2021

A Model to measure online student engagement using eye tracking and body movement analysis.

Mido, Jude Austin
Faculty of Information Technology
Strathmore University

A Model to Measure Online Student Engagement using Eye Tracking and Body Movement Analysis.

By

Mido Jude Austin

A thesis submitted to the School of Computing and Engineering Sciences in partial fulfilment for the requirement of the degree of Master of Science in Information Technology of Strathmore University

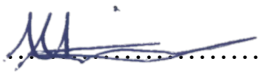
Strathmore University

August 2021

Declaration

I declare that this work has not been previously submitted and approved for the award of a degree by this or any other University. To the best of my knowledge and belief, the thesis contains no material previously published or written by another person except where due reference is made in the thesis itself.

Mido, Jude Austin



August 2021

Approval

The thesis of Mido, Jude Austin was reviewed and approved by the following:

Dr. Joseph Orero,

Senior Lecturer, School of Computing and Engineering Sciences,

Strathmore University

Dr. Julius Butime,

Dean, School of Computing and Engineering Sciences,

Strathmore University

Dr. Bernard Shibwabo,

Director of Graduate Studies,

Strathmore University

Abstract

Many institutions are adopting remote learning as way of expanding and offering their programs to mostly undergraduate students and adults seeking further education or training, and as a way of doing this at low costs; without constructing new buildings. However, measuring student engagement so as to focus attention on students who are struggling and manage students in a large class presents an extra challenge to teachers, when they have to do it virtually on eLearning platforms. The focus of this study was to build a model to measure student engagement using eye tracking and body movement analysis through web cameras, to help in tracking student engagement. The proposed solution is aimed at assisting in maintaining student engagement during remote classes, as it would be in a traditional classroom, and to enhance the learning process for both the student and the teacher. The proposed solution aimed to achieve this using computer vision algorithms; using computer vision to track the eyes and detect the hand, then analyze the movements to provide indices used to measure the engagement level. The model was tested using a prototype that analyzed recorded videos of students attending a remote class. The model achieved an accuracy of above 80%.

Keywords: Students Engagement, Online Student Engagement, Machine learning, Eye Tracking, Computer Vision, Body Movement Analysis.

Table of contents

List of figures

List of tables

List of code listings

List of abbreviations

EMH - Eye mind hypothesis

ET – Eye tracking

GA – Gesture analysis

HCI – Human Computer Interaction

HoF – Hands over Face

NSSE – National Survey of Student Engagement

OECD – Organisation for Economic Co-operation and Development

OOA – Object Oriented Analysis

OOD – Object Oriented Design

OOI – Object Oriented Implementation

Acknowledgements

I would like to sincerely thank my supervisor, Dr. Joseph O. Orero, for his constant support, patience and dedication in this study. His greatly appreciated guidance and commitment to the study was key to achieving the research objectives. I would also like to thank the members of the School of Computing and Engineering Sciences (SCES), for giving me the opportunity and support throughout the study. I would lastly, like to thank Diploma students at Strathmore Institute for their willingness to support the study.

Dedication

I dedicate this work to my mother Irene Mulemia, my siblings Anita Mido, Anthony Mido, and Angela Mido, my niece Maria Gabriella and most of all, to my late Dad Hesbon Mido.

Chapter 1: Introduction

1.1 Background

Technological advancements in computer science have revolutionized education in ways that cannot be ignored; the use of modern technological equipment and tools has increased learning and interaction among students (Raja and Nagasubramani, 2018). It is argued that technological advancements such as WIFI technology greatly helps students in learning, as it provides the means to access the large information available in the internet (Omar et al., 2018). The internet itself is seen to be the number one driver for technological advancements and is transforming the world into a global village, the education field not left behind. It has made it possible for students to access education from different geographical locations through remote learning, sometimes referred to as eLearning. Remote learning is an umbrella term for any distance learning that take place over the internet and not in a traditional class room (Stern, 2016.). Many institutions are adopting online learning as way of expanding and offering their programs to mostly undergraduate students and adults seeking further education or training , and as a way of doing this at low costs and without constructing new buildings (Meyer, 2014). eLearning is seemingly the future of education, as it is part of the big four agendas of the Kenyan government as reported by the world bank - one of the objectives is to have the Open University of Kenya by 2022, that will avail 30 percent of degree programmes through eLearning (World Bank, 2019). With so much effort being put into eLearning, a concern is how to maintain student engagement during remote classes and maintain the quality of education that students would have been comfortable with in a physical class.

It is important to keep track of student engagement levels since it, as shown by many researchers, has direct and strong relation with academic performance (Archambault et al., 2009; Hughes et al., 2008). The students participating in class are able to better understand the concepts and what is being taught. Proper student engagement has been seen to increase the motivation among students to learn and participate in school, with frequent online course dropout being attributed to the lack of it (Lee et al., 2015). Proper student engagement during the class would mean that teachers notice off-task behaviour such as a student disrupting a peer, or any indicator of emotional engagement such as boredom. Once this is seen, teachers are then able gauge and adjust their teaching methods to better capture the attention of the students (Kim et al., 2015a), which can improve learning and contribute to better academic performance.

With such importance of student engagement understood, many commercial platforms adopted for remote learning have tried to provide features to improve student engagement during the class sessions. For example, Zoom includes the use of white boards that teachers can use to explain certain concepts, and features such a student being able to raise their hand, or side chat the lecturer so as to seek clarification of a concept. Use of video grid view allows the instructor to see multiple participants/attendees to monitor and to help gauge their interest as they would in a traditional normal class. Another feature includes the use of priority view of the speaker such that, with the video on, the video of the main speaker is shown on the main screen so that viewers are aware of who is talking, and seeing the speaker also better captures there attention especially when there is no material being presented. All these features assist in managing a class but the biggest gap that makes it hard to utilize them is that the teacher has to concentrate on delivering the content he or she had planned. It is seemingly difficult to effectively deliver content while at the same time, monitor 25 participants, as the teacher would in a traditional face-to-face class. Research has shown that there is need to provide specialized support to students online, which can only be initiated by the teacher (Babcock et al., 2019). How then, can the teacher know which student in not engaged in the class or training session and be able to initiate the support?

1.2 Problem Statement

Remote learning means that the teachers are not physically present with the students and it is therefore, difficult to notice which students are present in the sense that, not only have they logged in, but are aware of what is being taught and are grasping the content. Teachers need to track student engagement levels and monitor students during remote classes and monitor early signs of distress (Revak, 2020). Existing solutions do not provide adequate platforms for remote learning when considering student engagement issues that follow. First, there is no maximum number of students that can be set for remote classes, so a teacher could be teaching fifteen, to fifty, to one hundred and fifty students in a remote class. In a small class, the teacher can easily monitor fifteen or twenty fifty students whose videos are shown on one screen, in a video grid but, high numbers would make it difficult to scroll through and be able to monitor what the students are doing (Kim et al., 2015b). This then makes it difficult to manage the class and stop students that may want to distract others through the chat messages or showing distracting content through their cameras. Study has shown that students believe that teacher and school restrictions provide the most effective method to limit distracting behaviour while learning (Kay et al., 2017). Second, students using a new platform may not be aware of the

features and how to use them on that particular platform. For example, a student used to raising their hand using a button on zoom, may not be able to do it on Skype, meaning it would be difficult to get the teachers attention during the class with a muted microphone. Unless there is a means where the camera could detect whether the student has raised the hand up and alert the teacher. Lastly, the teacher or instructor is central to engagement and therefore, needs to be aware of the student who is struggling to grasp the particular content and be able to approach them (Zepke and Leach, 2010). This is however, a challenge in a large class where the teacher has to manage the class and deliver content at the same time. Online courses can accommodate hundreds, if not thousands, of students in a single course (Lowenthal et al., 2019), and with the increased adoption of remote classes, the above problems could only get worse.

The focus of this study was to build a prototype to track student engagement using eye tracking and body movement analysis through web cameras, which would then help a teacher or instructor having a remote class session to identify students who are not engaged in the lesson, by highlighting the name of the student so the teacher can approach them virtually.

1.3 Objectives

1.3.1 General Objective

To design and develop a prototype to measure online student engagement using body movement and eye tracking analysis.

1.3.2 Specific Objectives

1. To investigate the problems associated with student engagement in remote learning.
2. To determine how the existing technologies measure student engagement in a remote environment.
3. To develop a model to assess student engagement in remote learning.
4. To test the reliability of the proposed prototype in assessing student engagement.

1.4 Research Questions

1. How can the problems associated with student engagement in remote learning be addressed?
2. How do existing technologies measure online student engagement?
3. How can a model be designed and a prototype developed to assess online student engagement?

4. How can the proposed prototype be tested for reliability is assessing online student engagement?

1.5 Justification

This research impacts teaching online classes by providing teachers with a way of measuring the engagement level of each student in the class, and therefore makes it easier to focus their attention on them by asking questions or use other ways of encouraging participation in the class, so as to avoid loss of interest. For the student, this intends to increase achievement as they are able to focus more in the class, develop positive behaviour of attending and remaining attentive during the class section and general aid in developing a sense of belonging so that they may remain in school (Ciric and Jovanovic, 2016a). Generally, this research impacts student engagement to make online learning more productive for the institution and more importantly, ensure that students are more successful as they pursue online studies (Meyer, 2014). Lastly, the research has chosen eye tracking and student engagement, which are assumed to easily be visible to a standard web camera, without demanding for any modifications or acquisitions of new devices.

1.6 Scope and limitation

1.6.1 Scope

This research focused on the design and development of a prototype to track student engagement during an online class by tracking the eye movements and detecting upper body movements (gestures) – such as when the student raises up their hand – and then give an engagement level index as a measure of their general presence in the virtual class. The prototype was to be tested on a small sample of videos recording during different class sessions, so as to examine how reliable it is in measuring the student engagement and ranking an index; the lowest index could then be brought to the teacher's attention. This research was limited to focus on students taking online classes to study part-time (evening) for undergraduate or postgraduate degrees, as they are seemingly more self-driven with the assumed intention and willingness to be in class.

1.6.2 Limitations

The working of the prototype was largely dependent on the collection of data in the form of images from video files. Sharing of videos over a network often requires a significant level of bandwidth and can be affected by poor internet connections. Additionally, as ethically required

for research, students whose data will be collected may act different during the video recording which could affect the reliability of the prototype in measuring student engagement.

Chapter 2: Literature review

2.1 Introduction

This chapter discusses the theoretical framework of student engagement; identifying the definition of the term to be used in this research, the dimensions of student engagement and useful indicators of each for the research, and how they can be tracked in an online class. Additionally, the two independent variables (eye tracking and body gestures) are discussed in this chapter. Similar studies on tracking student engagement are reviewed and assessed in the chapter as well as their contribution. Lastly, the chapter provides the conceptual framework of the proposed prototype to track student engagement in an online class, using eye tracking and body gestures.

2.2 Theoretical Framework

2.2.1 Definition of student engagement

Many researchers argue that the term student engagement does not have a standard definition (Ashwin and Mcvitty, 2015), it can used in general to refer to the effort and commitment that students have towards learning. The glossary of education reform, gives a more relevant definition of this as, "*the degree of attention, curiosity, interest, optimism, and passion that students show when they are learning or being taught*", which is the definition used in this study. Student engagement has three dimensions to it (Fredricks et al., 2016a): behavioural engagement which refers to the observable acts of students as seen in learning such as the attention and behaviour while in class (Cappella et al., 2013). The second is cognitive engagement, that refers to the extent to which students are willing to learn and take on tasks as shown by class attendance and participation in extra-curricular activities (Rotgans and Schmidt, 2011a). The third dimension is emotional engagement which refers to the student's level of interest, boredom, happiness, anxiety and other affective states. Researchers argue that behavioural engagement is the most important as it can be an indicator for the other two dimensions (Cappella et al., 2013).

While there are varying definitions and measurements of the term "student engagement" (Axelson and Flick, 2010; Martin and Torres, 2016), Many researchers seem to agree on its definition as the students active participation in a class lesson. Martin and Torres (2016) give a general definition as the term used to describe meaningful student involvement throughout the learning environment. The Glossary of Education Reform (2016) defined the term as the

degree of attention, curiosity, interest, optimism, and passion that students show when they are learning or being taught, similar to what several researchers had defined it in their research (Axelson and Flick, 2010; Bond et al., 2020; Martin and Torres, 2016).

2.2.2 Dimensions of Student Engagement

Recent researchers agree that the term is not a static concept, but has three dimensions to it, as described by previous research namely, behavioral student engagement, cognitive student engagement and emotional student engagement (Ciric and Jovanovic, 2016b; Fredricks et al., 2004). There are many sub-constructs of student engagement and one project could not possibly cover all of them but instead, develop one element without denying the existence of the others (Kahu, 2013). Additionally, it is important for every research project to begin with a clear definition of their own understanding (Boekaerts, 2016). Therefore, student engagement in this research is defined as "the energy and effort employed by a student during a class session, observable via any behavioral, cognitive or affective indicators across a continuum", as borrowed from similar research by Bond (Bond et al., 2020). While other research has shown the existence of many other elements and quite possibly a fourth dimension (Bowden et al., 2019; Fredricks et al., 2016b), this research relies on the three widely accepted dimensions – behavioral, cognitive and emotional engagement as well as their indicators – proposing that social engagement in online learning can only be observed in behavioral engagement during class for example; the student's participation during group activities.

Figure 2.1: Three dimensions of Student Engagement

Behavioural student engagement is defined as the easily noticeable participatory activities of a student (Bowden et al., 2019). A study to map research to educational technology revealed the top five indicators of behavioural engagement as; participation, interactions and being involved, achievement, confidence, study habits and lastly, attention or the focus of the student (Bond et al., 2020). For example, if the teacher asks a question, then participation and attention will be indicated by students answering through the chats or raising up their hands to attempt and answer the question, which can also be measured by tallying the number of students who raise up their hand to answer questions (Nyman, 2018). Additionally, the student watching/gazing at the teacher's presentation indicates attention and focus and this can be used as a measure, complementing login duration among other activity log data, based on a version of the National Survey of Student Engagement (NSSE) measures, modified for eLearning (Henrie et al., 2015).

Cognitive student engagement is the amount of effort that students put in to accomplish a given academic task and persistence in studying (Rotgans and Schmidt, 2011b). This is important as it enhances true understanding of the topic or interest in the topic as well as motivating the student (Sesmiyanti, 2016). Research done in 2020 reveals the top five indicators of cognitive engagement as learning from peers, deep learning, self-regulation, positive self-perception and critical thinking (Bond et al., 2020). It is argued that cognitive engagement cannot be measured directly, but through its effects (Pitterson et al., 2016). This can include; the student taking notes, paying attention to a presentation taking place in an online environment, gazing at the screen and asking questions.

Sometimes referred to as affective engagement, emotional engagement is defined as the affective reaction to the class (Fredricks et al., 2004). It is indicated by positive interaction with teachers and classmates, enjoyment, positive attitude, motivation and enthusiasm (Bond et al., 2020). It relates to the summative levels of emotion demonstrated through elation, happiness and joy or sadness and boredom shown by yawning (Bowden et al., 2019). Emotional engagement is important not only because it contributes on academic achievement of the student, but also because it creates social presence in an online environment which improves learning (D'Errico et al., 2016).

This research relied on the observable sub-constructs of student engagement such as body presence and gestures, as well as eye-tracking to determine how engaged the student is during the lesson in terms of attendance/presence, paying attention, participation and enjoyment.

2.3 Eye Tracking

2.3.1 Eye tracking in student engagement

Eye-tracking (ET) has been defined as a technique used to measure an individual's eye-movement with two objectives in mind; the first is to know where the person is looking at any given time and the second is to know the sequence in which the person's eyes are shifting from one location to another (Poole and Ball, 2006). ET relies on the eye-mind hypothesis (EMH) by Just and Carpenter that the eyes remain fixed on what the mind is processing at the given time. For example while reading, the eyes will remain fixed on a particular word, for as long as that given word is being processed (Just and Carpenter, 1980). EMH predominantly held in a research on domain specific interpretation of eye tracking data, where eye movements were related to attention (Schindler and Lilienthal, 2019). It is mainly based on EMH, that ET has found a variety of use in many fields such as analysing human behaviour in marketing,

neuroscience, human-computer interaction (HCI), visualization research, reading in education, scene perception and visual research (Blascheck et al., 2017).

On the other hand, student engagement consists of three major dimensions as discussed in section 2.2.2, one of which being the cognitive student engagement. It is mentioned that even though it cannot be measured directly, certain activities such as paying attention in a class is a good indicator of cognitive student engagement. Relying on this, this research related EMH to cognitive student engagement in that; the student who is paying attention would be looking at the screen, as they try to process what the instructor is teaching while presenting something on the screen. This meant that by keeping track of the student's eye movements, the student's engagement level can be tracked as well.

