking for the VR rig platform
10. Addition of Neuromorphic Vision Sensors

Here, a brief explanation of the steps above and what they entail is provided:

The most pressing improvement needed for the proposed system is the implementation of PID capabilities for motor commands. This is needed both for foveation and vergence performance, to more closely resemble human eye movement.

Due to lack of optimized saliency CNNs at the time of this project, DeepLab was used, even though optimal in runtime, this code is not a true saliency detection code and as such needs to be replaced with an optimized saliency CNN. This could be tackled in few different ways: re-writing of SAM-VGG for a TensorFlow backbone, experimentation with DeepLab architecture for a saliency map output or finding of a better suitable CNN code.

Vergence solution as it stands is not robust enough in terms of runtime and is susceptible to false positives when dealing with featureless and textureless visual settings. This is one of the shortcomings of cross-correlation. Cross-correlation does not perform optimally when dealing with featureless visual environments. Some solutions which should be explored are reinforcement learning and possible neural network designs which could aid with vergence control.

After these steps are completed, visual delivery could be dealt with. This task should be tackled in two separate delivery phases: 3D reconstruction and Virtual Reality output delivery. These tasks were briefly discussed in Chapter 3.

In my opinion, the first task to be tackled is the creation of a Virtual Reality output. This is more readily achievable since the availability of verged left and right video feeds, after the implementation of mentioned above steps, could lead to a viable VR output feed. There might be a need for transformations of the two feeds so they would conform to the VR output photogrammetric requirements.

Some additional implementations which could benefit the proposed system and the VR output are the addition of a neck component to the system. This then would extend the range of visual field for the proposed system and would also provide other benefits one of which is the possibility for addition of teleoperation movement command capabilities to the proposed system.

An IMU unit could be incorporated in the VR headset which then could be used for movement of neck component in a teleoperation command setting. This then would allow for a mix of conscious and subconscious operations of the system. Visual system would function in a subconscious context (autonomous) and neck conscious (teleoperated) context (or a mix of two modes).

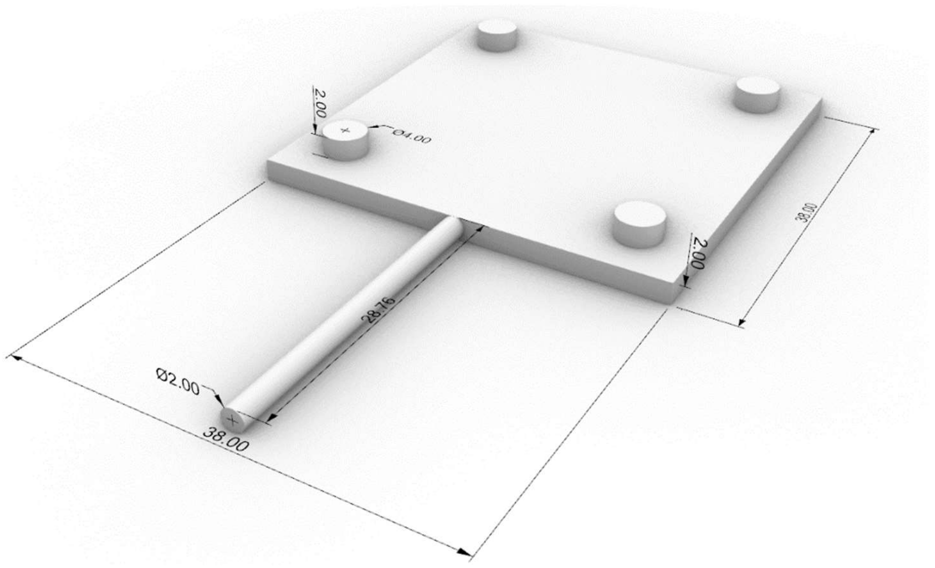
3D reconstruction would be the most challenging task to be fulfilled. This is due to the fact that to be able to produce a viable 3D reconstruction, there is a need for proper calibrations. As such, this step needs to be tackled last.

