Simple integrated electronic gantry-based timing system for school athletics

Final report: Supplementary examination

P.S. Schoeman
17025843

Submitted as partial fulfilment of the requirements of Project EPR400
in the Department of Electrical, Electronic and Computer Engineering
University of Pretoria

February 2021

Study leader: Prof. T. Hanekom

Electronic copy

Part 1. Preamble

This report describes work that I did in my final year project developing a simple integrated electronic gantry-based timing system for school athletics.

Project proposal and technical documentation
This main report contains a copy of the approved Project Proposal (as Part 2 of the report). Complete technical documentation appears as part of the additional material submitted on the electronic medium that accompanies this report and that has been submitted to my study leader.

Project history
This project makes use of software libraries developed by STMicroelectronics for controlling the peripheral functions of their micro-controllers. Where other authors' work has been used, it has been cited appropriately, and the rest of the work reported on here, is entirely my own.

Language editing
This document has been language edited by a knowledgeable person. By submitting this document in its present form, I declare that this is the written material that I wish to be examined on.

My language editor was Dr. Schalk Schoeman

01-02-2021

Language editor signature *Date*

Declaration

I, Pieter Schoeman understand what plagiarism is and have carefully studied the plagiarism policy of the University. I hereby declare that all the work described in this report is my own, except where explicitly indicated otherwise. Although I may have discussed the design and investigation with my study leader, fellow students or consulted various books, articles or the Internet, the design/investigative work is my own. I have mastered the design and I have made all the required calculations in my lab book (and/or they are reflected in this report) to authenticate this. I am not presenting a complete solution of someone else.

Wherever I have used information from other sources, I have given credit by proper and complete referencing of the source material so that it can be clearly discerned what is my own work and what was quoted from other sources. I acknowledge that failure to comply with the

instructions regarding referencing will be regarded as plagiarism. If there is any doubt about the authenticity of my work, I am willing to attend an oral ancillary examination/evaluation about the work.

I certify that the Project Proposal appearing as the Introduction section of the report is a verbatim copy of the approved Project Proposal.

01-02-2021

P.S. Schoeman Date

TABLE OF CONTENTS

LIST OF ABBREVIATIONS

ADC	Analog-to-digital converter
CCD	Charged coupled device
CMOS	Complimentary metal-oxide-semiconductor
CPU	Central processing unit
DC	Direct current
DAC	Digital-to-analog converter
DCT	Discrete cosine transform
DFT	Discrete Fourier transform
DSP	Digital signal processor
FFT	Fast Fourier transform
HAL	Hardware application layer
IAAF	International association of athletics federations
IC	Integrated circuit
IDCT	Inverse discrete cosine transform
IP	Internet protocol
JFIF	JPEG File Interchange Format
JPEG	Joint Photographic Experts Group
LED	Light emitting diode
MOSFET	Metal–oxide–semiconductor field-effect transistor
PCB	Printed circuit board
RTC	Real time clock
SD	Secure digital
TCP	Transmission control protocol
USB	Universal serial bus

Part 2. Project Definition: Approved Project Proposal

This section contains the problem identification in the form of the complete approved Project Proposal, unchanged from the final approved version.

For use by the project lecturer	Approved		Revision required	

Feedback

To be completed by the student						
PROJECT PROPOSAL 2020				Project no	TH2	Revision no 0
Title	Surname	Initials	Student no	Study leader (title, initials, surname)		
Mr	**Schoeman**	**PS**	17025843	Prof Tania Hanekom		
Project title						
Sport Science: Simple integrated electronic gantry-based timing system for school athletics.						

Language editor name	Language editor signature
Schalk Schoeman	
Student declaration I understand what plagiarism is and that I have to complete my project on my own.	**Study leader declaration** This is a clear and unambiguous description of what is required in this project
Student signature	Study leader signature and date
	see attachment

1. Project description

What is your project about? What does your system have to do? What is the problem to be solved?

The goal of this project is to design a low-cost system capable of automatically timing athletes in a primary school track-based athletics competition. Manual timing can lead to human error, which may cause unresolvable disagreements when no visual record of the race finish is available.