2.3.2 Eye movements in Eye Tracking

Eye movements are human behaviour that occur involuntarily or voluntarily on average, three times per second, indicating information that is being processed at the particular time by the subject (Soluch and Tarnowski, 2013). They can be divided into two major categories; saccades which are the rapid movements of the ye before it is fixated on a given object and fixation which is the period of time where the eye is kept aligned with the target object during processing (Blascheck et al., 2017). In eye tracking, the two categories are used to measure eye movements in three measures with sub-measures (Holmqvist et al., 2011). First are the movement measures for direction, amplitude, duration, velocity, acceleration and shape. Second, Position measures for basic position, dispersion, duration and pupil diameter and position data. Last are numerosity measures for saccades position, number and rates. Simply put, eye tracking measures eye movements using eye position (fixation), travel (saccades) and pupil size for specific time, and does this using optical methods of reflecting light to the eye and cameras capture changes in the reflection (Gonzalez-Sanchez et al., 2017).

In a typical eLearning environment, a student will use a laptop or smartphone with a web camera mounted. This research used the web camera to capture the student's eye movements and through ET analysis, determine whether the student is paying attention to the class or is not mentally present. The purpose relies on EMH and the assumption that the selected group of campus students – Undergraduate and part-time students – do not have any other window active on the desktop at the same time the class is running i.e., only the window showing the class is being viewed at the particular time.

2.4 Upper Body gestures

Body gestures or body language gestures refer to body movements that convey meaningful emotive expressions; important non-verbal cues that provide clues of intention, emotion and motivation (Tipper et al., 2015). As discussed in section 2.2.1, body gestures are a part of student engagement in a class. They are a part of non-verbal communication through body language, alongside facial expressions and eye movements and in this particular study, the focus is on the upper body gestures which include hand movement or gestures. Hands are commonly used in social interactions to non-verbally communicate simple things like direction while pointing to complex things like expressing feelings (Behera et al., 2020), or intention. For example, raising your hand up to get the presenters attention so as to as a question or to show intention to be a part of something. Additionally as part of hand movements and emotional body language, Hands over Face (HoF) gestures become a part of upper body gestures, not only as a way of enhancing facial expressions occluded by the hand that would otherwise be corrupted or erroneously detected, but as a way of interpreting the emotional state of the person (Mahmoud et al., 2014). Additionally, they provide a way of interpreting the cognitive mental states (Mahmoud et al., 2016). This can be based on The Definitive Book of Body Language, which attempts to identify the meaning of different HoF gestures (Behera et al., 2020). A sample of these HoF gestures is shown in figure 2.1.

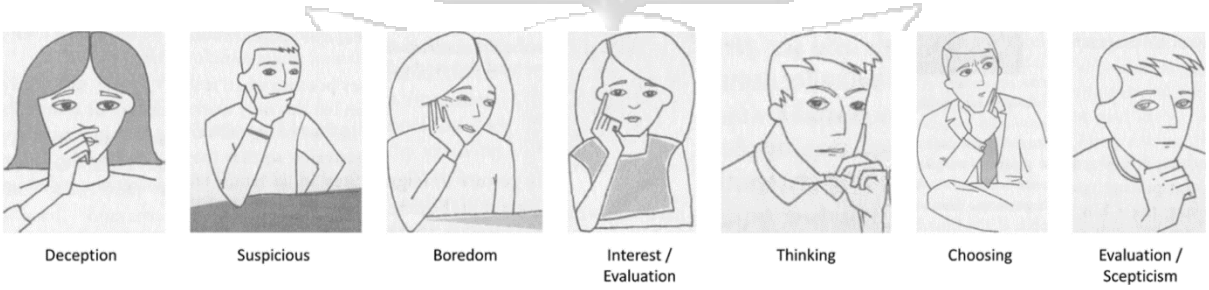


Figure 2.2: Identified meaning of HoF gestures (Pease and Pease, 2016)

As a way of keeping track of student engagement, upper body gestures are considerable indicators of cognitive, behavioral and emotional states, occurring at an average of 21.35% in 40 minutes during a class (Behera et al., 2020). Alongside simple hand movements such as raising and waving hands, this research intends to use this in tracking student engagement in an online class by pointing out students who raise hands to ask questions, wave to greet or interact with other class participants and also monitor their states through HOF.

2.5 Empirical Literature

2.5.1 Student Engagement using Eye movements, Head Movements and Facial Expressions

Researchers developed a prototype to track student engagement level in e-learning environments using three variables; movement of the eyes, movement of the head and facial expressions. The three can easily be captured in images from the web camera in a typical laptop. Analysis of these variables is done to determine the concentration level, which is then used to classify the student in one of the three levels of engagement; “very engaged”, “nominally engaged” and “not engaged at all”. The prototype was tested in a typical eLearning scenario with fifteen students and it correctly captured each period of time when the students were engaged at the three different levels. Additionally, it linked student with the best scores to high concentration index/ higher engagement level (Sharma et al., 2018). However, this system could not accurately capture facial expressions and the researchers suggested the use of 3D facial recognition, which would require a better camera than those found on a typical laptop computer.

2.5.2 Student Engagement using Eye Movements and Head Rotations

Krithika and Lakshmi (2016) developed a system to recognize student emotions on eLearning based on eye movements and head rotations, which they directly linked to the learner’s concentration level/student engagement during the class (emotional student engagement). The inputs were obtained from the images obtained from a web camera, and analysed to determine the concentration level as either high, medium or low. The system was then tested on 5 different students, shown a 10 minute lecture video and it was able to conclude the concentration levels accurately (Krithika L.B and Lakshmi Priya GG, 2016). The proposed system was efficient enough to detect negative emotion like boredom and lack of interest which however, presents a gap which the researchers present as a future work; “to capture other emotions” which can be said to be the positive emotions.

2.5.3 Student Engagement using Head Rotations

Following the social interactions in a classroom research. Raca and Dillenbourg (2014), researched on a system to translate head motion to attention levels of students during a classroom based on videos of the students taken. The research focused only on head movements - motion and pose - and through computer vision analysis captured the movements. Support vector machine (SVM) was used to classify the attention level. This system obtained an

accuracy level of 61.86% classification of student attention on a 3 point scale, and showed that drops in attention levels are reflected in decreased intensity of head movements (Raca and Dillenbourg, 2015). However, the system was not tested in an eLearning setup that could have reduced the effect of social dimension in a traditional classroom, as identified by the researcher. Additionally, the research did not include other measures (e.g. eye-tracking and HoF gestures) that could increase the accuracy though, the researchers present it as a future work on the system.

2.5.4 Student Engagement using Eye movements and EMH

Brian Miller (2015) researched on the measurement of cognitive student engagement based on reading times and eye movements, with the assumption that people look at images and words longer because they are thinking about them; that is where there attention is. The idea was that students look longer at images that they are trying to understand. For example, staring at the screen trying to understand the concept that is being taught. The research reveals that eye tracking can increase precision at micro levels of tracking student engagement.

2.5.5 Computer Vision Algorithms

2.5.5.1 Face detection and eye tracking Algorithms

OpenCV stands for Open Source Computer Vision library, which is an open source computer vision and machine learning library. It provides functions which use a pre-trained dataset to detect and recognize faces, classify human actions in videos and follow eye movements, among other uses for its 2500 plus optimized algorithms. Dlib is also an open source software that contains machine learning algorithms and can be applied in data analysis, as seen is Vardan Argwal's development of a simple algorithm that used this two – openCV and Dlib – so as to track real time algorithm (Agarwal, 2020). This algorithm was tailored to fit this research and used to track the eye movements and collect data for ET analysis.

To detect the eyes and track their movement, Argwal recommends detecting the face first, and then track the eyes. Kirthika defined 4 eye tracking algorithms to detect faces; Viola Jones, Local Binary Pattern (LBP), Ada Boost and Neural Networks (Krithika L.B and Lakshmi Priya GG, 2016).

Viola Jones provides high accuracy in face detection with less false positives, at the cost of a longer training time (D.N. et al., 2012). LBP labelled image features and used the labels to detect new faces and though it is fast, it is not as accurate. Ada boost performs more than one

iterations on visual features it takes, to generate a face detector that does not use past data. This makes it simple but greatly affected by outliers and noise in the data.

Lastly, the use of a neural network selects a network with the best performance to reduce the complexity in detecting images with no faces. Even though it requires less computing power, it is slow and provides less accurate results.

In using python to implement eye detection, the said algorithms are implemented using Haar Cascades for Viola Jones, Dlib frontal face detector for LBP and Multi-task Cascaded Convolutional Neural Network (MTCNN) for using a neural network. In this research, dlib was chosen as it is fast, and high accuracy was not a concern. It was also chosen as several open source datasets were available to be used with the said algorithm. More importantly, using Dlib allowed the use of the 68-facial key point dataset, which allowed the model to single out the eyes using specific key-points and tracking the movement within those areas.

2.5.5.2 Gestures and Object detection Algorithms

The gesture detection algorithm borrows from Gogul Ilango's algorithm for hand gesture recognition using python and OpenCV. The algorithm relies heavily on background subtraction technique, to find the hand and analyse its presence; this is achieved through the use of running averages (Ilango, 2017). Background subtraction is a method that is commonly used to isolate the moving parts from multiple frames, by segmenting into background and foreground (Turaga et al., 2010). To implement background subtraction, running averages is used in two steps; differencing step and the background modelling step which basically compares a new frame to a given number of frames, and the difference is the moving object (foreground) that we need to analyse (Sukhavasi et al., 2013). The algorithm was modified and used based on the following steps: The algorithm first segments the hand region using background subtraction method – done by providing the computer with 30 frames so that computes the running average for the background coordinates and have everything in black. This was accomplished by a running-average function.

2.6 Summary of Literature Reviewed

Based on the sections above, the literature can be summarized in that student engagement in an online environment is still important, and can be tracked using the web camera found on many laptops. The web camera can be used to capture images from which we can extract useful information variables such as eye movements, upper body gestures and head movements.

Measuring head movements alone gives a considerable accuracy which however, can be improved by the addition of eye movement. Measuring the saccades and fixations of the eye - as used by Krithika and Lakshmi research, as well as Sharma's research – can be used to increase the accuracy of interpreting the students' attention level, also supported by Miller. While Sharma et al points out the limitations when using facial expressions to analyse concentration levels of the student, Bond et al, provides the use of HoF gestures to provide more accurate results and reflect positive emotions which Krithika and Lakshmi present as a limitation of their proposed system. Lastly, research done by Brian miller points out the effectiveness of using eye tracking to track student engagement at a micro level, when complimented with reading times that alter the eye movements.

2.7 Conceptual Framework

The proposed system aimed to work with four modules. The first module being the eye tracking module that collects and analyses data of the eye movements (saccades and fixations) and then provides an index as a result, when requested. The second module is the body gestures analysis module that detects and analyses presence of a hand and also provides an index result for the same – indicating whether the hand is raised or not – when requested. Third and main module is the analysis module that requests, obtains and analyses indices from the two modules and provides a final index that is used to rank the engagement level of different students according to low level (red), average (yellow) and high level (green). Last is the reports module, which provides a summary of the engagement levels at the end of the class, based on the different indices given during the study.

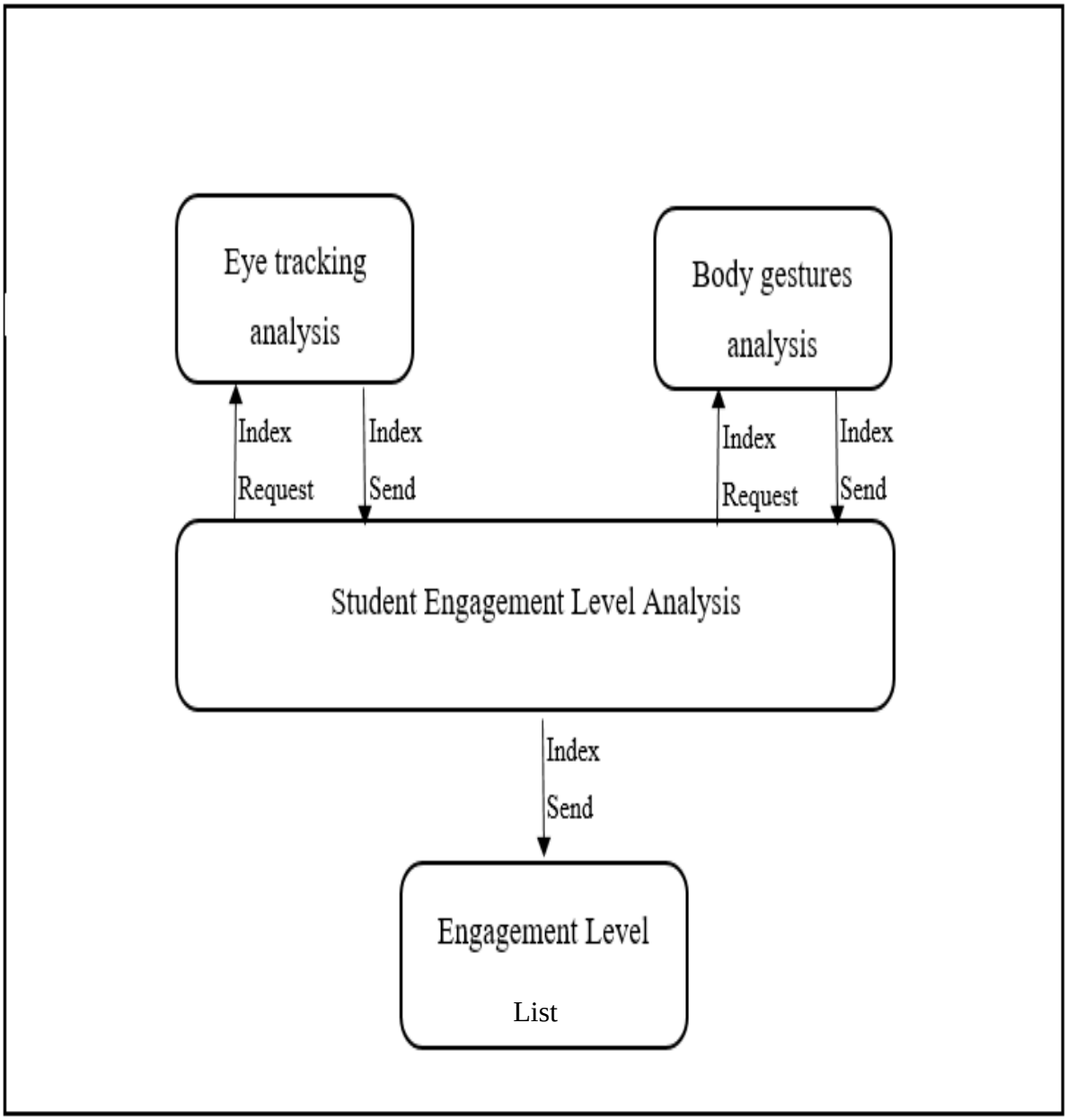


Figure 2.3: Conceptual framework of the proposed system

Chapter 3: Methodology

3.1 Introduction

Kothari describes Methodology as a way to systematically solve the research problem (Kothari, 2004). In this study, different methodologies were used with the intention of developing a prototype for tracking student engagement levels, as defined for this study in chapter 2. Student engagement analysis involved the relationship between the body and mind such that; the body shows what is going in the mind. Library research on student engagement was used to obtain information which was used to guide and determine how student engagement could be measured; in addition to how existing technologies and research could be used. Objected Oriented of System Design was applied to analyse and design the prototype, as well as to test the reliability of the prototype.

3.2 System Development Methodology

As mentioned above, Object Oriented methodology was chosen to be used in developing the system in this research. Object Oriented Methodology breaks down a new complex system into small components with different outputs that can be shared and reused. This makes it easier to build a higher quality solution - or more valid solution – quickly (Bordoloi and Lee, 2008). Particularly in this study, Eye tracking was one components of the system, body movement tracking was done by another component and lastly, student engagement levels were tracked by a different component. Additionally, it allowed for the modifications in one component without having to change the others, which made it easier to make adjustments so as to increase the usefulness of the prototype to be developed. Object oriented methodology Involves three macro stages; Object Oriented Analysis (OOA), Object Oriented Design (OOD) and Object Oriented implementation (OOI). Each of these phases overlap with each other.

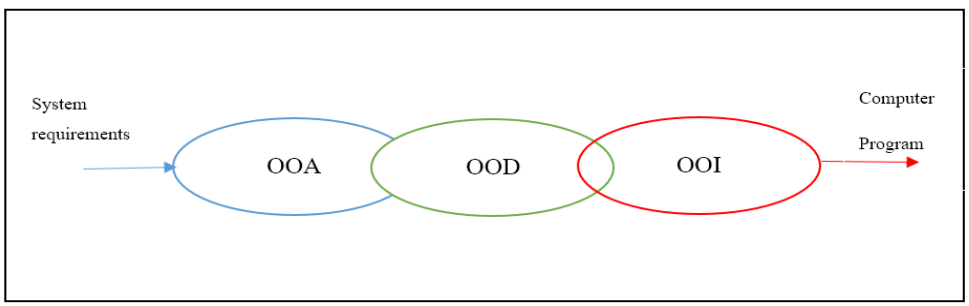


Figure 3.1: Major stages in object oriented methodology

3.2.1 Object Oriented Analysis (OOA)

OOA involves creating a model that represents the user's view of the system, which builds a use-case model. This is based on identifying different classes in the system and how the relate to one another. In this research, systems requirements was gathered from library research and used to create the use-case diagram. The use-case diagram helped in identifying and demonstrating the different ways that a user was expected to interact with the system.