5.4 Closing Remarks

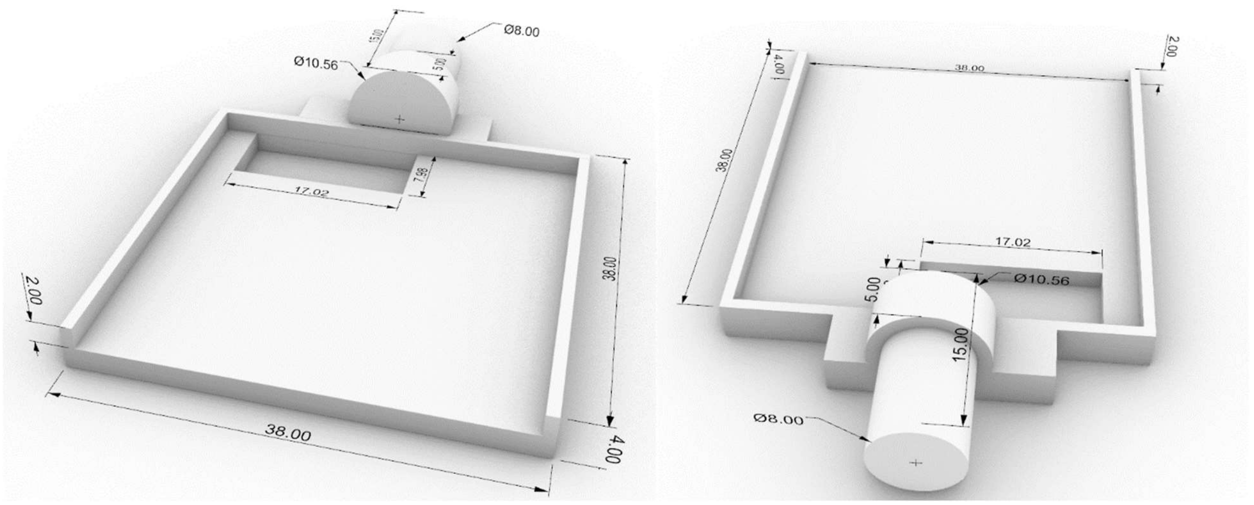
This thesis study has been conducted in the hopes for a better understanding of the human visual organism in order to determine possible solutions for mimicry of such an organism. Through efforts put forward, valuable lessons have been learned and needed future work has been determined. There are major additions and upgrades needed in order to finalize the present manifestation of the designed system into a viable product ready for industry use. My hope is to be able to pursue this further in the coming years.

Appendix

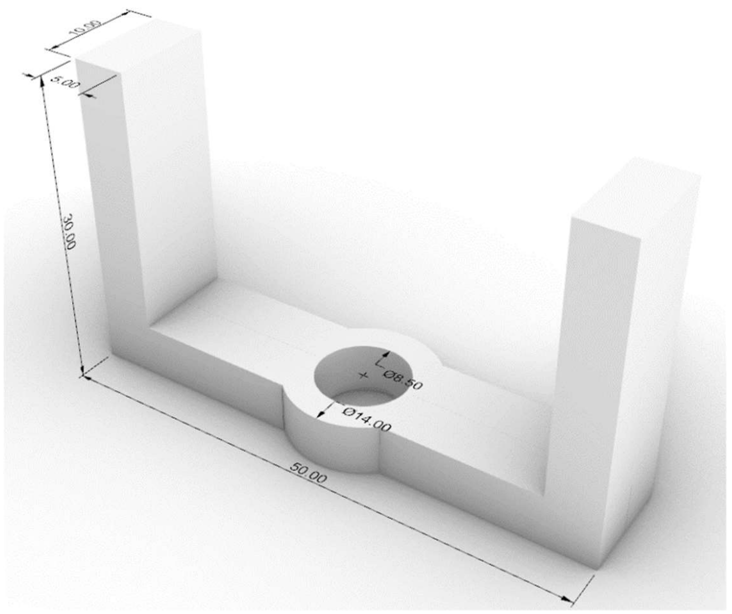
Appendix A – System Drawings



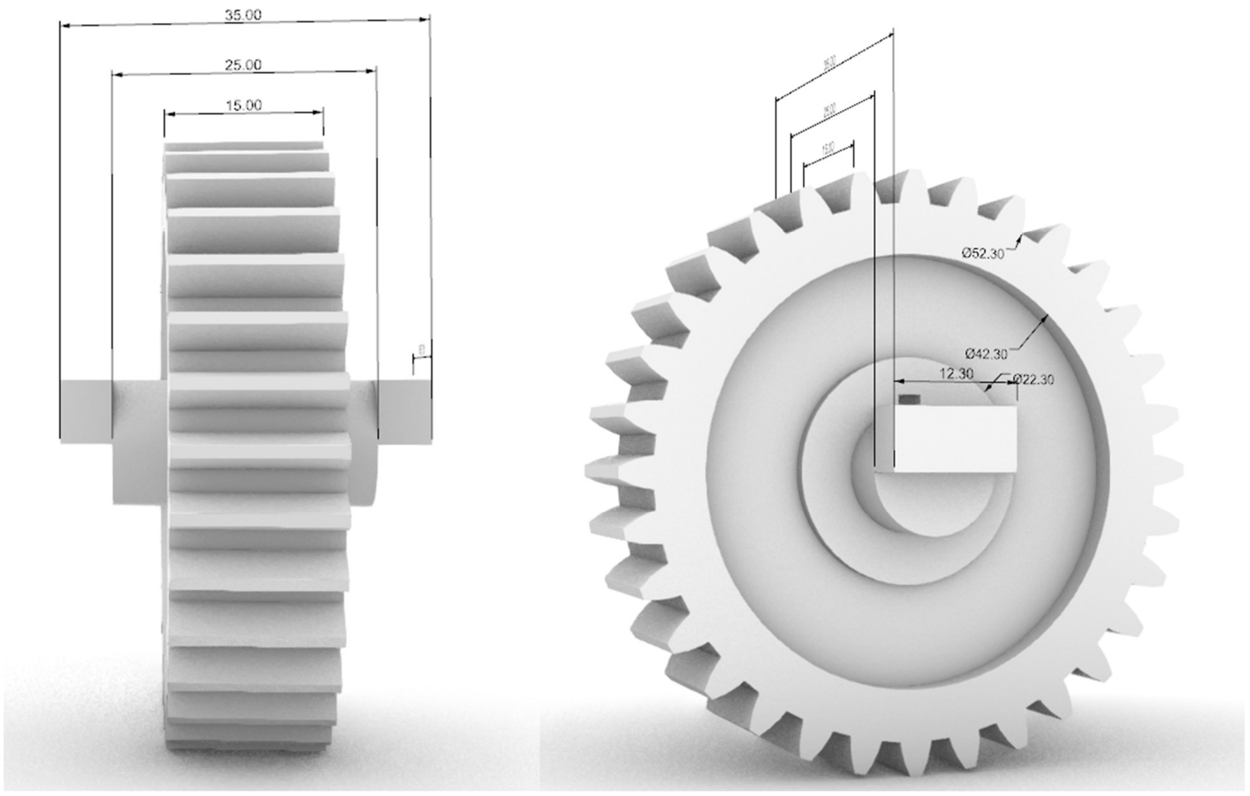
Appendix 1 First attempt at the camera platform design



