DEEP LEARNING INTEGRATED VIDEO IMAGE-BASED POSTURE ASSESSMENT PLATFORM FOR MANUAL ASSEMBLY WORKERS

Developed by: Amir Saman

Completed May 2023

Contents

Table of Figures

Chapter 1 Introduction

1.1. Research Background

The posture assessment platform is aimed at manual assembly workers which is intended to prevent the potential development of posture related work-related musculoskeletal disorders (WMSDs). Carrying out repetitive tasks on a daily basis involving arm and hand movements such as reaching, holding, straightening, gripping, clenching and bending where they are going against force and repetition which lead to WMSDs without them even noticing early on. If the awkward and unhealthy actions that lead to WMSDs are prevented as soon as they are performed every time. The worker will perhaps develop a positive habit where they would have a lower chance at attempting to that the harmful action again.

The real-time ergonomic feedback of this technology, which will be developed and documented throughout this project, will alert the user whenever they start to perform a harmful action so they can be prompted and made aware they need to focus harder on performing that movements in the correct way to avoid injury. There will also be an analysis output of the session in the form of a graph which will show posture performance against time so the worker will start to pick up on patterns they can reflect on to improve their postural performance for the upcoming sessions. There will also be a score rating for every session which the worker can use as a metric for their performance which the user can reflect on and perhaps encourage improvement.

1.2. Research Motivation

The main motivation behind this project is that there is a gap in research of solutions that give real-time ergonomic feedback where it is simple and inexpensive to set up. There are many solutions available that give real-time feedback on postural feedback but very little to none provide a cost-effective solution where posture assessment accuracy isn’t much sacrificed in the process. The proposed solution this paper is developing is cost efficient and doesn’t require advanced hardware and programming to the end user which gives it reach to more potential users. In this research, it is aimed to build a cost-effective solution in capturing human posture in real-time and provide ergonomic feedback and session analysis to reflect on at any time.

1.3. Research Questions

- What are the important features of an industry-ready real-time posture capture and ergonomics analysis tool?
- How can a camera-integrated deep learning solution can be developed in order to understand posture in real-time to give feedback?
- How effective can an inexpensive solution be at lowering the risk of developing work-related musculoskeletal disorders?
- What are the advantages and disadvantages or using this posture assessment platform in a manual assembly environment?

1.4. Research Aims and Objectives

The primary aim of this research is to develop a real-time video capture-based ergonomic posture assessment platform for manufacturing assembly industries. The focus of this platform is to capture the user’s posture while giving real-time feedback without the need for the user to wear anything. Another focus is to generate a dashboard with the user's postural data for each session with an ergonomic risk score.

The main objective is to prevent the development of work-related musculoskeletal disorders on manual assembly workers. This platform is intended to notify users when their posture starts to worsen anticipating they would fix their stance. The user can also look back on each session posture graph data and score so they can perhaps find patterns which would assist them on finding ways to prevent getting a bad score (e.g., the graph shows worse posture every session after x number of hours so the user can get a break in hopes of getting refreshed so bad posture can get prevented).

1.5. Research Scope

This project's testing will be conducted within London South Bank University which means the results of this project may not include edge cases which weren't applicable or seen in the testing stages due to the nature of the environment. The University doesn't have access to every manual assembly machine that is being used in a manufacturing environment meaning that there might exist a scenario where this system might not work as intended by producing inaccurate results.

The main focus of this project is to assess torso movements and angles relative to surrounding body parts. This means assessment of finger and wrist movements are outside the scope of this project. Although, the assessment of head position relative to the shoulders is also outside the scope of this project, attempts of implementing that form of assessment in this solution will be attempted if there is time to do at the end.

Testing will be done on human manual assembly operations meaning that the application of this program outside this use case is outside the scope for this project.

1.6. Structure of the Report

This report includes a comprehensive literature review that examines the systems and technologies developed by other projects, along with an analysis of their respective successes and limitations. This chapter presents a technical review, which involves an evaluation of existing technologies and frameworks. The purpose of this review is to provide a justified rationale for the selection of components used in the development of the system that is the focus of this project. The subsequent section of this paper delves into the research methodology, elucidating the procedural steps undertaken to accomplish the project and expounding on the implementation of crucial features. In the subsequent section, a case study shall be conducted, and the results shall be documented. In conclusion, this report will detail the limitations of the application and outline potential avenues for future research.

Chapter 2 Literature Review

2.1. Posture Extraction Methods

This section summarises posture-related data collection methods for manual labourers and discusses their advantages and disadvantages. Posture-related data collection can be divided into two categories: deep learning-based methods and traditional methods. Deep learning-based methods usually involve having technological devices as their main device for the data collection which includes contact sensors and non-contact sensors, on the other hand, traditional methods acquire posture-related data manually.

2.1.1. Motion Capture Technologies

2.1.1.1. Wearable Insole Pressure System

(Antwi-Afari, et al., 2018) evaluated the use of foot plantar pressure distribution data continuously captured by wearable insole pressure system to identify and classify unfavourable work postures linked to work-related musculoskeletal disorders among labour workers. **Figure 1** shows the overview of this system mainly highlighting that there are 13 pressure points that are measured on the insole which is wirelessly connected to a desktop computer to share its data. Five different awkward working postures were carried out in a simulated laboratory experiment to confirm feasibility of implementing the proposed approach. These postures were: overhead working, crouching, slouching, half crouching and kneeling on one leg. These postures are performed by a construction worker as seen in **Figure 2** and their pressure distribution on the insole are shown in **Figure 3**.

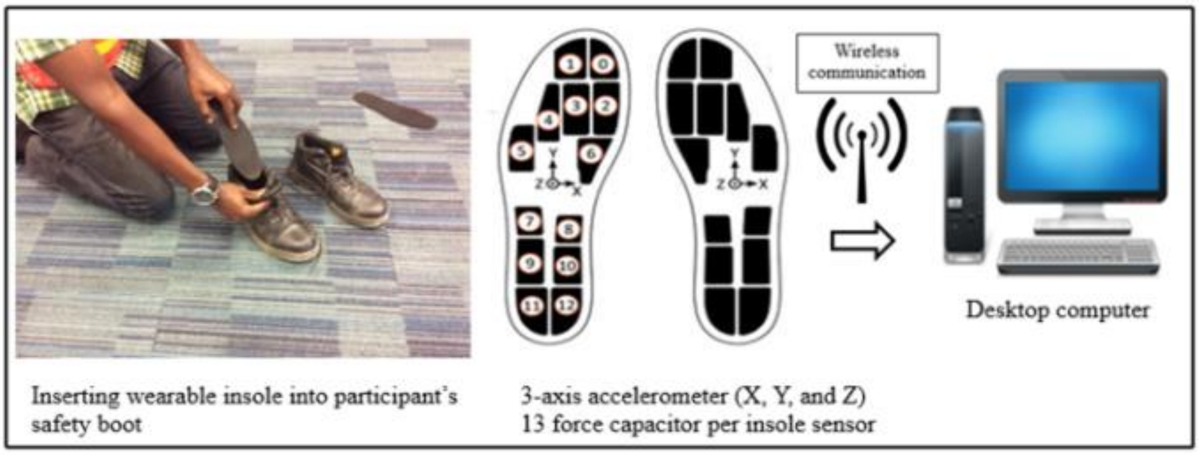

Wireless communication
Desktop computer
Inserting wearable insole into participant's safety boot
3-axis accelerometer (X, Y, and Z)
13 force capacitor per insole sensor

Table 1. An overview of the wearable insole pressure system (Source: (Antwi-Afari, et al., 2018)).

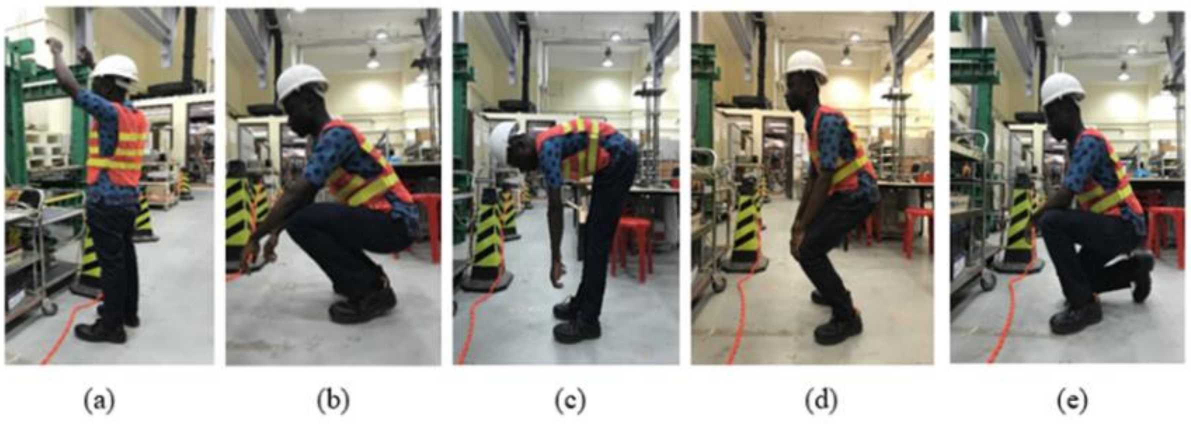


Figure 1. Laboratory experimental setup of awkward working postures: (a) overhead working; (b) squatting; (c) stooping; (d) semi-squatting; and (e) one-legged (Source: Antwi-Afari, et al., 2018).

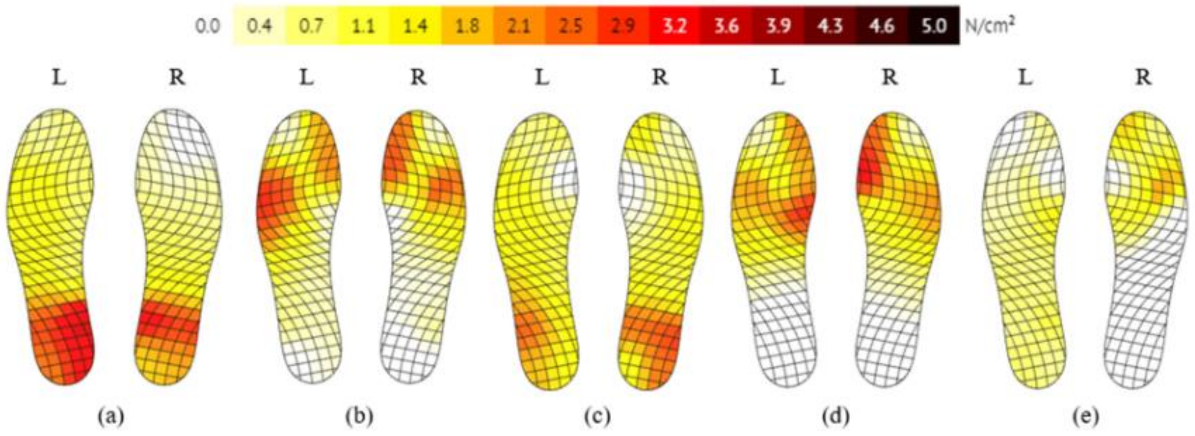


Figure 2. Foot plantar pressure distribution of different types of awkward working postures: (a) overhead working; (b) squatting; (c) stooping; (d) semi-squatting; and (e) one-legged kneeling. L and R are left and right foot, respectively. (Source: (Antwi-Afari, et al., 2018))

Four types of supervised machine learning classifiers (ANN, DT, KNN, SVM, etc.) had their classification performances compared to select the best classifier using a 0.32 second window size. Cross-validation results show that the SVM classifier achieved the best results with an accuracy of 99.70%, and a sensitivity of correct classification of all work postures was greater than 99.00%. This study outlines the feasibility and possible applications of such a wearable system for ergonomic risk assessment of WMSDs under construction (Antwi-Afari, et al., 2018).

2.1.1.2. Wearable Sensors

Inertial measurements unit (IMU) is a sensor system that uses measurement systems, such as gyroscopic sensors and accelerometers, to estimate relative position, velocity, and acceleration (Ahmad, et al., 2013). IMU can be attached with workers' joints to collect accurate data regarding the joint kinematic data for behavioural analysis. For example, if you attach an IMU to a worker's waistline you could measure three-axis acceleration for postural stability. Multiple IMUs attracted to key human body joints could collect the worker's full-body posture related data for behavioural analysis. If an IMU system which consists of eight IMU's covering the arms, legs (one on upper and one of lower leg), back (upper and lower back) could estimate joint angles and be able to work out if the user is standing up, stooping, and squatting. 17 IMU sensors were used in another system to create an IMU based suit that uses those sensors to collect 3D poses consisting of 28 joint centre locations (Yu, et al., 2021).

Wearable sensors can be useful since they are automatic, continuous, and accurate (Huang, et al., 2017). They are automatic and continuous because when you set them up on a user, they start the data collection automatically until turned off. When it is on and working it has a consistent frequency so continuous results are acquired.

However, they are uncomfortable and intrusive. They do only weigh in at about 70 grams (Yu, et al., 2021) but if this technology is used it will often be dozens at a time which makes the weight on you increase. This means assembly workers will be fatigued quicker while also being uncomfortable while they work. Developed a way of body posture identification using wireless networked proximity sensors, their network and wireless links are outlined in **Figure 4**.

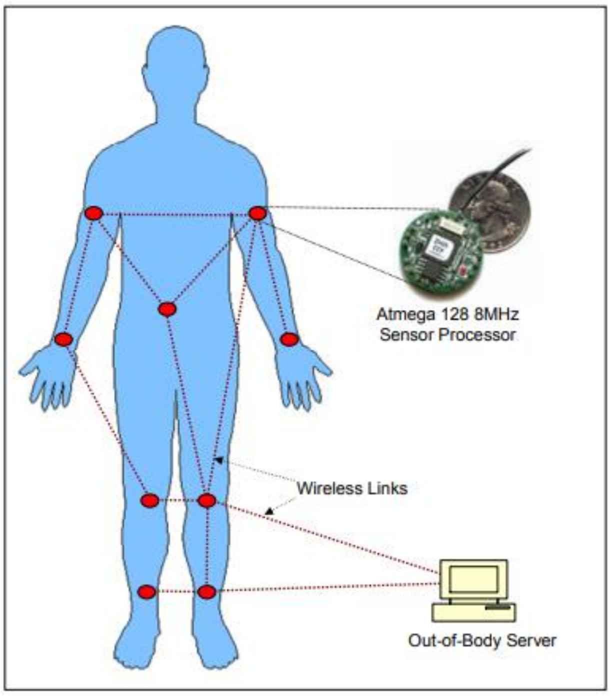


Figure 3. Wireless wearable sensor network (Source: (Quwaider & Biswas, 2010))

2.1.1.3. Depth Camera

Depth cameras could either generate 3D point clouds or range images. The 3D point clouds consist of a set of data points in space which may represent a 3D object (Bello, et al., 2020). Range images are a unique class of digital images where each pixel expresses the distance between a known reference frame and visible point in the scene, essentially it is reproducing the 3D structure of a scene. Time of flight depth cameras/sensors is an example of one of a type of depth cameras. They determine depth by computing the time it takes light to return to the device which is then multiplied by the speed of light to obtain the depth. Another type of depth cameras is the stereo depth camera which perceive depth by capturing images with at least two image sensors. Then depth is calculated by estimating disparities between marching key points in the images (Yu, et al., 2021).

Depth cameras are non-invasive to ongoing work, automatic and continuous. The fact it is non-invasive to ongoing work because nothing is being set up on your body means the user isn't going to be disturbed when their posture is being extracted. However, they are less accurate than contact sensors, have a high computation expense and they can interfere with illuminations. Since they are not directly attached to the user this makes this technology less accurate than a contact-based-method. All this data that is feeding into the camera is raw and it needs to be processed to make sense of it. When working in an assembly environment the room are well illuminated and there's a lot of light around which is a weakness for a depth camera meaning it will not be a good fit for an assembly environment (Yu, et al., 2021).

2.1.2. Traditional Methods

2.1.2.1. Self-Report

Self-report involves gathering data where workers provide information themselves by asking them to remember their postures during the manual assembly tasks and answer questions about how their posture changed. This can be executed by conducting questionnaires or interviews to collect the self-report data (Yu, et al., 2021).

This method is easy to conduct and has low initial cost. It is easy to conduct since it can be carried out by asking the workers to recall their postures during a shift and the cost is very little since it’s a simple procedure that only includes written or verbal communication. They are carried frequently but there exist several shortcomings of self-report methods such as their subjective and non-continuous results and this method can become labour and time consuming if there is an increasing number of participants. Since conducting this method requires questioning the worker, it is more than likely they are going to provide a subjective answer (Yu, et al., 2021) as to how their posture was during that session which doesn’t make this self-report very reliable. Continuous results are not expected when carrying out this method since the human memory is unlikely to remember the posture at every specific time which makes real-time feedback impossible. Questioning an increasing number of participants is going to make this procedure impractical as well as being labour intensive and time consuming making it impossible to gather big data from assembly manufacturing sites (Yu, et al., 2021).

2.1.2.2. Observation

Systematic observation is a method of quantitative data collection that involves one or more observers that observe the workers’ when they are manufacturing. Observers record an unbiased view in order to obtain reliable data (New York Chichester, 2007). Video tapes, pen and paper and computer aided analysis can be the methods used to record the working postures of assembly workers. Systematic observation methods specify which and how variables should be recorded to make the results more objective and comparable. In essence, systematic observation includes specific rules of data and recording, which increases the accuracy and the level of detail of the collected data (Yu, et al., 2021).

Method	Advantage	Disadvantage

Wearable Insole Pressure System	Automatic Accurate	Intrusive Expensive on a large scale
Wearable Sensors	Automatic Continuous Accurate	Intrusive Uncomfortable
Depth Camera	Less intrusive Automatic Continuous	Infrared-based depth cameras are sensitive to sunlight
Self-Report	Inexpensive Easy to implement	Subjective results Non-continuous Labour intensive and time consuming on a bigger scale
Observation	Inexpensive Easy to implement	Subjective results Non-continuous Labour intensive and time consuming on a bigger scale

Table 2. Advantages and disadvantages of posture- related data collection methods for labour workers (Yu, et al., 2021) (Antwi-Afari, et al., 2018).

2.2. Deep Learning Models in Capturing Human Posture Information

2.2.1. 2D Human Pose Estimation

The location of human joints from monocular images or videos is calculated by 2D human pose estimation (HPE). Before deep learning conducts a sizable impact on vision based HPE. traditional 2D human pose estimation algorithms use a specialised feature extraction and a refined body models to achieve local representations and global pose postures (Chen & Yuille, 2014). Deep learning-based models have been grouped into single person pose estimation and multi-person pose estimation on this occasion.

2.2.1.1. Single Person Pose Estimation

2D single person pose estimation is to circumscribe body joint positions of an individual in an image which derived from an input. If there are multiple people in an image, then pre-processing is required to edit the image in a way that it is segmented into more images where there is only one person per image (Chen, et al., 2020). To carry out this kind of pre-processing there is a full-body available to use by (Ren, et al., 2015).

There are two categories when it comes to HPE methods using convolutional neural networks (CNN) which are regression-based methods and detection-based methods (Chen, et al., 2020). Regression-based methods utilise an end-to-end framework to primarily produce joint coordinates by trying to learn a mapping from an image to kinematic body joint coordinates (Toshev & Szegedy, 2014). Meanwhile, detection-based methods predict approximate locations of body parts or joints, they are regularly supervised by a rectangular window (where each include a specific body part) or heatmaps where each indicate one joint position (Chen, et al., 2020).

2.2.1.2. Multi-Person Pose Estimation

Main difference to 2D single person pose estimation, 2D multi-person pose estimation “need or handle both detection and localisation tasks since there is no prompt of how many persons in the input images” (Chen, et al., 2020). HPE methods can be categorised into top-down methods and bottom-up methods (Chen, et al., 2020).

Top-down methods utilise existing single-person pose estimation methods to predict human poses after it gets person detectors (the predicted poses are heavily dependent on their precision) to retrieve a set of the bounding box of people in the image that has been inputted (Chen, et al., 2020). More people in the input image would proportionally affect the runtime for the entirety of the system. The bottom-up methods are more dependent on prediction, they directly predict the 2D joints of everyone in the input image before it builds them into separate skeletons. It is difficult to precisely group the joint point in a complex environment.

2.3. Posture Risk Assessment Methods

2.3.1. RULA

“The Rapid Upper Limb Assessment (RULA) is one of the most adopted observational methods for assessing working posture” (Li & Xu, 2019) and work-related musculoskeletal disorders’ (WMSD) risks in practice. In essence, RULA manipulates the worker joints’ angle through observing the worker’s posture. The potential risk of a task is evaluated from the score calculated from the formulae. The three factors assessed by this method are: the posture of the different area of the body, the load or force exerted and the muscle activity from static posture or repetitive movements. RULA focuses on two groups of body regions (Gómez-Galán, et al., 2020):

- Group A: arm, forearm, wrist, and wrist turn.
- Group B: neck torso and legs.

Its simplicity and effectiveness are the primary reason for its popularity in industrial practice (Roman-Liu, 2014). The RULA method can be considered to be an evolved version of Ovako Working Posture Analysis System (OWAS) method used for a closer examination of the upper limb (Yazdanirad, et al., 2018).

Advantages of RULA include:

- Fast and easy to use (Sanchez-Lite, et al., 2013).
- For assessing repetitive tasks in the upper limbs, it is a reliable method (Gómez-Galán, et al., 2020).
- It has been adopted on workers in very industry-diverse fields (Gómez-Galán, et al., 2020).

- Minimal to no experience is needed to apply RULA during the observation phase (McAtamney & Corlett, 1993).
- The result is a value which makes it easily comparable (Sanchez-Lite, et al., 2013).
- Its amplitude of simplicity makes its use very accessible (Takala, et al., 2010).
- It is software-aided meaning it can be implemented with the help of software (Takala, et al., 2010) (Sanchez-Lite, et al., 2013).

Limitation of RULA include:

- The left and right side of the body are assessed independently (Takala, et al., 2010).
- The amount of time the worker requires to complete the task is not considered (Takala, et al., 2010).
- Experience evaluators are needed for the evaluation thus decreasing its cost effectiveness (Li & Xu, 2019).
- The subjectivity of the evaluators can result in inconsistent final scores (Li & Xu, 2019).

It is worth mentioning that the two latter limitations can be tackled by adopting wearable sensors, IMUs and electromyography (EMG) sensors as seen in this study (Yan, et al., 2017) where they resulted in real-time data collection. The fact that wearable sensors are intrusive to natural body motion, as stated previously has to be taken into account when evaluating wearable equipment's because workers would be required to wear them for extended periods of time. Moreover, the application of wearable sensors, IMUs and EMGs wouldn’t be feasible for large-scale monitoring purpose (Manghisi, et al., 2017).

2.3.2. NERPA

Novel Ergonomic Postural Assessment Method (NERPA) was presented for the first time in 2013 as one of the “last working posture assessment methods”, which does a better assessment of the physical condition (Sanchez-Lite, et al., 2013). Although NERPA and RULA really have the same structure, NERPA underwent more significant adjustments than RULA. Finally, depending on the posture score, this method establishes four action levels, which is comparable to many observational posture assessment methods that use a scoring system based on unique physical variables for reporting the final score (Khandan, et al., 2018).