3.2.2 Object Oriented Design (OOD)

OOD generally involves designing the classes and describing the attributes, methods and structures based on system requirements for the user interface, implementation and data processing, and data access. This research intended to use tools such as Star UML to design the different diagrams such as the Use case and data flow diagrams, and then used Microsoft Word to document the research.

3.2.3 Object Oriented Implementation (OOI).

OOI involves translating the developed design into code using a given programming language or languages and then incrementally testing. In this kind of testing, the system is tested at different stages of development, so as to increase chances of having a working system. In this research, different components which represented the system modules were implemented using PyCharm and tested individually. Once they passed, they were integrated and tested as a single system.

3.3 Research Design

Experimental research design was used in this study, as it is focused on causal validity such that; eye tacking plus upper body gestures (independent variables) caused the measurement of student engagement (dependent variable). An experimental research design is used where the researcher has defined the research questions that they are to address, made an accurate and reliable measure for the variables and realized the population to be sampled, as well as the technique to be used (Mitchell, 2015). Data for the variables was obtained from experiments that had been conducted, and a pre-trained model developed for use, while observing the eye movements of different students during an online class, as well as the detection of upper body gestures and lastly, the engagement level that has to do with; how much the student was paying attention and they seem to put in to grasp the content for example, raising their hands to answer questions.

3.4 Research Population

This research was centred on university students who were doing their course part-time. It was assumed that these students were generally self-driven, with an objective to gain the most out of the classes that they attend such that, they would try as much as possible to pay attention in class. This population was used test the working prototype, and to gauge how well it tracks student engagement. To enable faster working and debugging of the system and because the study intended to only use the data for testing, the population was sampled and then used as described in the following sections. The Organisation for Economic Co-operation and Development (OECD) average for the number of students in a typical class room is 21.4 students (Rampell, 2009). Compared to this, a study done in Kenya recommended that an average class student-teacher ratio of 40:1 (Nyiwa et al., 2017). Based on these two facts, and domain knowledge by the researcher, the study assumed an average population of 30 students, and then sampled from the same population size.

3.5 Environment Set Up

The research intended to take advantage of the COVID-19 pandemic, which had led to an increased adoption of online classes – in the case of Strathmore University, where the research was carried out. Simple random sampling was used to choose three to five students to participate in the research at will, because it is easy to use and lacks bias (Levy and Lemeshow, 2013). The three to five students' number covers a least 10% to 16% of the population, which the researcher considered to be ideal for efficient testing of the prototype thus, eased the cost of acquiring computing resources needed for the research. The random selection was done on a list of names found on the class – list of students who had registered for that particular class. While the students were monitored during the class through videos from their web, ET and gesture features were then extracted, and used in the developed prototype.

3.6 Data Collection

Data collection was done via video cameras or web cameras or phones and laptops. The participants chosen to get involved in the research were requested to allow for a recording of themselves to be used, as they attend the online classes. The videos were then obtained by the researcher, who then extracted the needed data by passing them through the implemented prototype so as to acquire the Engagement level as an output. Several instructors were then chosen to evaluate the videos individually and provide their feedback. Their analysis was compared to that done by the prototype to determine the accuracy. Similarly, a group of

students were chosen to analyse the effectiveness of prototype, which helped in determining what to change and how to adjust the system.

3.7 Data Analysis

Once the data had been acquired as described above, the eye movements were analysed by the ET analysis algorithm so as to extract a numerical index on how engaged the student was, based on the eye movements alone. Secondly, the body gestures were analysed independently and a numerical index extracted to gauge the engagement index for the student during the lesson. The eye tracking index (ET index) was considered to have more weight than the Gesture Analysis Index (GA Index) due to the level of analysis that was implemented and so, the GA index served a more complementary role in determining the student's engagement level– estimated engagement level. On the other hand, the rating given by the student and the teacher were also averaged and were then considered to be the actual engagement level for the student; grouped according to table 2.1. Considering that the experimental research is done by observing and analysing students, the qualitative data was grouped using an ordinal scale that would then provide a meaningful interpretation of the data analysed; the data was divided into 3 categories where students who did not meet a third of the expected engagement level were shown to have a low engagement level while the students who had a value above two-thirds of the expected engagement level were said to have a high engagement level. In between, were the students who were said to have an average engagement level.

Average Engagement Index Score	Student Engagement Level	Indicator Colour
6.7 - 10	High	Green
3.4 – 6.6	Average	Yellow
1 – 3.3	Low	Red

Table 2.1: Proposed student engagement rating

3.8 Description of Algorithms used

3.8.1 Real-time eye tracking using OpenCV and Dlib

This algorithm was tailored to fit this research and used to track the eye movements and collect data for ET analysis. A special shape file was added to the environment which provided key-points for 68 facial landmarks that were then used by the Dlib functions to detect the faces

using a pre-trained model, in any stream of images that were passed (Kazemi and Sullivan, 2014).

The ET algorithm follows 12 steps to extract the eye pupil coordinates which are then used in the analysis. The steps are as follows: The Dlib, OpenCV and Numpy libraries are imported into the environment. The detector and predictor variables are loaded; Dlib frontal face detector was chosen since the focus is on the student looking at the object that they are thinking about. If the student was to look away from the computer, then this would mean that they are not really focusing on what is being presented – suggested by the EMH discussed in chapter 2. The predictor variable holds the shape predictor file for face detection. The default web camera was opened to capture a video from which, images are obtained and converted to grayscale, so as to reduce information and enhance the processes that followed. A face is detected from the image and the coordinates stored in a shape file which is then converted to a Numpy array. A function to convert the shape to a Numpy array is created to do this. The values are then used to draw a green rectangle around the face to show that it has been detected. Key-points 36 to 47 are used to get the coordinates, which are used to create a mask of the image. The mask segments out the eyes (area of interest) by drawing the areas outside the eyes in black and the eyes in white. The key-points were drawn in blue to indicate that the eyes had been found. The white areas are expanded using cv2.dilate, and then cv2.bitwise is used to convert the image such that, the eyeballs remain to appear in black. The resulting image is again converted to grayscale so that thresholding can be done.

Thresholding is done using three steps of erosion of the image – to sharpen the boundaries in the image, blurring the image – to reduce noise from the image – and lastly, dilation – to increase the size. The image was inverted to make the pupils white to prepare the image for contouring. Cv2.contouring is used to find contours which are sorted from largest to smallest, with the assumption that the largest contour in that region would be the iris, and thus the centre would be the pupil. The centres of our iris are then calculated using cv2.moments to find the coordinates, which would provide the coordinates for our pupils. Using this as the centre, red circles are drawn around this area to indicate that the position had been detected. The pupils' x and y coordinates are compared to the key-points over a period of time, to evaluate whether the student is engaged in class and an index for the same given.

3.8.2 Body gesture recognition using OpenCV and background subtraction

This algorithm borrowed from Gogul Ilango's algorithm for hand gesture recognition using python and OpenCV. The algorithm relies heavily on background subtraction technique, to find the hand and analyse its presence; this is achieved through the use of running averages (Ilango, 2017). The algorithm was modified and used based on the following steps: The algorithm first segments the hand region using background subtraction method – done by providing the computer with 30 frames so that computes the running average for the background coordinates and have everything in black. This is accomplished by a running-average function. A new frame is then introduced which has the hand present to serve as the object in the foreground. To separate this, the absolute difference between the frames is calculated and the object is then be separated. This is accomplished by a segment function. Motion detection then follows so that the hand region could be detected. Thresholding is done so that, the background of the image remains in black while the foreground (hand) remains to be the only white area left; so as to make contouring effective. Contours were extracted from the hand region and used for gesture analysis.

The scope of this research reduced the use of this algorithm to the detection of the hand when a student raised their hand to get the instructors attention. The presence of the hand gave a Gesture Analysis index of 10 that complements the ET index.

3.8.3 Student engagement analysis using Numpy and Tkinter

Student engagement analysis mainly involves analysing the values and using them to order the list of students. To serve this purpose, Numpy package for arrays in python is used as it provides the necessary functions that include; averaging functions and sorting functions to sort the values and calculate the array. Additionally, the insert and delete functions are used to create a function to update a FIFO data-structured queue. On the other hand, Tkinter is the standard GUI package for python. It is used in the implementation of this prototype to demonstrate how the names of the students would be color-coded based on their analysed engagement level indices and displayed to the instructor as the final output.

3.9 Research Quality

Research Quality can be evaluated by the concept of validity and reliability; validity refers to the accuracy of the measure, method or technique used while reliability is about the consistency of the same used in producing results in the same conditions. In this research, the estimated

level of engagement was compared to the actual values as obtained from the student and from the instructors, so as to measure the accuracy and thus, determine how valid the algorithms were, as used in this research. Lastly, so as to test the reliability, the research was repeated for different recorded videos to determine whether it would give the anticipated results with the same level of accuracy repeatedly.

3.10 Ethical Considerations

Since the main data for this research was obtained from observing and analysing the recordings of student while attending an online class, the researcher used the recorded videos with permission of the students and owners of the videos – many of which were permitted to be used for academic research purposes. The students were also made aware that the teachers were involved in assessing the videos and give a score on their engagement levels at that time. On the other hand, so as to not interfere with the results, the videos were chosen at random (whenever possible), to reduce awareness that the students are being watched and therefore, get more truthful results. The students were also informed before the lesson.

Chapter 4: System Analysis, Design and Architecture

4.1 Introduction

This chapter describes the Analysis, Design and Architecture of the prototype to measure online student engagement. System analysis section has three subsections that bring out the gathering of requirements for the prototype, functional requirement analysis of the prototype as well as the non-functional requirement analysis of the prototype. The system architecture is sketched out to visualize the function of the system, which is then elaborated on in the system design section using Unified Modelling Language (UML) diagrams; Use case diagram, sequence diagram, Entity Relationship Diagram (ERD), Database diagram, class diagram and a Wireframe of the system.

4.2 System Analysis

System analysis is used to define a methodology that attempts to understand the system requirements, then gives a quantifying and well-ordered approach to assessment of well-articulated concerns for the researcher to consider, among other alternatives (Wong, 2010). It is done with the main intention of developing a logical representation of the components that build up a system (Tilley and Rosenblatt, 2016). It is important that the system effective performs as desired and thus, questions such as; who will use the system, what will they want the system to achieve, where will they want this and when (Dennis et al., 2015). With this in mind, system analysis was done when developing a prototype to measure student engagement so as to establish the desired outcomes. It involved the analysis of requirements gathered from literature on student engagement evaluation, as well as reports of similar systems established before. It was done in 2 stages of requirement gathering and analysis of functional and non-functional requirements.

4.2.1 Requirement gathering

Requirement gathering is the user expectation of what the prototype should do and how it should do it. Requirements gathering for the prototype to measure student engagement was done in line with the objectives in chapter 1, and the data analysed in light of the research questions as listed.

4.2.2 Functional Requirements

Functional requirements analysis specifies what the prototype should do at the very least in regard to function, behaviour, input and output, so as to achieve the desired goal of measuring student engagement. The functional requirements of this prototype included:

i. The prototype should collect eye movements and body gestures data of different students
ii. The prototype should process the data collected and create an engagement level index for each user
iii. The prototype should order the list of users according to lowest engagement level index to the highest
iv. The prototype should present a summarised report for individual student engagement levels
v. The prototype should show when a certain user has not been captured for engagement level measuring (not present in class).
vi. The prototype should allow the system administrator to access recently analysed data.

4.2.3 Non-functional Requirements

Non-functional requirements are the qualitative attributes the prototype must have to ensure the goal is achieved. Though these attributes are intangible and cannot be represented in the prototype directly, not having them will be detrimental to the usability of the prototype. For the prototype, they include:

i. Availability**;** describing the quality of being ready to be used when required, the prototype should be available throughout a given session such as, a full online class session.
ii. Reliability**;** describing the quality of consistency in performance, the prototype should be reliable in identifying students with low engagement levels every time it is used.
iii. Accuracy**;** describing the quality of a measurement being close to what is true, the prototype should be accurate in measuring the engagement level of the student and classifying correctly.

iv. Usability**;** describing the quality of simplicity in terms of user interaction with the system interface, the prototype should have a simple interface with a few hidden menus to improve usability.

v. Security**;** describing the quality of restricted access to user information, the system should allow only the system administrator to access the report for a given session.

vi. Performance**;** describing the quality of time taken to take an input, process data and give the desired output, the prototype should take the least amount of time to capture gestures and eye movements, process and measure a single student's engagement level

4.3 System Architecture

Once a student requests to join the class, the request is sent to the instructor who admits the student into the class. The admission list module then keeps a record of students who are in attendance during the session so as to provide the camera ids or computer ids, which are used to sequentially get data from the web cameras. Both algorithms take in this data and break down the videos into images from which, the features are extracted and analysed. The images are turned into grayscale, then processed for thresholding and lastly, contours are found to be used for feature extraction. The Eye Tracking and Gesture Analysis algorithms, each functioning as separate modules, provide an eye index and gesture index respectively which are used to rate the engagement level. The rating is used by a forth module which sorts the engagement indices for all students, so as to rank the different students in a color-coded format. Figure 4.1 depicts how students 1, 2, 3 and 4 are ranked according to an ascending level of engagement. The student engagement analysis module the collects and compares the indices for all the students. The admission list module gets the ordered array list to create a view for the instructor, which shows the students engagement levels, while maintaining the order in the original, separate list for the collection of the data.

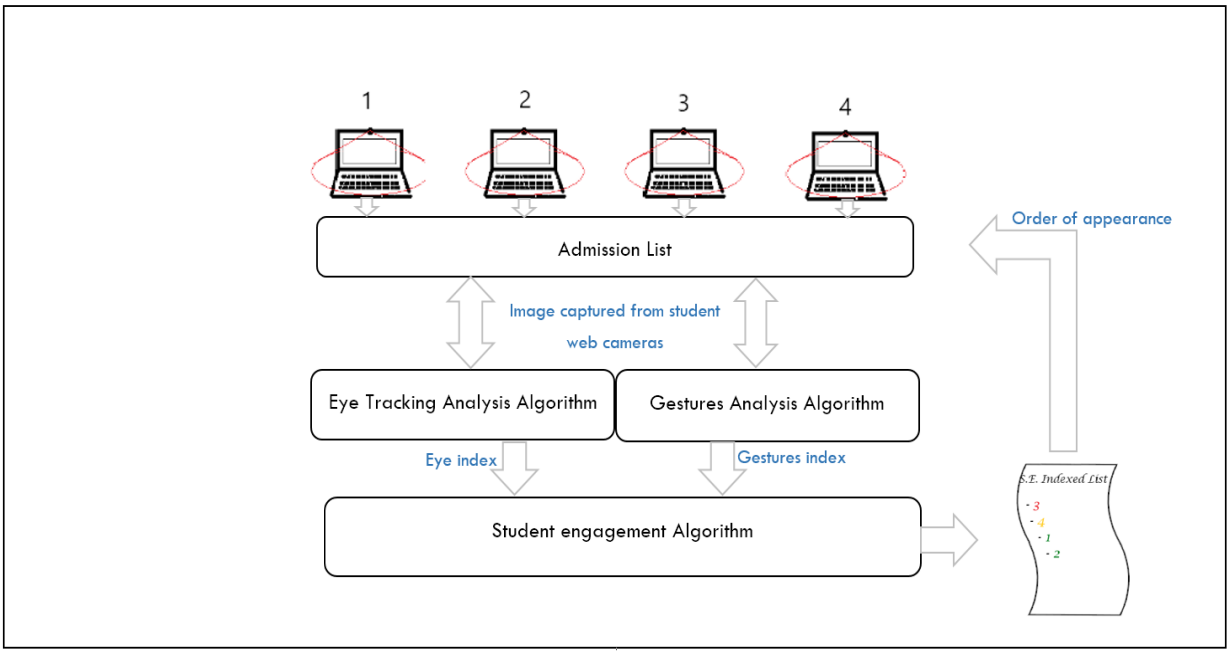


Figure 4.1: System Architecture

4.4 System Design Diagrams

4.4.1 Use Case Diagram

4.4.1.1 Primary use cases

Turn on camera: The student turns on his or her camera which then changes the status in the system, so as to allow capturing of images.

Admission record: Since the analysis is done sequentially, this admission record provides a list of all students that are to be monitored. It includes uses cases for the student to request to join the class (register) and for the instructor to grant access.