The proposed solution is an automatic timing system that will reduce disputes after races and eliminate human error. The system will use an array of low resolution and low frame rate cameras to photograph the finish line at a high frame rate for a fraction of the cost of a high-speed camera. A height sensor array will be used in conjunction with the cameras to provide depth data and increase the accuracy of the analysis algorithm. The sensor units will be placed on a gantry, extending over the finish line, on top of the lane to be measured. The system will be extendable with up to eight sensor units that can cover all eight lanes of a standard athletics track. A software system running on another embedded platform will then merge the data provided by the modules from each lane and determine each athlete's relative position. The images and places of the athletes will then be relayed to a computer running a software suite that will display it to the timekeepers.

2. Technical challenges in this project

Describe the technical challenges that are *beyond* those encountered up to the end of third year and in other final year modules.

2.1 Primary design challenges

1. Ensuring the camera array will continue to function under the varying lighting conditions normally present on an athletics track.
2. Ensuring the system will correctly identify the thorax of an athlete, given that there are many ways in which an athlete can cross the finish line.
3. Ensuring that the different points of view of the cameras in the array does not affect the performance and accuracy of the image processing system.

2.2 Primary implementation challenges

1. The implementation of the height sensor array.
2. The implementation of an image processing algorithm that will determine the position of an athletes thorax.
3. The optimization of the image processing algorithm to complete all data handling within one minute of the end of a race.