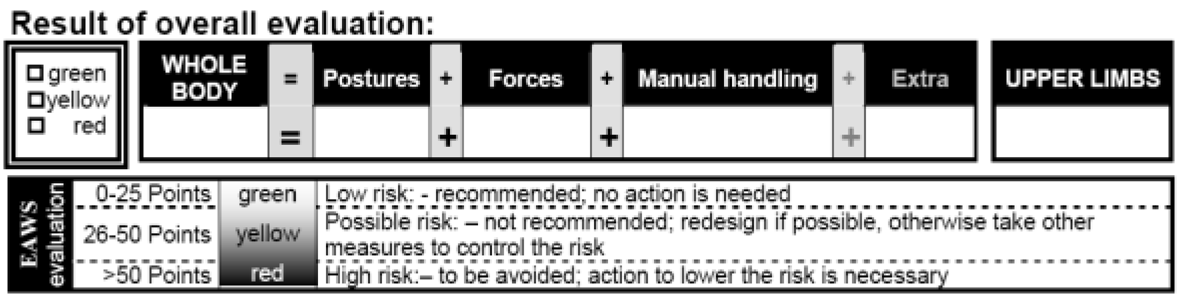
2.3.3. LUBA

Loading on the Upper Body Assessment (LUBA) is an observational and macro-postural methodology. This approach is based on empirical data contrasted with experienced pain and given as a numerical ratio for a series of five precise body motions, including those of the wrists, elbows, shoulders, neck, and waist. Each joint's pain is quantified at five different levels: 0%, 25%, 50%, 75%, and 100% of the joint's range of motion (Kee & Karwowski, 2001). According to the relevant tables, pain points are allocated to each joint's motions in the chosen postures. The postural load is then calculated by adding the scores of each joint. The postural load score corresponds to activity levels ranging from 1 to 4 (Yazdanirad, et al., 2018).

2.3.4. EAWS

The Ergonomic Assessment Worksheet evaluates physical workload on a three-zone rating system (red, yellow, and green) (Schaub, et al., 2013). “This is of limited value for practical applications, as real tasks in industry do not contain either ‘working postures’ or ‘manual materials handling’ but might be a combination of all four types” (Schaub, et al., 2012). A focus of EAWS is existing CEN and ISO standards (Schaub, et al., 2012), which considers four sections for the evaluation of:

- Repetitive loads of the upper limbs (Schaub, et al., 2013)
- Action forces of the entire body or hand-finger system (Schaub, et al., 2013)
- Manual material handling (Schaub, et al., 2012)
- Working postures and movements with low additional physical efforts (<30–40 N or 3–4 kg respectively) (Schaub, et al., 2013).



Result of overall evaluation:

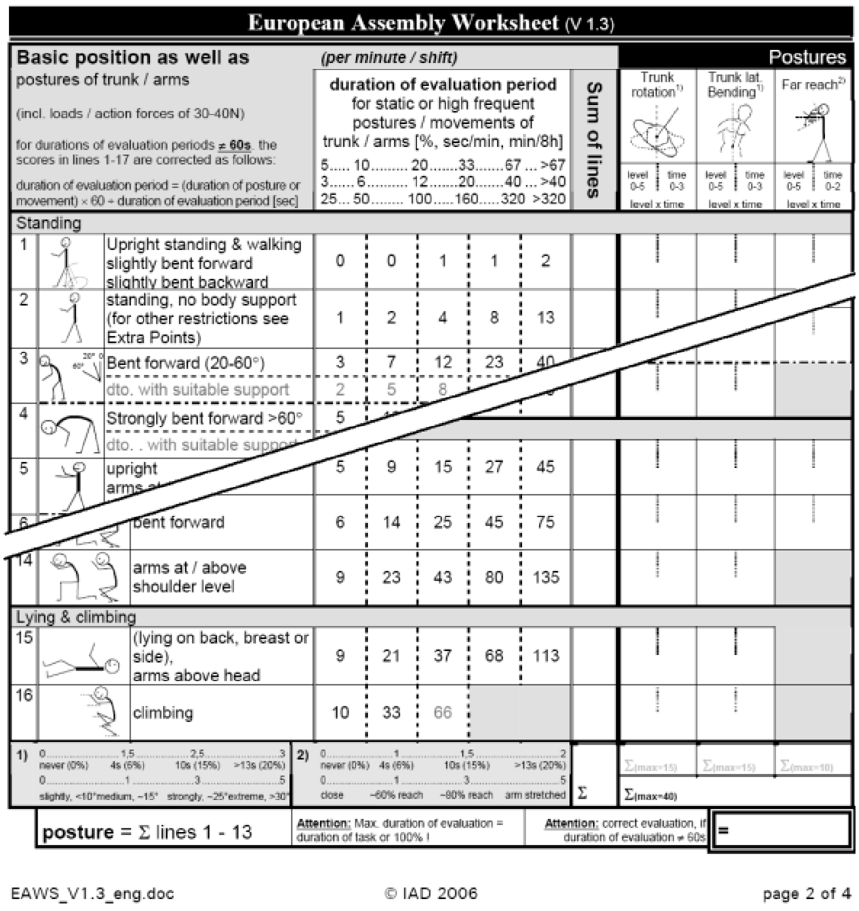
	WHOLE BODY	=	Postures	+	Forces	+	Manual handling	+	Extra	UPPER LIMBS
☐ green ☐ yellow ☐ red		=		+		+		+		

EAWS evaluation			
	0-25 Points	green	Low risk: - recommended; no action is needed
	26-50 Points	yellow	Possible risk: – not recommended; redesign if possible, otherwise take other measures to control the risk
	>50 Points	red	High risk:– to be avoided; action to lower the risk is necessary

Figure 4. EAWS overall evaluation scheme (Schaub, et al., 2012).

A risk assessment is offered by the one of the standards for each of those types of physical workloads (Schaub, et al., 2012). The nature of labour jobs in the industry means that real tasks will not contain either ‘working postures’ or ‘manual material handling’, but it commonly could be a combination of all four types for evaluation (Schaub, et al., 2012) which would limit the value for practical applications of EAWS. So, if a yellow working posture and a yellow manual materials handling are assessed, will it produce a total yellow assessment, aggregate to a red, or might compensate to a green total? The main purpose of development of tools like EAWS, posture risk assessment methods is to solve questions like these.

In the working posture section of the EAWS seen in **Figure 6**, working postures and high frequent movements are estimated. On the left-side symmetric working postures for standing, sitting, kneeling & crouching, and lying & climbing are rated; on the right-side asymmetric effects like rotation, lateral bending and far reach are considered as seen in **Figure 6**. The longer the time spent in unfavourable conditions; the higher the score associated (Schaub, et al., 2012). This section is crosschecked with other international accepted methods and re-levant CEN (EN 1005-4 [12]) and ISO (ISO 11226 [15]) standards (Schaub, et al., 2012). In this section low physical effort (less than 30-40 N or 3-4 kg respectively) is already included in the evaluation.



European Assembly Worksheet (V 1.3)

Basic position as well as postures of trunk / arms (incl. loads / action forces of 30-40N) for durations of evaluation periods ≠ 60s, the scores in lines 1-17 are corrected as follows: duration of evaluation period = (duration of posture or movement) × 60 ÷ duration of evaluation period [sec]			*(per minute / shift)* **duration of evaluation period** for static or high frequent postures / movements of trunk / arms [%, sec/min, min/8h]					**Sum of lines**	**Postures** Trunk rotation[1) level 0-5 / time 0-3 level x time	Trunk lat. Bending[1) level 0-5 / time 0-3 level x time	Far reach[2) level 0-5 / time 0-2 level x time
			5..... 3...... 25...	10......... 6.......... 50........	20....... 12....... 100.....	33....... 20....... 160.....	67 ... >67 40 ... >40 320 >320				
Standing											
1		Upright standing & walking slightly bent forward slightly bent backward	0	0	1	1	2				
2		standing, no body support (for other restrictions see Extra Points)	1	2	4	8	13				
3		Bent forward (20-60°)	3	7	12	23	40				
		dto. with suitable support	2	5	8						
4		Strongly bent forward >60°	5								
		dto. . with suitable suppo									
5		upright arms a	5	9	15	27	45				
6		bent forward	6	14	25	45	75				
14		arms at / above shoulder level	9	23	43	80	135				
Lying & climbing											
15		(lying on back, breast or side), arms above head	9	21	37	68	113				
16		climbing	10	33	66						
1) 0 ... 1,5 ... 2,5 ... 3 never (0%) 4s (6%) 10s (15%) >13s (20%) 0 ... 1 ... 3 ... 5 slightly, <10° medium, ~15° strongly, ~25° extreme, >30°			2) 0 ... 1 ... 1,5 ... 2 never (0%) 4s (6%) 10s (15%) >13s (20%) 0 ... 1 ... 3 ... 5 close ~60% reach ~80% reach arm stretched					Σ	Σ(max=15) Σ(max=40)	Σ(max=15)	Σ(max=10)
	posture = Σ lines 1 - 13		**Attention:** Max. duration of evaluation = duration of task or 100% !				**Attention:** correct evaluation, if duration of evaluation ≠ 60s		=		

EAWS_V1.3_eng.doc © IAD 2006 page 2 of 4

Figure 5. Extract of the EAWS working posture evaluation (Schaub, et al., 2012).

2.4. Gap Analysis

- A lot of the articles focus on construction workers and not on manual assembly workers.
- Accurate solutions are not cost effective.
- My solution is a video image-based posture assessment solution which means it could work with just a camera and a raspberry pie regarding hardware which makes it an inexpensive solution.

Chapter 3 Technical Review

3.1. Video Capture Devices

3.1.1. 1080p 30FPS USB Webcam

A USB webcam is a small digital video camera that can directly be connected to a PC or any device that has a female USB port and drivers that support the device. USB webcams are affordable, for the most part, and very portable (Lander, 2022). The nature of this device makes it solution that isn't very resource intensive. A study (Harish, et al., 2021) used a lower quality webcam and still achieved head pose estimation, mouth opening detection, and eye tracking with reasonable accuracy in a non-intrusive manner.

The limitation of this device is primarily the accuracy. The reason for this limitation is the nature of the device, it's a device that is designed for video calls. This means the lack of high-quality sensors which do not lead to a high-quality video output. The lack of high-quality sensors means it would need to be in a well-lit place to achieve the best results (Hernández, et al., 2021).

3.1.2. Mobile Phone Camera

A lot of mobile phones nowadays possess a rear-facing camera that can record videos at a high quality with some phones being able to record at 8K resolution with 24 frames per second (GSMArena, 2022) and 4K resolution with 60 frames per second being the standard on a flagship mobile phone. They have a flashlight built in so in low-light scenarios they can utilise it to brighten up the target they are recording. 91% of the world’s population owns a mobile phone which means it is a device the user is more than likely to have already (BankMyCell, 2022).

The limitation of using a mobile phone camera in a manual assembly environment is that unless someone is willing to give up a mobile phone then a new one needs to be purchased. If a user isn’t willing to use their personal phone for work purposes, then they would be required to purchase a new phone just for this occasion which would come at a high expense for a phone that would have a decent camera, better than that of a webcam.

3.1.3. Conclusion

A webcam would be the video capture device of choice. This is due to its affordability and still being able to give results that are accurate enough for the scope of this project. Its limitation concerning lighting doesn’t affect this project since manual assembly workspaces are well-lit. To reduce the effect of the other limitation, being accuracy, it will be aimed to place the camera as close to the user as possible so video quality of the human itself will be more than sufficient for the scope of this project.

3.2. Computer Vision Frameworks

3.2.1. Amazon Rekognition

Amazon Rekognition is a computer vision software as a service solution developed by Amazon Web Services (Amazon Web Services, 2022). This service enables you to identify scenes, text, objects, scenes, and activities from a provided image or video (Amazon Web Services, 2022) in real-time. According to their documentation (Amazon Web Services, 2022), computer vision or deep learning expertise is not required to utilise the image and video analysis that this service provides.

According to a study (Sharma, 2022), the image analysis limitations with Amazon Rekognition is that it only supports JPEG and PNG image formats, and the image size is restricted to 15MB when taken as S3 object and 5MB as a raw image. Amazon Rekognition is a paid service which opposes this project's objective of attempting to be as affordable as possible.

3.2.2. OpenCV

OpenCV is an open-source computer vision and machine learning software library with more than 2500 optimised algorithms (OpenCV, 2022). These algorithms enable you to identify objects, follow eye movements, extract 3D models of objects, track moving objects, produce 3D point clouds from stereo cameras as well as many more ways of implementations (OpenCV, 2022). OpenCV has a module that provides GPU acceleration using CUDA (Koriukina, 2020) which has a massive performance upgrade over CPU as seen in **Figure 7**.

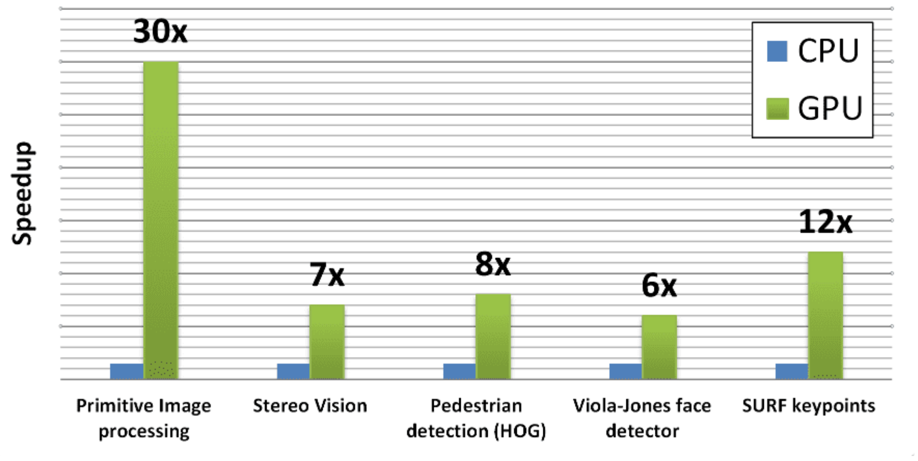


Figure 6. Comparison between OpenCV algorithms on CPU and with CUDA (Koriukina, 2020).

3.2.3. Conclusion

Affordability is one of the focus points which extends to computer vision frameworks as well. For this reason, OpenCV is going to the computer vision framework of choice as it is open source and completely free to use while having all the features needed for this project with no restrictions on content formats and content sizes. OpenCV is also the computer vision framework of choice for most pose estimation project reviewed in the literature review. This increasing user-base means there is more assistance available when it comes to troubleshooting errors. OpenCV also has much better documentation compared to Amazon Rekognition (OpenCV, 2022).

3.3. Human Pose Estimation Frameworks

3.3.1. OpenPose

OpenPose is “first real-time multi-person system to jointly detect human body, foot, hand and facial keypoints on single images” (Martınez, 2019). OpenPose overcomes problems that are seen in other 2D body pose estimation libraries; such as user requires to implement most of the pipeline, frame reader, facial and body keypoints not combined and, results require a separate display library to be visualised; by having a framework that having workarounds built into OpenPose instead of having a different library for each purpose (Martınez, 2019). It is compatible with a variety of operating systems including Windows, Mac OSX, Ubuntu, and embedded systems like the Nvidia Tegra TX2) (Martınez, 2019). OpenPose also supports 2D pose estimation where it had been applied in sport settings (Kitamura, et al., 2022). OpenPose is trained using the COCO dataset for body keypoint estimation which allowed them to excel at single person pose estimation and allowed multi-person pose estimation to be possible (Martınez, 2019).

OpenPose can accurately detect poses of multiple people in real time from an image, but it struggles in extreme poses, like inverted position which is evident in (Kitamura, et al., 2022). The COCO (Common Objects in Context) data set used for training

OpenPose’s network has insufficient unusual poses such as the poses mentioned earlier (Lin, et al., 2014). It requires high performance hardware to achieve the results mentioned because its combined body-foot model ran at ~22 frames per second when an Nvidia GTX 1080 Ti (a high-end GPU) achieved results with “high accuracy” (Martınez, 2019). Since OpenPose is a slow pipeline approach (Moryossef, et al., 2021) it would be disadvantageous since a framework which enables the ability to build high performing numerous pipelines would be more suited for this project due to the numerous inputs and outputs the system needs to perform.

3.3.2. MediaPipe

MediaPipe is a framework that infers from arbitrary sensory data by building pipelines developed and used by Google for 10 years that has been “immensely successful” (Lugaresi, et al., 2019). This solution is designed for machine learning practitioners who implement production-ready machine learning application, build technology prototypes, and publish code accompanying research work (Lugaresi, et al., 2019). MediaPipe can create pipelines that can be accelerated with GPU compute while reusing most of the pipeline configuration and MediaPipe makes it easily to configure a pipeline where a branch is performed on GPU while another branch is running on CPU for potentially better overall performance (Lugaresi, et al., 2019). MediaPipe allows its users to develop in an agile way because it has incremental improvements to pipelines in mind when it was developed. This is possible due to its “rich configuration language and evaluation tools” (Lugaresi, et al., 2019). MediaPipe Holistic utilises the video stream to calculate the skeletal joints for the body and hands which are estimated using regression rather than classification in 3D (Moryossef, et al., 2021).

A Study (Moryossef, et al., 2021) showed MediaPipe can be unreliable because in some of their tests the keypoints for the hands weren't available at all. This occurred because MediaPipe can exclude the hands if it fails to detect it (Moryossef, et al., 2021).

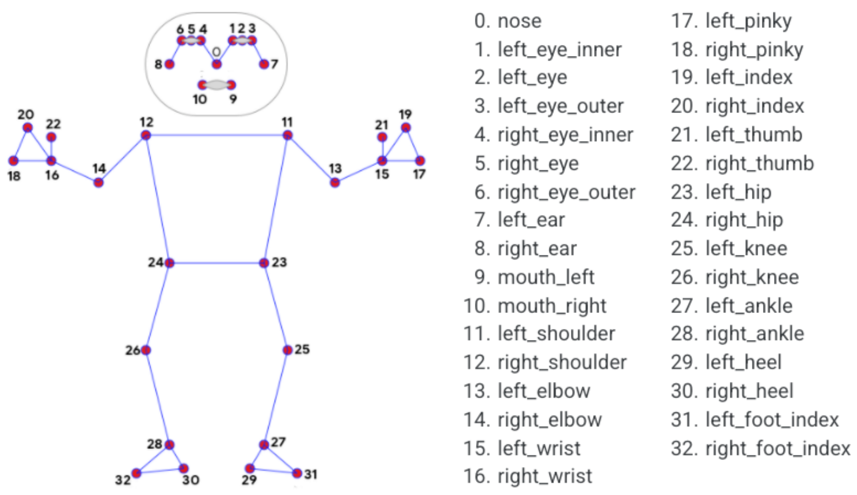


Figure 7. Pose landmarks (Google, 2022)

3.3.3. Conclusion

For this project, MediaPipe will be the human pose estimation of choice because of its focus on pipeline architecture. The project is going to have several inputs from the computer vision framework and the process will output more data that will need to be input for another process and so on which means utilising pipelines will be crucial. MediaPipe also offers more optimisation possibilities compared to OpenPose which means there is room for improvement in the future to make it run more efficiently. A recent study (Amaliya, et al., 2021) involving hand keypoints has shown that MediaPipe's results has been more accurate compared to OpenPose when the hand is detected. MediaPipe's limitation in unreliability in detecting the hand has already been formerly mentioned but since wrist and hand detection is beyond the scope of this project it will not affect the results and usability. This proves to show that human pose estimation is more accurate than OpenPose when keypoints are detected on the body which MediaPipe doesn’t prove to have a disadvantage in compared to OpenPose. The MediaPipe documentation is more abundant than OpenCV which is another advantage, **Figure 8** comes from the documentation stating the different pose landmarks seen by MediaPipe.

3.4. Programming Language

After reviewing the frameworks that are planned to be used, OpenCV as the computer vision framework and MediPipe as the human pose estimation framework, the programming languages can be used can be narrowed down to a few. According to the OpenCV documentation (OpenCV, 2022), OpenCV supports:

- Java
- C++
- Python
- MATLAB

According to the MediaPipe documentation (Google, 2022), MediaPipe supports:

- C++
- JavaScript
- Python
- Coral

C++ and Python are the languages that both of these frameworks support. Due to the deadline of this project, learning and developing with two different languages from scratch will not be realistic so only one language will be chosen.

Table 3. Comparison between C++ and Python (devjobsscanner, 2022) (United States Data Science Institute, 2022) (Schaffer, 2022) (BrainStation, 2022) (Microsoft, 2022) (The Python Software Foundation, 2022).

Description	C++	Python
Open Source	Yes	Yes
Learning Curve	Steep	Shallow
Industry Popularity	Medium	High
Documentation	Good	Excellent
Community Support	Good	Excellent

Python is going to be the language of choice for this project. Python has a shallow learning curve which means completing this project intime will be more realistic since I would need to learn both languages from scratch either way and the better documentation from Python, as seen in Table 2, will assist me in doing so. The higher industry popularity means that it will benefit me in the future with more career opportunities. The fact that C++ is a lower-level language, leading to better performance of applications even when developing an OpenCV application (OpenCV, 2022), needs to be mentioned but due to the reasons formerly mentioned C++ is still going to be dismissed.

Chapter 4 Requirements

This section describes posture evaluation platform functional and non-functional requirements. Functional requirements specify the platform's features and functions. These abilities include assessing body posture, providing quick input, and tracking progress over time. Non-functional criteria address the platform's performance, dependability, and usability. These conditions outline the posture evaluation platform's functions and performance benchmarks.

This section describes posture assessment platform functional and non-functional needs. Functional requirements list the platform's necessary features and capabilities. These include posture assessment, real-time feedback, and progress tracking. Non-functional requirements dictate platform quality, such as performance, reliability, and usability. These requirements outline the posture assessment platform's functions and performance.

4.1. Functional Requirements

The platform needs precise computer vision posture evaluation. The posture evaluation platform must use computer vision to accurately measure and assess an individual's posture. User feedback and recommendations must be precise.

The platform must provide immediate posture feedback and remedial solutions. This requires the platform to quickly provide posture feedback and suggest corrections. This feature lets users make real-time modifications, speeding up posture improvement.

The platform must allow users to track their posture improvements over time. This requires a platform feature that lets users track their progress and posture improvement over time. This feature may motivate users and highlight the platform's benefits.

To enable user interaction and access, the platform's interface should be user-friendly. The platform must have an intuitive interface for users to interact with and utilise its features. Enhancing user experience may increase platform engagement and retention.

4.2. Non-Functional Requirements

Performance: This non-functional requirement concerns posture assessment platform speed and efficiency. The platform must process posture assessments fast and give users instantaneous feedback. This ensures a seamless user experience.

Reliability: This non-functional need refers to posture assessment platform consistency and accuracy. The platform must consistently deliver accurate results. This helps consumers trust the platform and receive accurate posture feedback.

Usability: This non-functional requirement addresses posture assessment platform usability. The platform must have an intuitive and user-friendly interface that allows users of all skill levels to interact with the software and access its capabilities. This enhances user experience and encourages platform adoption.

Scalability: This non-functional criterion means the posture assessment platform can manage many users and assessments without performance deterioration. The platform must scale up without affecting performance or reliability to handle usage increases.