Eye tracking analysis: The system uses this to breakdown the captured or grabbed images and extract the eye features that it uses to create an engagement level index based on the eyes only; referred to as the eye index.

Gesture analysis: The system uses this to again breakdown the captured or grabbed images and instead, extracts features in the images that it uses to create another engagement level index based on the gestures only; referred to as the gesture index.

Student engagement level: The system uses this to rate the final engagement level of the student based on the eye index and gesture index. It updates the algorithm to be used for this,

as well retrieving analysed data which form the basis on which the rating is done. The uses case results in analysed data for storing and viewing by the instructor.

4.4.1.2 Actors

Four actors are considered in this use case. The actors are; *student* being monitored, the *instructor* who needs to know the engagement levels of the students and lastly, the *system administrator* that modifies the algorithms accordingly to maintain the system.

4.4.1.3 Assumptions

Several assumptions that have been made are first, the student actor represents more than one student. Second, the prototype itself is treated like an actor that interacts with different use cases so as to rate the students' engagement levels. Lastly, considering that this is a prototype, the developer also interacts with the system to modify the algorithms to improve its accuracy and fix other issues that may arise.

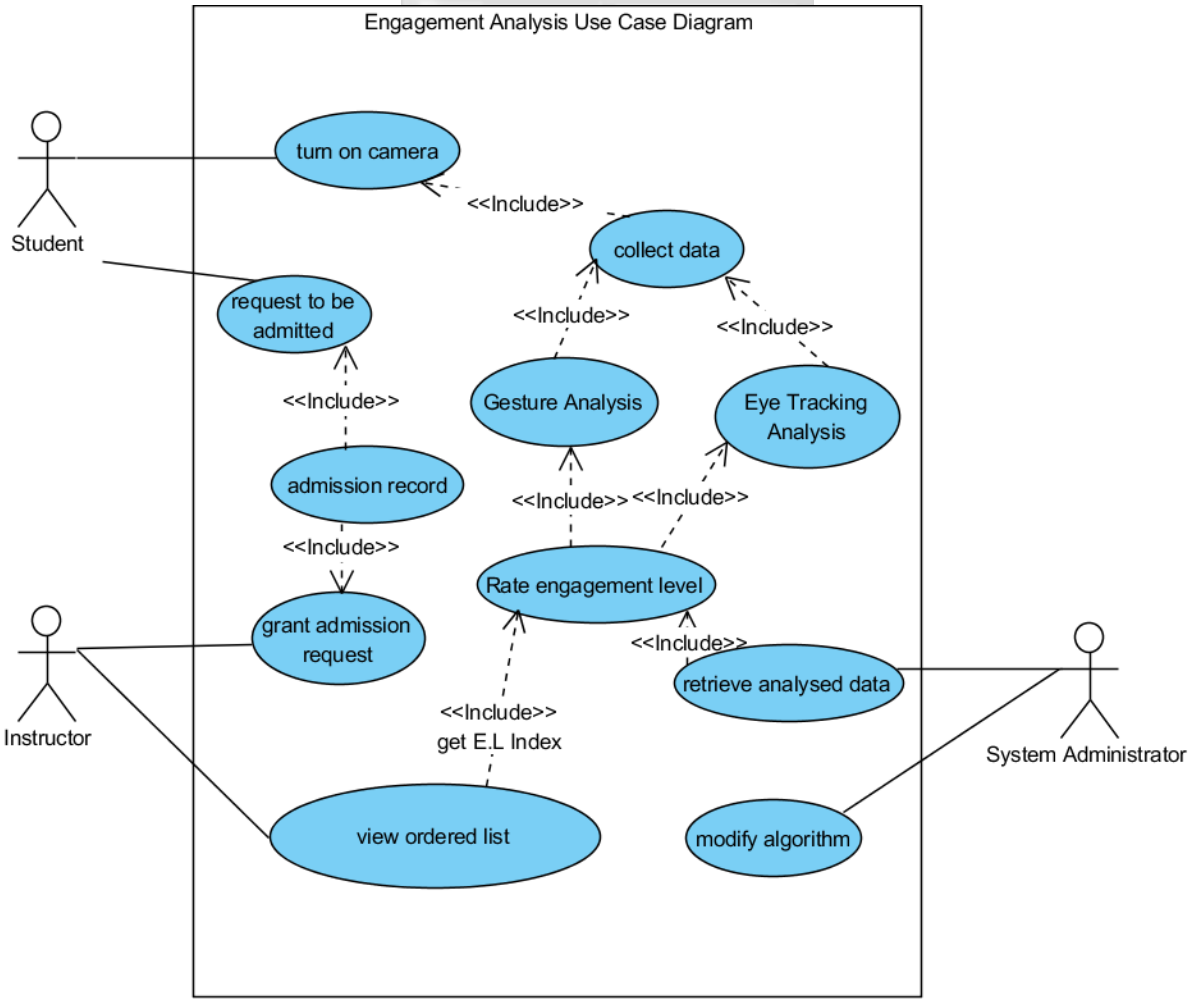


Figure 4.2: Use Case Diagram

4.4.1.4 Fully dressed model

Primary Actor:

- Student

Stakeholders and interests:

- Student**:** wants the instructor to know when they have understood the concept or not
- Instructor**:** wants to know which student requires special attention and how many are engaged in the class.
- Developer: An off-stage actor that wants to ensure the system is monitoring individual student engagement level as effectively as possible.

*Preconditions***:**

- The student requests to join the class.
- The instructor admits, and student is added to list.
- Student camera status is on.

Success Guarantee (Post conditions)

- The student's engagement level is listed.

Main Success Scenario

- Student turns on camera
- Student requests to be admitted
- Instructor admits student
- System monitors student's eyes
- System monitors student's upper body gestures
- System calculates student's engagement level
- Instructors views an ordered list of student's engagement level

4.4.2 Sequence Diagram

Interaction diagrams are used in OOD to show how different objects of the system interact with one another and among them, is a sequence diagram that illustrates the interactions in fence like format. It has been chosen as it simple and easy to represent the time-ordering (sequencing) of messages. The student request for admission into the session. The system forwards the request to the instructor who accepts and the name of the student is added to the list which is visible to the teacher. The engagement level analysis queries from a looped list in admission, the identity of the next student whose data is to be collected. The id is then used to capture the

recording from the particular student's camera, and images are then passed to eye tracking analysis (ET Analysis) and Gesture Analysis. Both reply with indices that are then used to analyse the engagement level and a final index is sent to admission, to be used to order the list of attendees in ascending order, according to their engagement levels.

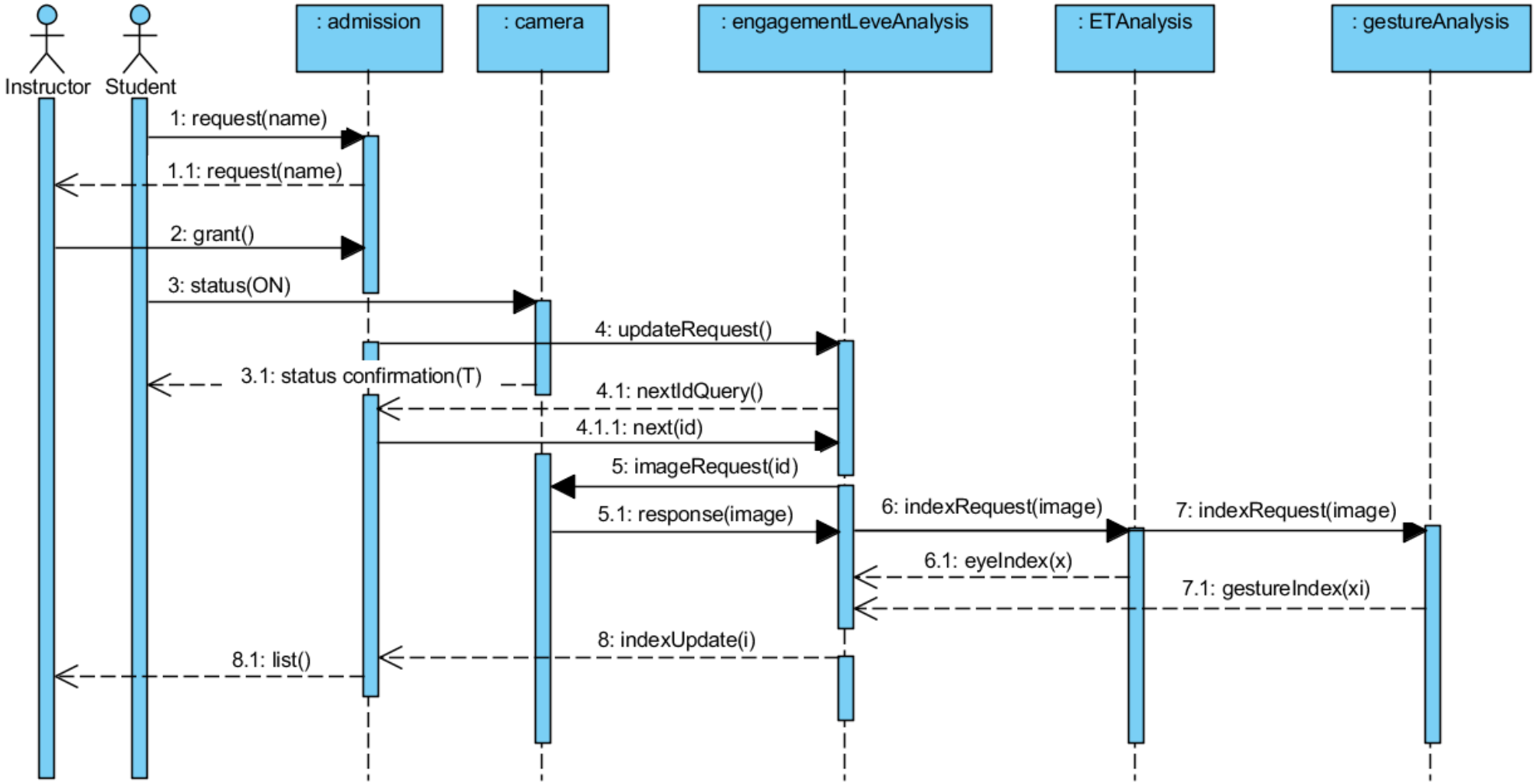


Figure 4.3: Sequence Diagram

4.4.3 Process Model Diagrams

4.4.3.1 Context Diagram

Figure 4.5 below, is a context diagram for the engagement level monitoring prototype and the immediate external entities that interact with prototype. The student sends an admission request which is granted by the instructor in charge of the given session. The prototype then gets input data in video format, breaks it down and analyses to get the engagement levels of the different students. It retrieved data to aid in classification and then adds the new data it has analysed. Finally, the instructor gets a list highlighting the engagement level of students.

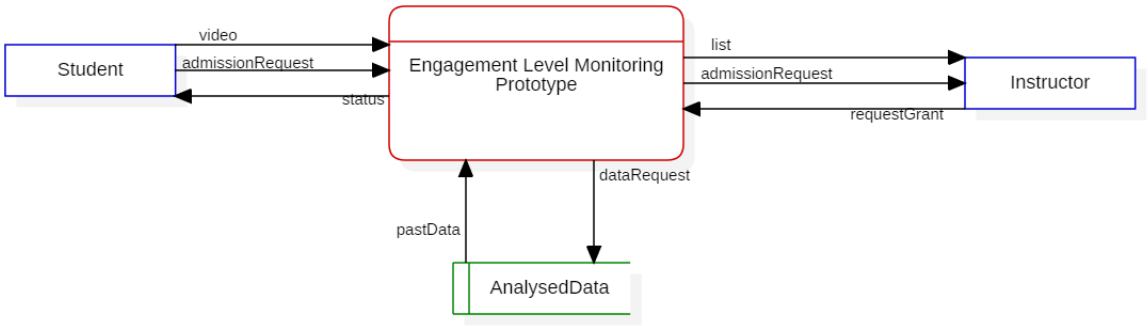


Figure 4.4: Context Diagram

4.4.3.2 Level 1 Data Flow Diagram

The context diagram in figure 4.5 is expanded so as to elaborate the second level processes of the engagement level monitoring prototype. This is shown in a figure 4.6, a level 1 data flow diagram providing a detailed breakdown of the prototype. The data flow through 5 main process which; maintain the admission list or list of present students to be monitored, manage data retrieval from web cameras, calculate the ET index and gesture index and lastly, rate and list the engagement level of students, which is the presented to the students. Once the student has been granted access into the sessions, the engagement level prototype begins to sequentially monitor all the students in the list in a loop. The list has a camera id for each student which is used to find the camera and get footage. The camera input data is then passed to 2 different processes to get the ET index of the student, as well as the gesture index from the other student. The engagement tracking analysis then loads the analyzed data to build the classification model it will use to classify or rate the engagement level of the student based on the ET and gesture indices. The rating is then passed to the admission list which will reorder the list in ascending order, with the lowest rating representing the least engaged student.

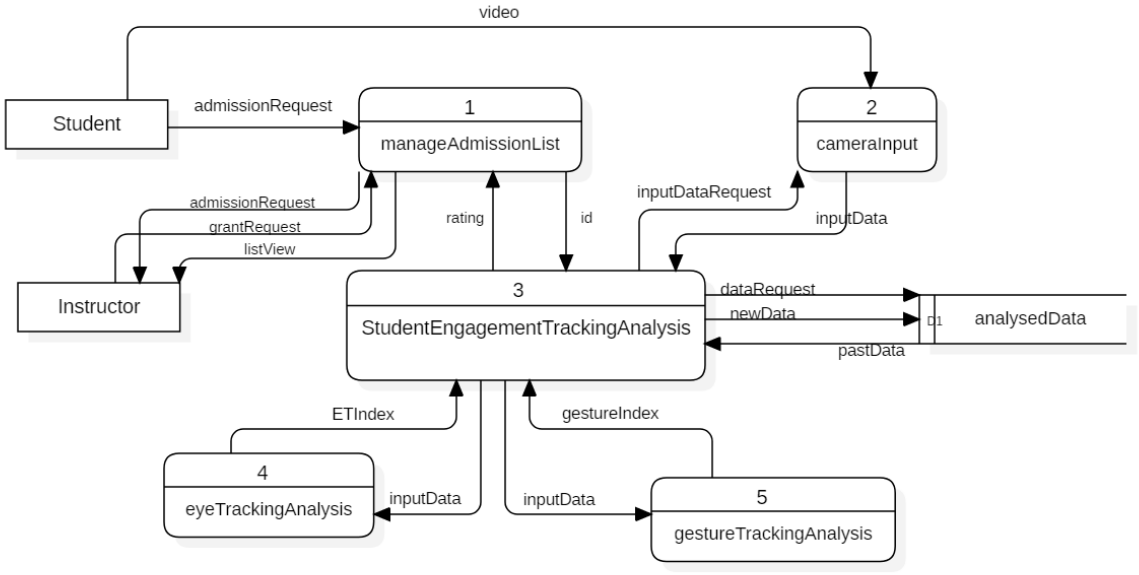


Figure 4.5: Level 1 Data Flow Diagram

4.4.4 Partial Domain Model

Figure 4.6 is a domain model diagram representing the real-world objects, also known as conceptual classes of the prototype to be used, as well as their attributes and the associations between the different classes. The domain model comprises of 5 classes which are; AdmissionList, Camera, ELAnalysis, GestureAnalysis and ET analysis. The admission list serves as interface for the instructor to admits students into the class and it provides a list for the prototype's algorithm to use to track the student engagement. Once a student is added, the name is added to the bottom of the list and tracking is done sequential in a First-in-First-Out (FIFO) order. The Camera class is used to obtain multiple images or video from different student cameras, using the IDs provided by the admission list. The Engagement Level Analysis class or ELAnalysis class calculates the engagement level of the student using the ET index from the Eye-Tracking analysis class of ETAnalysis class, and the GA index from the GestureAnalysis class. It then provides the Engagement Index (EI) to the admission list where a view has been created for the instructor.