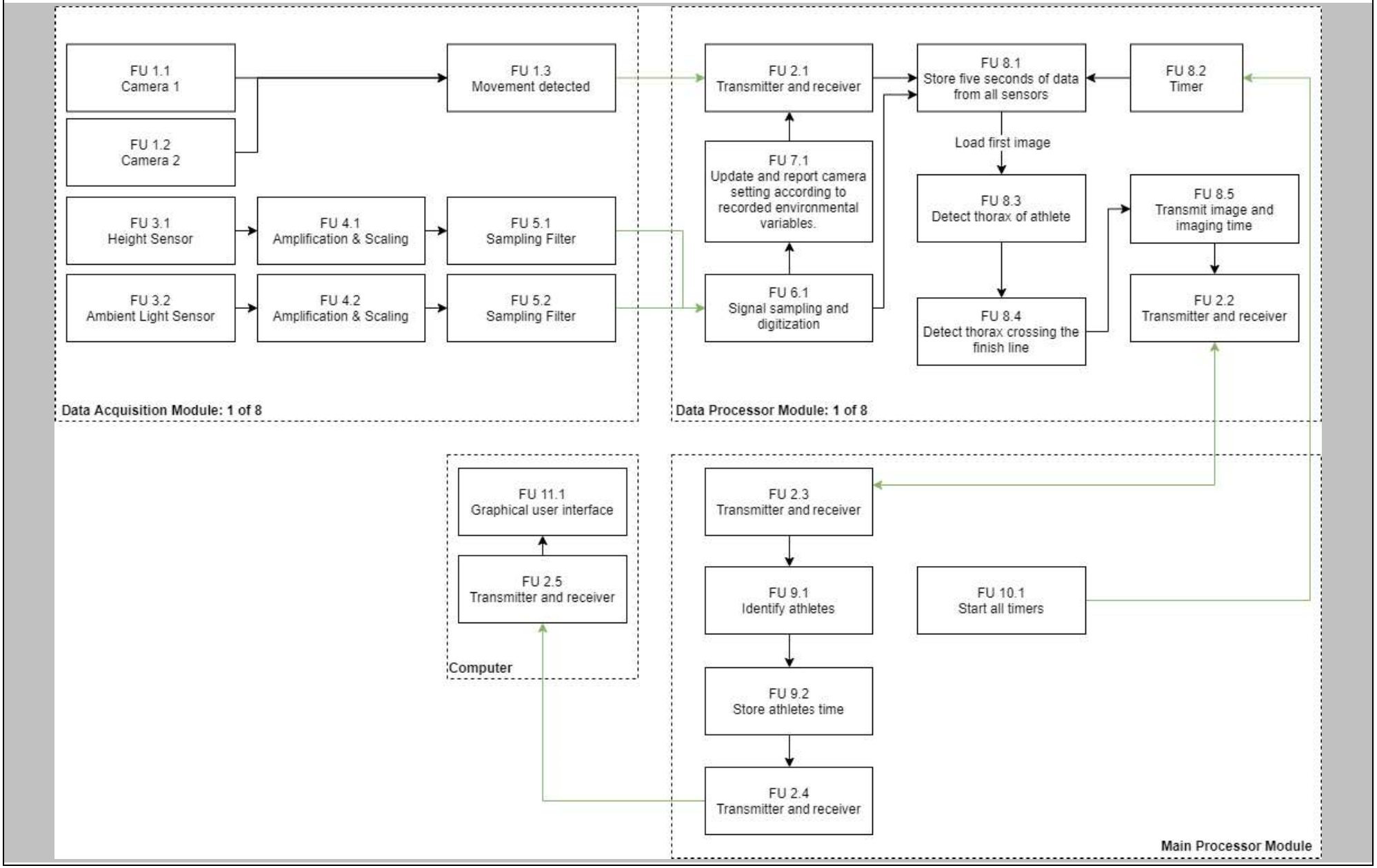
3. Functional analysis

3.1 Functional description

Describe the design in terms of system functions as shown on the functional block diagram in section 3.2. This description should be in narrative format.

A gantry will be placed over the finish line, with the horizontal component being approximately two meters above the ground. Each lane will be equipped with a data acquisition module facing downwards as well as a data processing module connected directly to it. The timers on the data acquisition modules are all started as soon as the starting gun is fired (FU 11.1). The instant an athlete in a specific lane enters the field of view of a camera (FU 1.1, FU 1.2, FU 1.3 & 2.1), the data processing module starts capturing and saving (FU 8.1) data for a set amount of time. The data includes the images from the cameras, timestamps from the timer (FU 8.2), depth information from the height sensor array (FU 3.1, FU 4.1, FU 5.1 & FU 6.1) and ambient light data (FU 3.1, FU 4.2, FU 5.2 & FU 6.1) from the light sensors. Depth data will later be used to aid discrimination among the head and other parts of the body. Ambient light data will be used to correct the image processing algorithm when lighting conditions change. After data acquisition, the system will analyze each frame by calculating the position of the athlete's shoulders. If a line drawn between the two shoulders crosses the finish line in any way, that photo and timestamp will be transmitted to the main processing module (FU 8.3, FU 8.4, FU 8.5 & FU 2.2). The main processing module will store the athlete's images and times (FU 9.1 & 9.2). It will also provide access to them through a connected computer (FU 2.4). The winning photos of each lane together with their time will then be displayed on the graphical user interface of a computer (FU 12.1).

3.2 Functional block diagram



4. System requirements and specifications

These are the core requirements of the system or product (the mission-critical requirements) summarised in table format .

	Requirement 1: fundamental functional and performance requirement	**Requirement 2**	**Requirement 3**
1. Core mission requirements of the system or product. Solution of the problem will be the most important requirement. Capture this in the set of requirements.	The system should be capable of determining the time an athlete's thorax crosses the finish line with a very high degree of accuracy.	The system should always correctly report the position of a white line painted on a grass field (finish line and lane markers).	The system should ensure the data acquisition modules are in the correct place before the race starts.
2. What is the target specification (in measurable terms) to be met in order to achieve this requirement?	A camera array will be created capable of filming the finish line at a speed of at least 100 frames per second.	The reported position of the finish line should always be inside the dimensions of the 1.27 x 0.05-meter rectangle of the real finish line, or within the 0.05-meter width of the lane markers.	The system should use the data from the line identification algorithm to determine if the vertical space above the top lane marker and the space below the bottom lane marker are even (+-10%).
3. Motivation: how will meeting this specification solve the problem?	At 100 frames per second, each new frame is 0.01 seconds after the previous frame. Thus, if an athlete's thorax is not over the finish line in one frame but over in the next, the time at which the second frame is taken will be used.	The finish line is 0.05 meters wide and 1.27 meters in length. Line markers are 0.05 meters wide and will extend through the cameras entire field of view. Ensuring the system places each line in these boxes will give an accurate result.	It will ensure that the camera system is correctly positioned and has a large enough field of view to accurately track the athlete.
4. How will you *demonstrate* at the examination that this requirement has been met?	The monitoring software will be placed in test mode. The system will then be set to run at 100 fps for 5 seconds. The number of photos taken will be counted and displayed on screen as an average frames per second value.	The system will be placed into test mode. The data acquisition system will be placed over the finish line. The system will display an image with lines drawn where the finish line and lane markings were identified.	The system will be placed into calibration mode. The system will then output whether the cameras are in the correct position. If not, the system will display how the module should be moved to place it correctly.
5. What is the deliverable? What are the aspects that you will design and implement yourself to meet this requirement? If none, indicate clearly.	Two cameras will be integrated into an array. A microprocessor and memory unit will trigger the camera to start capturing images as well as read and store the returned data. The recorded jpeg images will be decoded using custom firmware.	An algorithm for determining the position of white lines on a grass field.	All aspects of the calibration module. All firmware will be developed from first principles.
6. What are the aspects to be taken off the shelf to meet this requirement? If none, indicate clearly.	Cameras, Microprocessor, Memory.	None.	None.