Chapter 5 Design

5.1. System Architecture Diagram

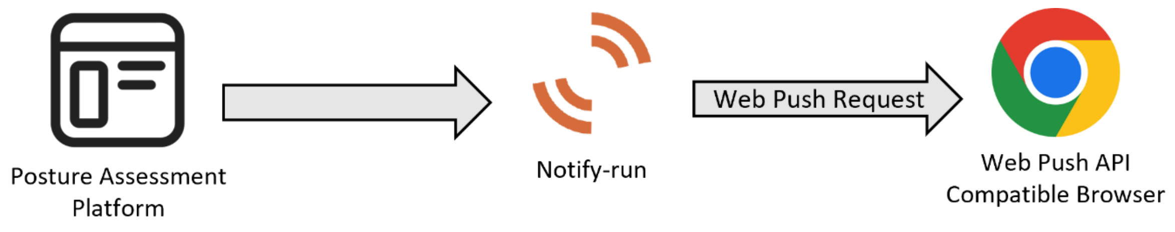


Figure 8. System architecture diagram for the platform.

As shown in in **Figure 8**, the only reason the application would need to interact with any external systems is for the notification alert feature. Notify-run is the library of choice to allow the implementation of alerts and it sends notifications using web push requests. To tell notify-run to send that request the HTTPS endpoint (channel string) is appointed when the app is launched.

5.2. Flow Chart

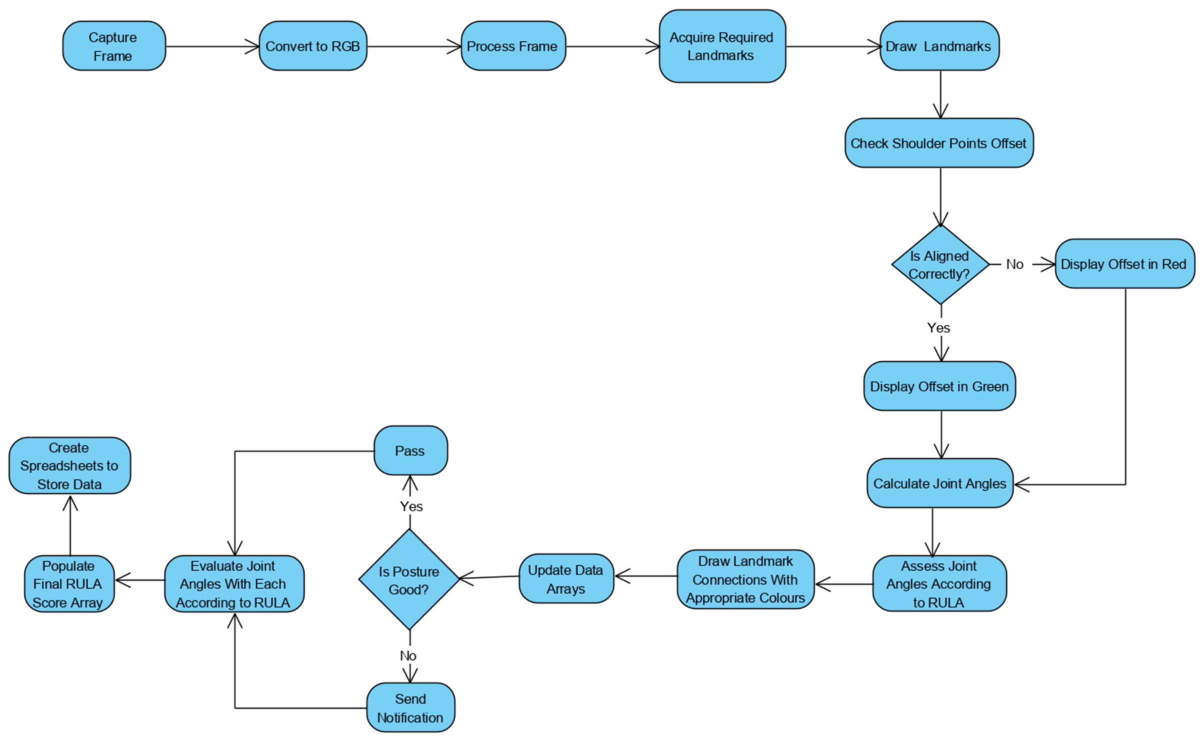

Capture Frame
Convert to RGB
Process Frame
Acquire Required Landmarks
Draw Landmarks
Check Shoulder Points Offset
Is Aligned Correctly?
No
Display Offset in Red
Yes
Display Offset in Green
Calculate Joint Angles
Assess Joint Angles According to RULA
Draw Landmark Connections With Appropriate Colours
Update Data Arrays
Is Posture Good?
Yes
Pass
No
Send Notification
Evaluate Joint Angles With Each According to RULA
Populate Final RULA Score Array
Create Spreadsheets to Store Data

Figure 9. Activity diagram for the platform.

Figure 9 shows a flow chart which essentially depicts the order of execution. The whole thing is a loop, and a more in-depth view of the loop is seen in another flow chart explained in the following chapter.

The diagram is going to be briefly explained, a more in-depth view of the loop can be found in the upcoming chapter. When the frame is captured, it is first converted from BGR to RGB due to historical conventions and differences in colour channel ordering between the libraries. The frame is then process and the landmarks are acquired and drawn again in every loop. This is so the user is receiving real-time feedback. The shoulder offset is checked for calibration reasons and the user will receive a visual message to let them know if they are aligned. The angles of the joints are calculated then evaluated in accordance with RULA so the connection lines between joints can be drawn according to the score. If posture is unfavourable an alert is sent and when the loop is broken using the specified key (which is ‘Q’) the final RULA score is calculated for every frame and store into an array so it can be used to create Excel spreadsheets.

Chapter 6 Research Methodology

6.1. Implementation Methodology

The research project aims to develop a posture assessment platform for manual assembly workers by integrating deep-learning video image-based processing. The approach taken to achieve this is to follow the waterfall model with some modification.

The waterfall model is a sequential development process, in which the progress flows through requirements, design, implementation, testing, deployment and maintenance (Kramer, 2018). The requirements of the system are clear which makes this a favourable approach. Testing is done after the implementation and development is complete which is the best practice in this situation. This is because testing will not be a quick stage to get through due to the nature of the project and the resources does not allow multiple testing stages to be advantageous. Deploying the project in a public or private/internal manner will purely depend on the testing outcome. Maintenance of the software isn't planned but depending on the outcome of the project it might be carried out in the future but will not be documented on this report.

GitHub is going to be the version control hosting service of choice. The implementation progress and code used to make this project can be found here:

https://github.com/amir-saman/Posture-Assessment-Platform

Commit messages on GitHub will have more description of the features and the steps taken to get to the current state of the software.

6.1.1. Implementation Plan

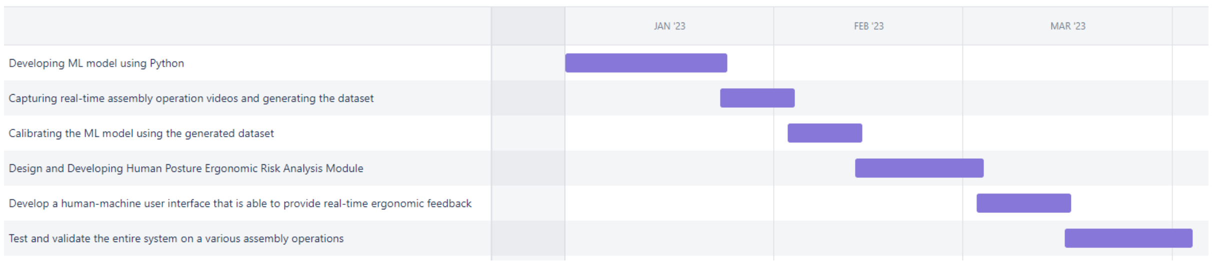


Figure 10. Gantt chart showcasing estimated time for implementation and testing tasks.

For the project, developing a machine learning model is crucial to complete the objectives and it is estimated to take the longest time. The time taken for each task is an estimation, this is because the developer is new to Python and machine learning so getting an accurate duration for each task is not possible. The timeline starts at January the 1st because that's when the implementation will start taking place since the developer is still learning enough knowledge for the topics and programming languages before starting the implementation. The tasks are very broad at the moment due to the same reason but are planned to be broken down to more specific tasks to achieve more clarity for the developer.

The first 10 weeks of the project was primarily used for subject knowledge and understanding. The developer of the project doesn't have prior experience with this subject of study so learning tasks were given to ensure project success. A way to understand this field of study was to get started on this report to research and understand systems that achieved a similar outcome.

6.1. AI Methodology

MediaPipe Pose is a system that estimates the pose of an individual using the BlazePose 33 landmark topology. The BlazePose superset comprises the COCO keypoints, Blaze Palm, and Blaze Face topologies. The procedure involves two phases: identification and pursuit of potential opportunities. MediaPipe achieves faster inference than other solutions by skipping detection between frames (LearnOpenCV, 2023).

MediaPipe employs a machine learning pipeline consisting of a detector and tracker in two stages to estimate posture. The first step of this process involves identifying the pose region-of-interest (ROI) within the frame using a detector. Subsequently, the tracker utilises the ROI to generate forecasts for all 33 pose keypoints. It should be noted that in video applications, the detector is only applied to the initial frame. The pose keypoints detected in the previous frame are utilised to create the region of interest (ROI) for the subsequent frames (Google Research, 2020).

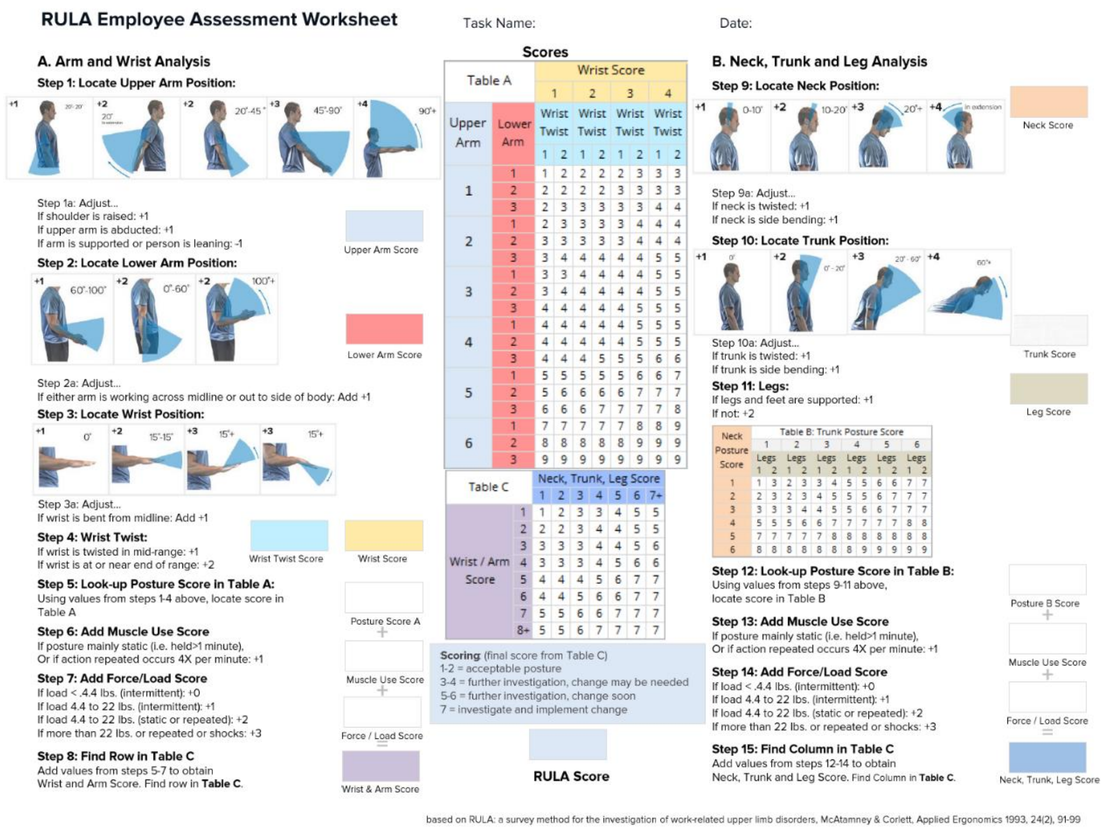
6.1.1. BlazePose

Google Research is responsible for the development of the BlazePose method for recognising human body posture. It employs machine learning to identify 33 2D markers of a body from a single image of the body. The BlazePose model is ideally adapted for use in fitness applications because it can accurately localise more keypoints than other pose models based on the standard COCO topology. This distinguishes it from other models (Google Research, 2020).

The 33 human body keypoints included in the BlazePose topology constitute a superset of the COCO, BlazeFace, and BlazePalm topologies. Due to this, it can determine the body's semantics based solely on the predicted posture, which is compatible with face and hand models. In addition to scale and rotation, the additional keypoints provide crucial information regarding the location of the face, hands, and feet (Google Research, 2020).

BlazePose uses a two-step detector-tracker machine learning workflow to estimate poses. Using a detector, the pose region-of-interest (ROI) is identified within the frame to initiate this pipeline. Based on this ROI, the tracker then makes predictions for all 33 pose keypoints (Google Research, 2020).

6.2. Implementation of Joint Angle Calculation



RULA Employee Assessment Worksheet

Task Name:

Date:

A. Arm and Wrist Analysis

Step 1: Locate Upper Arm Position:

+1 20°-20°
+2 20°
+2 20°-45°
+3 45°-90°
+4 90°+

Step 1a: Adjust...
If shoulder is raised: +1
If upper arm is abducted: +1
If arm is supported or person is leaning: -1

Upper Arm Score

Step 2: Locate Lower Arm Position:

+1 60°-100°
+2 0°-60°
+2 100°+

Lower Arm Score

Step 2a: Adjust...
If either arm is working across midline or out to side of body: Add +1

Step 3: Locate Wrist Position:

+1 0°
+2 15°-15°
+3 15°+
+3 15°+

Step 3a: Adjust...
If wrist is bent from midline: Add +1

Step 4: Wrist Twist:
If wrist is twisted in mid-range: +1
If wrist is at or near end of range: +2

Wrist Twist Score

Wrist Score

Step 5: Look-up Posture Score in Table A:
Using values from steps 1-4 above, locate score in Table A

Posture Score A
+

Step 6: Add Muscle Use Score
If posture mainly static (i.e. held>1 minute),
Or if action repeated occurs 4X per minute: +1

Muscle Use Score
+

Step 7: Add Force/Load Score
If load < .4.4 lbs. (intermittent): +0
If load 4.4 to 22 lbs. (intermittent): +1
If load 4.4 to 22 lbs. (static or repeated): +2
If more than 22 lbs. or repeated or shocks: +3

Force / Load Score
=

Step 8: Find Row in Table C
Add values from steps 5-7 to obtain
Wrist and Arm Score. Find row in **Table C**.

Wrist & Arm Score

Scores

Table A		Wrist Score							
		1		2		3		4	
Upper Arm	Lower Arm	Wrist Twist		Wrist Twist		Wrist Twist		Wrist Twist	
		1	2	1	2	1	2	1	2
1	1	1	2	2	2	2	3	3	3
	2	2	2	2	2	3	3	3	3
	3	2	3	3	3	3	3	4	4
2	1	2	3	3	3	3	4	4	4
	2	3	3	3	3	3	4	4	4
	3	3	4	4	4	4	4	5	5
3	1	3	3	4	4	4	4	5	5
	2	3	4	4	4	4	4	5	5
	3	4	4	4	4	4	5	5	5
4	1	4	4	4	4	4	5	5	5
	2	4	4	4	4	4	5	5	5
	3	4	4	4	5	5	5	6	6
5	1	5	5	5	5	5	6	6	7
	2	5	6	6	6	6	7	7	7
	3	6	6	6	7	7	7	7	8
6	1	7	7	7	7	7	8	8	9
	2	8	8	8	8	8	9	9	9
	3	9	9	9	9	9	9	9	9

Table C		Neck, Trunk, Leg Score						
		1	2	3	4	5	6	7+
Wrist / Arm Score	1	1	2	3	3	4	5	5
	2	2	2	3	4	4	5	5
	3	3	3	3	4	4	5	6
	4	3	3	3	4	5	6	6
	5	4	4	4	5	6	7	7
	6	4	4	5	6	6	7	7
	7	5	5	6	6	7	7	7
	8+	5	5	6	7	7	7	7

Scoring (final score from Table C)
1-2 = acceptable posture
3-4 = further investigation, change may be needed
5-6 = further investigation, change soon
7 = investigate and implement change

RULA Score

B. Neck, Trunk and Leg Analysis

Step 9: Locate Neck Position:

+1 0-10°
+2 10-20°
+3 20°+
+4 in extension

Neck Score

Step 9a: Adjust...
If neck is twisted: +1
If neck is side bending: +1

Step 10: Locate Trunk Position:

+1 0°
+2 0°-20°
+3 20°-60°
+4 60°+

Trunk Score

Step 10a: Adjust...
If trunk is twisted: +1
If trunk is side bending: +1

Step 11: Legs:
If legs and feet are supported: +1
If not: +2

Leg Score

Neck Posture Score	Table B: Trunk Posture Score											
	1		2		3		4		5		6	
	Legs		Legs		Legs		Legs		Legs		Legs	
	1	2	1	2	1	2	1	2	1	2	1	2
1	1	3	2	3	3	4	5	5	6	6	7	7
2	2	3	2	3	4	5	5	5	6	7	7	7
3	3	3	3	4	4	5	5	6	6	7	7	7
4	5	5	5	6	6	7	7	7	7	7	8	8
5	7	7	7	7	7	8	8	8	8	8	8	8
6	8	8	8	8	8	8	8	9	9	9	9	9

Step 12: Look-up Posture Score in Table B:
Using values from steps 9-11 above,
locate score in Table B

Posture B Score
+

Step 13: Add Muscle Use Score
If posture mainly static (i.e. held>1 minute),
Or if action repeated occurs 4X per minute: +1

Muscle Use Score
+

Step 14: Add Force/Load Score
If load < .4.4 lbs. (intermittent): +0
If load 4.4 to 22 lbs. (intermittent): +1
If load 4.4 to 22 lbs. (static or repeated): +2
If more than 22 lbs. or repeated or shocks: +3

Force / Load Score
=

Step 15: Find Column in Table C
Add values from steps 12-14 to obtain
Neck, Trunk and Leg Score. Find Column in **Table C**.

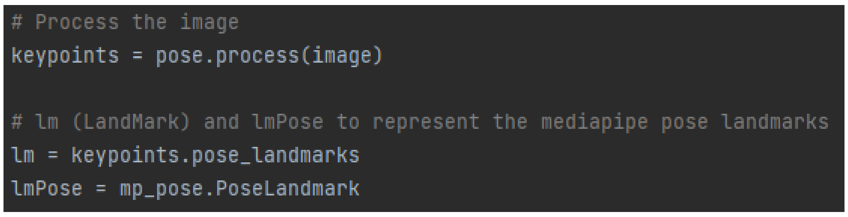
Neck, Trunk, Leg Score

based on RULA: a survey method for the investigation of work-related upper limb disorders, McAtamney & Corlett, Applied Ergonomics 1993, 24(2), 91-99

Figure 11. RULA Worksheet (Source: (Middlesworth, 2023)).

From **Figure 11**, it is apparent that the calculation of angles between joints is an essential implementation for this project. A library, MediaPipe, provides joint detection estimation and what is necessary is to calculate the coordinates of these joints. The joint coordinate calculations are required so there are numerical values available to use to calculate joint angles. From these joint angles, the posture performance can be assessed in accordance with RULA.

6.2.1. Joint Coordinate Estimation



```
# Process the image
keypoints = pose.process(image)

# lm (LandMark) and lmPose to represent the mediapipe pose landmarks
lm = keypoints.pose_landmarks
lmPose = mp_pose.PoseLandmark
```

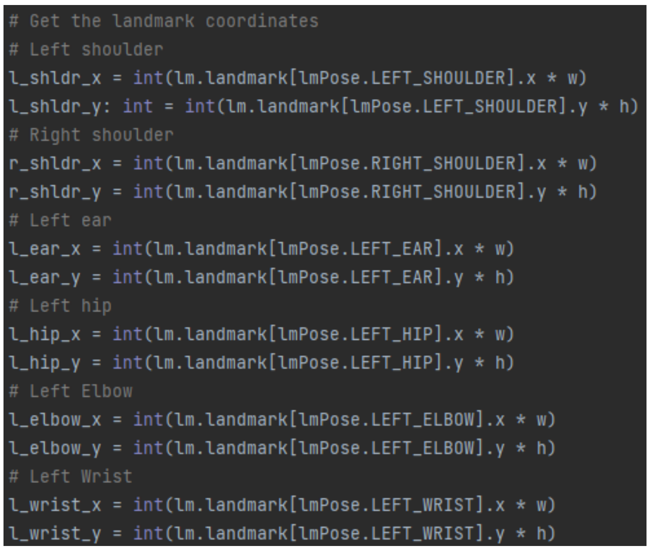
Figure 12. Code to get landmarks.

This code snippet, labelled as **Figure 12**, demonstrates how to use the MediaPipe library to perform pose estimation on an image. This code snippet is designed to generate keypoints, also known as landmarks, for the detected pose. After extracting the keypoints, the specific pose landmarks can be accessed and assigned to the variable named "lm". This enables you to manipulate and analyse the pose landmarks as needed. The variable lmPose is an enumeration that allows you to refer to specific body parts within a pose estimation model.

keys = pose.process(picture): The pose estimation model represented by the pose object is used to process the image in this line. It performs an image analysis and extracts keypoints (landmarks) that stand in for the image's identified attitude. The keypoints variable holds the final keypoints.

keypoints = lm.pose_landmarks: This line gives the variable lm the pose landmarks that were taken from the keypoints object. The major elements of the identified pose in the image are represented by the pose landmarks. You can acquire details on certain stance landmarks, such as the location, orientation, and visibility of particular body parts, by gaining access to the lm object.

lmPose equals mp_pose.PoseLandmark: This line sets the variable lmPose to the PoseLandmark class from the mp_pose module. The enumerated values for pose landmarks are represented by the PoseLandmark class. It gives the pose estimation model a way to reference various body parts. Use lmPose as an illustration.use the identifier LEFT_SHOULDER to refer to the left shoulder.



```
# Get the landmark coordinates
# Left shoulder
l_shldr_x = int(lm.landmark[lmPose.LEFT_SHOULDER].x * w)
l_shldr_y: int = int(lm.landmark[lmPose.LEFT_SHOULDER].y * h)
# Right shoulder
r_shldr_x = int(lm.landmark[lmPose.RIGHT_SHOULDER].x * w)
r_shldr_y = int(lm.landmark[lmPose.RIGHT_SHOULDER].y * h)
# Left ear
l_ear_x = int(lm.landmark[lmPose.LEFT_EAR].x * w)
l_ear_y = int(lm.landmark[lmPose.LEFT_EAR].y * h)
# Left hip
l_hip_x = int(lm.landmark[lmPose.LEFT_HIP].x * w)
l_hip_y = int(lm.landmark[lmPose.LEFT_HIP].y * h)
# Left Elbow
l_elbow_x = int(lm.landmark[lmPose.LEFT_ELBOW].x * w)
l_elbow_y = int(lm.landmark[lmPose.LEFT_ELBOW].y * h)
# Left Wrist
l_wrist_x = int(lm.landmark[lmPose.LEFT_WRIST].x * w)
l_wrist_y = int(lm.landmark[lmPose.LEFT_WRIST].y * h)
```

Figure 13. Code to estimate joint coordinates.

This code snippet in **Figure 13** extracts the coordinates of various landmarks from a pose estimation model output.