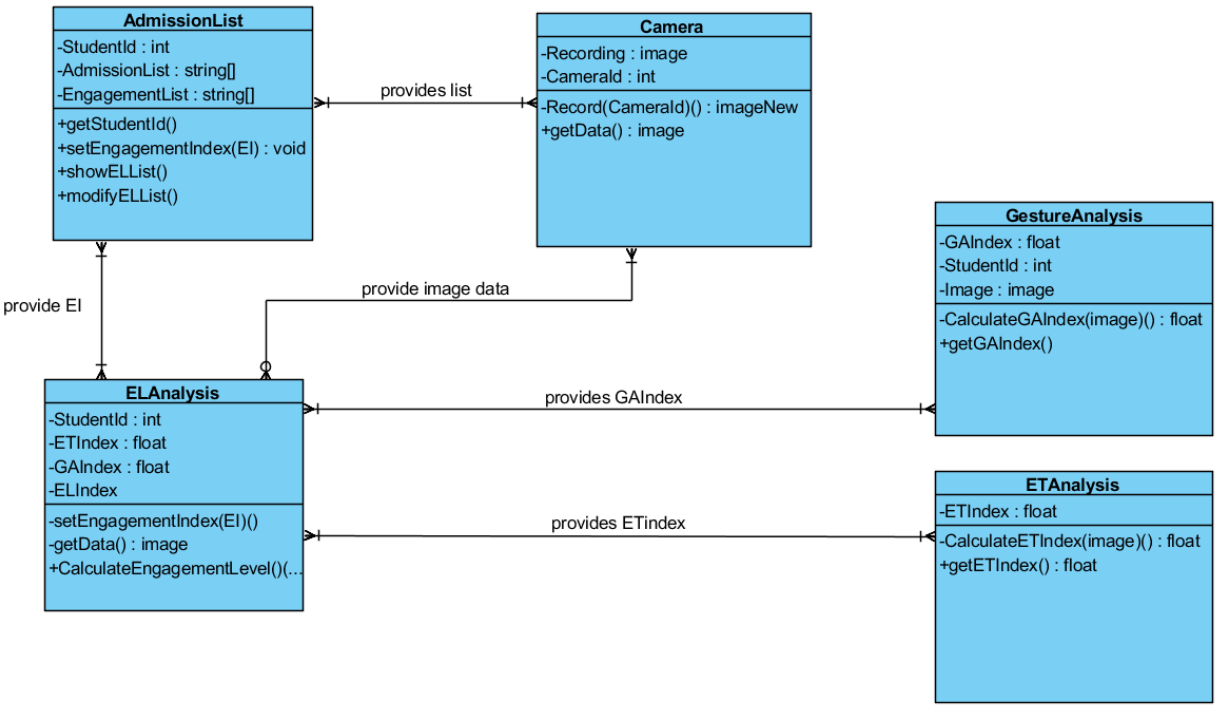


Figure 4.6: Partial Domain Model

Chapter 5: System Implementation and Testing

5.1 Introduction

This chapter delves into the process that was used in creating a prototype to measure online student engagement, using eye tracking and body movement analysis. It begins by demonstrating how the algorithm explained in Chapter 3, is used to process the raw data captured from the video camera. The system implementation section is broken down into the 4 main modules to describe how they process data individually, then it concludes by summarizing how modules work together as one. Secondly, it explains the testing phase of the system and provides the accuracy levels attained. Lastly, it explains how the system validation process that was carried out.

5.2 System Implementation

System implementation was done using Component-Based Development (CBD) software development process. This involved splitting the entire system into modules that could then be implemented and tested separately. The four modules are described in the previous chapter and were implemented as seen in the sections that follow.

5.2.1 Eye Tracking Module

In tracking the eyes, the algorithm first had to detect the face, then single out the eye regions, so that the algorithm can continue from there. The environment was prepared for eye-tracking using Dlib version 19.21.1 to achieve this as it most importantly, provided 68 facial landmarks that were used to detect the full frontal face from images captured from the webcam. First, the detector function was loaded to the environment and stored in the variable called detector. Secondly, the face predictor file consisting of the 68 facial landmarks was loaded into the environment and stored in the predictor variable. Finally, a variable called Rects was created, where all the faces detected in an image were stored, and a rectangle drawn around the detected face in green. The face detected were stored in a shape file, which then needed to be converted into an array so as to enhance use of the coordinates in python. For this reason, a function called *shape_to_np* was coded to take in the shape key-points coordinates and the data-type for conversion, and then return an array of coordinates (x and y) for the all the facial key-points detected. The code segment that follows demonstrates how this was accomplished using python and PyCharm coding environment:

```
detector = dlib.get_frontal_face_detector()
predictor = dlib.shape_predictor('C:/Users/Mido
Austin/AppData/Local/Programs/Python/Python38/shape_predictor_68_face_landmarks.dat
')
rects = detector(gray, 1)
for rect in rects:
    #draw rectange around face
    x1 = rect.left()
    y1 = rect.top()
    x2 = rect.right()
    y2 = rect.bottom()
    cv2.rectangle(img, (x1, y1), (x2, y2), (0, 255, 0), 3)
    face_pos = cv2.rectangle(img, (x1, y1), (x2, y2), (0, 255, 0), 3)
```

Listing 1: Code to load predictor and detectors files

```
#this function converts the facial landmarks from the dlib predictor into a numpy
array
def shape_to_np(shape, dtype="int"):
    # initialize the list of (x, y)-coordinates
    coords = np.zeros((68, 2), dtype=dtype)

    # loop over the 68 facial landmarks and convert them
    # to a 2-tuple of (x, y)-coordinates
    for i in range(0, 68):
        coords[i] = (shape.part(i).x, shape.part(i).y)
    # return the list of (x, y)-coordinates
    return coords
```

Listing 2: Code to convert shape file to array of coordinates

Once the environment had been set, the next step involved capturing and processing the image so that the eyes and particularly the pupils, could be segmented out for tracking. This involved creating a video object using the video capture function provided by openCV, which was stored in cap, that was set to capture images from default camera (web camera) every 100 milliseconds so as to ease the load of processing images – the zero parameter passed indicated the use of the default camera. The read function from the loaded video capture object then gave two outputs consisting of the capture image – in form of array values - and a Boolean value for the captured image. the wait key function in cv2 was used to detect when a key has been pressed to terminate the image capturing process using the release function as seen.

```
cap = cv2.VideoCapture(0)
ret, img = cap.read()
thresh = img.copy()
cv2.imshow("image", thresh)
if cv2.waitKey(1) & 0xFF == ord('q'):
    break
cap.release()
```

Listing 3: Code to capture images

Figure 5.1: Detected face image

The captured image was stored in a variable called thresh, as this was then used to process the image by thresholding, to extract the position of the pupils to be used in the eye tracking analysis. The faces predicted from the captured images are then stored in shapes which is then converted into a numpy array using the function in listing 2. The left eye and right eye positions (obtained from the key points 36 to 48 stored in left and right variables, as seen in listing 4) are masked and segmented out such that, everything else outside the eye region was converted into black. The processing followed the steps explained in chapter 4 of converting the image to grey scale, thresholding and then using the cv2.cvtcolour function to create a grey image using the COLOR_BGR2GRAY value. The black and white image was then passed through the thresholding function provided by cv2 so as to provide a binary image. The erode function from cv2 eroded the binary image to eliminate the background. The dilate function was used to add pixels to the boundaries identified in the image and then the image was blurred so as to remove noise from the image, leaving a more clear image. The clear binary image was inverted so that the eye region appear as white and the pupils and everything else, as black as seen in figure 5.3.

```
left = [36, 37, 38, 39, 40, 41]
right = [42, 43, 44, 45, 46, 47]
```

Listing 4: Code to set key-points for the left and right eye

Figure 5.2: Key-points predicted on detected face image

Figure 5.3: Eyes after Thresholding and inversion

So as to proceed with contouring, which requires the object to be detected to be in white – which in this case, is our eyeballs, seen in figure 5.3. Contouring then followed using the contour function in listing 5. The function took in 4 parameters; the processed image in thresh, the mid region between the eyes, the image capture from the camera and the side of the right eye. The function used the cv2.findcontours function from the opencv library, to find the two largest contours within the region – these would be assumed to be the iris. Next, the function used cv2.moments function to capture the centre of this contours, which then provided the estimated coordinates of the pupils in the image, shown by drawing red circles around the pupils in the first image capture from the webcam. This block of code was put in try statement such that, if no eyes were detected from contouring, then nothing would be done because there would be no contours to find the centres from. Lastly, everything was included in a while loop for the purpose of streaming

```
def contouring(thresh, mid, img, right=False):
    cnts, _ = cv2.findContours(thresh, cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_NONE)
    try:
        cnt = max(cnts, key=cv2.contourArea)
        M = cv2.moments(cnt)
        cx = int(M['m10'] / M['m00'])
        cy = int(M['m01'] / M['m00'])
        if right:
            cx += mid
        cv2.circle(img, (cx, cy), 4, (0, 0, 255), 2)
        print(cx, cy)
    except:
        pass
        print(0,0)
```

Listing 5: Code to find contours in image

```
shape = predictor(gray, rect)
shape = shape_to_np(shape)
mask = np.zeros(img.shape[:2], dtype=np.uint8)
mask = eye_on_mask(mask, left)
mask = eye_on_mask(mask, right)
mask = cv2.dilate(mask, kernel, 5)
eyes = cv2.bitwise_and(img, img, mask=mask)
mask = (eyes == [0, 0, 0]).all(axis=2)
eyes[mask] = [255, 255, 255]
mid = (shape[42][0] + shape[39][0]) // 2
eyes_gray = cv2.cvtColor(eyes, cv2.COLOR_BGR2GRAY)
threshold = cv2.getTrackbarPos('threshold', 'image')
```

```
_, thresh = cv2.threshold(eyes_gray, threshold, 255, cv2.THRESH_BINARY)
thresh = cv2.erode(thresh, None, iterations=2)  # 1
thresh = cv2.dilate(thresh, None, iterations=4)  # 2
thresh = cv2.medianBlur(thresh, 3)  # 3
thresh = cv2.bitwise_not(thresh)
contouring(thresh[:, 0:mid], mid, img)
contouring(thresh[:, mid:], mid, img, True)
for (x, y) in shape[36:48]:
   eyes = cv2.circle(img, (x, y), 2, (255, 0, 0), -1)
   print(x,y)
```

Listing 6: Code to process image and extract coordinates

Figure 5.4: Coordinates shown on captured image

Once the coordinates for the pupils and the 12 key-points for the eyes could be extracted, Analysis of the tracked eyes followed. Since the predictor used was frontal face detection, it meant that the algorithm simply returned a null value if the student face was not detected, suggesting that they may not be engaged. If a face was detected, then the saccades and fixations were analyzed to determine the student engagement index, based on ETH discussed in Chapter 2, section 2.3. The saccades and fixations were analyzed by first, creating an array to hold the positions of the eye at a given time, and then checking to the new positions coming in to determine if they have moved in either direction or if not. If the eyes are detected the ET index was set to 5 and if the saccades are detected, then the ET index is set to 10. If none of the conditions are met, then the ET index remained to be 0. The output of this module – the ET index, is then passed to the Engagement analysis module for analysis; explained in section 5.2.3.

5.2.2 Gesture Analysis Module

The upper body gesture analysis module was implemented in two major steps; segmenting the hand region from a video sequence and the finding the position of the fingers in the video sequence. Using the face detector from dlib described in the above section, the face region was detected and a rectangular boundary draw to define the region to estimate where the hand would be when the student attending the class raises up their hand.

After getting the regions of interest, the first step of segmentation was carried out. This involved background subtraction where the foreground and background of the image was

detected. Running averages was used to detect the background of our image, by making the program analyse frames capture by the webcam, and then detecting an object in a new frame as the foreground. The absolute difference between the background and foreground was used to recognize the object/the hand. All these was encapsulated in a function which takes in a frame and a value to compute the running average by. It checks if it is the first frame or not, and then does the computation as shown in listing 5.6:

```
# global variables
bg = None

#--------------------------------------------------
# To find the running average over the background
#--------------------------------------------------
def run_avg(image, aWeight):
    global bg
    # initialize the background
    if bg is None:
        bg = image.copy().astype("float")
        return

    # compute weighted average, accumulate it and update the background
    cv2.accumulateWeighted(image, bg, aWeight)
```

Listing 7: Code to compute running averages

The next step was to segment the hand from video captured from the camera. This involved using cv2.absdiff() to calculate the absolute difference between the new frames captured (foreground) and the background obtained by calculating the running average. After this, thresholding was done on the image to reveal the object/hand only in the image, then contouring was done, and only the contours with the largest area were taken. This was also encapsulated in a function called segment which takes in a frame and a thresholding value and returns the segmented image, as well as well the frame after thresholding. This can be seen on listing 8.

```
def segment(image, threshold=25):
    global bg
    # find the absolute difference between background and current frame
    diff = cv2.absdiff(bg.astype("uint8"), image)
    # threshold the diff image so that we get the foreground
    thresholded = cv2.threshold(diff, threshold, 255, cv2.THRESH_BINARY)[1]

    # get the contours in the thresholded image
    (_, cnts, _) = cv2.findContours(thresholded.copy(), cv2.RETR_EXTERNAL,
cv2.CHAIN_APPROX_SIMPLE)

    # return None, if no contours detected
    if len(cnts) == 0:
        return
    else:
        # based on contour area, get the maximum contour which is the hand
        segmented = max(cnts, key=cv2.contourArea)
        return (thresholded, segmented)
```

Listing 8: Code to segment the hand region

With the two functions in place, the main code for the module was then written which initialized the weight for the running average function to 0.5 so as to set the number of frames of which we want to use to compute the running average. The camera was then opened and used to get the frames. Once the frames were obtained, the face detector was used to locate the region of interest (ROI), where we would expect to find our hand – on the face or beside the face. Once detected, the coordinates were stored and used to highlight this area, and the areas surrounding it. Imutils was then used to resize the image around the ROI for image processing. The image was then converted to grayscale, blurred to remove noise, eroded to sharpen the edges and then it was passed to the segmenting function. The thresholded segment was then drawn over the image and the center coordinates stored in the hand array. The full code is included in the appendix section as it was similar to the code used in eye tracking.

The detection of an object/ a hand being recognized in the image then called the analysis function which can prompt that the user requires attention, and consequently, passed a gesture analysis (GA) index of value of 10, and 0 if no hand is detected within a given number of frames. The GA index is the passed to then Student Engagement analysis module explained in section 5.2.3.

5.2.3 Engagement Level Analysis Module

The engagement analysis module analysed the engagement level for a particular student by calling for the ET index and GA index. This module took in the indices and added them into a numpy array, which then used them to calculate the average figure. The numpy array was implemented as a FIFO data structure such that; it is declared as a single dimension array that holds a maximum of 100 indices, all initialised to zeros and then function called fifoinsert was used to maintain the array by inserting a new element at the last index (99) and deleting the value at the first index (0). The module would then output the average of the values, which would then provide the Engagement Level index, used to list and color-coded the names of the students, according to the index.

```
# declare numpy array to hold 100 values
engtLVL = np.zeros((100,), np.int8)
# declare variable to hold the index
studentEL = 10

# array to manage FIFO list
def fifoInsert(coords, ar):
    NP = np.append(ar, coords)
    for x in range(1):
        NP = np.delete(NP, 0)
    return NP
```

```
# calculate the average to get the studentEL
studentEL = np.average(engtLVL)
```

Listing 9: Code to calculate the engagement level

5.2.4 Admission List Module

The admission management module served as an interface that the instructor used to monitor the engagement level of their students. It basically used the Engagement level index provided to divide the students into three groups; High engagement level for values above 7 indicated by a green colour, average engagement level for values between 4 and 6.9 indicated by a yellow colour and finally, a low engagement level for values below 3.9, indicated by the red colour. The individual student index is linked according to the student name. The module should have then listed the names of the students with low engagement levels at the top of the list so that the instructor can easily see them. To achieve this, a two dimensional array was created to hold the names of the students and their engagement level index values. The array was then sorted in ascending order, according to the index of each student. A canvas from the tkinter library was then created which would allow for the names to be listed in the different colours. The code for this can be seen in the listing that follows.

```
window = tk.Tk()
window.configure(background='black')
window.title("Students Engagement")

ws = window.winfo_screenwidth()
hs = window.winfo_screenheight()
w = 200  # width for the Tk root
h = 500  # height for the Tk root
x = (ws / 2) - (w / 2)
y = (hs / 2) - (h / 2)



window.geometry('%dx%d+%d+%d' % (w, h, x, y))
canvas = tk.Canvas(window, bg="black", width=900, height=900, highlightthickness=0)
canvas.pack()
canvas_scroll = tk.Scrollbar(canvas, command=canvas.yview)
canvas_scroll.place(relx=1, rely=0, relheight=1, anchor=tk.NE)
canvas.configure(yscrollcommand=canvas_scroll.set, scrollregion=())

# sort array in ascending order according to the second element (engagement level)
op.sort(key=sortSecond)

# list the students according to their engagement levels
def listStudents():
    n = len(op)
    for i in range(n):
        if op[i][1] <= 3.9:
            canvas.create_text(10, 33+(i*20), anchor='w', text=str(i+1)+'.
'+op[i][0], fill = 'red')
        elif op[i][1] >= 4.0 and op[i][1] <= 6.9:
            canvas.create_text(10, 33 + (i * 20), anchor='w', text=str(i+1)+'.
'+op[i][0], fill='yellow')
        elif op[i][1] >= 7.0:
            canvas.create_text(10, 33 + (i * 20), anchor='w', text=str(i+1)+'.
```

```
'+op[i][0], fill='green')
        if not op[i][1] >= 0.0 and op[i][1] <= 10.0:
            canvas.create_text(10, 33 + (i * 20), anchor='w', text=str(i + 1) + '.
' + op[i][0], fill='white')

listStudents()


window.mainloop()
```

Listing 10: Code to display colour coded student name

5.3 System Testing

Testing of the implemented system to track student engagement in an online class was tested following the three steps used in OOAD. First, Unit testing was done where each class was tested as single units; the relevant attributes were tested and the functions checked to confirm that it accepted the necessary parameters, and then processed the data to provide the desired output. The classes were then tested to check that they provided the desired output. Second was integration testing; where the classes were instantiated (object creation) to test how messages was passed from one class or module to the other and in particular, how data was passed from the images were captured from the web camera, to how it was used to order the final list. Lastly, system testing was done where a video was passed and changes in the value of the student engagement levels were observed.