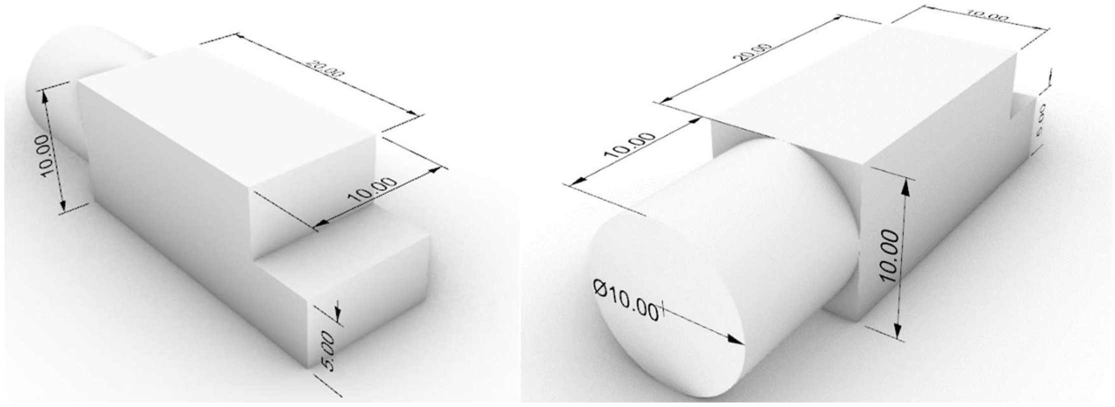
Appendix 2 Second attempt at the camera platform design