System requirements and specifications (continued)

	Requirement 4	Requirement 5	Requirement 6
1. Core mission requirements of the system or product. Solution of the problem will be the most important requirement. Capture this in the set of requirements.	It should not take more than one minute from the last of eight athletes crossing the line to process the results.		
2. What is the target specification (in measurable terms) to be met in order to achieve this requirement?	The camera has a resolution of X by Y. A data processing module capable of processing the data involved with decoding and analyzing (500/60) XY pixels per second is required.		
3. Motivation: how will meeting this specification solve the problem?	At a resolution of X by Y, 500XY pixels will be present at most (assuming raw quality) after a 5 second recording interval at 100 frames per second. The algorithm should thus process, at most, (500/60)XY pixels per second.		
4. How will you *demonstrate* at the examination that this requirement has been met?	The system will be placed into test mode. A person will run underneath the gantry and trigger the camera. The processing time will be displayed on the user interface.		
5. What is the deliverable? What are the aspects that you will design and implement yourself to meet this requirement? If none, indicate clearly.	An optimized computer algorithm capable of identifying the time an athlete crosses the finish line by processing the recorded data in under one minute.		
6. What are the aspects to be taken off the shelf to meet this requirement? If none, indicate clearly.	None.		

5. Field conditions

These are the real world conditions under which your project has to work and has to be demonstrated.

	Field condition 1	Field condition 2	Field condition 3
Field condition requirement. In which field conditions does the system have to operate? Indicate the one, two or three most important field conditions.	The system will have to operate outside on an athletics field during an active event.	The system should be able to work on any athlete, irrespective of physical differences.	The system should be capable of operating during temperature ranges common in South Africa.
Field condition specification. What is the specification (in measurable terms) for this field condition?	The system should be able to handle lighting conditions ranging from sunrise to sunset. Operating in rainy conditions is not required.	The system should be able to function with any type or color of athlete's clothing. The length or ethnicity of the athlete shouldn't have any effect.	The system should be able to withstand temperatures ranging from 0 to 45 degrees Celsius.

6. Student tasks

6.1 Design and implementation tasks

List your primary design and implementation tasks in bullet list format (5-10 bullets). These are *not* product requirements, but *your* tasks.

- Run simulations on photos using Python libraries to identify the processing techniques that will be necessary to solve the problem.
- Build a sensor array with a camera capable of taking at least 100 pictures per second.
- Decode the jpeg data from the camera array into usable data for the thorax identification algorithm. Development will start in Python and will then be ported to C or Assembly.
- Develop the line identification algorithm. Development will start in Python and will then be ported to C or Assembly.
- Develop the thorax identification algorithm. Development will start in Python and will then be ported to C or Assembly.
- Test the system under ideal conditions and improve where necessary
- Develop the firmware for the main processor module. This will involve the timekeeping system. Development will start in Python and will then be ported to C or Assembly.
- Develop the software for displaying the results as well as the final finish photo on a computer.
- Build the gantry that will be placed over the finish line and test the integrated system under field conditions. Improve where necessary.