This line of code assigns the integer value of the left shoulder landmark's x-coordinate multiplied by the width to the variable l_shldr_x. The lmPose and lm variables are assumed to have been previously defined. This line computes the value of the x-coordinate associated with the left shoulder landmark. The code accesses the "landmark" attribute of the "lm" object. This attribute represents the landmarks of the detected pose. The code uses "lmPose.LEFT_SHOULDER" as an index to retrieve the landmark of the left shoulder. The code multiplies the x-coordinate with the value of w, which is assumed to be the width of the image or frame. After that, the resulting value is converted to an integer using the int() function. The value is assigned to the variable l_shldr_x.

l_shldr_y: int = int(lm.landmark[lmPose.LEFT_SHOULDER].y * h): This line computes the y-coordinate of the left shoulder landmark in a way that is similar to the previous line. This code retrieves the left shoulder landmark from the landmark attribute of the lm object. It uses lmPose.LEFT_SHOULDER as the index to access the specific landmark. The resulting y-coordinate is multiplied by h (presumably the height of the image or frame) and converted to an integer. The value is assigned to the variable l_shldr_y.

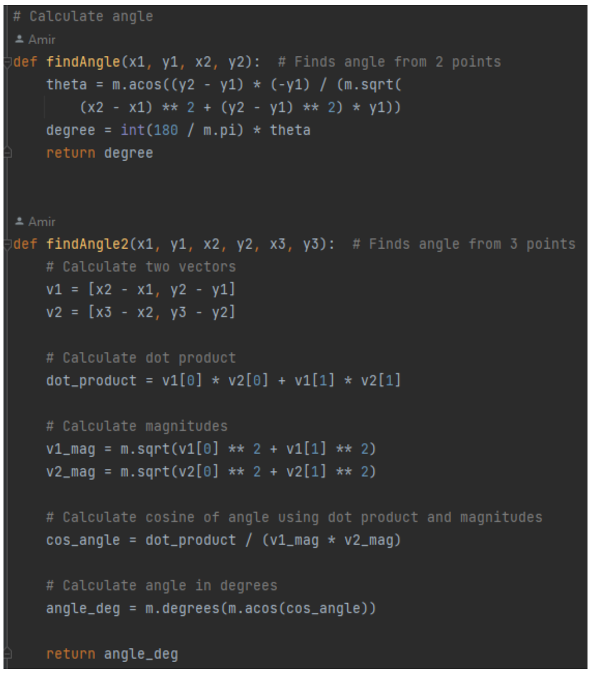
A pattern is repeated in the following lines to extract coordinates for various landmarks. These landmarks include the right shoulder, left ear, left hip, left elbow, and left wrist. To calculate the coordinates, the code accesses landmarks from the list called "lm.landmark" using the values assigned to the "lmPose" enumeration. The x-coordinates are multiplied by w, and the y-coordinates are multiplied by h. The values obtained are converted into integers and then stored in their respective variables (r_shldr_x, r_shldr_y, l_ear_x, l_ear_y, l_hip_x, l_hip_y, l_elbow_x, l_elbow_y, l_wrist_x, l_wrist_y).

This code extracts the x and y coordinates of landmarks from the output of a pose estimation model. This code accesses landmarks by utilising the lmPose enumeration values as indices in the lm.landmark list. The resulting coordinates are scaled by the width (w) and height (h) of the image or frame and stored in separate variables. The variables are then transformed into integers and assigned to the respective variables (r_shldr_x, r_shldr_y, l_ear_x, l_ear_y, l_hip_x, l_hip_y, l_elbow_x, l_elbow_y, and l_wrist_x, l_wrist_y) which are the coordinates of the landmarks (joints) which will later go through a function that calculates angles.

In conclusion, this code snippet extracts from the output of a posture estimate model the x and y coordinates of numerous landmarks. By using the associated lmPose enumeration values as indices in the lm.landmark list, it may access the landmarks. The generated coordinates are scaled by the image or frame's width (w) and height (h), and they are kept in different variables.

6.2.2. Joint Angle Calculation

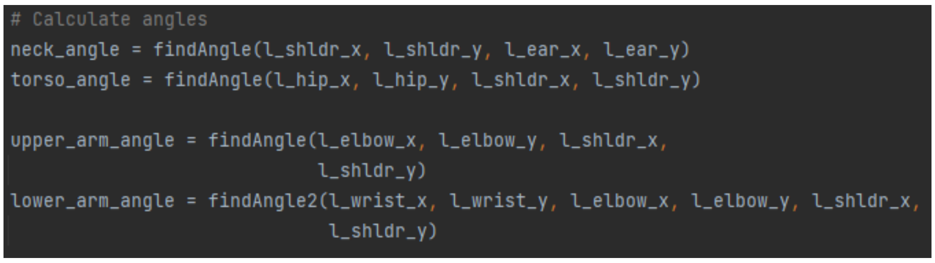
There are two ways angles are calculated as depicted in **Figure 14**. The first function called ‘findAngle’ calculates angles from two points (but from four coordinates because each x and y coordinate make up a point) and ‘findAngle2’ calculates the angle from three points (but from six coordinates). These functions apply basic Pythagoras Theorem principles to achieve the desired outcome.



```python
# Calculate angle
Amir
def findAngle(x1, y1, x2, y2):  # Finds angle from 2 points
    theta = m.acos((y2 - y1) * (-y1) / (m.sqrt(
        (x2 - x1) ** 2 + (y2 - y1) ** 2) * y1))
    degree = int(180 / m.pi) * theta
    return degree


Amir
def findAngle2(x1, y1, x2, y2, x3, y3):  # Finds angle from 3 points
    # Calculate two vectors
    v1 = [x2 - x1, y2 - y1]
    v2 = [x3 - x2, y3 - y2]

    # Calculate dot product
    dot_product = v1[0] * v2[0] + v1[1] * v2[1]

    # Calculate magnitudes
    v1_mag = m.sqrt(v1[0] ** 2 + v1[1] ** 2)
    v2_mag = m.sqrt(v2[0] ** 2 + v2[1] ** 2)

    # Calculate cosine of angle using dot product and magnitudes
    cos_angle = dot_product / (v1_mag * v2_mag)

    # Calculate angle in degrees
    angle_deg = m.degrees(m.acos(cos_angle))

    return angle_deg
```

Figure 14. Two ways to calculate joint angles.

The former function is used to calculate the joint angle for neck, torso, and upper arm. The latter function is used to calculate the joint angle only for lower arm. Their method of implementation is seen in **Figure 15**.



```
# Calculate angles
neck_angle = findAngle(l_shldr_x, l_shldr_y, l_ear_x, l_ear_y)
torso_angle = findAngle(l_hip_x, l_hip_y, l_shldr_x, l_shldr_y)

upper_arm_angle = findAngle(l_elbow_x, l_elbow_y, l_shldr_x,
                            l_shldr_y)
lower_arm_angle = findAngle2(l_wrist_x, l_wrist_y, l_elbow_x, l_elbow_y, l_shldr_x,
                             l_shldr_y)
```

Figure 15. Code that calculates joint angles.

The number of points used to calculate angles depends on the specific features and the angle measurement that is needed.

For the neck angle and chest angle, which are measured between two body parts, you only need two points. To figure out the neck angle, measure from the left shoulder to the left ear. To figure out the trunk angle, measure from the left hip to the left shoulder. You can figure out these angles by looking at where two body parts are in relation to each other.

The lower arm angle, on the other hand, needs three points because the arm bends at the elbow joint. To get a good idea of the slope of the lower arm, the left wrist, the left elbow, and the left shoulder are all considered. By taking these three things into account, the right position at the elbow joint can be found.

Whether you use two or three points relies on the body part and how you want to measure the angle. Depending on the shape of the body part or joint, you may need a different number of points to correctly show and measure the angles.

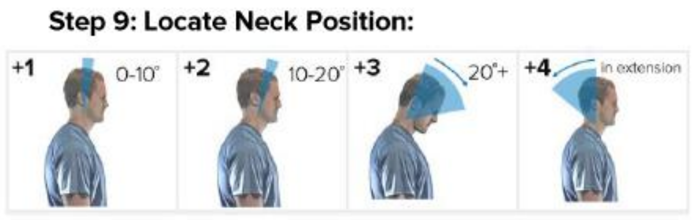
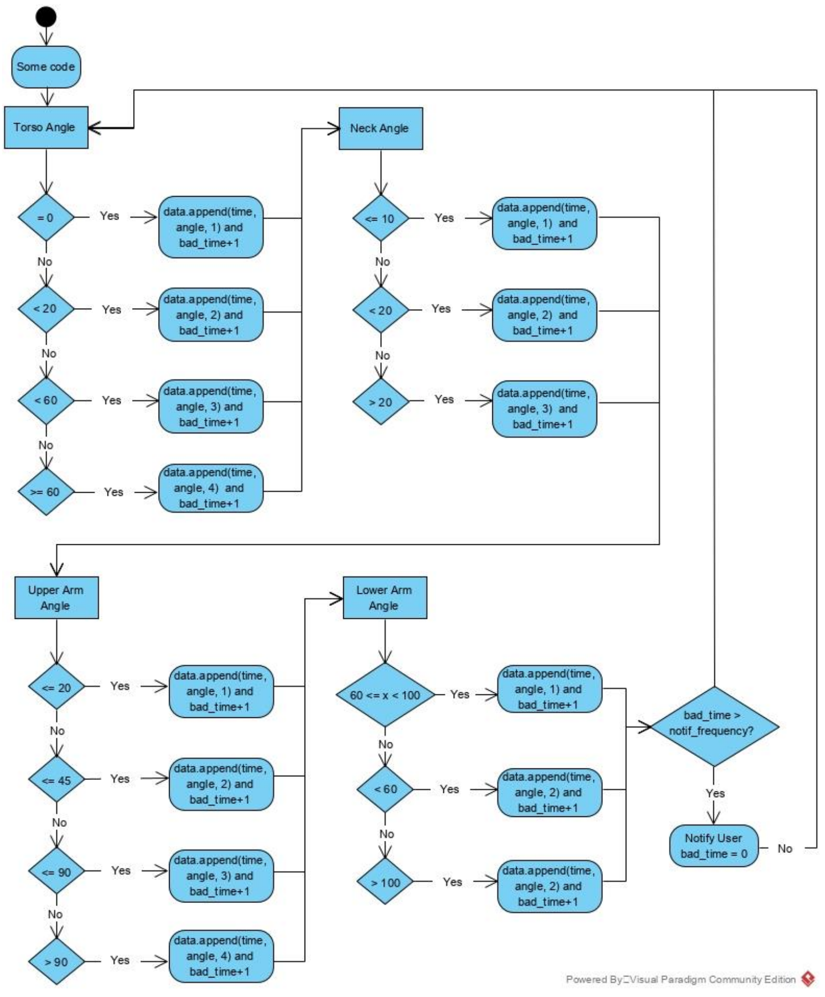


Figure 16. Neck position angle according to RULA (Source: (Middlesworth, 2023)).

A line of code, in **Figure 15**, defines a variable called "neck_angle" and assigns it the value returned by the function "findAngle". The function takes four arguments: the x and y coordinates of the left shoulder ("l_shldr_x" and "l_shldr_y") and the x and y coordinates of the left ear ("l_ear_x" and "l_ear_y"). The purpose of the function is to calculate the angle between the left shoulder and left ear, and the resulting value is stored in the "neck_angle" variable This line computes the angle between two points: the left shoulder (l_shldr_x, l_shldr_y) and the left ear (l_ear_x, l_ear_y). The code invokes the findAngle() function, passing in the coordinates of two points, and stores the resulting angle in the variable neck_angle. The code follows the RULA guidelines, as shown in **Figure 16**, which specify that the ear and shoulder should be used in the calculation.

A similar implementation has been done for torso and upper arm. For torso joint angle calculation, the hip and shoulder points were used, however, for upper arm it was elbow and shoulder. A different approach was taken for lower arm joint angle calculation, which was to use three points, the reason for this was formerly explained. The points for that were the wrist, elbow, and shoulder. The reason for these decisions regarding the points used was to be in accordance with RULA guideline, which are shown in **Figure 11**.

6.3. Implementation of Scoring System



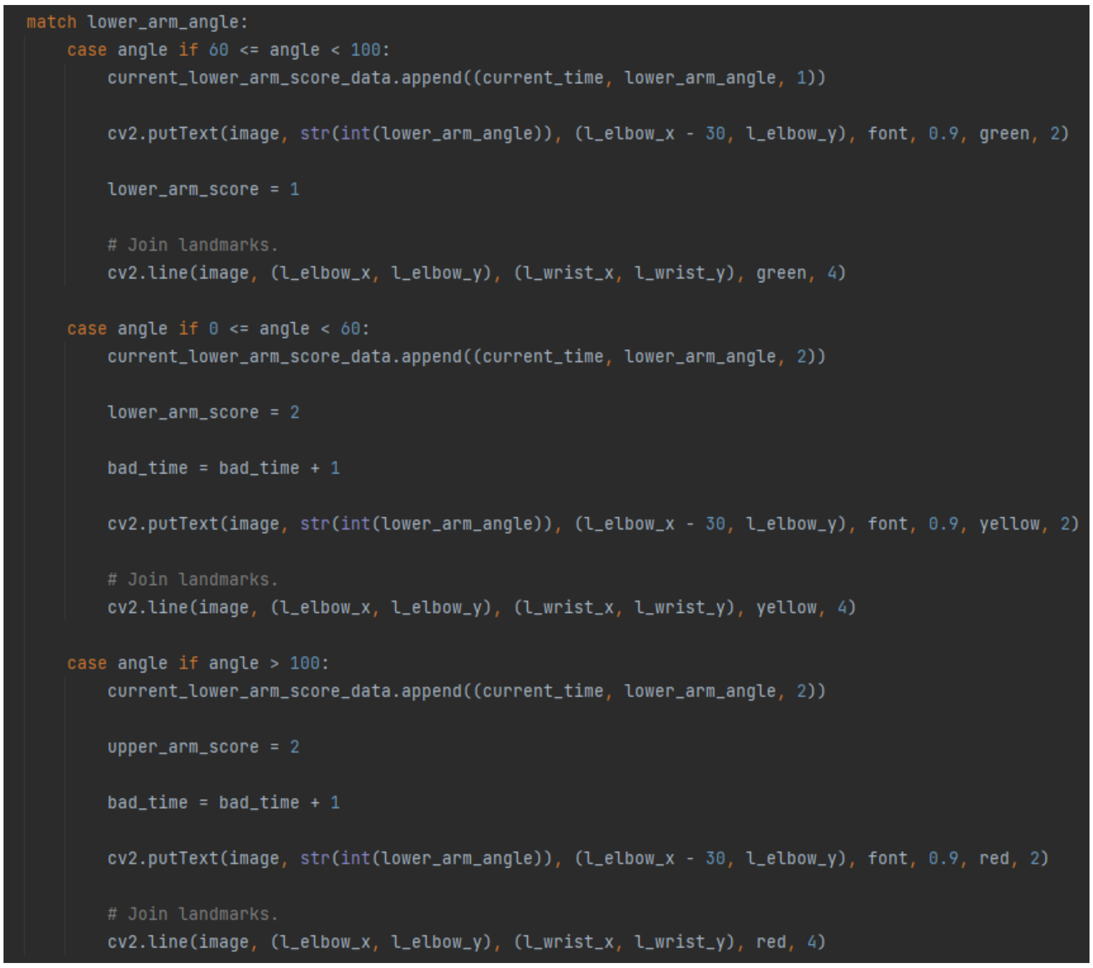
Some code
Torso Angle
= 0
Yes
data.append(time, angle, 1) and bad_time+1
No
< 20
Yes
data.append(time, angle, 2) and bad_time+1
No
< 60
Yes
data.append(time, angle, 3) and bad_time+1
No
>= 60
Yes
data.append(time, angle, 4) and bad_time+1
Neck Angle
<= 10
Yes
data.append(time, angle, 1) and bad_time+1
No
< 20
Yes
data.append(time, angle, 2) and bad_time+1
No
> 20
Yes
data.append(time, angle, 3) and bad_time+1
Upper Arm Angle
<= 20
Yes
data.append(time, angle, 1) and bad_time+1
No
<= 45
Yes
data.append(time, angle, 2) and bad_time+1
No
<= 90
Yes
data.append(time, angle, 3) and bad_time+1
No
> 90
Yes
data.append(time, angle, 4) and bad_time+1
Lower Arm Angle
60 <= x < 100
Yes
data.append(time, angle, 1) and bad_time+1
No
< 60
Yes
data.append(time, angle, 2) and bad_time+1
No
> 100
Yes
data.append(time, angle, 2) and bad_time+1
bad_time > notif_frequency?
Yes
Notify User bad_time = 0
No
Powered By Visual Paradigm Community Edition

Figure 17. Code inside the loop, simplified to show the scoring system implementation.

The scoring system's implementation is depicted in a simplified manner in **Figure 17**. The flow chart presented is an oversimplification as it omits crucial code elements such as variable initializations and extraneous code that serves other purposes.

The programme initially evaluates the computed torso angle and subsequently generates varying results based on the obtained value. In the event that the value is equal to zero, the array that accommodates the torso information at a specific moment will be augmented with the present elapsed time, torso angle, and a score of one for the torso. If the previous condition is not met, the subsequent possibility will be evaluated. The aforementioned process is executed through a case switch statement, as depicted in **Figure 18**. The aforementioned approach is implemented with the aim of optimising the program's performance.

Upon termination of the programme through the activation of the 'Q' key on the keyboard, all scores are exported to a Microsoft Excel spreadsheet. The selection of Excel as the preferred software is based on the high probability of its availability to the user, as well as its capacity to store data in a space-efficient manner, given that the data solely comprises numerical lists.



```python
match lower_arm_angle:
    case angle if 60 <= angle < 100:
        current_lower_arm_score_data.append((current_time, lower_arm_angle, 1))

        cv2.putText(image, str(int(lower_arm_angle)), (l_elbow_x - 30, l_elbow_y), font, 0.9, green, 2)

        lower_arm_score = 1

        # Join landmarks.
        cv2.line(image, (l_elbow_x, l_elbow_y), (l_wrist_x, l_wrist_y), green, 4)

    case angle if 0 <= angle < 60:
        current_lower_arm_score_data.append((current_time, lower_arm_angle, 2))

        lower_arm_score = 2

        bad_time = bad_time + 1

        cv2.putText(image, str(int(lower_arm_angle)), (l_elbow_x - 30, l_elbow_y), font, 0.9, yellow, 2)

        # Join landmarks.
        cv2.line(image, (l_elbow_x, l_elbow_y), (l_wrist_x, l_wrist_y), yellow, 4)

    case angle if angle > 100:
        current_lower_arm_score_data.append((current_time, lower_arm_angle, 2))

        upper_arm_score = 2

        bad_time = bad_time + 1

        cv2.putText(image, str(int(lower_arm_angle)), (l_elbow_x - 30, l_elbow_y), font, 0.9, red, 2)

        # Join landmarks.
        cv2.line(image, (l_elbow_x, l_elbow_y), (l_wrist_x, l_wrist_y), red, 4)
```

Figure 18. Example of one of the case switch statements, lower arm in this case).

The scoring mechanism endeavours to automate the process of posture assessment using the Rapid Upper Limb Assessment (RULA) worksheet, thereby rendering the scoring figures entirely dependent on the said worksheet. **Figure 17** depicts the assessment of the angles at which the neck, torso, upper and lower arm joints join. This was undertaken to automate the entire Rapid Upper Limb Assessment (RULA) worksheet. However, it should be noted that this objective exceeds the scope of the present project. The project's focus was limited to the evaluation of the neck and torso, as well as the completion of Table B illustrated in **Figure 11**.

6.3.1. Implementation of Table A of RULA Worksheet

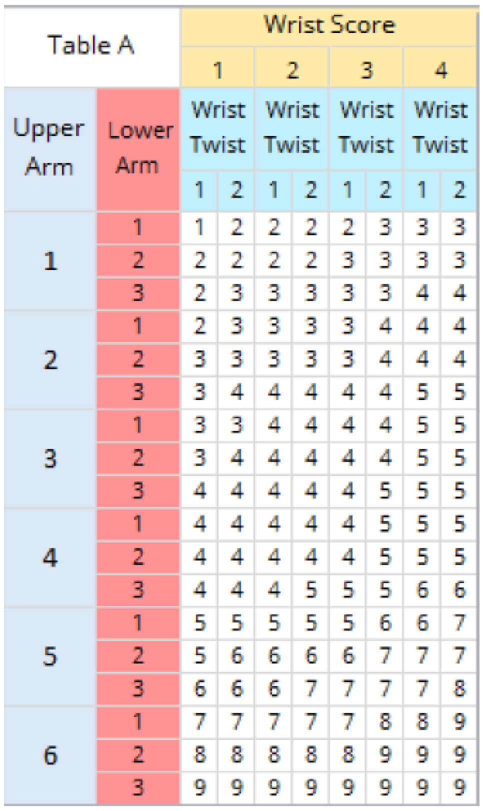
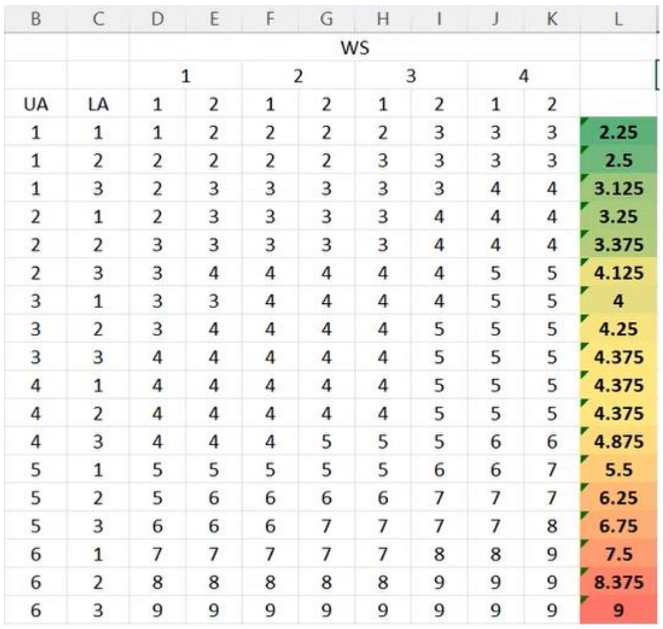


Table A		Wrist Score							
		1		2		3		4	
Upper Arm	Lower Arm	Wrist Twist		Wrist Twist		Wrist Twist		Wrist Twist	
		1	2	1	2	1	2	1	2
1	1	1	2	2	2	2	3	3	3
	2	2	2	2	2	3	3	3	3
	3	2	3	3	3	3	3	4	4
2	1	2	3	3	3	3	4	4	4
	2	3	3	3	3	3	4	4	4
	3	3	4	4	4	4	4	5	5
3	1	3	3	4	4	4	4	5	5
	2	3	4	4	4	4	4	5	5
	3	4	4	4	4	4	5	5	5
4	1	4	4	4	4	4	5	5	5
	2	4	4	4	4	4	5	5	5
	3	4	4	4	5	5	5	6	6
5	1	5	5	5	5	5	6	6	7
	2	5	6	6	6	6	7	7	7
	3	6	6	6	7	7	7	7	8
6	1	7	7	7	7	7	8	8	9
	2	8	8	8	8	8	9	9	9
	3	9	9	9	9	9	9	9	9

Figure 19. Table A of RULA Worksheet (Source: (Middlesworth, 2023)).

There were instances in which the library's use of the RULA spreadsheet was constrained. To evaluate the wrist position, for instance, an import of another library was necessary, but there was a compatibility fault with the other library, making it impossible to evaluate the wrist score and wrist twist.