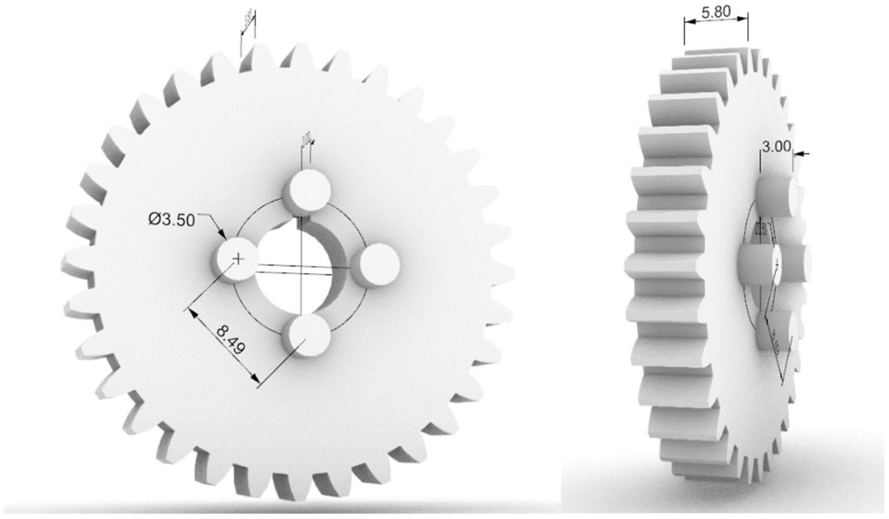
Appendix 3 a section of main shaft



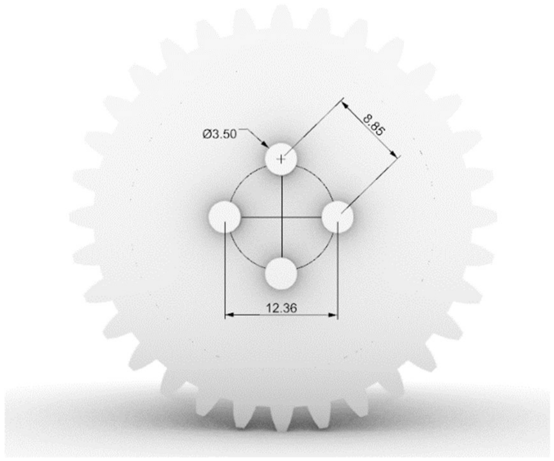
Appendix 4 Main shaft gear design



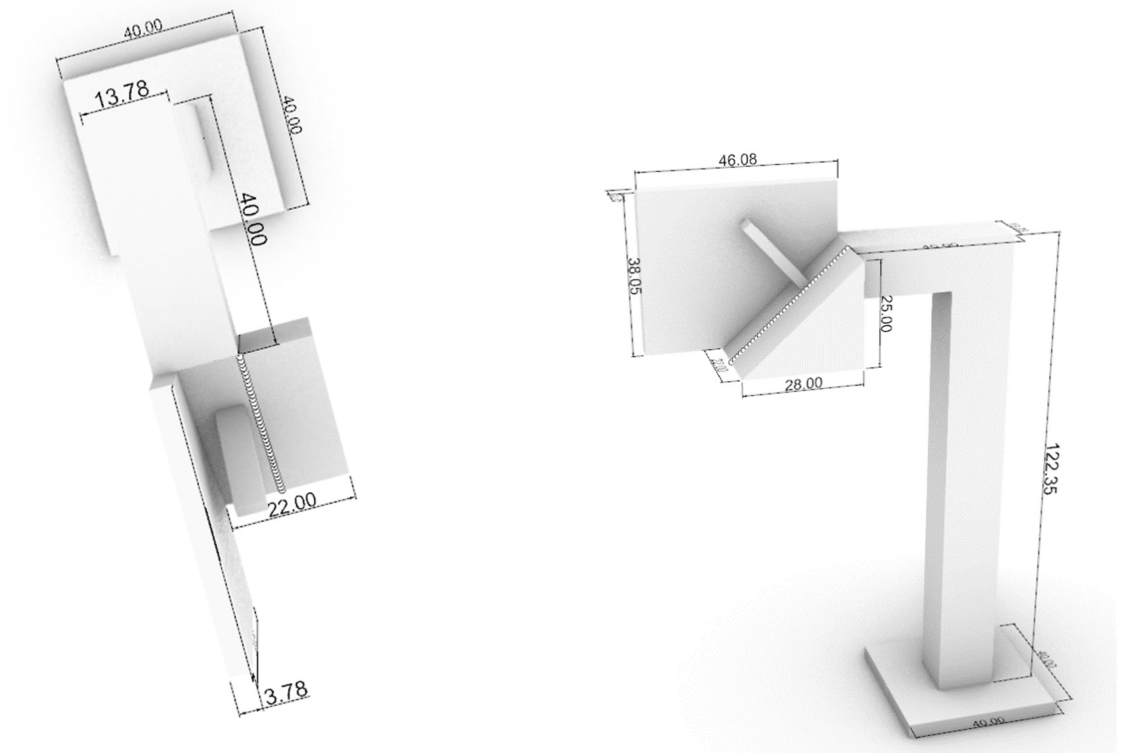
Appendix 5



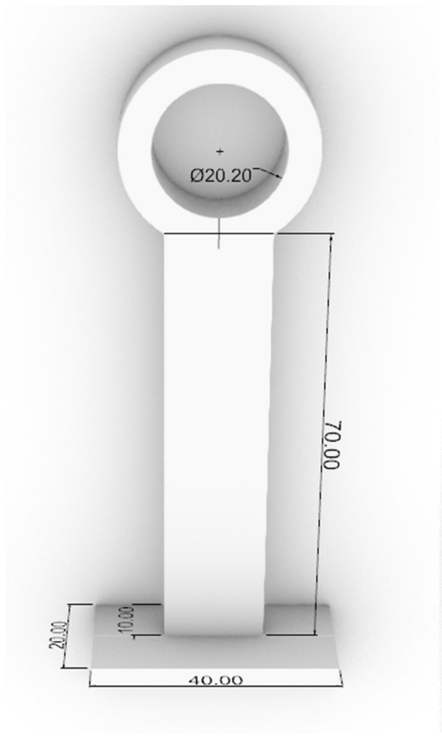
Appendix 6 gear support for the pan motor

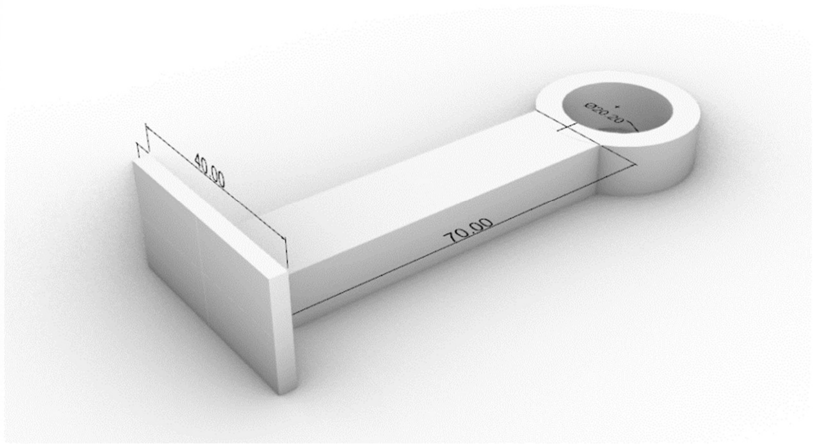


Appendix 7 gear support for the tilt motor



Appendix 8 Tilt motor platform





Appendix 9 Pillars for main shaft support

References

[1] Purves D, Augustine GJ, and Fitzpatrick D, *Neuroscience*, 2nd ed. Sinauer Associates, 2001.