6.2 New knowledge to be acquired

Describe what the theoretical foundation to the project is, and which new knowledge you will acquire (beyond that covered in any other undergraduate modules).

- To decrease the amount of data that needs to be transferred between the camera and the processor jpeg encoding will be used. Knowledge of jpeg encoding, and decoding will be required to interpret the received data.
- Identifying the thorax of athletes as well as lines on the ground will require advanced knowledge of image processing techniques. Emphasis will be placed on edge detection and surface area integration techniques.
- Knowledge of ARM32 assembly language will be required to optimize sections of the thorax detection, line identification and jpeg decoding algorithms.
- Designing the graphical user interface will require knowledge of the qt framework.
- A thorough understanding of the rules set by the International Association of Athletics Federations (IAAF) will be required to design the system.

Part 3. Main Report

1. Literature study

The design of the simple integrated electronic gantry-based timing system for school athletics merged knowledge from the fields of image processing, object detection and depth analysis with electronic design and efficient firmware engineering. It also required knowledge of athletics rules set by the international association of athletics federations (IAAF). The important work and background technical information that served as an inspiration and guide during the design of this system is discussed below.

The research into the the system started with getting acquainted with the manual timing procedures set out by the IAAF [1]. The procedure of multiple manual timers given in the rules is however not followed by most schools as it is often wrongly considered redundant. This may lead to inaccurate times being recorded. The photo finish guidelines [2] released by the IAAF defined the horizontal end of the torso as the outer end of the collarbone. The vertical end of the torso is identified as a horizontal line drawn through the hip line. The IAAF classifies an athlete as having finished if any part of their torso crosses the finish line.

The IAAF allows for the use of fully automatic timing systems using cameras and / or transponders [1]. For a system to qualify it should have a frame rate of no less that 100 frames per second in local competitions. Frame rate requirements for international competitions are much more stringent. The system should also be automatically started using a starting gun with a delay of less than 0.001 seconds. Suitable user interface elements should be present to show the placement of the lane markings. It was also defined that a system which operates automatically at the finish line but is started by hand produces times no more accurate than the standard hand timing technique. A automatic timing system that starts the timer but requires manual intervention at the finish line is not allowed at any IAAF athletics competition.

The IAAF documentation [1] defined the size of each lane in a standard athletics track lane to be 1.17m wide excluding the lane markings. All lane markings are set at a width of 50mm. The full width of an athletics track is given as 9.76m. Further information was then gathered on how the timing system could be automated while complying with the above stated rules.

Koceski and Koceska (2010) [3] developed a freight ramp control system that responded to hand gestures made by a remote operator. A video camera placed in front of the operator captured their hand and body gestures. A computer system then proceeded to decoded these gestures to control the loading system. The technique the researchers used was based of the idea of background subtraction. A reference frame was prepared based on a running average taken from each incoming frame. The weighting of each new frame decreases as time continues. The newest incoming frame is then subtracted from this reference frame to reveal the foreground. A large subtraction result indicates that the pixels in the newest incoming frame significantly differed from the pixels in the reference frame. The result of the subtraction is then passed through a threshold algorithm, where significant differences are changed to white and negligible differences are transformed to black. The distribution of white pixels in the horizontal and vertical axis of the image was then matched to a predetermined library of distributions corresponding to specific gestures. Correlation was then used to determine which library distribution fitted the observed distribution best. It was thought that

this technique could be extended to determine the position of an athletes body as the moved over the finish line in accordance with IAAF rules.

Lin et al (2007) [4] and Chadha et al (2011) [5] investigated the use of template matching for the purposes of human feature detection. Chadha's work focused on the frequency domain while Lin's work focused on the spatial domain. Template matching in the spatial domain aims to determine whether a pre-prepared template is present in a frame of video. The template can represent any object that the operator wishes to detect. Multiple templates can be used to search for a variety of objects in a single frame. For this technique to properly work, multiple templates of a single object is required in all it's possible orientations. The frequency domain analysis used the discrete cosine transform (DCT) to transform the image from the spatial domain into the frequency domain. These types of transforms are very efficient as they can be optimised in the same manner as the Fast Fourier transform (FFT) (Samra et al, 2004 [6]). While the frequency domain analysis was primary focused on the detection of facial features it may be extended to the athlete detection problem. Chadha et al obtained very good results with detection percentages in the low nineties using this approach.