5.3.1 Unit Testing

First, was the eye tracking class (ETAnalysis module) that was expected to get an image, process it and find the position of the pupils. The positions of the pupils captured over time were then to be used in analysing the engagement level of the student, from the provided ET index. To test this, the web camera was opened and the algorithm allowed to capture images. Then two windows were used to observe the capture image as seen in figure 5.1, section 5.2.2. The algorithm was then expected to first detect the eyes using the keypoints and then indicate this by marking blue-points in the captured image as seen in figure 5.2, using the imshow() function in cv2. At this point, the algorithm was expected to call the contouring function which would threshold the image and find the contours so as to extract the position of the pupils. The result of this were the coordinates of the of the left and right pupils, coupled with the coordinates of one of the key points for each eye, as seen in figure 5.5 that follows.

Figure 5.5: ET coordinates extracted

An error was detected at this point which involved the lighting conditions such that, if the thresholding value was not good enough, then the algorithm would crash. This was solved by adding an "if statement" that printed 'face not detected' if the lighting was not good enough and the thresholding value was not high enough to extract the pupils else, the analysis would continue and values provided as seen in the highlighted section of figure 5.5. The last function that was to create the ET index was tested with this coordinates to confirm that it gave an output of 10 to show that the student is engaged, and 0 if not – The module always provided an ET index when the pupil coordinates were detected, and always returned a zero if either the face was not detected or the pupils could not be analysed.

Second was the gesture analysis module that was tested to detect when the student raised their hand up or not, as well as movements in the facial regions. The module was then used to extract images from video captured from the web camera. The running average function was called from the main class to confirm that it provided the running average for the background detection. The segment function was tested next to confirm that given a new image from the ones used to create the background and a threshold value, it could segment the hand out and show it, again using the imshow() function. The module would then create a GA index of 10 when the hand was detected and 0 if not. Figure 5.6 shows the background that was captured and stored, then the introduction of the hand or a new object that provided the foreground and lastly, the image as seen by the computer with the detected object (segmented hand) seen in white.

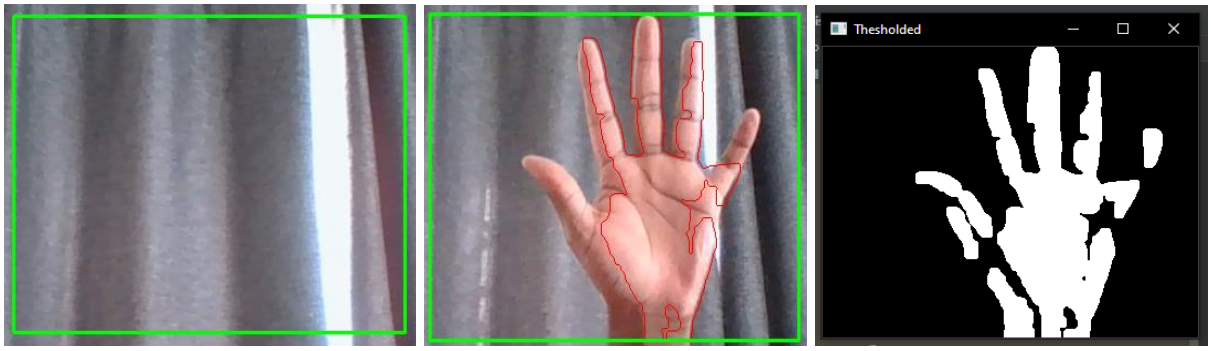


Figure 5.6: Background, object detection and segmented hand

The third unit testing involved testing the Student engagement and admission management modules. The student engagement module was tested to confirm that an array holding a maximum of 100 values could be averaged to find the students engagement level. The fifoinsert() function was tested to see that it could add a new value to the array and remove the oldest value in the array at index 0. A for loop was used to add 50 figures to the array full of 100 zeros to check that only 50 zeros would remain and the new values added to the array, resulting in only 100 elements throughout; as seen in figure 5.7.

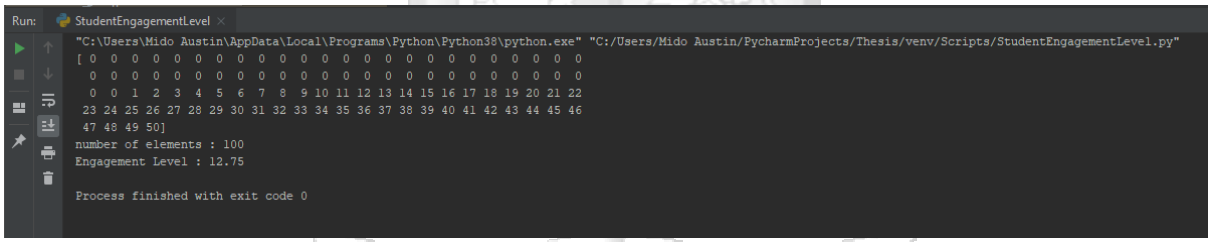


Figure 5.7: Student engagement analysis output

The student engagement level was tested using a 3 dimensional array the held dummy data of three students: the student name, student engagement index and a value for whether or not the student's hand was up. This data was used to test the sort() function which was to sort the array using the second column in each row (element at index 1) and then test the listSudents() function to check that the five students could be listed in a color coded format. The result can be seen in figure 5.8.

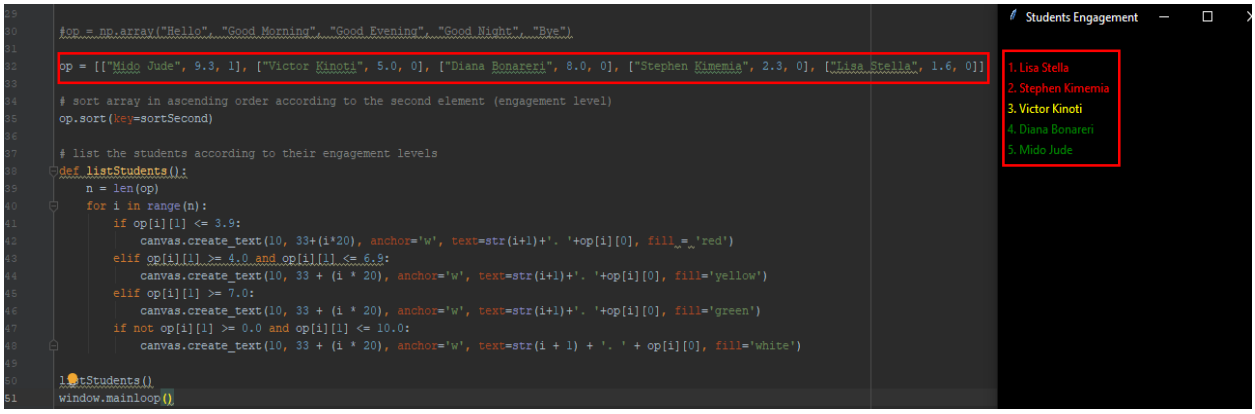


Figure 5.8: List of students based on engagement levels

5.3.2 Integration and System Testing

The integration and system testing was done based on the functional requirement stated in chapter 4, section 4.2.2. The results of this testing were as seen in the table 5.1.

Test	**Importance**	**Result**
The prototype should collect eye movements and body gestures data of different students	High	Pass. ET and GA modules provide the ET indices and GA indices
The prototype should process the data collected and create an engagement level index for each user	High	Pass. As seen in figure 5.7
The prototype should order the list of users according to lowest engagement level index to the highest	High	Pass. As seen in figure 5.8 (top right)
The prototype should present a summarised report for individual student engagement levels	Medium	Pass. The data is stored in an array that can be printed in a comma separated values (.csv) file.
The prototype should show when a certain user has not been captured for engagement level measuring (not present in class).	High	Pass.

		Indicates face not detected and returns 0 engagement level indices
The prototype should allow the system administrator to access recently analysed data.	Medium	Pass. Upon exit, the system can provide a comma separated values (.csv) file or text file.

Table 5.1: Summary of integration and system testing

5.4 System Validation

The prototype was validated using video recorded of students attending an online class. Before the recording was done, the prototype was demonstrated to the students who were attending a programming class and it was explained how the image processing was done from the point where images are obtained from the web camera to the point where the eye coordinates are extracted and analysed to provide the engagement level index. It was also demonstrated how their names would appear on the instructor's screen. The students were then asked to fill a form that would be used to analyse the data that was collected, if they approved. As seen in figure 5.5, 25 students responded and 84% of the students agreed to be a part of the student engagement, for as long as their data was not availed outside the research and used for anything else other than testing the prototype.

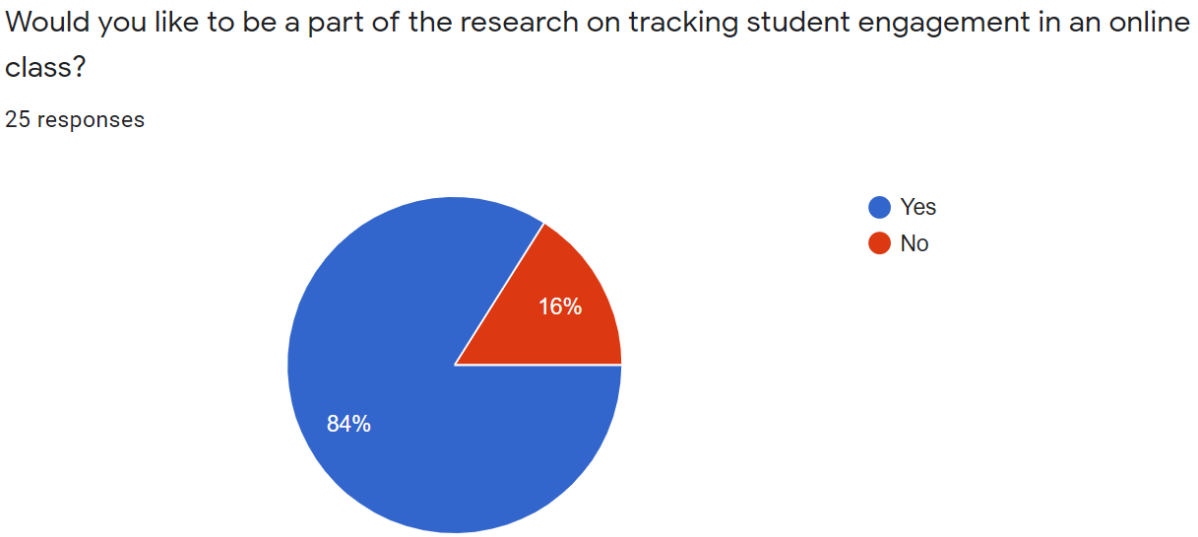


Figure 5.9: Student participation

Once the recording for the entire class was obtained, the test algorithm was modified to focus on the particular area where the videos of the particular students would be placed. The entire video was then edited to cut out the necessary video section for the students who had agreed to participate in the research. Once everything was ready, the validation test was ran and the student engagement levels were analyzed and compared to what the students had rated for themselves; as seen in figure 5.10 for all the 25 students. The section that was considered to carry the most weight was when three students had their cameras on and only one of them was presenting, which then gave the base line figures for a student who was fully engaged. The other two students observed the presentation. Then a student who had already made a presentation was told that they were free to do whatever they wanted, which then provided the base line for the student who was not very engaged in the lesson. The prototype was able to give engagement levels to those that were provided by the students, and the class instructor, as seen in table 5.2.

Students	Student rating	Instructor rating	Prototype analysis
Student presenter	10 (5/5)	10	9.2
Group member 1	8 (4/5)	7	7.4
Group member 2	8 (4/5)	8	7.8
Already presented	6 (4/5)	5	5.8

Table 5.2: Student engagement ratings validation

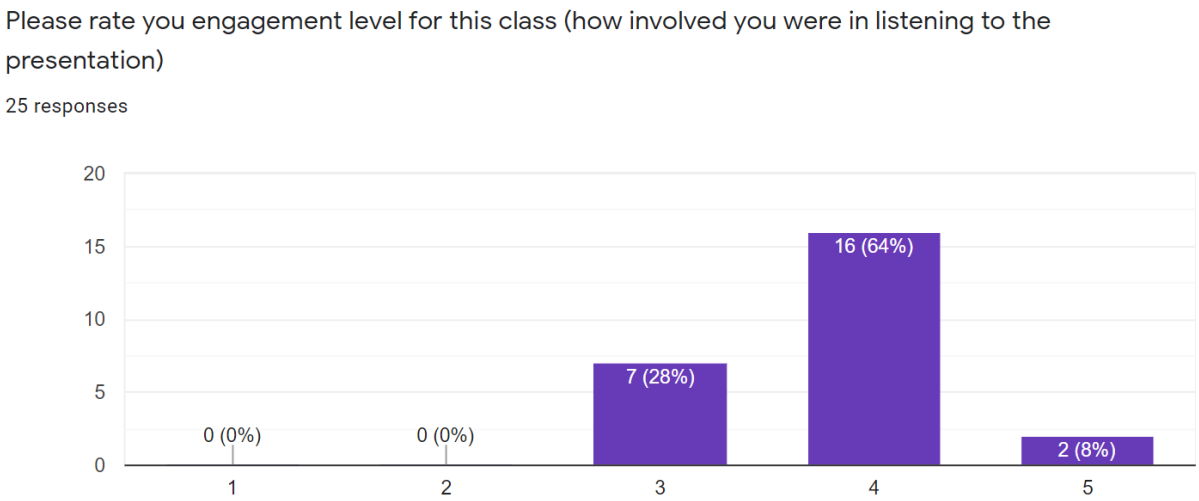


Figure 5.10: Summary of the students' engagement levels

After the tests to validate the prototype were completed, the participants were asked to state and briefly explain what they felt about the system – its effectiveness in tracking student engagement in the class – as well as what improvements they felt could be added to make the system more effective. All the participants accepted the system with 72% percent of the students indicating that it was at least 80 % good at tracking student engagement levels in an online class, as seen in figure 5.11.

Figure 5.11: Summary of user acceptance and feedback

Chapter 6: Discussions

6.1 Introduction

This chapter discusses the results of the research in light of the main research objective and the specific objectives, questions and aim discussed in chapter one. The objective of this research was to design and develop a prototype to measure online student engagement using eye tracking analysis and body gesture. The sections that follow discuss how the Result of the study and how the objectives were met, centered on the specific objectives and research questions; how student engagement can be measured in an online class,

6.2 Results of the Study

The results of this study were analysed in two divisions. The first division was to check whether the working prototype could actually be created then second, to check whether the prototype could actually work with data from a real world scenario. The results revealed that a system could be designed and developed to track student engagement in an online class – student engagement meaning how well the student is paying attention in class. This definition was formulated in chapter 2, based on the recommendation by Boekaerts among other researchers, that the study should have a clear definition of the term 'student engagement'. Analysis of the reviewed literature in chapter 2 section 2.3 revealed that student involvement in an online class could be analyzed primarily by tracking the eyes, which can then be complimented by other variables. This study then complimented the eye tracking with body gestures, and focused on the student raising their hand during a lesson. Using this, the study was successful in tracking student engagement, as discussed in section 5.4 in chapter 5.

6.3 Existing technologies and Prototype Development

The researcher chose to centre their research on technologies that were being used during the Covid-19 pandemic – which forced some learning institutions such as Strathmore University and other organizations, to switch to remote working and learning. These technologies mainly used a device's web camera and microphone to connect users. Based on this, the researcher then chose to develop the prototype using Python programming language. This was seen to provide benefits such as; the use of pre-trained datasets that would enable face detection and eye-tracking without having to collect new data and create a new model. Second, python being an interpreter language, made it easier to create a prototype that would be less resource intensive and thus, work more efficiently in processing data. Lastly, the idea behind such a

system was that it would be used in a distributed environment. The tracking components of the system could then be installed in a user's device – just like any other application – and then the analysis component could be retained on the server. This meant that the heavy load of image processing would have been removed from the server and instead, be distributed among the different clients. This brought about two benefits; first, the prototype would function more efficiently in processing the data and providing the student engagement analysis by using less bandwidth and having an increased latency, as image processing is done locally and then the small data sent to the server. Second, a less powerful server would be needed as the weight of the data to be processed is reduced.