[2] C. F. Martin and L. Schovanec, “Muscle mechanics and dynamics of ocular motion,” *J. Math. Syst. Estim. Control*, vol. 8, pp. 233–236, 1998.

[3] R. Dodge, “Five types of eye movement in the horizontal meridian plane of the field of regard,” *Am. J. Physiol.-Leg. Content*, vol. 8, no. 4, pp. 307–329, 1903.

[4] G. WESTHEIMER, “MECHANISM OF SACCADIC EYE MOVEMENTS,” *JAMA Ophthalmol.*, vol. 52, no. 5, pp. 710–724, Nov. 1954.

[5] J. G. Thomas, “The dynamics of small saccadic eye movements,” *J. Physiol.*, vol. 200, no. 1, pp. 109–127, Jan. 1969.

[6] D. A. Robinson, “The mechanics of human saccadic eye movement,” *J. Physiol.*, vol. 174, no. 2, pp. 245–264, 1964.

[7] C. Rashbass, “The relationship between saccadic and smooth tracking eye movements,” *J. Physiol.*, vol. 159, no. 2, pp. 326–338, 1961.

[8] Q. Yang, M. P. Bucci, and Z. Kapoula, “The latency of saccades, vergence, and combined eye movements in children and in adults,” *Invest. Ophthalmol. Vis. Sci.*, vol. 43, no. 9, pp. 2939–2949, 2002.

[9] J. H. Darrien, K. Herd, L.-J. Starling, J. R. Rosenberg, and J. D. Morrison, “An analysis of the dependence of saccadic latency on target position and target characteristics in human subjects,” *BMC Neurosci.*, vol. 2, no. 1, p. 13, 2001.

[10] A. D. Ruedemann, “Plastic eye implant,” *Am. J. Ophthalmol.*, vol. 29, no. 8, pp. 947–952, 1946.

[11] D. H. Ballard, “Behavioural constraints on animate vision,” *Image Vis. Comput.*, vol. 7, no. 1, pp. 3–9, Feb. 1989.

[12] D. H. Ballard, “Animate vision,” *Artif. Intell.*, vol. 48, no. 1, pp. 57–86, Feb. 1991.

[13] B. Scassellati, “Eye Finding via Face Detection for a Foveated, Active Vision System,” in *Proceedings of the Fifteenth National/Tenth Conference on Artificial Intelligence/Innovative Applications of Artificial Intelligence*, Menlo Park, CA, USA, 1998, pp. 969–976.

[14] B. Scassellati, “A Binocular, Foveated Active Vision System,” Massachusetts Institute of Technology, Cambridge, MA, USA, 1998.

[15] P. Sinha, “Perceiving and recognizing three-dimensional forms,” PhD Thesis, Massachusetts Institute of Technology, 1995.

[16] Y. Bar-Cohen and C. Breazeal, “Biologically inspired intelligent robots,” in *Smart Structures and Materials 2003: Electroactive Polymer Actuators and Devices (EAPAD)*, 2003, vol. 5051, pp. 14–20.

[17] C. Breazeal and P. Fitzpatrick, “That certain look: Social amplification of animate vision,” in *Proceedings of the AAAI fall symposium on society of intelligence agents—the human in the loop*, 2000.

[18] C. Breazeal, A. Edsinger, P. Fitzpatrick, B. Scassellati, and P. Varchavskaia, “Social constraints on animate vision,” *IEEE Intell. Syst. Their Appl.*, vol. 15, no. 4, pp. 32–37, 2000.

[19] C. Breazeal and B. Scassellati, "A context-dependent attention system for a social robot," *rn*, vol. 255, p. 3, 1999.

[20] C. Breazeal and B. Scassellati, "How to build robots that make friends and influence people," in *Proceedings 1999 IEEE/RSJ International Conference on Intelligent Robots and Systems. Human and Environment Friendly Robots with High Intelligence and Emotional Quotients (Cat. No. 99CH36289)*, 1999, vol. 2, pp. 858–863.

[21] N. G. Tsagarakis *et al.*, "iCub: the design and realization of an open humanoid platform for cognitive and neuroscience research," *Adv. Robot.*, vol. 21, pp. 1151–1175, 2007.

[22] J. Leitner, S. Harding, M. Frank, A. Förster, and J. Schmidhuber, "An integrated, modular framework for computer vision and cognitive robotics research (icVision)," in *Biologically Inspired Cognitive Architectures 2012*, Springer, 2013, pp. 205–210.

[23] C. Bartolozzi *et al.*, "Embedded neuromorphic vision for humanoid robots," in *CVPR 2011 WORKSHOPS*, 2011, pp. 129–135.

[24] S. Fanello *et al.*, "icub world: Friendly robots help building good vision data-sets," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2013, pp. 700–705.