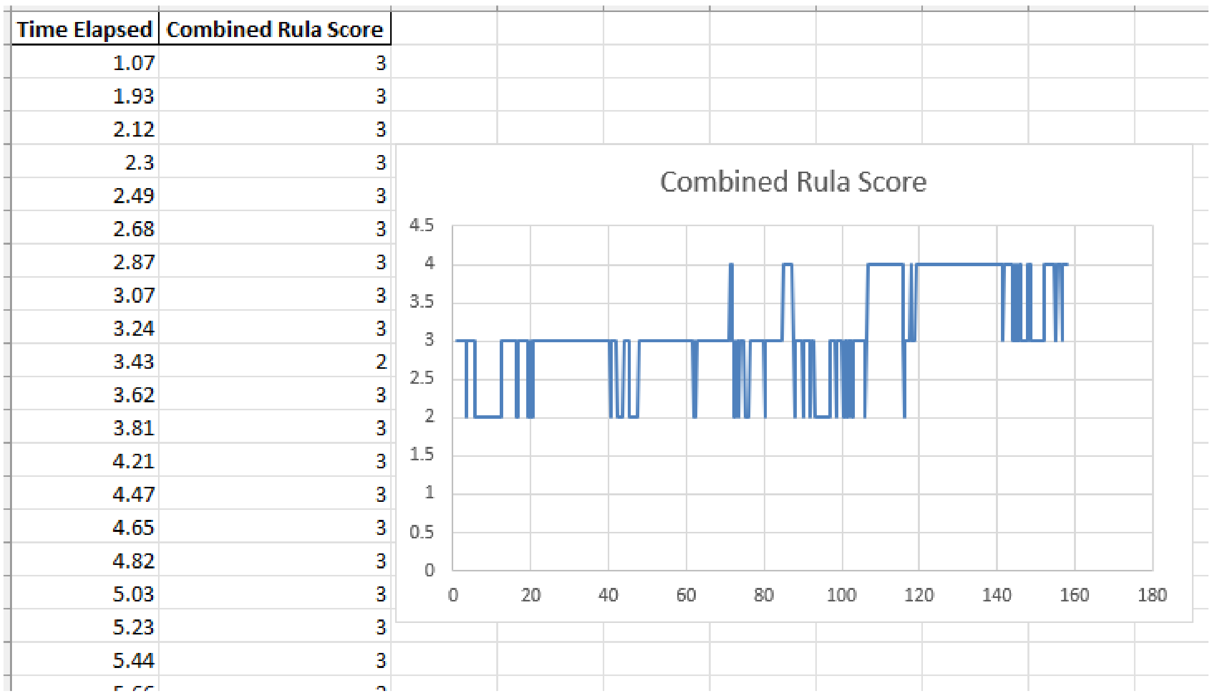


B	C	D	E	F	G	H	I	J	K	L
		WS								
		1		2		3		4		
UA	LA	1	2	1	2	1	2	1	2	
1	1	1	2	2	2	2	3	3	3	**2.25**
1	2	2	2	2	2	3	3	3	3	**2.5**
1	3	2	3	3	3	3	3	4	4	**3.125**
2	1	2	3	3	3	3	4	4	4	**3.25**
2	2	3	3	3	3	3	4	4	4	**3.375**
2	3	3	4	4	4	4	4	5	5	**4.125**
3	1	3	3	4	4	4	4	5	5	**4**
3	2	3	4	4	4	4	5	5	5	**4.25**
3	3	4	4	4	4	4	5	5	5	**4.375**
4	1	4	4	4	4	4	5	5	5	**4.375**
4	2	4	4	4	4	4	5	5	5	**4.375**
4	3	4	4	4	5	5	5	6	6	**4.875**
5	1	5	5	5	5	5	6	6	7	**5.5**
5	2	5	6	6	6	6	7	7	7	**6.25**
5	3	6	6	6	7	7	7	7	8	**6.75**
6	1	7	7	7	7	7	8	8	9	**7.5**
6	2	8	8	8	8	8	9	9	9	**8.375**
6	3	9	9	9	9	9	9	9	9	**9**

Figure 20. Table that averages every column.

To combat this problem, every row was averaged out, and the result is seen in column L in Figure 17, so there would still be numbers that could be worked with. **Figure 20** shows how the table was averaged. This way the scores collected from upper arm and lower arm can still be used to contribute towards the final score because if Table A was to be rejected completely then there wouldn't be scores that could be used for Table C to get the final RULA score.

Figure 17 shows that there was some data appended into an array. There are multiple arrays being populated with data (for cumulative lower and upper arm score data and combined score from **Figure 19**. There is also data at any given point which are more important). The data is being populated via case switch statements depending on the angle and such seen in the flow chart in **Figure 17**.

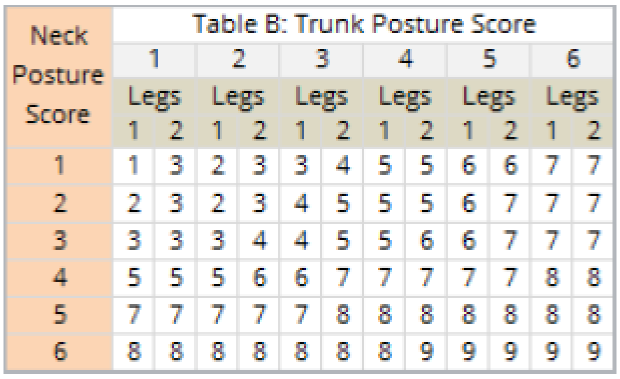


Time Elapsed	Combined Rula Score
1.07	3
1.93	3
2.12	3
2.3	3
2.49	3
2.68	3
2.87	3
3.07	3
3.24	3
3.43	2
3.62	3
3.81	3
4.21	3
4.47	3
4.65	3
4.82	3
5.03	3
5.23	3
5.44	3

Figure 21. Graph to output arm and wrist combined score.

Figure 21 shows the performance of arm and wrist posture in a session, the lower the score the better. It shows the score at any given point/time which is more useful than cumulative data in all cases in this program. This is because the user can then identify at which point in time, they are having their posture compromised and they can act upon that to improve their postural performance for the next session.

6.3.2. Implementation of Table B of RULA Worksheet

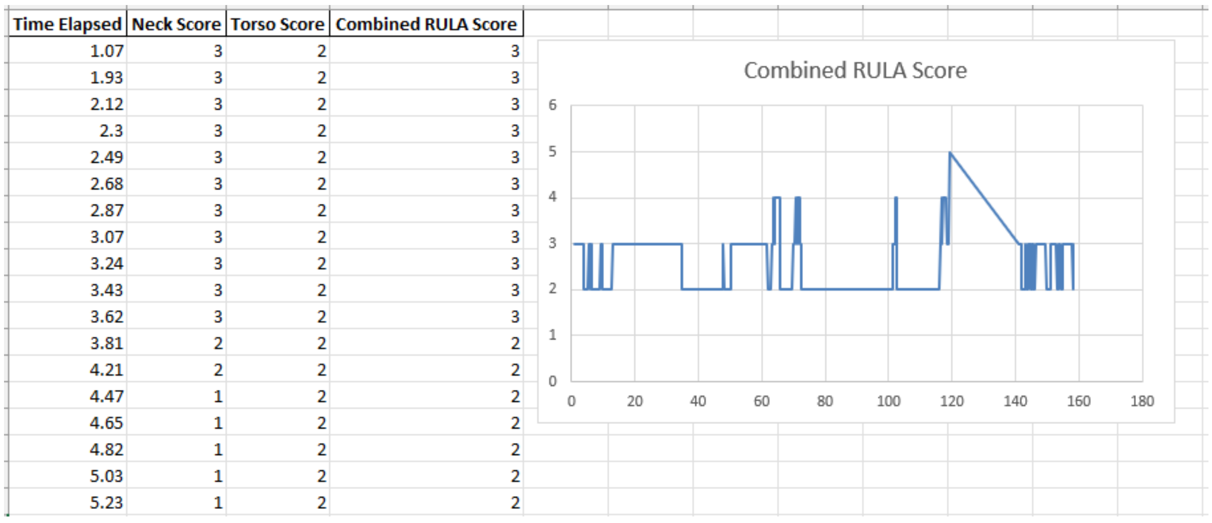


Neck Posture Score	Table B: Trunk Posture Score											
	1		2		3		4		5		6	
	Legs		Legs		Legs		Legs		Legs		Legs	
	1	2	1	2	1	2	1	2	1	2	1	2
1	1	3	2	3	3	4	5	5	6	6	7	7
2	2	3	2	3	4	5	5	5	6	7	7	7
3	3	3	3	4	4	5	5	6	6	7	7	7
4	5	5	5	6	6	7	7	7	7	7	8	8
5	7	7	7	7	7	8	8	8	8	8	8	8
6	8	8	8	8	8	8	8	9	9	9	9	9

Figure 22. Table B of RULA worksheet (Source: (Middlesworth, 2023)).

The Table of RULA spreadsheet, as depicted in **Figure 22**, presents a notable advantage over Table A in terms of ease of application, owing to the accumulation of neck and torso posture scores. According to Step 11 depicted in **Figure 11**, the assigned score for legs can vary between 1 and 2, contingent upon whether they were supported or not. Given that the programme is designed for tasks involving manual assembly or office work, which typically entail a seated posture, the programme consistently assigns a value of 1.

In order to execute this implementation, it is necessary to populate the array with data that is dependent on Table B for the resultant output. Therefore, the preferred method for this task was the utilisation of case switches. Subsequently, the scores are recorded in arrays and subsequently transferred to Excel spreadsheets. The Excel spreadsheet for the combined neck and torso Rapid Upper Limb Assessment (RULA) score at any given point is depicted in **Figure 23**.



Time Elapsed	Neck Score	Torso Score	Combined RULA Score
1.07	3	2	3
1.93	3	2	3
2.12	3	2	3
2.3	3	2	3
2.49	3	2	3
2.68	3	2	3
2.87	3	2	3
3.07	3	2	3
3.24	3	2	3
3.43	3	2	3
3.62	3	2	3
3.81	2	2	2
4.21	2	2	2
4.47	1	2	2
4.65	1	2	2
4.82	1	2	2
5.03	1	2	2
5.23	1	2	2

Figure 23. Graph to output arm and wrist combined score.

6.3.3. Implementation of Table C of RULA Worksheet

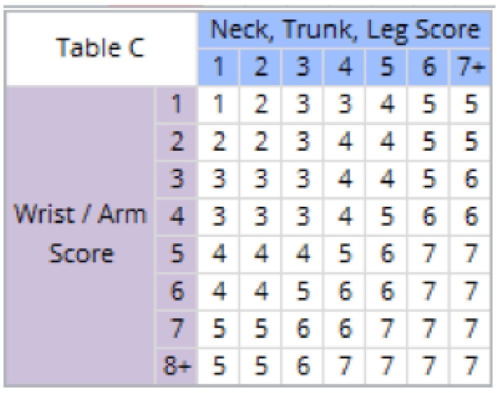
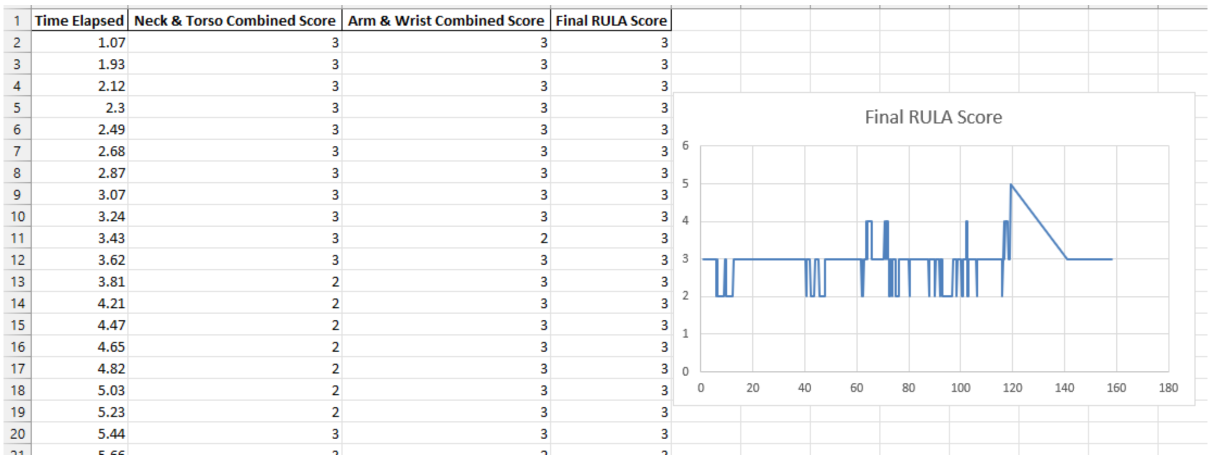


Table C		Neck, Trunk, Leg Score						
		1	2	3	4	5	6	7+
Wrist / Arm Score	1	1	2	3	3	4	5	5
	2	2	2	3	4	4	5	5
	3	3	3	3	4	4	5	6
	4	3	3	3	4	5	6	6
	5	4	4	4	5	6	7	7
	6	4	4	5	6	6	7	7
	7	5	5	6	6	7	7	7
	8+	5	5	6	7	7	7	7

Figure 24. Table C of RULA worksheet (Source: (Middlesworth, 2023)).

Upon collating the scores obtained from Tables A and B, a comparative analysis is conducted to derive the ultimate Rapid Upper Limb Assessment (RULA) score. Table C, as depicted in **Figure 24**, displays the outcomes of the score comparison. Table C presents the ultimate RULA score, where a lower score indicates a superior posture. Case switch statements were employed to address this issue and execute the implementation.

The resultant RULA score array is exported as an Excel spreadsheet to facilitate graphical representation and long-term retention for potential postural improvement assessment. **Figure 25** depicts the visual representation of the process of evaluating postural performance.

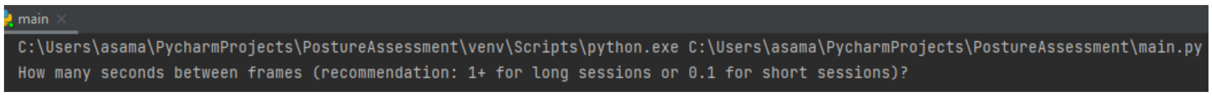


1	Time Elapsed	Neck & Torso Combined Score	Arm & Wrist Combined Score	Final RULA Score
2	1.07	3	3	3
3	1.93	3	3	3
4	2.12	3	3	3
5	2.3	3	3	3
6	2.49	3	3	3
7	2.68	3	3	3
8	2.87	3	3	3
9	3.07	3	3	3
10	3.24	3	3	3
11	3.43	3	2	3
12	3.62	3	3	3
13	3.81	2	3	3
14	4.21	2	3	3
15	4.47	2	3	3
16	4.65	2	3	3
17	4.82	2	3	3
18	5.03	2	3	3
19	5.23	2	3	3
20	5.44	3	3	3
21	5.66	[illegible]	[illegible]	[illegible]

Figure 25. Graph to output final RULA score.

6.4. Implementation of Optimisation for Low-End Systems

The primary objective of the project is to offer a cost-efficient and efficacious approach to posture assessment. The present programme is executable on any system that can run Python code and possesses a functional webcam. To ensure the efficacy of a cost-effective system, which is typically associated with lower-end technology, the user is prompted to input the desired number of seconds between frames upon programme execution, as depicted in **Figure 24**.



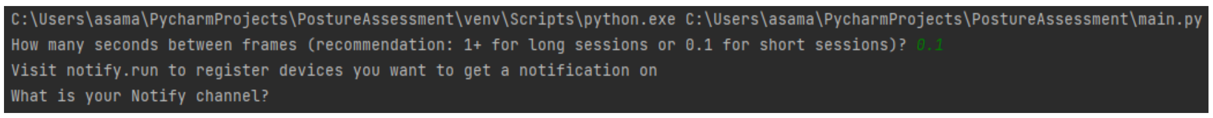
```
C:\Users\asama\PycharmProjects\PostureAssessment\venv\Scripts\python.exe C:\Users\asama\PycharmProjects\PostureAssessment\main.py
How many seconds between frames (recommendation: 1+ for long sessions or 0.1 for short sessions)?
```

Figure 26. Prompt asking for the desired seconds between frames.

A higher value of this parameter results in a decrease in the frames per second, indicating a preference for performance optimisation over precision. The present programme is designed to operate efficiently on low-end systems, without necessitating high performance. The greater the numerical value assigned to a variable, the lesser the amount of data that will be present in the resulting Excel spreadsheet. The prompt suggests that longer sessions should have a higher number of seconds between frames. The rationale behind increasing the interval for posture monitoring from every tenth of a second to approximately 10 seconds in a 4-hour session is to reduce the amount of superfluous data in the output file. This adjustment would result in a 100-fold reduction in data, which would otherwise be of limited value. If more consistent values are preferred and resources was not an issue, then having a lower value would be better.

6.5. Implementation of the Notification Feature

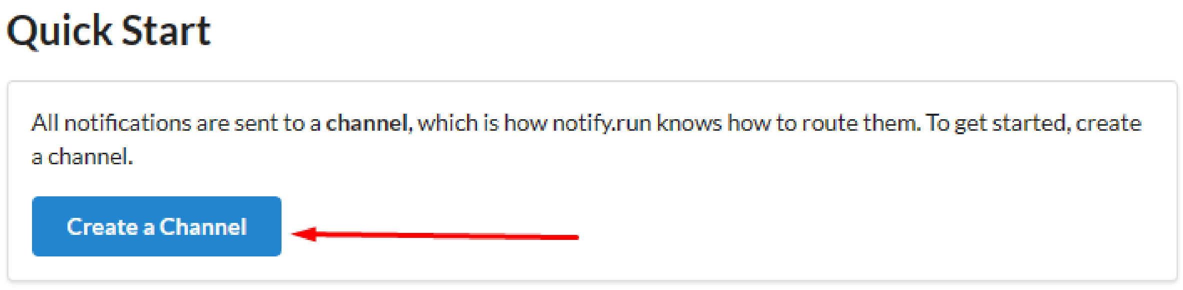
A library called ‘notify.run’ is imported to enable the notification feature for this application. A random string identifies notify.run users' channels. Chrome, Firefox, and Android devices that support the Web Push API can subscribe to channel notifications and display pop-up notifications. iOS Safari and other devices don't support background notifications, but they can display recent notifications. All channels have HTTPS endpoints. Notify when posting to that endpoint. Run broadcasts to all channel subscribers. notify.run sends and receives notifications using standard APIs, so no software is needed (notify.run, 2023).



```
C:\Users\asama\PycharmProjects\PostureAssessment\venv\Scripts\python.exe C:\Users\asama\PycharmProjects\PostureAssessment\main.py
How many seconds between frames (recommendation: 1+ for long sessions or 0.1 for short sessions)? 0.1
Visit notify.run to register devices you want to get a notification on
What is your Notify channel?
```

Figure 27. Prompt to enter Notify channel.

After the program asks the user for the number of seconds between frames, it prompt you to enter a notify channel link, seen in **Figure 26**. On the notify.run website you can go there to create a channel in the quick start section of the homepage as seen in **Figure 28**.

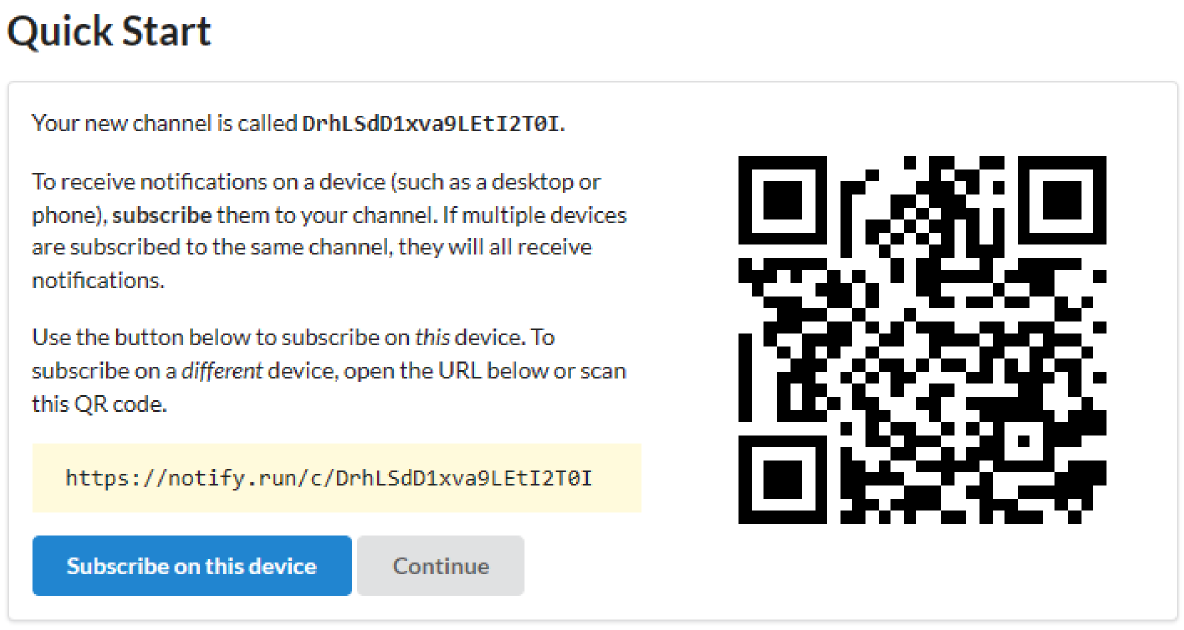


Quick Start

All notifications are sent to a **channel**, which is how notify.run knows how to route them. To get started, create a channel.

Figure 28. How to create a Notify channel.

Channel string will be provided and that can be used to navigate to the webpage that enables you to subscribe to the notifications. The user can create the channel on the device then use the barcode provided, as seen in **Figure 29**, to subscribe on another device. This way the user can easily copy and paste the channel string into the program while being able to receive notification on any device like their phone for example.



Quick Start

Your new channel is called `DrhLSdD1xva9LEtI2T0I`.

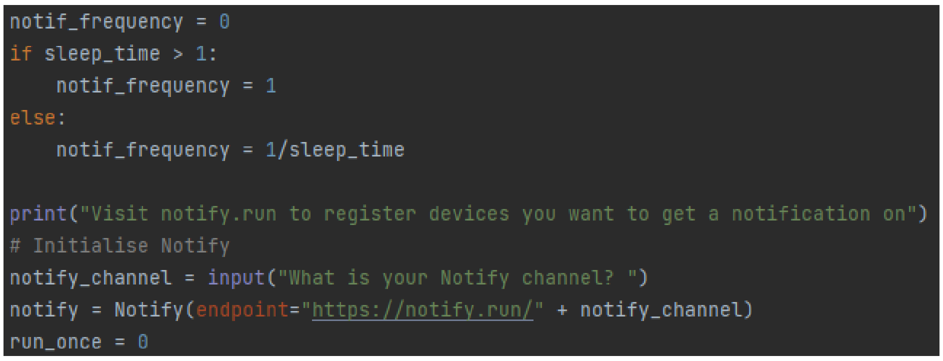
To receive notifications on a device (such as a desktop or phone), **subscribe** them to your channel. If multiple devices are subscribed to the same channel, they will all receive notifications.