This was the rationale that was used in designing and developing the prototype, which then resulted in a much 'lighter' system – lighter referring to the uses of resources. This then suggested that prototype could be included as plug in for already existing technologies, which could then increases their efficiency. In the case of an in-house development of a system, the prototype could be used as one of the components to build an entire system to be used in tracking student engagement.

6.4 Validity of the developed prototype

Several systems have been adopted for the purpose of conduction online classes. Here in Kenya, examples of such systems are zoom, Skype, Microsoft teams and Google class room. Each provided benefits such as; the instructor being able to share a screen to display the content they are teaching, students being able to share their screens during a presentation to show content for their presentations, useful features that include quick reactions and a way to manage participants – by admitting and removing them. However, majority of these systems apply to the general needs of the population, and are designed to be used for a variety of interactions from cooperate meetings, trainings, learning and friendly chats among friends. None of the mentioned systems truly monitor the student activities.

The developed prototype aims to satisfy the desire to know how well the students are engaged in the class, to ensure they properly grasp the content being taught. It purposes to achieve this by using the web camera to monitor the eye movements and gestures. Instructors or teachers are able to rely on this prototype to know how well the students are paying attention in class, and are able to engage those who appear to not be engaged in what is being taught.

6.5 Research Assumptions

The developed prototype had a few limitations that would otherwise, affect its effectiveness

i. This research assumed that all students who attend online classes, do so using devices – either a laptop or a smartphone/tablet – that come with a web camera integrated as standard. This may not be the case for students who use desktops, that include monitors which are commonly not sold with an integrated web camera
ii. This relied on the assumption that all students attend class while in comfortable environment that is at the very least, well lit.
iii. The prototype was designed to only monitor one student during a given lesson, assuming that all students will be attending the class on a single computer.
iv. The prototype worked with the assumption that students will log in to a class session in appropriate attire such that; they would not be wearing hats/caps that would otherwise, hinder face detection.
v. This research assumed that students' eyes will at all times, be visible to the camera while in the real world, the student might look away while taking notes. This could have a negligible effect on the engagement level rating.

Chapter 7: Conclusions, Recommendations and Future Work

7.1 Conclusions

Online classes holds the potential to increase the number of students undertaking education at a tertiary level, as well as at secondary and primary levels by; providing part-time students with a way to attend classes without having to commute from home to work, to school, and then back home, which ends up demanding more hours from their day. Allowing learning institutions to expand more easily by not having to construct new building to hold face to face classes and lastly, allowing students to access better education opportunities from schools that are located in faraway geographic locations. However, online classes come with their own challenges to the instructors and disadvantages to the student.

This research has presented a working prototype that provides a solution to the perceived problem of tracking the students' engagement while conducting an online class, by tracking the eyes and upper body gestures. This was the main objective of the system. The developed prototype is able to monitor each student remotely using the web camera that is found on many laptops and mobile phones/tablets, and then present a colour coded list to the instructor, who can then easily if majority of the student are engaged, and also see which students in particular, are not paying attention.

7.2 Recommendations

The researcher having conducted the study and achieved the set objective and specific objectives, recommends the following for the developed prototype:

i. Implementation of the prototype in a distributed environment. This means that the image processing components of the system could be implemented in the client side, and the rest of the system on the server side thus, images will be process locally on a student's computer and the indices then sent to the server where they will analysed, sorted and used to create the list. This will greatly reduce the load by sending numbers instead of images, and avoid the need for a large internet bandwidth and increased latency.

ii. In cases where speed is preferred over accuracy, then the researcher recommends that the 68 facial key-points be switched with a 5 facial key-point shape fail that was created by dlib author, Davis King. A test on the algorithms showed that use 5

instead of 68 key-points performed 8 to 10 percent faster and was 10 times smaller than the original dataset (Rosebrock, 2018).

iii. The prototype can be implemented as plug in to existing technologies and systems such as zoom, which adds the extra functionality to instructors who wish to know how well the student is engaged.

7.3 Future Work

The scope of this research only allowed for the development of a prototype that would measure student engagement by tracking the eyes mainly, and detecting when the student raises up their hand. In light of this, future studies could:

i. Expand this prototype to detect and analyse HoF gestures to further improve the measuring of student engagement, in addition to eye tracking analysis.

ii. The prototype could be improved to tracking multiple students using the web camera, instead of one.

iii. Different methods of background subtraction could be used to improve the accuracy or efficiency of the model; especially when using HoF gestures.

References

Agarwal, V., 2020. Real-time eye tracking using OpenCV and Dlib [WWW Document]. Medium. URL https://towardsdatascience.com/real-time-eye-tracking-using-opencv-and-dlib-b504ca724ac6 (accessed 9.2.20).

Archambault, I., Janosz, M., Morizot, J., Pagani, L., 2009. Adolescent behavioral, affective, and cognitive engagement in school: relationship to dropout. J. Sch. Health 79, 408–415. https://doi.org/10.1111/j.1746-1561.2009.00428.x

Ashwin, P., Mcvitty, D., 2015. The Meanings of Student Engagement: Implications for Policies and Practices. pp. 343–359. https://doi.org/10.1007/978-3-319-20877-0_23

Axelson, R.D., Flick, A., 2010. Defining Student Engagement. Change Mag. High. Learn. 43, 38–43. https://doi.org/10.1080/00091383.2011.533096

Babcock, A., Lehan, T., Hussey, H., 2019. Mind the Gaps: An Online Learning Center's Needs Assessment 24, 27–58.

Behera, A., Matthew, P., Keidel, A., Vangorp, P., Fang, H., Canning, S., 2020. Associating Facial Expressions and Upper-Body Gestures with Learning Tasks for Enhancing Intelligent Tutoring Systems. Int. J. Artif. Intell. Educ. 30, 236–270. https://doi.org/10.1007/s40593-020-00195-2

Blascheck, T., Kurzhals, K., Raschke, M., Burch, M., Weiskopf, D., Ertl, T., 2017. Visualization of Eye Tracking Data: A Taxonomy and Survey. Comput. Graph. Forum 36, 260–284. https://doi.org/10.1111/cgf.13079

Boekaerts, M., 2016. Engagement as an inherent aspect of the learning process. Learn. Instr., Special Issue: Student engagement and learning: theoretical and methodological advances 43, 76–83. https://doi.org/10.1016/j.learninstruc.2016.02.001

Bond, M., Buntins, K., Bedenlier, S., Zawacki-Richter, O., Kerres, M., 2020. Mapping research in student engagement and educational technology in higher education: a systematic evidence map. Int. J. Educ. Technol. High. Educ. 17, 1–30. https://doi.org/10.1186/s41239-019-0176-8

Bordoloi, B., Lee, M.-H., 2008. Adopting and Implementing Object-Oriented Methodologies 16.

Bowden, J.L.-H., Tickle, L., Naumann, K., 2019. The four pillars of tertiary student engagement and success: a holistic measurement approach. Stud. High. Educ. 0, 1–18. https://doi.org/10.1080/03075079.2019.1672647

Cappella, E., Kim, H.Y., Neal, J.W., Jackson, D.R., 2013. Classroom Peer Relationships and Behavioral Engagement in Elementary School: The Role of Social Network Equity. Am. J. Community Psychol. 52, 367–379. https://doi.org/10.1007/s10464-013-9603-5

Ciric, M., Jovanovic, D., 2016a. Student Engagement as a Multidimensional Concept. Presented at the WLC 2016 World LUMEN Congress. Logos Universality Mentality Education, pp. 187–194. https://doi.org/10.15405/epsbs.2016.09.24

Ciric, M., Jovanovic, D., 2016b. Student Engagement as a Multidimensional Concept. Presented at the WLC 2016 World LUMEN Congress. Logos Universality Mentality Education, pp. 187–194. https://doi.org/10.15405/epsbs.2016.09.24

Dennis, A., Wixom, B., Tegarden, D., 2015. Systems Analysis and Design: An Object-Oriented Approach with UML. John Wiley & Sons.

D'Errico, F., Paciello, M., Cerniglia, L., 2016. When emotions enhance students' engagement in e-learning processes. J. E-Learn. Knowl. Soc. 12.

D.N., C., G., A., M., R., 2012. Face Detection Using a Boosted Cascade of Features Using OpenCV, in: Venugopal, K.R., Patnaik, L.M. (Eds.), Wireless Networks and

Computational Intelligence, Communications in Computer and Information Science. Springer, Berlin, Heidelberg, pp. 399–404. https://doi.org/10.1007/978-3-642-31686-9_46

Fredricks, J.A., Blumenfeld, P.C., Paris, A.H., 2016a. School Engagement: Potential of the Concept, State of the Evidence: Rev. Educ. Res. 74, 59–109. https://doi.org/10.3102/00346543074001059

Fredricks, J.A., Blumenfeld, P.C., Paris, A.H., 2004. School Engagement: Potential of the Concept, State of the Evidence. Rev. Educ. Res. 74, 59–109. https://doi.org/10.3102/00346543074001059

Fredricks, J.A., Filsecker, M., Lawson, M.A., 2016b. Student engagement, context, and adjustment: Addressing definitional, measurement, and methodological issues. Learn. Instr., Special Issue: Student engagement and learning: theoretical and methodological advances 43, 1–4. https://doi.org/10.1016/j.learninstruc.2016.02.002

Gonzalez-Sanchez, J., Baydogan, M., Chavez-Echeagaray, M.E., Atkinson, R.K., Burleson, W., 2017. Chapter 11 - Affect Measurement: A Roadmap Through Approaches, Technologies, and Data Analysis, in: Jeon, M. (Ed.), Emotions and Affect in Human Factors and Human-Computer Interaction. Academic Press, San Diego, pp. 255–288. https://doi.org/10.1016/B978-0-12-801851-4.00011-2

Henrie, C.R., Bodily, R., Manwaring, K.C., Graham, C.R., 2015. Exploring intensive longitudinal measures of student engagement in blended learning. Int. Rev. Res. Open Distrib. Learn. 16. https://doi.org/10.19173/irrodl.v16i3.2015

Holmqvist, K., Nystrom, M., Anderson, R., Dewhurst, R., van de Weijer, J., Duchowski, A., Bojko, A., 2011. TEXT BOOKS/ REFERENCES: 2.

Hughes, J.N., Luo, W., Kwok, O.-M., Loyd, L.K., 2008. Teacher–Student Support, Effortful Engagement, and Achievement: A 3-Year Longitudinal Study. J. Educ. Psychol. 100, 1–14. https://doi.org/10.1037/0022-0663.100.1.1

Ilango, G., 2017. Hand Gesture Recognition using Python and OpenCV - Part 1 – Gogul Ilango [WWW Document]. URL https://gogul.dev/software/hand-gesture-recognition-p1 (accessed 9.2.20).

Just, M.A., Carpenter, P.A., 1980. A theory of reading: From eye fixations to comprehension. Psychol. Rev. 87, 329–354. https://doi.org/10.1037/0033-295X.87.4.329

Kahu, E.R., 2013. Framing student engagement in higher education. Stud. High. Educ. 38, 758–773. https://doi.org/10.1080/03075079.2011.598505

Kay, R., Benzimra, D., Li, J., 2017. Exploring Factors That Influence Technology-Based Distractions in Bring Your Own Device Classrooms. J. Educ. Comput. Res. 55, 073563311769000. https://doi.org/10.1177/0735633117690004

Kazemi, V., Sullivan, J., 2014. One millisecond face alignment with an ensemble of regression trees. 2014 IEEE Conf. Comput. Vis. Pattern Recognit. https://doi.org/10.1109/CVPR.2014.241

Kim, C., Park, S.W., Cozart, J., Lee, H., 2015a. From Motivation to Engagement: The Role of Effort Regulation of Virtual High School Students in Mathematics Courses 12.

Kim, C., Park, S.W., Cozart, J., Lee, H., 2015b. From motivation to engagement: The role of effort regulation of virtual high school students in mathematics courses. J. Educ. Technol. Soc. 18, 261–272.

Kothari, C.R., 2004. Research Methodology: Methods and Techniques. New Age International.

Krithika L.B, Lakshmi Priya GG, 2016. Student Emotion Recognition System (SERS) for e-learning Improvement Based on Learner Concentration Metric. Procedia Comput. Sci., International Conference on Computational Modelling and Security (CMS 2016) 85, 767–776. https://doi.org/10.1016/j.procs.2016.05.264

Lee, E., Pate, J.A., Cozart, D., 2015. Autonomy Support for Online Students. TechTrends 59, 54–61. https://doi.org/10.1007/s11528-015-0871-9

Levy, P.S., Lemeshow, S., 2013. Sampling of Populations: Methods and Applications. John Wiley & Sons.

Lowenthal, P.R., Nyland, R., Jung, E., Dunlap, J.C., Kepka, J., 2019. Does Class Size Matter? An Exploration into Faculty Perceptions of Teaching High-Enrollment Online Courses. Am. J. Distance Educ. 33, 152–168. https://doi.org/10.1080/08923647.2019.1610262

Mahmoud, M., Baltrušaitis, T., Robinson, P., 2016. Automatic Analysis of Naturalistic Hand-Over-Face Gestures. ACM Trans. Interact. Intell. Syst. 6, 19:1-19:18. https://doi.org/10.1145/2946796

Mahmoud, M.M., Baltrušaitis, T., Robinson, P., 2014. Automatic Detection of Naturalistic Hand-over-Face Gesture Descriptors, in: Proceedings of the 16th International Conference on Multimodal Interaction, ICMI '14. Association for Computing Machinery, Istanbul, Turkey, pp. 319–326. https://doi.org/10.1145/2663204.2663258

Martin, J., Torres, A., 2016. I. WHAT IS STUDENT ENGAGEMENT AND WHY IS IT IMPORTANT? Natl. Assoc. Indep. Sch. 3.

Meyer, K.A., 2014. Student Engagement in Online Learning: What Works and Why: Student Engagement Online. ASHE High. Educ. Rep. 40, 1–114. https://doi.org/10.1002/aehe.20018

Miller, B.W., 2015. Using Reading Times and Eye-Movements to Measure Cognitive Engagement. Educ. Psychol. 50, 31–42. https://doi.org/10.1080/00461520.2015.1004068

Mitchell, O., 2015. Experimental Research Design, in: The Encyclopedia of Crime & Punishment. American Cancer Society, pp. 1–6. https://doi.org/10.1002/9781118519639.wbecpx113

Nyiwa, K., Maithya, R., Gathumbi, A.M., 2017. Influence of Pupil-teacher Ratio on Performance in Kenya Certificate of Primary Education in Makueni County, Kenya 5.

Nyman, R., 2018. Indicators of student engagement: What teachers notice during introductory algebra lessons 17.

Omar, M., Ahmad, M., Yasin, A., Ibrahim, H., Ghazali, O., Khamis, S., 2018. The impact of Wi-Fi usage on students' academic performance. Int. J. Eng. Technol. 7, 240–244.

Pease, A., Pease, B., 2016. The Definitive Book of Body Language: How to read others' attitudes by their gestures. Orion.

Pitterson, N.P., Brown, S., Pascoe, J., Fisher, K.Q., 2016. Measuring cognitive engagement through interactive, constructive, active and passive learning activities, in: 2016 IEEE Frontiers in Education Conference (FIE). Presented at the 2016 IEEE Frontiers in Education Conference (FIE), pp. 1–6. https://doi.org/10.1109/FIE.2016.7757733

Poole, A., Ball, L.J., 2006. Eye Tracking in HCI and Usability Research, in: Encyclopedia of Human Computer Interaction. IGI Global, pp. 211–219. https://doi.org/10.4018/978-1-59140-562-7.ch034

Raca, M., Dillenbourg, P., 2015. Translating Head Motion into Attention - Towards Processing of Student's Body-Language 7.

Raca, M., Dillenbourg, P., 2014. Classroom Social Signal Analysis. J. Learn. Anal. 1, 176–178.

Raja, R., Nagasubramani, P.C., 2018. Impact of modern technology in education. J. Appl. Adv. Res. 3, 33. https://doi.org/10.21839/jaar.2018.v3iS1.165

Rampell, C., 2009. Class Size Around the World. Econ. Blog. URL https://economix.blogs.nytimes.com/2009/09/11/class-size-around-the-world/ (accessed 4.19.21).

Revak, M.A., 2020. When the Tide Goes Out: Identifying and Supporting Struggling Students in Online Courses | Faculty Focus. Fac. Focus High. Ed Teach. Learn. URL https://www.facultyfocus.com/articles/online-education/identifying-and-supporting-struggling-students-in-online-courses/ (accessed 7.24.20).

Rosebrock, A., 2018. (Faster) Facial landmark detector with dlib. PyImageSearch. URL https://www.pyimagesearch.com/2018/04/02/faster-facial-landmark-detector-with-dlib/ (accessed 4.20.21).