[25] H. Kim, G. York, G. Burton, E. Murphy-Chutorian, and J. Triesch, "Design of an anthropomorphic robot head for studying autonomous development and learning," in *IEEE International Conference on Robotics and Automation, 2004. Proceedings. ICRA '04. 2004*, 2004, vol. 4, pp. 3506-3511 Vol.4.

[26] T. Villgrattner, "Design and Control of Compact High Dynamic Camera Orientation Systems," Dissertation, Technische Universität München, München, 2010.

[27] G. Csurka and F. Perronnin, "A Simple High Performance Approach to Semantic Segmentation.," in *BMVC*, 2008, pp. 1–10.

[28] B. W. Tatler, N. J. Wade, H. Kwan, J. M. Findlay, and B. M. Velichkovsky, "Yarbus, eye movements, and vision," *-Percept.*, vol. 1, no. 1, pp. 7–27, Jul. 2010.

[29] Alfred L. Yarbus, *Eye movements and vision*. Plenum Press, 1967.

[30] N. D. B. Bruce, C. Catton, and S. Janjic, "A Deeper Look at Saliency: Feature Contrast, Semantics, and Beyond," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 516–524.

[31] T. Judd, K. Ehinger, F. Durand, and A. Torralba, "Learning to predict where humans look," in *2009 IEEE 12th International Conference on Computer Vision*, 2009, pp. 2106–2113.

[32] X. Hou and L. Zhang, "Saliency detection: A spectral residual approach," in *2007 IEEE Conference on Computer Vision and Pattern Recognition*, 2007, pp. 1–8.

[33] A. Borji, "Saliency Prediction in the Deep Learning Era: An Empirical Investigation," *ArXiv181003716 Cs*, Oct. 2018.

[34] J. Tsotsos and N. Bruce, "Saliency based on information maximization," presented at the Advances in Neural Information Processing Systems 18, 2006, pp. 155–162.

[35] K. Simonyan and A. Zisserman, *Very Deep Convolutional Networks for Large-Scale Image Recognition*. 2014.

[36] M. Cornia, L. Baraldi, G. Serra, and R. Cucchiara, "Predicting Human Eye Fixations via an LSTM-Based Saliency Attentive Model," *IEEE Trans. Image Process.*, vol. 27, no. 10, pp. 5142–5154, Oct. 2018.

[37] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. The MIT Press, 2016.

[38] "Understanding LSTM Networks -- colah's blog." [Online]. Available: http://colah.github.io/posts/2015-08-Understanding-LSTMs/. [Accessed: 08-Dec-2018].

[39] “Kullback-Leibler Divergence Explained,” *Count Bayesie*. [Online]. Available: http://www.countbayesie.com/blog/2017/5/9/kullback-leibler-divergence-explained. [Accessed: 08-Dec-2018].

[40] Liang-Chieh Chen and Z. Yukun, “Semantic Image Segmentation with DeepLab in TensorFlow,” *Google AI Blog*. .

[41] L.-C. Chen, G. Papandreou, F. Schroff, and H. Adam, “Rethinking Atrous Convolution for Semantic Image Segmentation,” *ArXiv*, vol. abs/1706.05587, 2017.

[42] P. Viola and M. Jones, “Rapid object detection using a boosted cascade of simple features,” *CVPR 1*, vol. 1, no. 511–518, p. 3, 2001.

[43] A. Mohamed, P. F. Culverhouse, A. Cangelosi, and C. Yang, “Depth Estimation Based on Pyramid Normalized Cross-Correlation Algorithm for Vergence Control,” *IEEE Access*, vol. 6, pp. 65199–65211, 2018.

[44] C. Capurro, F. Panerai, and G. Sandini, “Dynamic vergence using log-polar images,” *Int. J. Comput. Vis.*, vol. 24, no. 1, pp. 79–94, 1997.

[45] N. Kyriakoulis, A. Gasteratos, and S. G. Mouroutsos, “Fuzzy vergence control for an active binocular vision system,” in *2008 7th IEEE International Conference on Cybernetic Intelligent Systems*, 2008, pp. 1–5.

[46] A. X. Zhang and A. L. Tay, “Baseline independent binocular vergence control of 2 dof pan-tilt cameras using a visual cortical model,” in *2006 9th International Conference on Control, Automation, Robotics and Vision*, 2006, pp. 1–6.

[47] A. Bernardino and J. Santos-Victor, “Vergence control for robotic heads using log-polar images,” in *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems. IROS ’96*, 1996, vol. 3, pp. 1264–1271 vol.3.

[48] H.-J. Kim, M.-H. Yoo, and S.-W. Lee, “Dynamic Vergence Using Disparity Flux,” in *Biologically Motivated Computer Vision*, 2000, pp. 179–188.

[49] F. Mutti, C. Alessandro, M. Angioletti, A. Bianchi, and G. Gini, “Learning and evaluation of a vergence control system inspired by Hering’s law,” in *2012 4th IEEE RAS & EMBS International Conference on Biomedical Robotics and Biomechatronics (BioRob)*, 2012, pp. 931–936.