Use the button below to subscribe on *this* device. To subscribe on a *different* device, open the URL below or scan this QR code.

```
https://notify.run/c/DrhLSdD1xva9LEtI2T0I
```

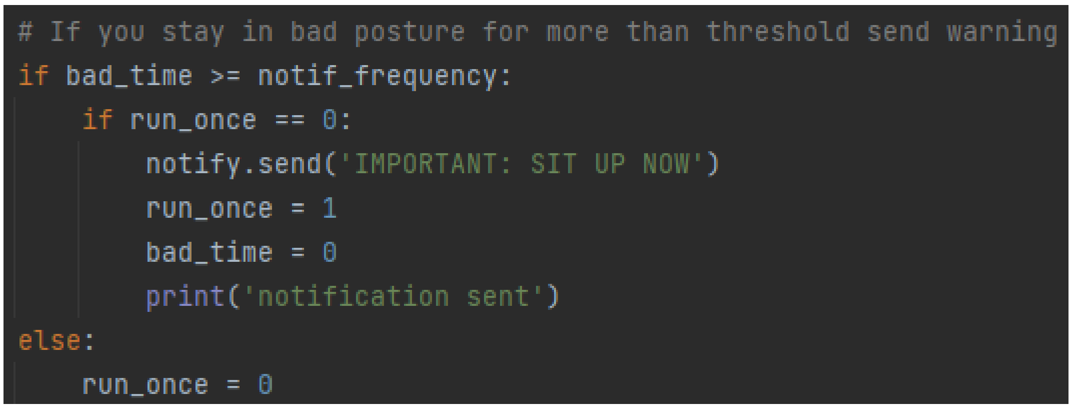
Figure 29. Notify subscription feature.

Upon pasting the string, it is subsequently designated as the endpoint, as illustrated in **Figure 30**'s code. The Notify object is created and the endpoint is assigned during instantiation. The initial lines depicted in **Figure 30** are predominantly intended for establishing the frequency of notifications. The system guarantees the reception of a notification on a per-second basis in cases where the time interval between frames is less than one second. Conversely, if the time interval between frames exceeds one second, the notification frequency is adjusted to every frame. The notification sending system implementation is depicted in **Figure 31**. If the number of frames exhibiting poor posture exceeds or equals the notification frequency threshold, a notification will be dispatched. The modification of the notification message can be readily achieved by altering the string. This approach is employed to prevent overwhelming the user with an excessive number of notifications.



```python
notif_frequency = 0
if sleep_time > 1:
    notif_frequency = 1
else:
    notif_frequency = 1/sleep_time

print("Visit notify.run to register devices you want to get a notification on")
# Initialise Notify
notify_channel = input("What is your Notify channel? ")
notify = Notify(endpoint="https://notify.run/" + notify_channel)
run_once = 0
```

Figure 30. Code for Notify.



```python
# If you stay in bad posture for more than threshold send warning
if bad_time >= notif_frequency:
    if run_once == 0:
        notify.send('IMPORTANT: SIT UP NOW')
        run_once = 1
        bad_time = 0
        print('notification sent')
else:
    run_once = 0
```

Figure 31. Code responsible for sending the notification.

6.6. Implementation of Real-Time Feedback

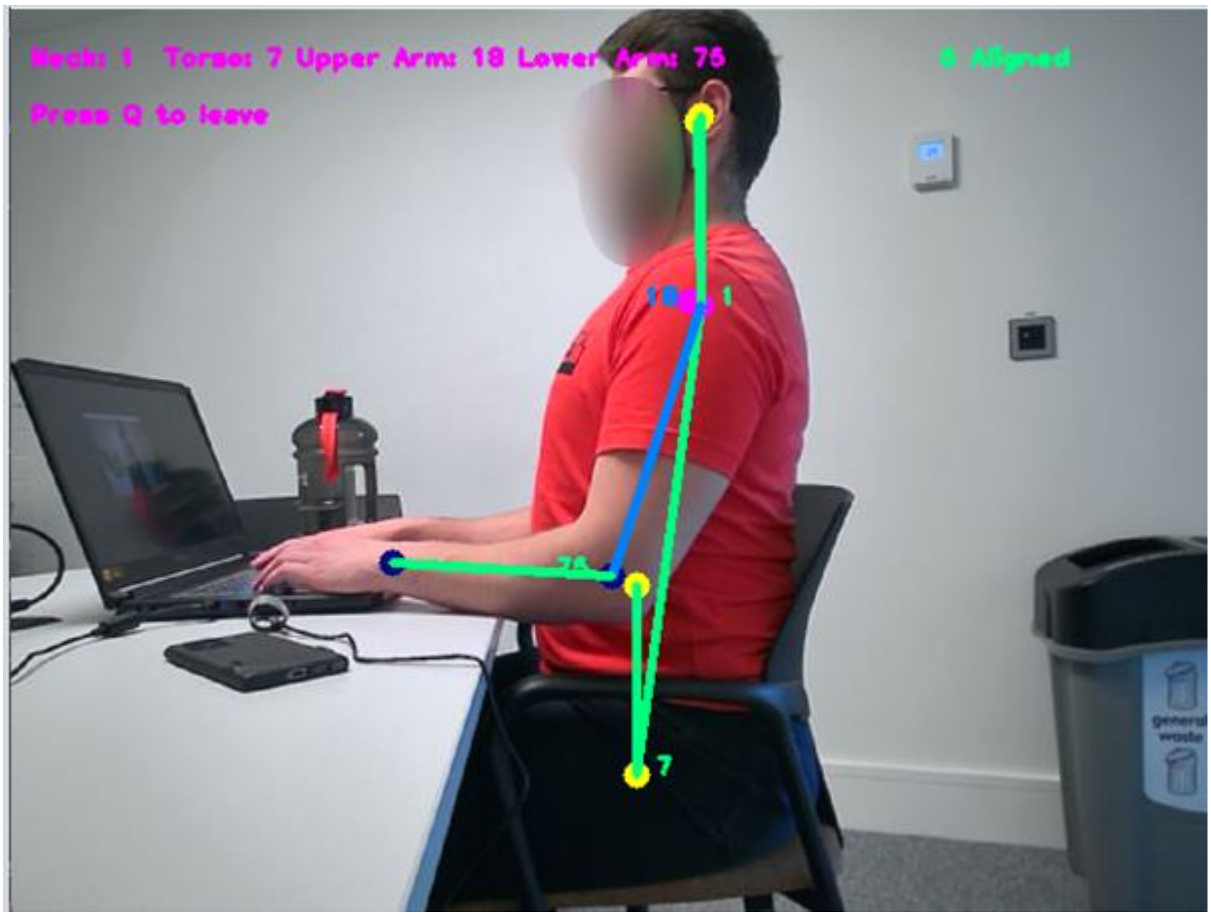
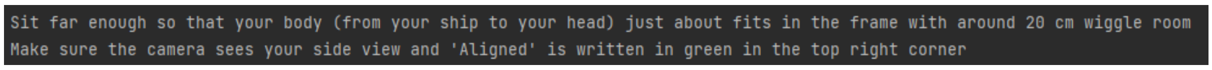


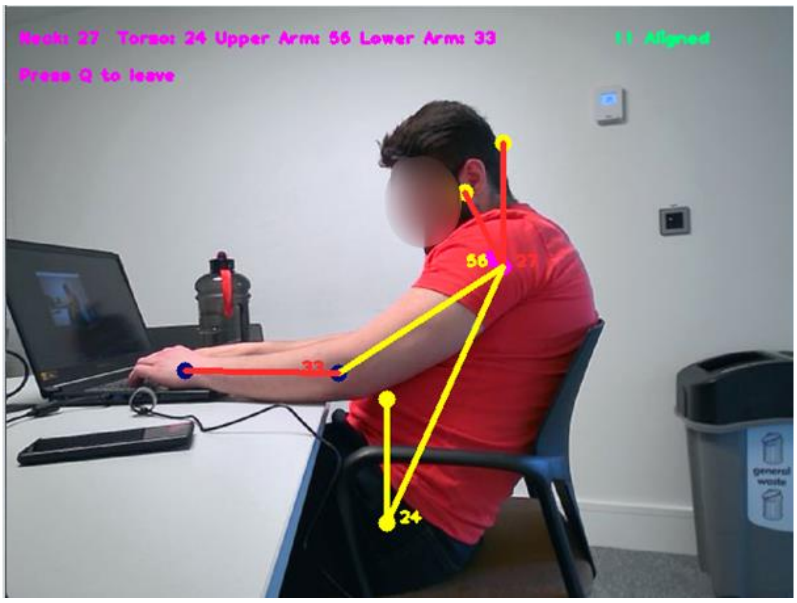
Figure 32. Best case scenario in terms of colour coding.

Figure 32 illustrates how the system employs colours to inform the user of their posture performance. The illustration presented in Figure 30 depicts a scenario that is considered unfavourable. The system employs a colour scheme based on traffic lights, wherein the colour green signifies optimal performance, yellow indicates room for improvement, and red denotes suboptimal performance. The upper arm may exhibit a blue hue, which is considered optimal. This assertion is supported by **Figure 11**, which illustrates that the upper arm can manifest four possible outcomes. The upper left section of the display presents the angles of the pertinent joints, while also providing guidance on the appropriate manner in which to conclude the session. Additionally, the upper right section of the interface indicates whether the user has achieved proper alignment. The programme provides guidance on the ideal position upon initiation, following the provision of the Notify channel, as depicted in **Figure 33**.



```
Sit far enough so that your body (from your ship to your head) just about fits in the frame with around 20 cm wiggle room
Make sure the camera sees your side view and 'Aligned' is written in green in the top right corner
```

Figure 33. Alignment Instructions.



```
Neck: 27  Torso: 24 Upper Arm: 56 Lower Arm: 33          11 Aligned
Press Q to leave
```

Figure 34. Bad posture colour evaluation.

6.7. Implementation of the User Interface

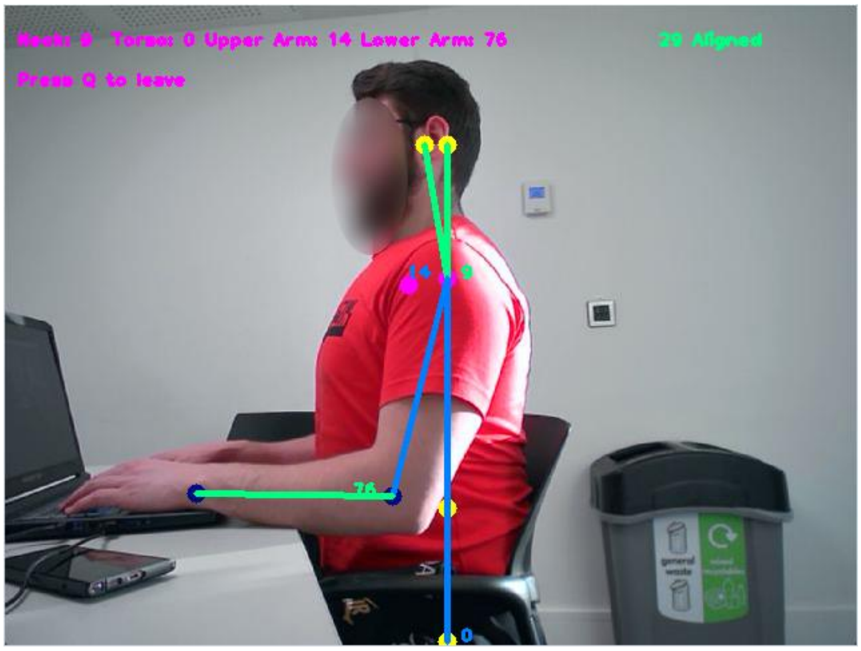


Figure 35. Demonstrates the UI.

The implementation of the user interface should provide real-time feedback to the user. The optimal approach was to demonstrate the feedback visually, using lines connecting the joints that change colour according to the joint posture performance, as depicted in **Figure 35**. The colour blue is considered best but it is limited to the torso and upper arm due to the difficulty in reaching perfect posture on those joints. Green is good, while yellow indicates a decline in posture quality and red represents the lowest quality.

The alignment indicator, located at the top right of the application window, displays additional useful information. The significance of this feature lies in its ability to guide the user towards an optimal camera angle, which in turn enhances the accuracy of the program's output.

The upper left section of the window displays the angles of each joint in a centralised location. The upper left corner contains instructions on how to properly exit the program to ensure correct data saving.

Chapter 7 Results and Discussion

Figure 36. Cheap 1080p 30fps USB webcam

This section will conduct a case study. The webcam depicted in **Figure 36** and a mid-range laptop that is 5 years old will be utilised to run the program. The experimental setup is depicted in **Figure 37**. The subject will be seated on the chair located on the right side of the image, facing the laptop. The webcam, shown in **Figure 38**, is mounted on the chair about one metre away. The setup process is quick and easy.

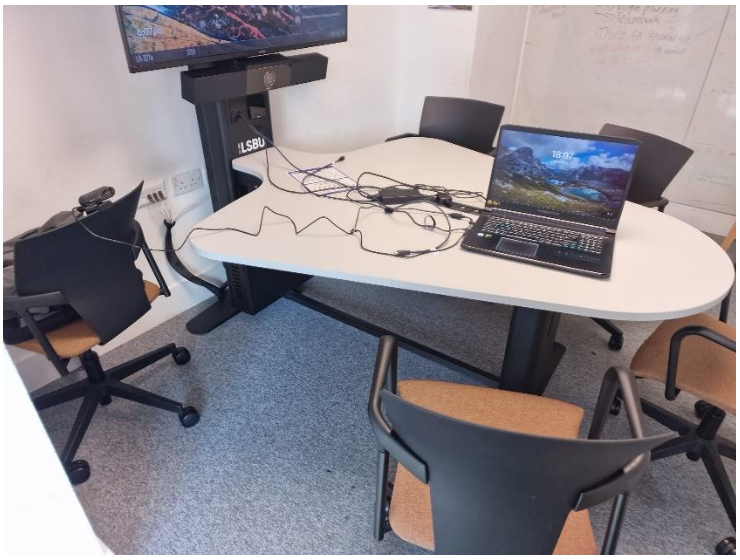


Figure 37. Setting of the experiment.

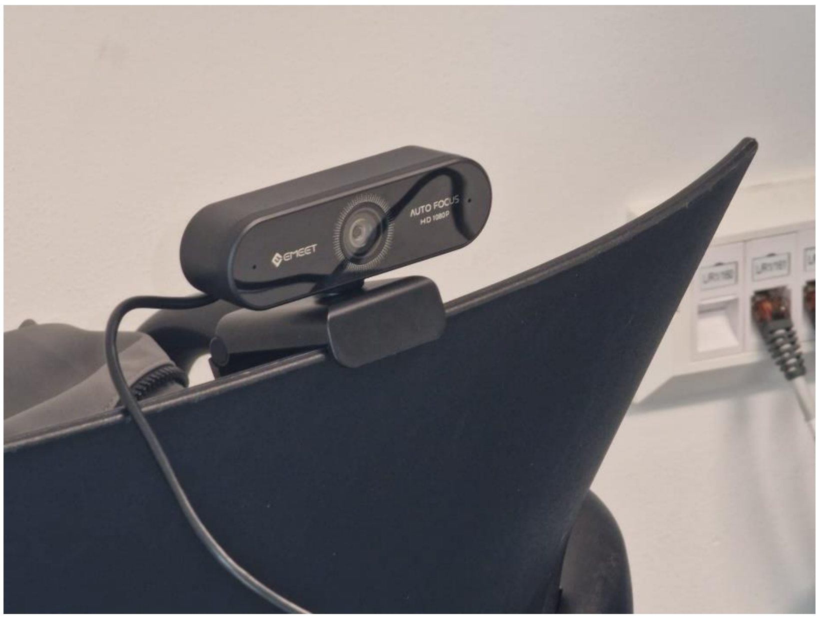


Figure 38. Webcam mounted.

7.1. Case Study – Office Work

In contemporary office work settings, maintaining correct posture is essential for the holistic maintenance of physical and mental health. Adverse body positioning may result in a range of health complications, such as lumbar discomfort, cervical discomfort, and headaches. The prevalence of sedentary occupations has led to an increasing demand for tools and resources that can assist individuals in enhancing their posture and mitigating associated concerns.

The Posture Assessment Platform is a software application that has been developed to assess an individual's posture and offer constructive feedback and suggestions for enhancement whilst notifying the user when their posture is not preferable. The present case study aims to test the Posture Assessment Platform in an office setting.

The research entailed a singular participant who underwent a 20-minute assessment of office-related tasks, 10 minutes with notifications on and 10 minutes with it off. The reason it was ran with notification off is to see how the program effects the posture performance. Throughout this period, the participant utilised the Posture Assessment Platform to evaluate their posture and obtain guidance on maintaining good posture in the form of notifications. The aim of this research is to assess the effectiveness of the Posture Assessment Platform in enhancing the postural alignment of individuals in an office environment.

7.1.1. Case Study Results

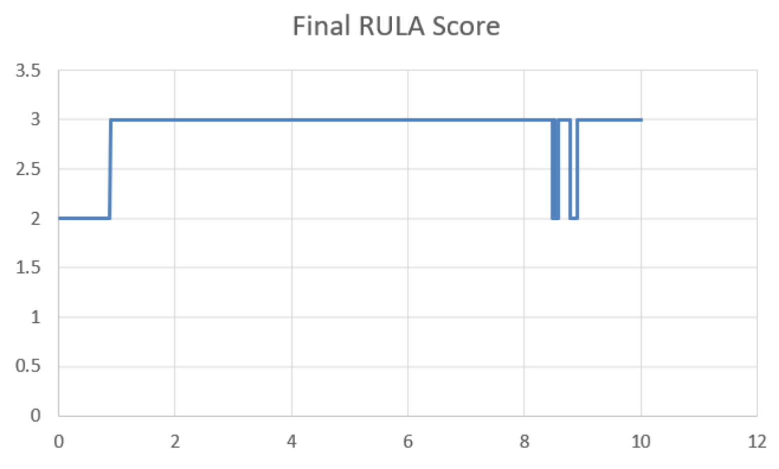


Figure 39. Office work with notifications off (Y-Axis is RULA Score (lower is better) and X-Axis is minutes elapsed).

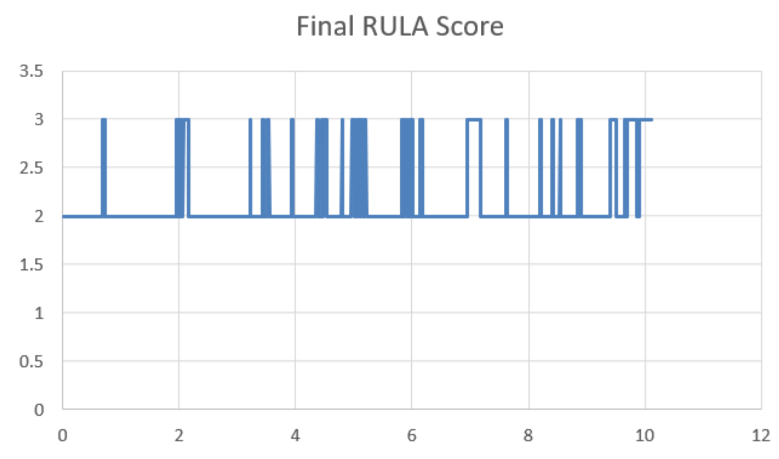


Figure 40. Office work with notifications on (Y-Axis is RULA Score (lower is better) and X-Axis is minutes elapsed).

According to **Figure 39**, the subject initially exhibited good posture for approximately one minute, but subsequently demonstrated a decline in posture quality that persisted. At the time of the incident, notifications were disabled, thereby rendering the subject unaware of their poor posture. Analogously to the findings presented in **Figure 39**, the results depicted in **Figure 38** indicate that the participant initially exhibited proper posture for approximately one minute, but subsequently demonstrated an improvement in posture after a period of poor posture. The evidence suggests that the individual has exhibited poor posture on multiple occasions while receiving notifications yet has made efforts to rectify their posture with each occurrence.

The average score when notifications are off is 2.98 while the average score when notifications are on is 2.15. This shows a major improvement and good influence towards the user to improve their posture.

7.1.1.1. Screenshots From the Case Study

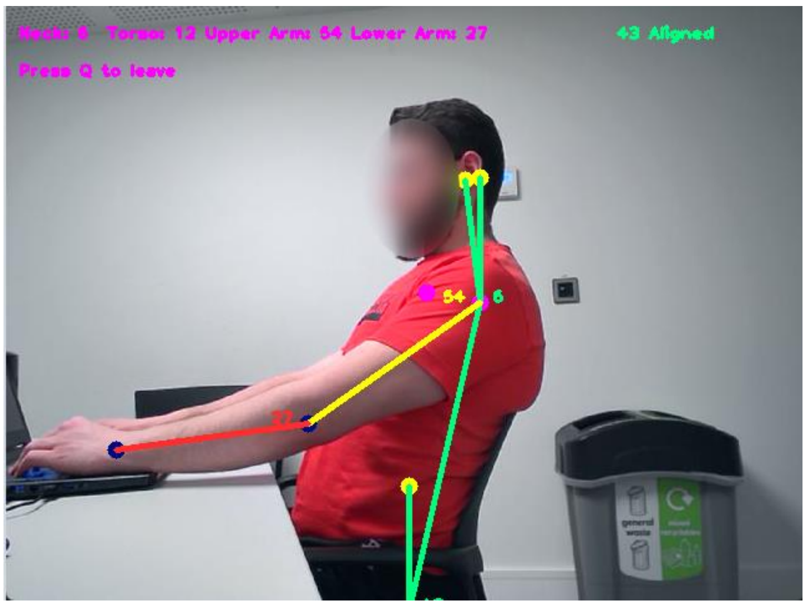


Figure 41. Screenshot of a frame from the scenario with notifications (Score: 3).

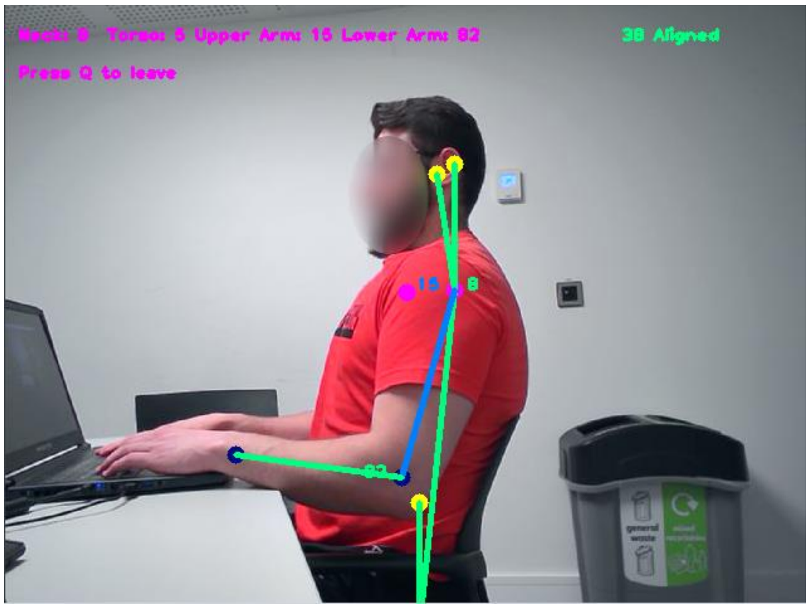


Figure 42. Screenshot of a frame from the scenario with notifications (Score: 2).