Rotgans, J.I., Schmidt, H.G., 2011a. Cognitive engagement in the problem-based learning classroom. Adv. Health Sci. Educ. 16, 465–479. https://doi.org/10.1007/s10459-011-9272-9

Rotgans, J.I., Schmidt, H.G., 2011b. Cognitive engagement in the problem-based learning classroom. Adv. Health Sci. Educ. 16, 465–479. https://doi.org/10.1007/s10459-011-9272-9

Schindler, M., Lilienthal, A.J., 2019. Domain-specific interpretation of eye tracking data: towards a refined use of the eye-mind hypothesis for the field of geometry. Educ. Stud. Math. 101, 123–139. https://doi.org/10.1007/s10649-019-9878-z

Sesmiyanti, S., 2016. Student's Cognitive Engagement in Learning Process. J. Polingua Sci. J. Linguist. Lit. Educ. 5, 48–51. https://doi.org/10.30630/polingua.v5i2.34

Sharma, P., Esengönül, M., Khanal, S.R., Khanal, T.T., Filipe, V., Reis, M.J.C.S., 2018. Student Concentration Evaluation Index in an E-learning Context Using Facial Emotion Analysis, in: Technology and Innovation in Learning, Teaching and Education. Presented at the International Conference on Technology and Innovation in Learning, Teaching and Education, Springer, Cham, pp. 529–538. https://doi.org/10.1007/978-3-030-20954-4_40

Soluch, P., Tarnowski, A., 2013. Eye-Tracking Methods and Measures: Paweł Soluch, Adam Tarnowski, Translation Studies and Eye-Tracking Analysis.

Stern, J., n.d. Introduction to Online Teaching and Learning 10.

Sukhavasi, S.B., Sukhavasi, S., Khan, H., Chowdary, K., 2013. Implementation of Running Average Background Subtraction Algorithm in FPGA for Image Processing Applications. Int. J. Comput. Appl. 73, 41–46. https://doi.org/10.5120/13022-0259

Tilley, S., Rosenblatt, H.J., 2016. Systems Analysis and Design. Cengage Learning.

Tipper, C.M., Signorini, G., Grafton, S.T., 2015. Body language in the brain: constructing meaning from expressive movement. Front. Hum. Neurosci. 9. https://doi.org/10.3389/fnhum.2015.00450

Turaga, P., Chellappa, R., Veeraraghavan, A., 2010. Advances in Video-Based Human Activity Analysis: Challenges and Approaches, in: Zelkowitz, M.V. (Ed.), Advances in Computers, Advances in Computers. Elsevier, pp. 237–290. https://doi.org/10.1016/S0065-2458(10)80007-5

Wong, J.Y., 2010. Chapter 1 - Introduction, in: Wong, J.Y. (Ed.), Terramechanics and Off-Road Vehicle Engineering (Second Edition). Butterworth-Heinemann, Oxford, pp. 1–19. https://doi.org/10.1016/B978-0-7506-8561-0.00001-4

World Bank, 2019. Improving Higher Education Performance in Kenya: A Policy Report (Education No. AUS0001105), A Policy Report.

Zepke, N., Leach, L., 2010. Improving student engagement: Ten proposals for action: Act. Learn. High. Educ. https://doi.org/10.1177/1469787410379680

Appendix

Appendix 1: Originality Report

Curiginal

Entire Document

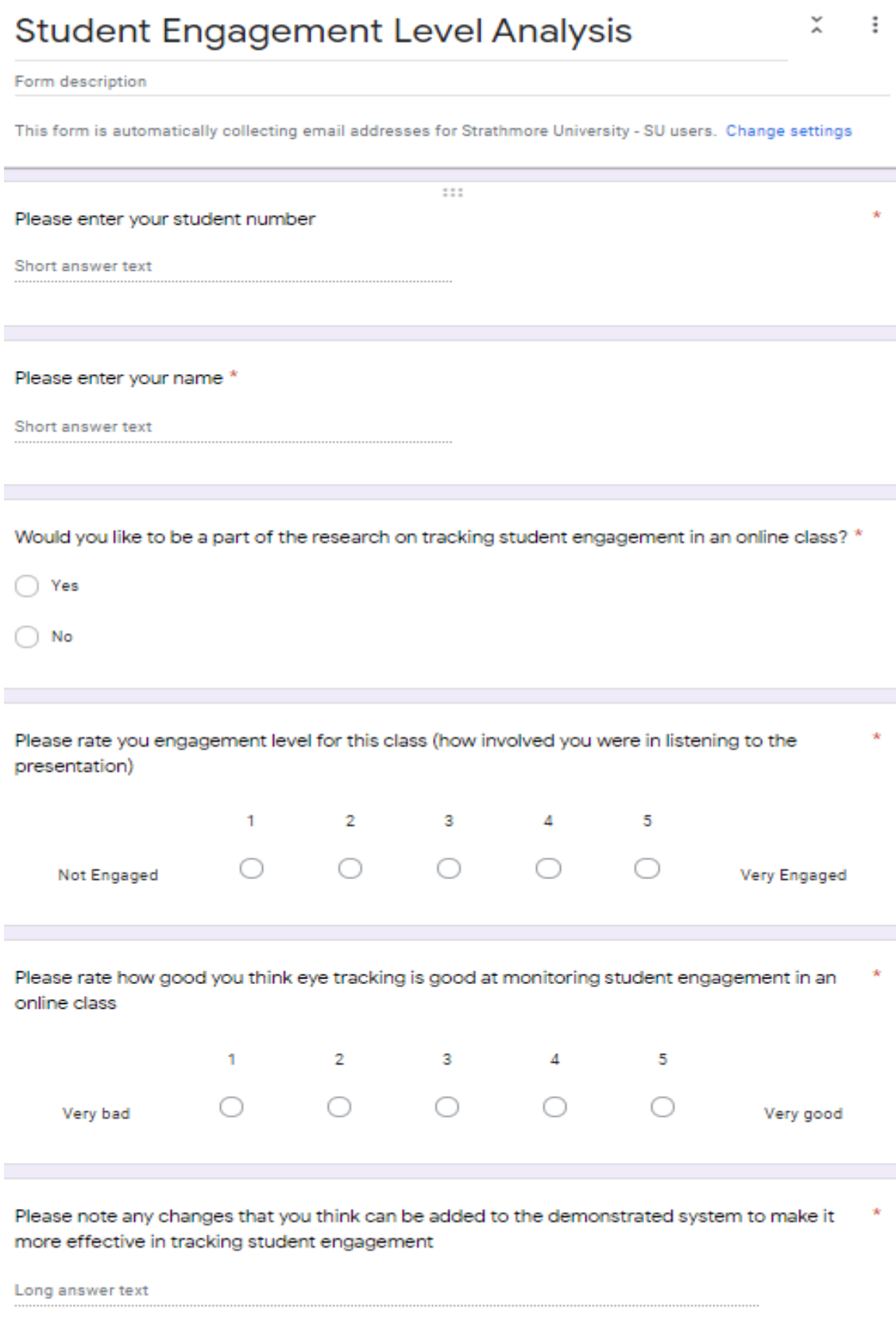
A Model to Measure Online Student Engagement using Eye Tracking and Body Movement Analysis.

By

Mido Jude Austin

Submitter email	jude.mido@strathmore.edu
Similarity	8%
Analysis address	library.strath@analysis.urkund.com

Appendix 2: Participant Feedback Form



Student Engagement Level Analysis

Form description

This form is automatically collecting email addresses for Strathmore University - SU users. Change settings

Please enter your student number *

Short answer text

Please enter your name *

Short answer text

Would you like to be a part of the research on tracking student engagement in an online class? *

- Yes
- No

Please rate you engagement level for this class (how involved you were in listening to the presentation) *

	1	2	3	4	5	
Not Engaged						Very Engaged

Please rate how good you think eye tracking is good at monitoring student engagement in an online class *

	1	2	3	4	5	
Very bad						Very good

Please note any changes that you think can be added to the demonstrated system to make it more effective in tracking student engagement *

Long answer text

Appendix 3: NACOSTI Research License

REPUBLIC OF KENYA

NATIONAL COMMISSION FOR SCIENCE,TECHNOLOGY & INNOVATION

Ref No: **918583**

Date of Issue: **29/April/2021**

RESEARCH LICENSE

This is to Certify that Mr.. Jude Mido Austin of Strathmore University, has been licensed to conduct research in Nairobi on the topic: Measuring Online Student Engagement using Eye Tracking and Body Movement Analysis for the period ending : 29/April/2022.

License No: **NACOSTI/P/21/10202**

918583

Applicant Identification Number

Director General
NATIONAL COMMISSION FOR SCIENCE,TECHNOLOGY & INNOVATION

Verification QR Code

NOTE: This is a computer generated License. To verify the authenticity of this document, Scan the QR Code using QR scanner application.

Appendix 4: Eye Tracking Analysis Code

```
import cv2
import dlib
import numpy as np

img = cv2.imread('C:/image.jpg')

#if img is None:
    #print('Could not open or find the image:', args.input)
    #exit(0)

gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY) # convert to grayscale detector =
dlib.get_frontal_face_detector()
detector = dlib.get_frontal_face_detector()
rects = detector(gray, 1) # rects contains all the faces detected

#draw rectangle around detected face
for rect in rects:
    x1 = rect.left()
    y1 = rect.top()
    x2 = rect.right()
    y2 = rect.bottom()
    cv2.rectangle(img, (x1, y1), (x2, y2), (0, 255, 0), 3)

def shape_to_np(shape, dtype="int"):
    coords = np.zeros((68, 2), dtype=dtype)
    for i in range(0, 68):
        coords[i] = (shape.part(i).x, shape.part(i).y)
    return coords

predictor = dlib.shape_predictor('C:/Users/Mido
Austin/AppData/Local/Programs/Python/Python38/shape_predictor_68_face_landmarks.dat
')
for (i, rect) in enumerate(rects):
    shape = predictor(gray, rect)
    shape = shape_to_np(shape)
    for (x, y) in shape:
        cv2.circle(img, (x, y), 2, (0, 0, 255), -1)


cap = cv2.VideoCapture(0)
while(True):
    ret, img = cap.read()
    cv2.imshow("Output", img)
    if cv2.waitKey(1) & 0xFF == ord('q'): # escape when q is pressed
        break


def eye_on_mask(mask, side):
    points = [shape[i] for i in side]
    points = np.array(points, dtype=np.int32)
    mask = cv2.fillConvexPoly(mask, points, 255)
    return mask
left = [36, 37, 38, 39, 40, 41] # keypoint indices for left eye
right = [42, 43, 44, 45, 46, 47] # keypoint indices for right eye

mask = np.zeros(img.shape[:2], dtype=np.uint8)
mask = eye_on_mask(mask, left)
mask = eye_on_mask(mask, right)

kernel = np.ones((9, 9), np.uint8)
mask = cv2.dilate(mask, kernel, 5)
eyes = cv2.bitwise_and(img, img, mask=mask)
mask = (eyes == [0, 0, 0]).all(axis=2)
eyes[mask] = [255, 255, 255]
eyes_gray = cv2.cvtColor(eyes, cv2.COLOR_BGR2GRAY)
```

```
def nothing(x):
    pass
cv2.namedWindow('image')
cv2.createTrackbar('threshold', 'image', 0, 255, nothing)
threshold = cv2.getTrackbarPos('threshold', 'image')
_, thresh = cv2.threshold(eyes_gray, threshold, 255, cv2.THRESH_BINARY)
thresh = cv2.erode(thresh, None, iterations=2)
thresh = cv2.dilate(thresh, None, iterations=4)
thresh = cv2.medianBlur(thresh, 3)


def contouring(thresh, mid, img, right=False):
    cnts, _ = cv2.findContours(thresh, cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_NONE)
    cnt = max(cnts, key = cv2.contourArea) # finding contour with #maximum area
    M = cv2.moments(cnt)
    cx = int(M['m10']/M['m00'])
    cy = int(M['m01']/M['m00'])
    if right:
        cx += mid # Adding value of mid to x coordinate of centre of #right eye to
adjust for dividing into two parts
    cv2.circle(img, (cx, cy), 4, (0, 0, 255), 2)# drawing over #eyeball with red
mid = (shape[39][0] + shape[42][0]) // 2
contouring(thresh[:, 0:mid], mid, img)
contouring(thresh[:, mid:], mid, img, True)

def contouring(thresh, mid, img, right=False):
    cnts, _ = cv2.findContours(thresh, cv2.RETR_EXTERNAL,cv2.CHAIN_APPROX_NONE)
    try:
        cnt = max(cnts, key = cv2.contourArea)
        M = cv2.moments(cnt)
        cx = int(M['m10']/M['m00'])
        cy = int(M['m01']/M['m00'])
        if right:
            cx += mid
        cv2.circle(img, (cx, cy), 4, (0, 0, 255), 2)
    except:
        pass
```

Appendix 5: Gesture Analysis (Hand Detection) Code

```
import cv2
import dlib
import numpy as np
import imutils

cap = cv2.VideoCapture(0)
ret, img = cap.read()
detector = dlib.get_frontal_face_detector()
predictor = dlib.shape_predictor('C:/Users/Mido
Austin/AppData/Local/Programs/Python/Python38/shape_predictor_68_face_landmarks.dat
')

face = np.zeros(4)
num_frames = 0
aWeight = 0
#Add coordinates to face (ROI)
def fifoInsert(coords, ar):
    NP = np.append(ar, coords)
    for x in range(4):
        NP = np.delete(NP, 0)
    return NP

# global variables
bg = None

# RA FUnction

def run_avg(image, aWeight = 0.5):
    global bg
    # initialize the background
    if bg is None:
        bg = image.copy().astype("float")
        return

    # compute weighted average, accumulate it and update the background
    cv2.accumulateWeighted(image, bg, aWeight)


# segmenting fucntion
def segment(image, threshold=25):
    global bg

    diff = cv2.absdiff(bg.astype("uint8"), image)

    thresholded = cv2.threshold(diff, threshold, 255, cv2.THRESH_BINARY)[1]

    (_, cnts, _) = cv2.findContours(thresholded.copy(), cv2.RETR_EXTERNAL,
cv2.CHAIN_APPROX_SIMPLE)

    if len(cnts) == 0:
        return
    else:
        segmented = max(cnts, key=cv2.contourArea)
        return (thresholded, segmented)


while (True):
    ret, img = cap.read()
    gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
    rects = detector(gray, 1)

    #check if face has been detected
    if not rects:                   # if no face detected, set ET to Zero
coordiates
        print("no face detected")
```

```
    else:
        print(" face detected")
        x1 = x2 = x3 = y1 = y2 = y3 = 0
        for rect in rects:
            #draw rectange around face
            x1 = rect.left()
            y1 = rect.top()
            x2 = rect.right()
            y2 = rect.bottom()
            face = fifoInsert([x1,x2,y1,y2], face)
            cv2.rectangle(img, (x1, y1), (x2, y2), (0, 255, 0), 3)
            face_pos = cv2.rectangle(img, (x1, y1), (x2, y2), (0, 255, 0), 3)

            x3 = int(x1+((x2-x1)/2))
            y3 = int(y1+((y2-y1)/2))

            cv2.circle(img, (x1, y3), 4, (0, 0, 255), 2)
            cv2.rectangle(img, (x1, y3), (x3, y2), (0, 0, 255), 3)
            cv2.rectangle(img, (x3, y3), (x2, y2), (255, 0, 0), 3)

        # region of interest (ROI) coordinates
        top, right, bottom, left = y1, y2, x1, x2

        print(face)
        # resize the frame
        #frame = imutils.resize(img, width=700)

        # flip the frame so that it is not the mirror view
        #frame = cv2.flip(frame, 1)

        # clone the frame
        #clone = frame.copy()

        frame = img
        # get the height and width of the frame
        (height, width) = frame.shape[:2]

        # get the ROI
        # roi = frame[top:bottom, right:left]
        roi = frame[int(face[1]):int(face[3]), int(face[0]):int(face[2])]
        #roileft = frame[y3:y2, x3:x1]
        #roiright = frame[y3:y2, x2,x3]

        # convert the roi to grayscale and blur it
        gray1 = cv2.cvtColor(roi, cv2.COLOR_BGR2GRAY)
        gray1 = cv2.GaussianBlur(gray1, (7, 7), 0)

        '''grayleft = cv2.cvtColor(roileft, cv2.COLOR_BGR2GRAY)
        grayleft = cv2.GaussianBlur(grayleft, (7, 7), 0)

        grayright = cv2.cvtColor(roiright, cv2.COLOR_BGR2GRAY)
        grayright = cv2.GaussianBlur(grayright, (7, 7), 0)'''

        # to get the background, keep looking till a threshold is reached
        # so that our running average model gets calibrated
        if num_frames < 30:
            run_avg(gray, aWeight)
            #run_avg(grayleft, aWeight)
            #run_avg(grayright, aWeight)
        else:
            # segment the hand region
            hand = segment(gray1)
            #fingerleft = segment(grayleft)
            #fingerright = segment(grayright)

            # check whether hand region is segmented
            if hand is not None:
```

```
                # if yes, unpack the thresholded image and
x
```