[50] K.-C. Kwon, Y.-T. Lim, N. Kim, Y.-J. Song, and Y.-S. Choi, “Vergence Control of Binocular Stereoscopic Camera Using Disparity Information,” *J Opt Soc Korea*, vol. 13, no. 3, pp. 379–385, Sep. 2009.

[51] G. Westheimer, “The law of equal innervation of both eyes: Thomas Reid preceded Hering by a century. An historical note,” *Vision Res.*, vol. 101, pp. 32–33, Aug. 2014.

[52] P. G. Awade, R. Bodhula, and N. Chopade, “Implementation of barrel distortion correction on DSP in real time,” in *2016 International Conference on Computing Communication Control and automation (ICCUBEA)*, 2016, pp. 1–6.

[53] ST Coporation, “5.0 megapixel auto-focus camera module.” Oct-2015.

[54] “2001 Stereo Datasets.” [Online]. Available: http://vision.middlebury.edu/stereo/data/scenes2001/. [Accessed: 01-Aug-2019].

[55] O. Ghita, J. Mallon, and P. F. Whelan, “Epipolar line extraction using feature matching,” 2001.

[56] M. Jiang, S. Huang, J. Duan, and Q. Zhao, “SALICON: Saliency in Context,” in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015, pp. 1072–1080.

[57] Jackson Pollock, *Untitled*. 1950.

[58] George Sanen, *Conversation With Jackson Pollock No.41*. 2015.

[59] Y. Zhang *et al.*, “ActiveStereoNet: End-to-End Self-supervised Learning for Active Stereo Systems,” in *Computer Vision – ECCV 2018*, 2018, pp. 802–819.
[60] K. S. Vadivel, “Modeling Eye Tracking Data with Application to Object Detection,” 2014.
[61] “MPU-9250 | TDK.” [Online]. Available: https://www.invensense.com/products/motion-tracking/9-axis/mpu-9250/. [Accessed: 15-Dec-2018].
[62] M. Pedley, “Tilt Sensing Using a Three-Axis Accelerometer.” NXP, Mar-2013.
[63] V. Braitenberg, “Vehicles: Experiments in Synthetic Psychology,” *Philos. Rev.*, vol. 95, no. 1, pp. 137–139, 1986.
[64] S. Shostak and M. Landy, “How to Catch a Robot Rat: When Biology Inspires Innovation. By Agnès Guillot and Jean-Arcady Meyer,” *Eur. Leg.*, vol. 17, no. 4, pp. 560–561, 2012.
[65] H. Schoenefeldt, “The Crystal Palace, environmentally considered,” *Archit. Res. Q.*, vol. 12, no. 3–4, pp. 283–294, 2008.
[66] B. Orman, “Art Nouveau & Gaudí: The Way of Nature,” *JCCC Honors J.*, vol. 4, no. 1, p. 2, 2013.
[67] M. W. Joachim, “Ecotransology: integrated design for urban mobility,” Massachusetts Institute of Technology, 2006.
[68] M. Joachim, “A Century of Ecological Innovation,” *Archit. Des.*, vol. 85, no. 4, pp. 68–73, 2015.
[69] G. Alexandridis, “Sustainable Product Design Inspired from Nature,” 2016.
[70] H. Arwin, T. Berlind, B. Johs, and K. Järrendahl, “Cuticle structure of the scarab beetle Cetonia aurata analyzed by regression analysis of Mueller-matrix ellipsometric data,” *Opt. Express*, vol. 21, no. 19, pp. 22645–22656, 2013.
[71] A. J. Lilienthal and T. Duckett, “Experimental analysis of smelling Braitenberg vehicles,” in *IEEE international conference on advanced robotics (ICAR 2003), Coimbra, Portugal, June 30-July 3, 2003*, 2003, vol. 1, pp. 375–380.
[72] D. Cardoso, “Generative Craft A brief critical inquiry into design automation and design automata,” *Soc. Iberoam. Gráfica Digit.*, pp. 288–290, 2009.
[73] “The Electric Dog,” *Sci. Am. Munn Co.*, vol. Supplement No. 2267, pp. 376–377, Jun. 1919.
[74] C. Angle, “Genghis, a six legged autonomous walking robot,” Massachusetts Institute of Technology, 1989.
[75] Lingfeng Wang, Kay Chen Tan, and Chee Meng Chew, *Evolutionary Robotics: From Algorithms to Implementations*, vol. Volume 28. World Scientific Series in Robotics and Intelligent Systems, 2006.
[76] Serge Kernbach, *Handbook of Collective Robotics: Fundamentals and Challenges*. Pan Stanford, 2013.
[77] M. Dorigo, “SWARM-BOT: An experiment in swarm robotics,” in *Proceedings 2005 IEEE Swarm Intelligence Symposium, 2005. SIS 2005.*, 2005, pp. 192–200.
[78] M. Garrad, J. Rossiter, and H. Hauser, “Shaping Behavior With Adaptive Morphology,” *IEEE Robot. Autom. Lett.*, vol. 3, no. 3, pp. 2056–2062, Jul. 2018.
[79] S. C. Jacobsen *et al.*, “Research Robots for Applications in Artificial Intelligence, Teleoperation and Entertainment,” *Int. J. Robot. Res.*, vol. 23, no. 4–5, pp. 319–330, Apr. 2004.
[80] M. Sfakiotakis and D. P. Tsakiris, “Biomimetic Centering for Undulatory Robots,” *Int. J. Robot. Res.*, vol. 26, no. 11–12, pp. 1267–1282, Nov. 2007.