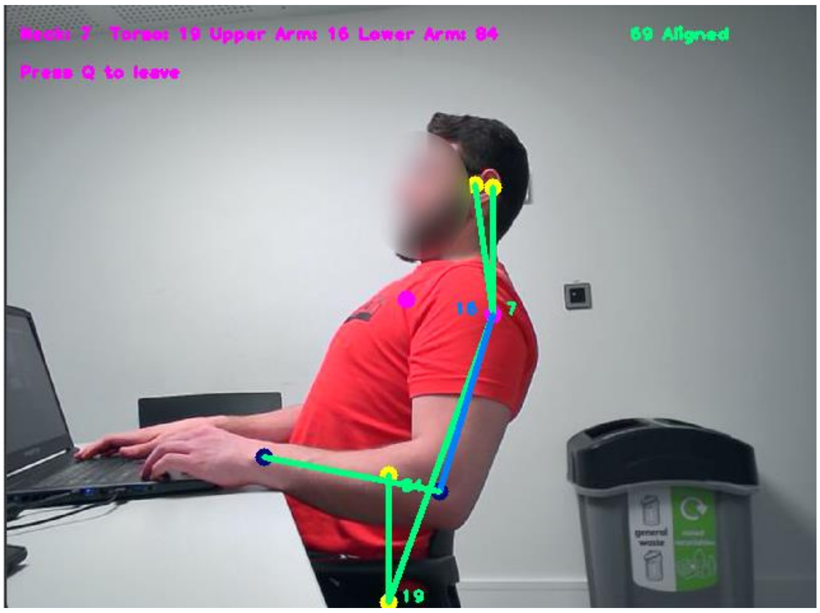


Figure 43. Screenshot of a frame from the scenario without notifications (Score: 2).

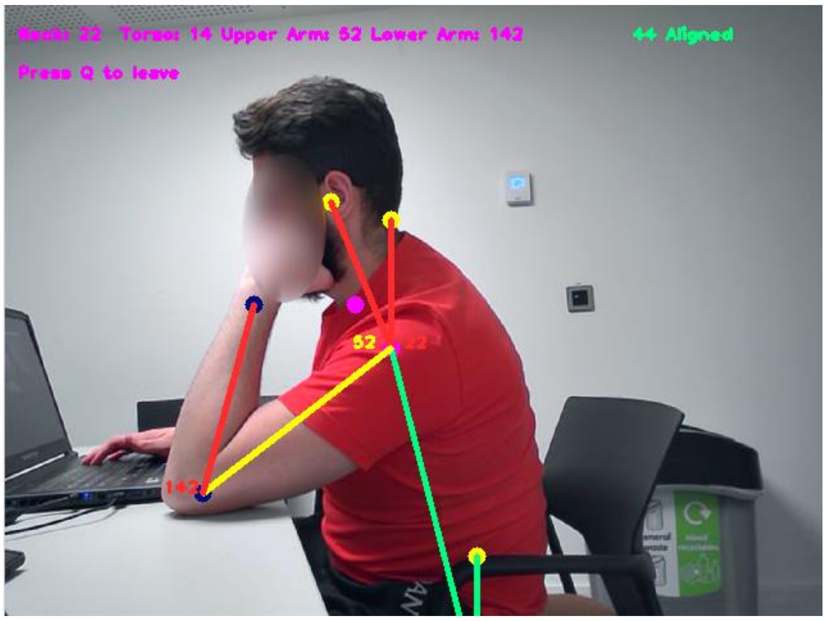


Figure 44. Screenshot of a frame from the scenario without notifications (Score: 3).

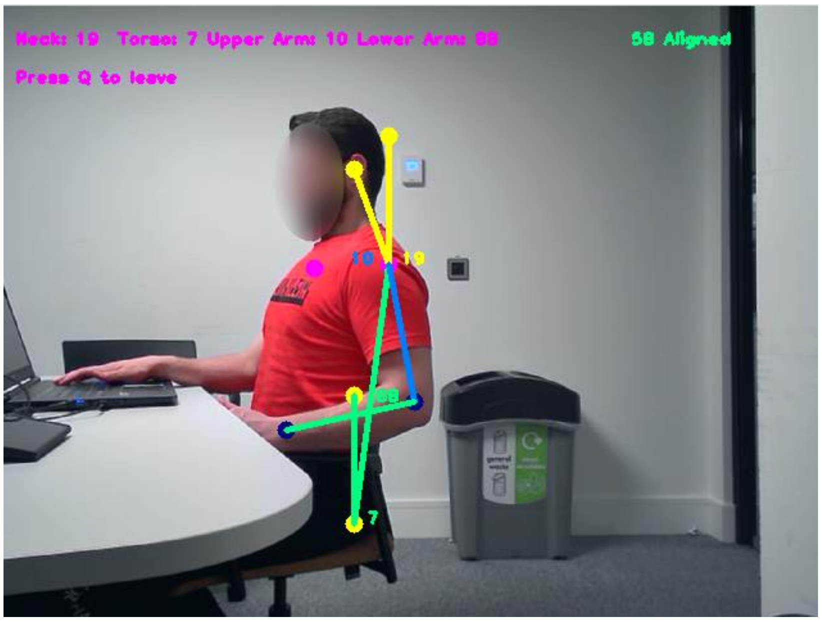


Figure 45. Screenshot of a frame from the scenario with notifications (Score: 3).

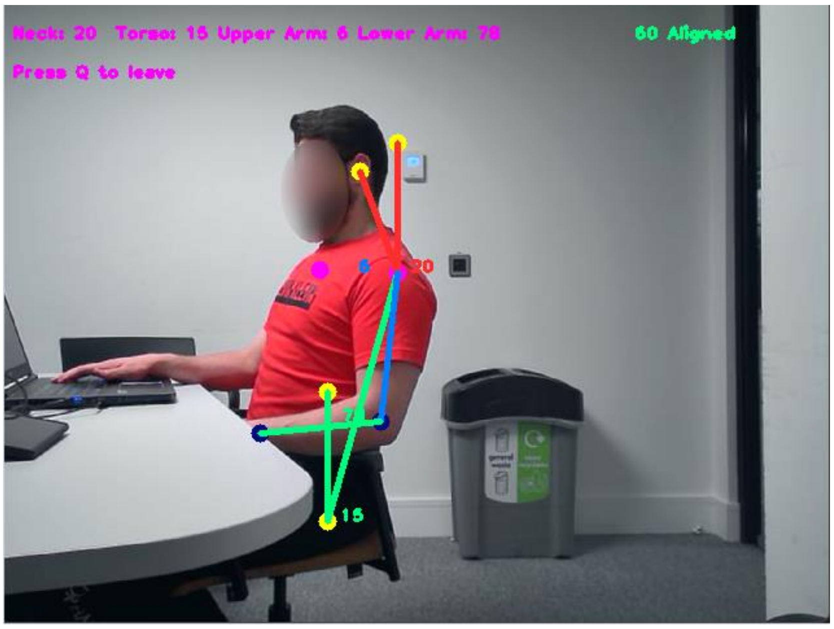


Figure 46. Screenshot of a frame from the scenario with notifications (Score: 3).

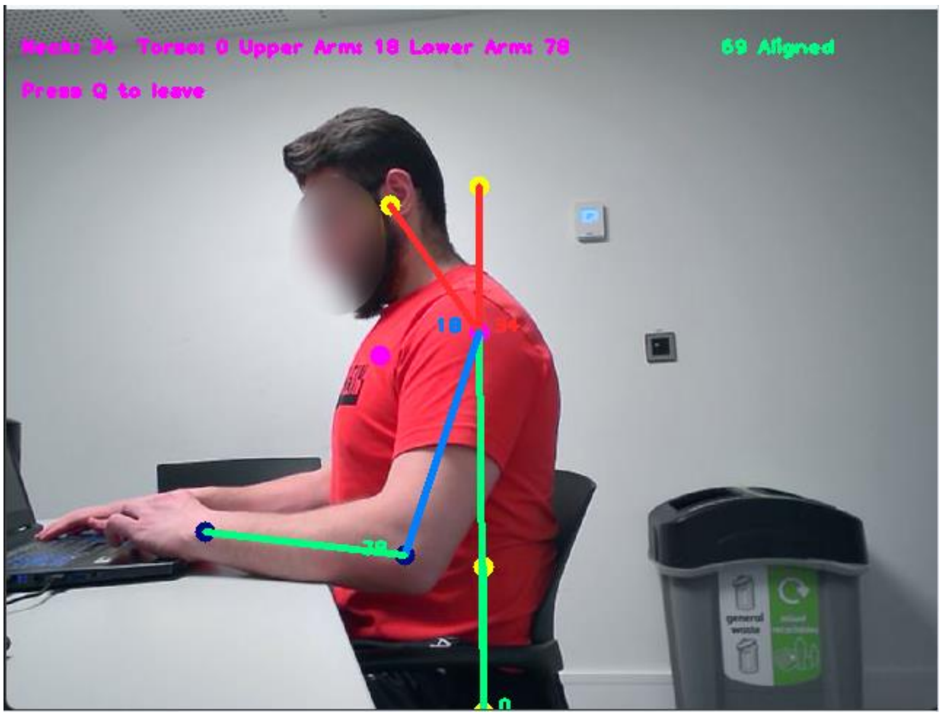
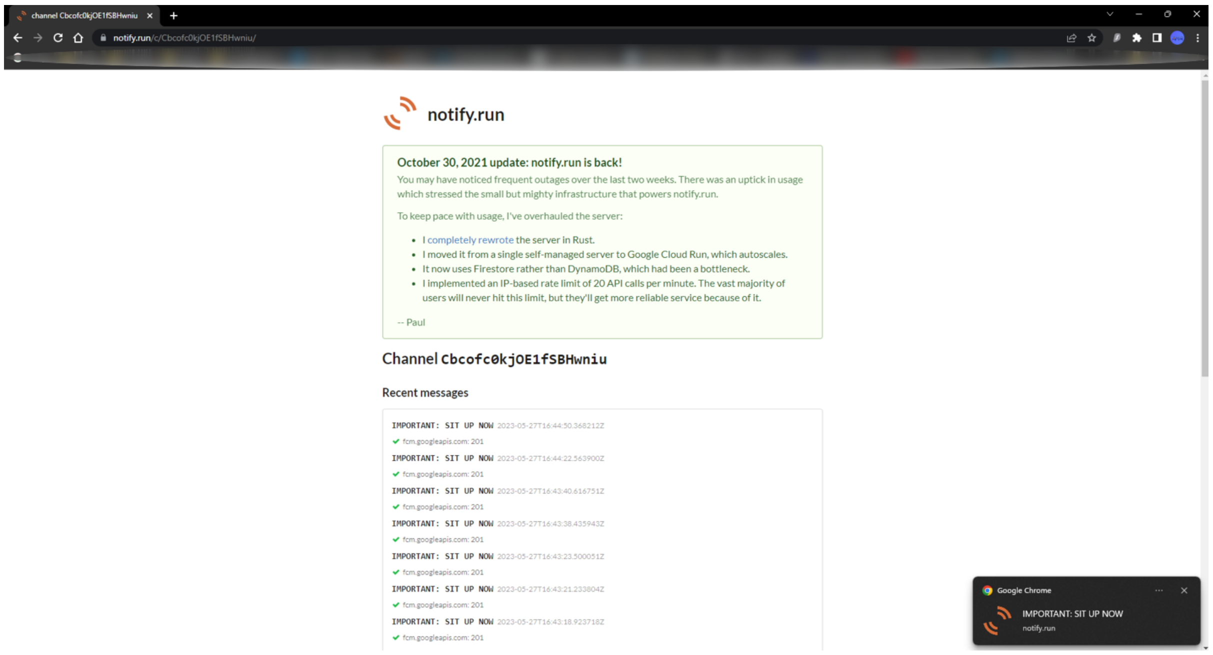


Figure 47. Screenshot of a frame from the scenario without notifications (Score: 3).

7.2. System Performance Validation

The findings of the case study indicate that the ergonomic posture assessment platform, which utilises real-time video capture technology, has been effectively created. This platform can monitor the user's posture and providing immediate feedback, without requiring the use of supplementary equipment such as sensors. The provision of instantaneous feedback is manifested in two distinct modalities, namely notifications and posture performance feedback that is chromatically differentiated. If the device selected by the user is a desktop PC or a laptop, the notification that is transmitted to the device remains visible and persistent. As depicted in **Figure 48**, the notification is characterised by a high degree of salience and persistence. Consequently, unless the user desires to continuously maintain the notification on their display, they would need to rectify their posture.


channel Cbcofc0kjOE1fSBHwniu
notify.run/c/Cbcofc0kjOE1fSBHwniu/
Google Chrome
IMPORTANT: SIT UP NOW
notify.run

notify.run

October 30, 2021 update: notify.run is back!
You may have noticed frequent outages over the last two weeks. There was an uptick in usage which stressed the small but mighty infrastructure that powers notify.run.

To keep pace with usage, I've overhauled the server:

- I completely rewrote the server in Rust.
- I moved it from a single self-managed server to Google Cloud Run, which autoscales.
- It now uses Firestore rather than DynamoDB, which had been a bottleneck.
- I implemented an IP-based rate limit of 20 API calls per minute. The vast majority of users will never hit this limit, but they'll get more reliable service because of it.

-- Paul

Channel `Cbcofc0kjOE1fSBHwniu`

Recent messages

```
IMPORTANT: SIT UP NOW 2023-05-27T16:44:50.368212Z
✓ fcm.googleapis.com: 201
IMPORTANT: SIT UP NOW 2023-05-27T16:44:22.563900Z
✓ fcm.googleapis.com: 201
IMPORTANT: SIT UP NOW 2023-05-27T16:43:40.616751Z
✓ fcm.googleapis.com: 201
IMPORTANT: SIT UP NOW 2023-05-27T16:43:38.435943Z
✓ fcm.googleapis.com: 201
IMPORTANT: SIT UP NOW 2023-05-27T16:43:23.500051Z
✓ fcm.googleapis.com: 201
IMPORTANT: SIT UP NOW 2023-05-27T16:43:21.233804Z
✓ fcm.googleapis.com: 201
IMPORTANT: SIT UP NOW 2023-05-27T16:43:18.923718Z
✓ fcm.googleapis.com: 201
```

Figure 48. Notification instructing user to sit up.

The implementation of the dashboard functionality was deemed too complex and encountered issues during the implementation phase, potentially due to conflicting libraries. An additional aim to enable users to retrospectively monitor their behaviour has been successfully executed. This is evidenced by the storage of posture performance data in Excel format, which can be conveniently duplicated and renamed to reflect the date and time of the corresponding session. As evidenced by prior illustrations, the graph implementation is extant and fulfils the function of facilitating user pattern recognition for the purpose of habit identification.

7.2.1. Validate the Ability to Detect Good/Poor Posture

It is important to note that in the initial 30 seconds of the first test, as depicted in **Figure 39**, the participant received a prompt to maintain good posture, whereas during the last 30 seconds of the same test, the participant was instructed to deliberately sit in a poor posture. This procedure was conducted to verify the functionality of the system, and the outcomes indicate that it is operating appropriately.

7.2.1.1. Validate Worst Possible Score

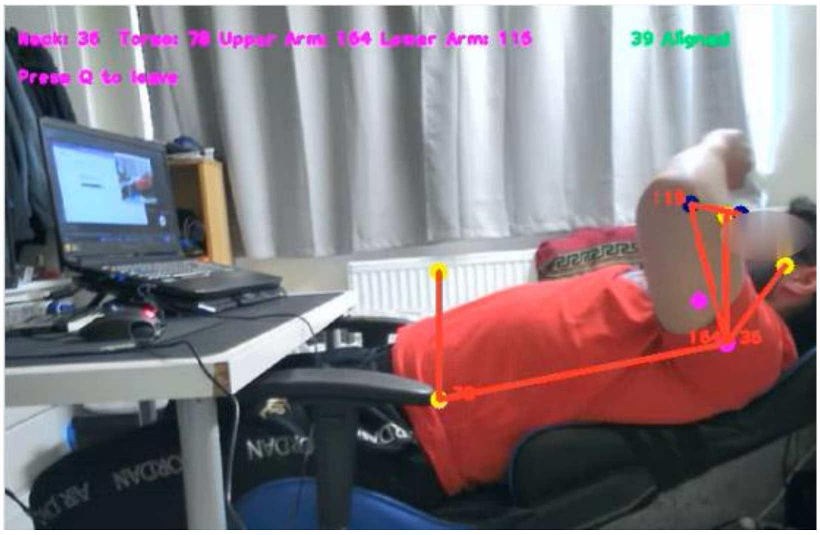
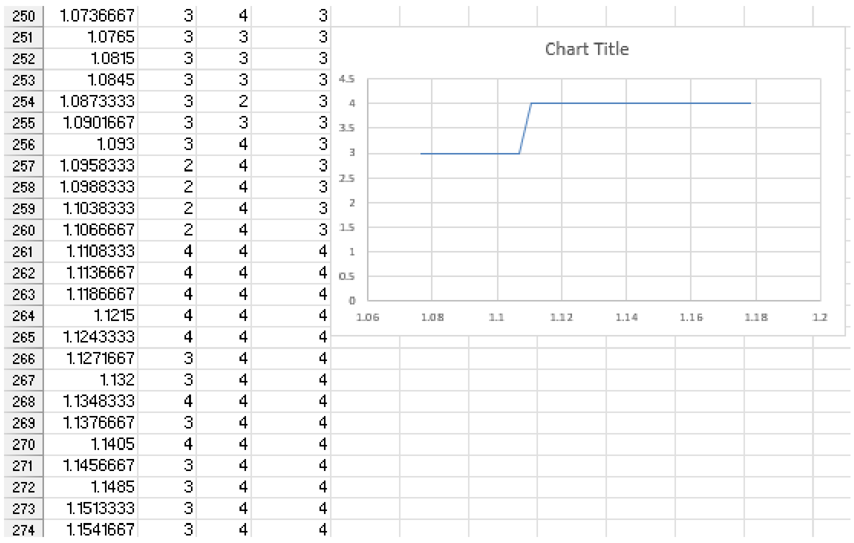


Figure 49. Worse score possible (Score: 4).

Figure 49 shows the worst posture possible which the platform would evaluate as a score of 4. It is apparent that this scenario is very unlikely in an office work scenario because the hands cannot reach the keyboard, which is why this score is not seen in the case study.



250	1.0736667	3	4	3
251	1.0765	3	3	3
252	1.0815	3	3	3
253	1.0845	3	3	3
254	1.0873333	3	2	3
255	1.0901667	3	3	3
256	1.093	3	4	3
257	1.0958333	2	4	3
258	1.0988333	2	4	3
259	1.1038333	2	4	3
260	1.1066667	2	4	3
261	1.1108333	4	4	4
262	1.1136667	4	4	4
263	1.1186667	4	4	4
264	1.1215	4	4	4
265	1.1243333	4	4	4
266	1.1271667	3	4	4
267	1.132	3	4	4
268	1.1348333	4	4	4
269	1.1376667	3	4	4
270	1.1405	4	4	4
271	1.1456667	3	4	4
272	1.1485	3	4	4
273	1.1513333	3	4	4
274	1.1541667	3	4	4

Figure 50. Worse score possible graph.

7.2.1.2. Validate Best Possible Score

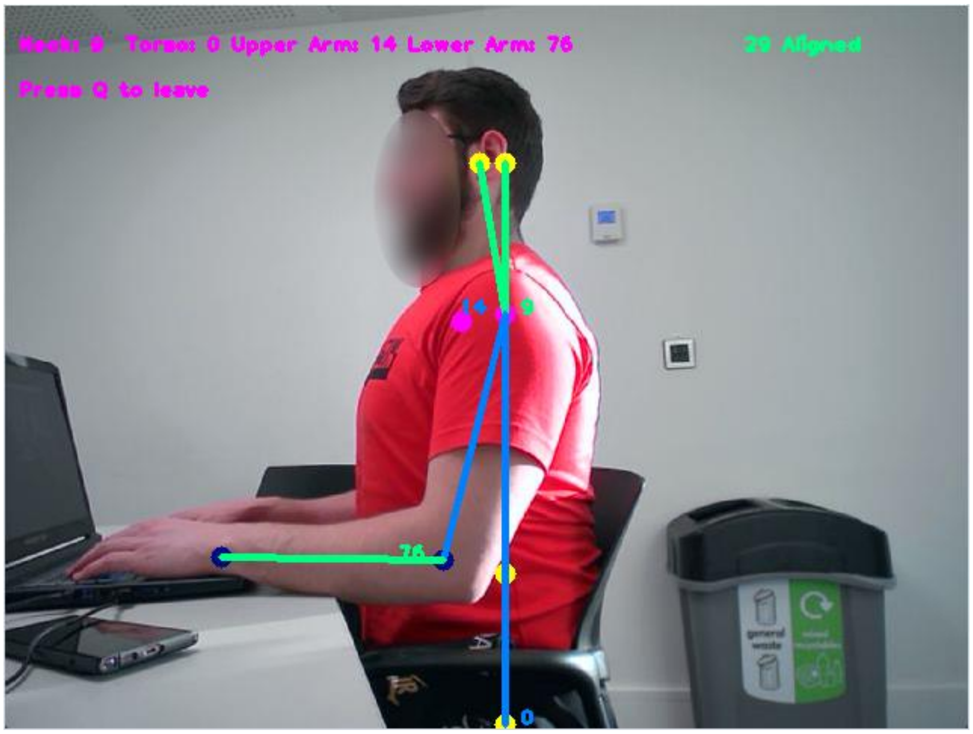
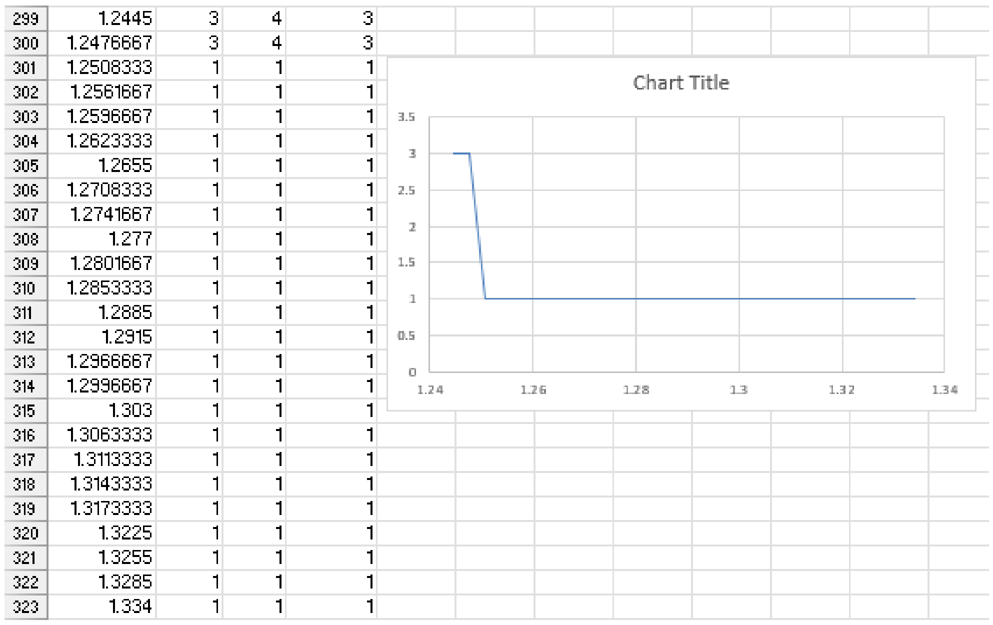


Figure 51. Best score possible (Score: 1)

Figure 51 shows the best possible posture which the platform would evaluate at a score of 1. The torso must be at exactly 0 degrees for this to happen which is very unlikely because it needs to be very precise, which is why the user will not get notified if they don't do this. It is also why it wasn't seen in the case study.