[81] J. J. Abbott *et al.*, “How Should Microrobots Swim?,” *Int. J. Robot. Res.*, vol. 28, no. 11–12, pp. 1434–1447, Jul. 2009.

[82] F. Schillebeeckx, F. De Mey, D. Vanderelst, and H. Peremans, “Biomimetic Sonar: Binaural 3D Localization using Artificial Bat Pinnae,” *Int. J. Robot. Res.*, vol. 30, no. 8, pp. 975–987, Sep. 2010.

[83] C. A. C. Parker and Hong Zhang, “Biologically inspired collective comparisons by robotic swarms,” *Int. J. Robot. Res.*, vol. 30, no. 5, pp. 524–535, Mar. 2011.

[84] H. Yamaguchi, “A cooperative hunting behavior by mobile robot troops,” in *Proceedings. 1998 IEEE International Conference on Robotics and Automation (Cat. No.98CH36146)*, 1998, vol. 4, pp. 3204–3209 vol.4.

[85] G. K. Klute, J. M. Czerniecki, and B. Hannaford, “Artificial Muscles: Actuators for Biorobotic Systems,” *Int. J. Robot. Res.*, vol. 21, no. 4, pp. 295–309, Apr. 2002.

[86] L. Kerhuel, S. Viollet, and N. Franceschini, “Steering by Gazing: An Efficient Biomimetic Control Strategy for Visually Guided Micro Aerial Vehicles,” *IEEE Trans. Robot.*, vol. 26, no. 2, pp. 307–319, Apr. 2010.

[87] Y. Bar-Cohen, “Biological Senses as Inspiring Model for Biomimetic Sensors,” *IEEE Sens. J.*, vol. 11, no. 12, pp. 3194–3201, Dec. 2011.

[88] J. Cabibihan, D. Joshi, Y. M. Srinivasa, M. A. Chan, and A. Muruganantham, “Illusory Sense of Human Touch From a Warm and Soft Artificial Hand,” *IEEE Trans. Neural Syst. Rehabil. Eng.*, vol. 23, no. 3, pp. 517–527, May 2015.

[89] Q. Shi, H. Ishii, Y. Sugahara, A. Takanishi, Q. Huang, and T. Fukuda, “Design and Control of a Biomimetic Robotic Rat for Interaction With Laboratory Rats,” *IEEEASME Trans. Mechatron.*, vol. 20, no. 4, pp. 1832–1842, Aug. 2015.

[90] E. Tidoni, P. Gergondet, G. Fusco, A. Kheddar, and S. M. Aglioti, “The Role of Audio-Visual Feedback in a Thought-Based Control of a Humanoid Robot: A BCI Study in Healthy and Spinal Cord Injured People,” *IEEE Trans. Neural Syst. Rehabil. Eng.*, vol. 25, no. 6, pp. 772–781, Jun. 2017.

[91] Z. Shen, J. Na, and Z. Wang, “A Biomimetic Underwater Soft Robot Inspired by Cephalopod Mollusc,” *IEEE Robot. Autom. Lett.*, vol. 2, no. 4, pp. 2217–2223, Oct. 2017.

[92] H. Ishii *et al.*, “Design and development of biomimetic quadruped robot for behavior studies of rats and mice,” in *2009 Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, 2009, pp. 7192–7195.

[93] C. Menon, M. Murphy, and M. Sitti, “Gecko Inspired Surface Climbing Robots,” in *2004 IEEE International Conference on Robotics and Biomimetics*, 2004, pp. 431–436.

[94] J. Reijniers and H. Peremans, “Biomimetic Sonar System Performing Spectrum-Based Localization,” *IEEE Trans. Robot.*, vol. 23, no. 6, pp. 1151–1159, Dec. 2007.

[95] Y. Fu, H. Li, and Y. Ma, “Path Planning of Cooperative Robotics and Robot Team,” in *2006 IEEE International Conference on Robotics and Biomimetics*, 2006, pp. 1250–1255.

[96] P. Turner and M. Dickinson, “HyKim - Development of a robot bear: Bringing the strength and robustness of a bear’s biomimetic features to a robot.,” in *2008 IEEE International Conference on Robotics and Biomimetics*, 2009, pp. 13–18.

[97] Feng Li, Lei Hu, Shaolong Kuang, Chuang Yang, and T. Wang, “A 5-DOF table-mounted surgical robot,” in *2007 IEEE International Conference on Robotics and Biomimetics (ROBIO)*, 2007, pp. 355–359.

[98] C. Xie, F. Kong, and H. Zhang, “Behavior-Based Motion Planning of Biomimetic Robot-Fish,” in *2010 International Conference on Measuring Technology and Mechatronics Automation*, 2010, vol. 1, pp. 749–751.