299	1.2445	3	4	3
300	1.2476667	3	4	3
301	1.2508333	1	1	1
302	1.2561667	1	1	1
303	1.2596667	1	1	1
304	1.2623333	1	1	1
305	1.2655	1	1	1
306	1.2708333	1	1	1
307	1.2741667	1	1	1
308	1.277	1	1	1
309	1.2801667	1	1	1
310	1.2853333	1	1	1
311	1.2885	1	1	1
312	1.2915	1	1	1
313	1.2966667	1	1	1
314	1.2996667	1	1	1
315	1.303	1	1	1
316	1.3063333	1	1	1
317	1.3113333	1	1	1
318	1.3143333	1	1	1
319	1.3173333	1	1	1
320	1.3225	1	1	1
321	1.3255	1	1	1
322	1.3285	1	1	1
323	1.334	1	1	1

Figure 52. Best score possible graph.

7.3. Discussion

The findings of this investigation indicate that the utilisation of the Posture Assessment Platform yields positive outcomes in terms of improving the postural alignment of individuals working in an office setting. The system had the capability to observe the user's body position and furnish prompt responses via notifications and chromatically differentiated feedback on posture performance. The notification that was sent to the device exhibited a noticeable and enduring quality, marked by a significant level of prominence and durability. The user was motivated to correct their body position to eliminate the notification from their visual interface.

The research findings indicate that the subject's posture quality exhibited a persistent decline when notifications were disabled. Nevertheless, upon enabling notifications, the individual endeavoured to correct their posture upon each instance of poor posture. This implies that the notifications of the platform are efficacious in prompting users to uphold proper posture.

It is noteworthy that the present investigation solely encompassed a solitary participant and was executed within a limited timeframe. To comprehensively evaluate the efficacy of the Posture Assessment Platform in improving postural alignment within an office setting, additional investigation utilising a more extensive sample size and a lengthier time frame would be required.

The software's output was limited to two possible scores, either two or three, as observed during the testing process. A binary system is characterised by the presence of two elements, whereas a ternary system is characterised by the presence of three elements. In this context, the former is considered favourable, while the latter is considered unfavourable. The lack of a known scale may result in confusion for the user, potentially leading them to misinterpret a score of 3 as satisfactory when it may not be so. Attaining a score of one on the posture scale is deemed unattainable due to the requirement of absolute perfect posture. Similarly, scoring above three is considered improbable as it would necessitate the individual to exhibit the most unfavourable posture, which is unlikely to occur in practical settings. It is unfeasible to obtain a score higher than four in the Rapid Upper Limb Assessment (RULA) worksheet, as certain steps in the assessment are omitted due to constraints within the library.

In summary, the present case study offers initial indications that the Posture Assessment Platform is effective in overseeing and enhancing postural alignment in a workplace setting by means of video capture technology that provides instantaneous feedback through notifications and posture performance feedback. Additional investigation is required to comprehensively evaluate its effectiveness. This form of testing would primarily be relevant to sedentary manual assembly tasks, given the characteristics of office work.

Chapter 8 Conclusion and Future Work

In summary, the development and documentation of a low-cost and non-intrusive posture assessment platform has yielded an effective solution. In conclusion, the primary impetus for undertaking this project was to address the escalating incidence of work-related musculoskeletal disorders (WMSDs) among manual assembly workers and other individuals engaged in prolonged repetitive tasks. In conclusion, this report has successfully addressed the research questions and demonstrated the implementation of software to achieve the stated objectives. In conclusion, it can be stated that all the requisite functional and non-functional requirements have been satisfactorily fulfilled.

In conclusion, the product's implementation exceeded expectations by successfully automating the entire RULA worksheet. In conclusion, the intended testing of the programme on manual assembly workers was not feasible as initially planned due to unforeseen complications. In conclusion, it can be inferred that office work shares similarities with prolonged repetitive work. Further research is required to confirm the effectiveness of office work in this regard.

The present study has effectively showcased the viability of utilising economical and unobtrusive technological solutions for the evaluation of body posture, thereby mitigating the risk of work-related musculoskeletal disorders (WMSDs) among labourers who engage in extended repetitive tasks. In conclusion, the current findings exhibit a promising outlook for the effectiveness of this approach in manual assembly workers. However, further testing is required to validate its efficacy. Nonetheless, the results suggest that this approach has the potential to significantly enhance worker health and safety.

8.1. Limitations of the Work

This product was able to automate eight out of the 15 steps from the Rapid Upper Limb Assessment (RULA) worksheet, as depicted in **Figure 11**. Therefore, the ultimate score was confined to a range of 1 to 4, as opposed to the anticipated range of 1 to 7. This outcome could potentially compromise the precision of the evaluation and certainly renders the score incomparable to that of the RULA.

The operationality of the notification feature is contingent upon the presence of an active internet connection. If the platform user experiences intermittent internet connectivity that results in occasional disruptions, it would present a significant drawback. This is due to the inconsistency with the real-time feedback feature.

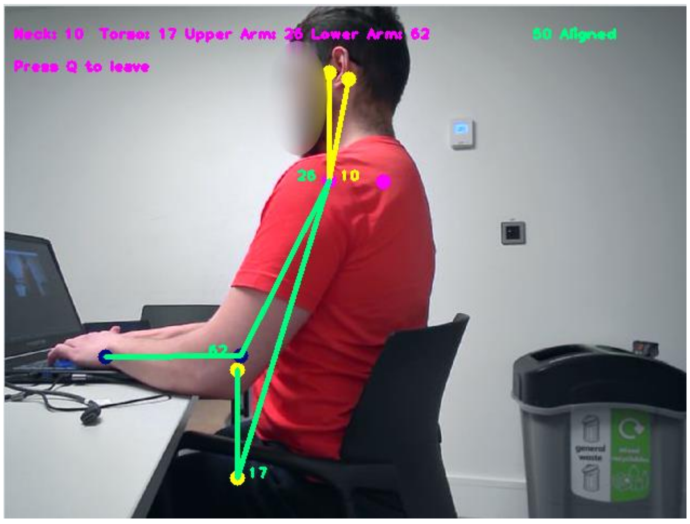


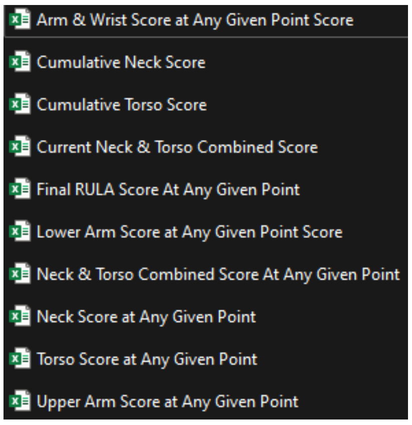
Figure 53. False positive alignment.

The calibration process applied to the product and the guidelines provided to the user during the initial setup are of a general nature. It is possible that certain scenarios or perspectives may result in the user interface indicating alignment, while it is providing inaccurate angles. An instance of this phenomenon is observable in **Figure 53**, wherein the user is executing a posture that would yield all green indicators, except for the neck joint indicator which appears yellow. The misalignment of the object in question is attributed to its orientation, whereby the rear aspect is inadvertently positioned towards the camera instead of the anterior aspect.

The assessment platform solely evaluates the left side of a human. Achieving identical outcomes cannot be guaranteed if the user relocates the webcam to the right-hand side. This issue could potentially pose a challenge for users who do not have a wall located to their left, as it may require a significant adjustment to their workstation configuration to effectively utilise the programme. This contravenes a fundamental criterion, namely accessibility.

8.2. Future Works

Minor adjustments could be made to optimise the user experience. As an illustration, the notification could be enhanced in terms of specificity by elucidating the precise aspect of the posture that necessitates correction. **Figure 54** displays a comprehensive inventory of the generated data. Further manipulation of the data can be conducted to obtain information tailored to specific scenarios.



Arm & Wrist Score at Any Given Point Score
Cumulative Neck Score
Cumulative Torso Score
Current Neck & Torso Combined Score
Final RULA Score At Any Given Point
Lower Arm Score at Any Given Point Score
Neck & Torso Combined Score At Any Given Point
Neck Score at Any Given Point
Torso Score at Any Given Point
Upper Arm Score at Any Given Point

Figure 54. Data outputs.

Subsequent endeavours may also encompass addressing certain constraints. The mitigation of certain limitations may be facilitated by the provision of additional time, such as the issue pertaining to the platform's singular evaluation of one side of the body. Certain constraints may require a prolonged duration to address, such as the incorporation of additional measures from the Rapid Upper Limb Assessment (RULA) worksheet.

The nature of office work bears resemblance to that of manual assembly work in that they both involve the execution of repetitive tasks over an extended duration. However, empirical testing is necessary to demonstrate that the programme can yield comparable results to manual assembly labour. In future endeavours, it is recommended to conduct testing on the platform in a wider range of scenarios to assess the software's versatility.

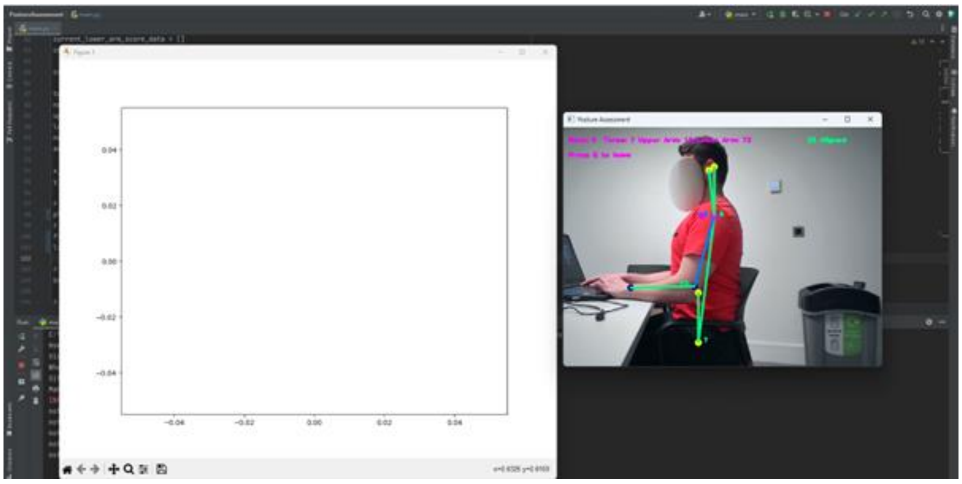
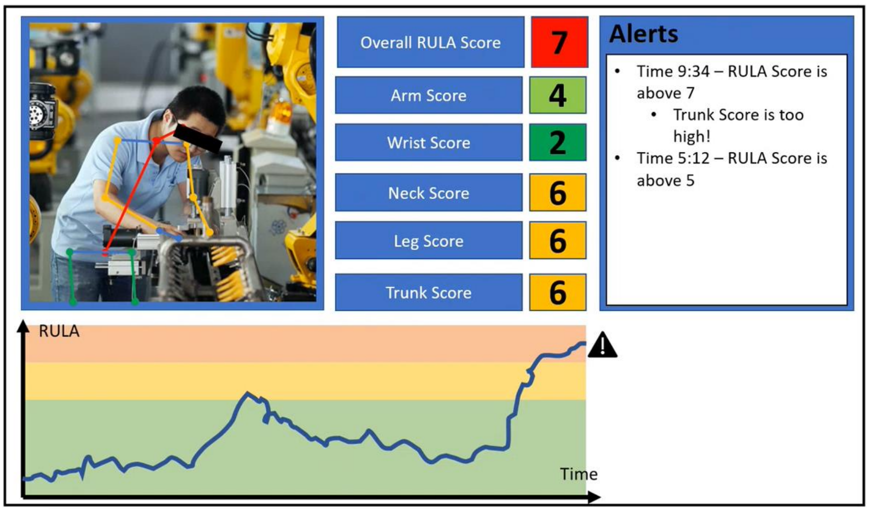


Figure 55. Attempt at real-time graph.

The real-time updating capability of a graph implementation can prove advantageous in certain scenarios. An endeavour was undertaken, as depicted in **Figure 55**, however, it did not yield the desired outcome. The endeavour remains present in the code as a commented-out section n, as it may prove advantageous to subsequent individuals who endeavour to undertake it. The issue at hand pertains to the automatic resizing of the graph. Upon conducting debugging procedures, it was determined that the graph is being generated in a real-time fashion. However, the issue of resizing is impeding the user's ability to visually perceive the graph.



- Time 9:34 – RULA Score is above 7
    - Trunk Score is too high!
- Time 5:12 – RULA Score is above 5

Figure 56. Dashboard design attempt.

The implementation of a dashboard has the potential to improve the overall user experience. **Figure 56** depicts a design attempt. The implementation of a dashboard can confer advantages to the user, as it enhances the user-friendliness of the software. The implementation process is not deemed arduous provided that the developer possesses proficiency in Python GUI. However, due to the time constraint, this task could not be accomplished despite being beyond the scope.

References

Ahmad, N., Ghazilla, R., Khairi, N. & Kasi, V., 2013. Reviews on various inertial measurement unit (IMU) sensor applications. *International Journal of Signal Processing Systems,* Volume 1, pp. 256-262.

Amaliya, S. et al., 2021. Study on Hand Keypoint Framework for Sign Language Recognition. *In 2021 7th International Conference on Electrical, Electronics and Information Engineering,* pp. 446-451.

Amazon Web Services, 2022. *What is Amazon Rekognition? - Amazon Rekognition.* [Online]
Available at: https://docs.aws.amazon.com/rekognition/latest/dg/what-is.html
[Accessed 30 11 2022].

Antwi-Afari, M., Li, H., Yu, Y. & Kong, L., 2018. Wearable insole pressure system for automated detection and classification of awkward working postures in construction workers. *Automation in Construction,* Volume 96, pp. 433-441.

BankMyCell, 2022. *How Many People Have Smartphones Worldwide (Dec 2022).* [Online]
Available at: https://www.bankmycell.com/blog/how-many-phones-are-in-the-world
[Accessed 30 11 2022].

Bello, S. et al., 2020. Deep learning on 3D point clouds. *Remote Sensing,* Volume 12, p. 1729.

BrainStation, 2022. *How Long Does it Take to Learn Python? (2022 Guide)* **|** *BrainStation®.* [Online]
Available at: https://brainstation.io/career-guides/how-long-does-it-take-to-learn-python
[Accessed 30 11 2022].

Chen, X. & Yuille, A., 2014. Advances in Neural Information Processing Systems. *Advances in Neural Information Processing Systems ,* Volume 2, pp. 1736-1744.

Chen, Y., Tian, Y. & He, M., 2020. Monocular human pose estimation: A survey of deep learning-based methods. *Computer Vision and Image Understanding,* Volume 192, p. 102897.

devjobsscanner, 2022. *Top 8 Most Demanded Programming Languages in 2022.* [Online]
Available at: https://www.devjobsscanner.com/blog/top-8-most-demanded-languages-in-2022/
[Accessed 30 11 2022].

Gómez-Galán, M. et al., 2020. Musculoskeletal Risks: RULA Bibliometric Review. *International Journal of Environmental Research and Public Health,* Volume 17, p. 4354.

Google Research, 2020. *MediaPipe Holistic — Simultaneous Face, Hand and Pose Prediction, on Device.* [Online]
Available at: https://ai.googleblog.com/2020/12/mediapipe-holistic-simultaneous-face.html
[Accessed 29 05 2023].

Google Research, 2020. *On-device, Real-time Body Pose Tracking with MediaPipe BlazePose.* [Online]
Available at: https://ai.googleblog.com/2020/08/on-device-real-time-body-pose-tracking.html
[Accessed 29 05 2023].

Google, 2022. *Home | mediapipe.* [Online]
Available at: https://google.github.io/mediapipe/
[Accessed 30 11 2022].

Google, 2022. *Pose | mediapipe.* [Online]
Available at: https://google.github.io/mediapipe/solutions/pose.html
[Accessed 01 12 2022].

GSMArena, 2022. *Samsung Galaxy S22 Ultra 5G - Full phone specifications.* [Online]
Available at: https://www.gsmarena.com/samsung_galaxy_s22_ultra_5g-11251.php
[Accessed 30 11 2022].

Harish, S. et al., 2021. New Features for Webcam Proctoring Using Python and Opencv. *REVISTA GEINTEC-GESTAO INOVACAO E TECNOLOGIAS,* Volume 11, pp. 1497-1513.

Hernández, Ó., Morell, V., Ramon, J. & Jara, C., 2021. Human pose detection for robotic-assisted and rehabilitation environments. *Applied Sciences,* Volume 11, p. 4183.

Huang, Y. et al., 2017. *A study on multiple wearable sensors for activity recognition.* Taipei, IEEE, pp. 449-452.

Kee, D. & Karwowski, W., 2001. LUBA: an assessment technique for postural loading on the upper body based on joint motion discomfort and maximum holding time.. *Applied ergonomics,* Volume 32, pp. 357-366.

Khandan, M. et al., 2018. Ergonomic assessment of posture risk factors among Iranian Workers: An alternative to conventional methods.. *Iranian Rehabilitation Journal,* Volume 16, pp. 11-16.

Kitamura, T., Teshima, H., Thomas, D. & Kawasaki, H., 2022. Refining OpenPose with a new sports dataset for robust 2D pose estimation. *In Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision,* pp. 672-681.

Koriukina, V., 2020. *Getting Started with OpenCV CUDA Module.* [Online] Available at: https://learnopencv.com/getting-started-opencv-cuda-module/
[Accessed 01 12 2022].

Kramer, M., 2018. Best practices in systems development lifecycle: An analyses based on the waterfall model. *Review of Business & Finance Studies,* Volume 9, pp. 77-84.

Lander, S., 2022. *Disadvantages & Advantages of Webcams.* [Online] Available at: https://smallbusiness.chron.com/disadvantages-advantages-webcams-67591.html
[Accessed 30 11 2022].

LearnOpenCV, 2023. *YOLOv7 Pose vs MediaPipe in Human Pose Estimation.* [Online] Available at: https://learnopencv.com/yolov7-pose-vs-mediapipe-in-human-pose-estimation/
[Accessed 29 05 2023].

Li, L. & Xu, X., 2019. A deep learning-based RULA method for working posture assessment. *In Proceedings of the Human Factors and Ergonomics Society Annual Meeting,* Volume 63, pp. 1090-1094.

Lin, T. et al., 2014. Microsoft coco: Common objects in context. *In European conference on computer vision,* pp. 740-755.

Lugaresi, C. et al., 2019. *Mediapipe: A framework for building perception pipelines,* s.l.: Google Research.

Manghisi, V. M. et al., 2017. Real time RULA assessment using Kinect v2 sensor. *Applied Ergonomics,* Volume 65, pp. 481-491.

Martınez, G., 2019. *OpenPose: Whole-body pose estimation,* (Doctoral dissertation, Master's Thesis, Carnegie Mellon University): Carnegie Mellon University.

McAtamney, L. & Corlett, E., 1993. RULA–A survey method for the investigation of work-related upper limb. *App. Ergon,* Volume 24, pp. 91-99.

Microsoft, 2022. *Microsoft C/C++ Documentation | Microsoft Learn.* [Online] Available at: https://learn.microsoft.com/en-us/cpp/?view=msvc-170 [Accessed 30 11 2022].

Middlesworth, M., 2023. *A Step-by-Step Guide to the RULA Assessment Tool.* [Online] Available at: https://ergo-plus.com/rula-assessment-tool-guide/ [Accessed 25 05 2023].

Moryossef, A. et al., 2021. Evaluating the immediate applicability of pose estimation for sign language recognition. *In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition,* pp. 3434-3440.

New York Chichester, W. S. C. U. P., 2007. 3. Systematic Observation. *Research Techniques for Clinical Social Workers,* pp. 51-69.

notify.run, 2023. *notify.run.* [Online] Available at: https://notify.run/ [Accessed 26 05 2023].

OpenCV, 2022. *About - OpenCV.* [Online] Available at: https://opencv.org/about/ [Accessed 30 11 2022].

OpenCV, 2022. *OpenCV: Introduction to OpenCV-Python Tutorials.* [Online] Available at: https://docs.opencv.org/4.x/d0/de3/tutorial_py_intro.html#:~:text=OpenCV%20supports%20a%20wide%20variety,are%20also%20under%20active%20development. [Accessed 30 11 2022].

Quwaider, M. & Biswas, S., 2010. Body posture identification using hidden Markov model with a wearable sensor network. *In 3rd International ICST Conference on Body Area Networks.*

Ren, S., He, K., Girshick, R. & Sun, J., 2015. Faster r-cnn: Towards real-time object detection with region proposal networks. *Advances in neural information processing systems,* Volume 28, pp. 91-99.

Roman-Liu, D., 2014. Comparison of concepts in easy-to-use methods for MSD risk assessment. *Applied Ergonomics,* Volume 45, p. 420–427.

Sanchez-Lite, A., Garcia, M., Domingo, R. & Angel Sebastian, M., 2013. Novel ergonomic postural assessment method (NERPA) using product-process computer aided engineering for ergonomic workplace design.. *PloS one,* Volume 8, p. e72703.

Schaffer, E., 2022. *Is C++ still a good language to learn for 2022?.* [Online] Available at: https://www.educative.io/blog/learn-cpp-for-2022 [Accessed 30 11 2022].

Schaub, K., Caragnano, G., Britzke, B. & Bruder, R., 2012. The European Assembly Worksheet. *Theoretical Issues in Ergonomics Science,* Volume 14, pp. 1-23.

Schaub, K., Caragnano, G., Britzke, B. & Bruder, R., 2013. The European assembly worksheet. *Theoretical Issues in Ergonomics Science,* Volume 14, pp. 616-639.

Schaub, K. et al., 2012. Ergonomic assessment of automotive assembly tasks with digital human modelling and the 'ergonomics assessment worksheet'(EAWS). *International Journal of Human Factors Modelling and Simulation,* Volume 3, pp. 398-426.

Sharma, V., 2022. Object Detection and Recognition using Amazon Rekognition with Boto3. *In 2022 6th International Conference on Trends in Electronics and Informatics (ICOEI),* pp. 727-732.

Takala, E. et al., 2010. Systematic evaluation of observational methods assessing biomechanical exposures at work. *Scand J Work Environ Health,* Volume 36, pp. 3-24.

The Python Software Foundation, 2022. *3.11.0 Documentation.* [Online] Available at: https://docs.python.org/3/ [Accessed 11 30 2022].

Toshev, A. & Szegedy, C., 2014. *Deeppose: Human pose estimation via deep neural networks.* Columbus, IEEE.

United States Data Science Institute, 2022. *Is Python still the language of Data Science in 2022?.* [Online]
Available at: https://www.usdsi.org/data-science-insights/is-python-still-the-language-of-data-science-in-2022
[Accessed 30 11 2022].

Yan, X., Li, H., Li, A. R. & Zhang, H., 2017. Wearable IMU-based real-time motion warning system for construction workers' musculoskeletal disorders prevention.. *Journal of Research in Medical Sciences,* Volume 74, pp. 2-11.

Yazdanirad, S. et al., 2018. Comparing the effectiveness of three ergonomic risk assessment methods RULA, LUBA, and NERPA to predict the upper extremity musculoskeletal disorders.. *Indian journal of occupational and environmental medicine,* Volume 22, p. 17.

Yu, Y., Umer, W. U., Yang, X. & Antwi-Afari, M. F., 2021. Posture-related data collection methods for construction workers: A review. *Automation in Construction,* 124(103538).