The DCT is also the transform around which the joint photographic experts groups (JPEG) [7] format is based. This encoding standard starts by dividing an image into its luminance and two chrominance components. The luminance values gives and indication of the brightness of an image and is derived from RGB using the weightings given in equation (1.1) [8].

$$Y = 0.299R + 0.587G + 0.114B \tag{1.1}$$

The chrominance values correspond to the hue and saturation of image and are calculated according to the weights given in equation (1.2) [8].

$$\begin{aligned} Cb &= -0.1687R - 0.3313G + 0.5B + 128 \\ Cr &= 0.5R - 0.4187G - 0.0813B + 128 \end{aligned} \tag{1.2}$$

The JPEG standard then divides the luminance and chrominance values, where each value represents a pixel, into 8×8 blocks. To prepare these 8×8 blocks for the DCT a constant of 128 is subtracted from each value. The two dimensional version of the DCT is then applied to each block. JPEG can encode colour in a different way using a technique called chroma sub-sampling. This essentially merges two 8×8 blocks into one 16×8 or 8×16 block. Every second row or column is then discarded to leave one 8×8 block on which the DCT is performed. The colour information is thus encoded with half the resolution of the luminance component. Research in the field of human colour perception found that the human eye is much less sensitive to changes in colour than it is to changes in brightness. The approach in image analysis is usually to ignore the colour component and focus primarily on the luminance component as it is where most of the detail is encoded.

The output of the DCT produces an 8×8 grid that represents amplitudes of cosine waves that, if added together in both the horizontal and vertical directions, will form the desired image through constructive and destructive interference. The value starting in the top left corner represents the zero frequency (DC) component of the block. This value gives an representation

of the overall brightness of this section of the image. The values located in each block to the right, bottom or diagonal of the DC value represents a half cycle increase in the frequency of the cosine wave.

The values in each of the 8×8 grids are then divided by another grid called the quantisation table. This table is a function of the quality of the desired output image. This is also where JPEG reduces the image detail. A common approach is to zero most of the high frequency components in each 8×8 block as the human eye is insensitive to the higher frequency components. The image is then encoded using a Huffman encoder and stored with the quantization and Huffman tables in a binary format. These tables are included into the image to enable the decoder to decode them as many cameras and image manipulation software packages use incompatible tables. Markers are then added to the data to identify the location of the different components in the binary file.

Ken Cabeen and Peter Gent (1999) [9] conducted a short study on how this quantization procedure affects the final image. At a quality level of 50% they found almost no visible loss in image quality. They did however notice a considerable decrease in the image size. They also found that decreasing the image quality below 50% results in a drastically reduced image quality with very little gain in size reduction.

Technical documentation released by STMicroelectronics [10] on creating a serial communication interface showed that the symbol rate and data rate of this communication medium is equal. Many encoding methods are available to decrease the size of an image transmitted using this link including WebP from Google and PNG. WebP produces images that are on average about 25-34% smaller than the equivalent JPEG images. JPEG is however still the de facto standard in the world of single board cameras. The same company [11] released an application note detailing the use of Ethernet frames in computer to microcontroller networks. This note detailed the considerable resources required when implementing a full transmission control protocol using the internet protocol (TCP/IP) on a microcontroller. It did however give details regarding how it can be implemented manually.

Two cameras positioned next to each other enabled Tian et al (2011) [12] to determine the distance to objects in the cameras field of view. While these cameras did not use serial as communication with the management computer they did use analog video, which is also split between luminance and chrominance components. The binocular imaging system was found to be a very effective way of approximating the location of a walking individual in addition to determining their speed. The system designed by the researchers then used this information to keep the camera centred on the individual. This technique demonstrated the use of interleaving video frames produced by two cameras. This can effectively double the output frame rate of such an array. The binocular configuration also enables depth detection which can help to identify different sections of an athletes body.

The work from Koceski and Koceska was applied in the design of the athlete's torso identification system. The same technique they applied to use a template image and then subtract subsequent images from that template was used to isolate the athlete in a moving frame. The idea of using the DCT for image analysis and the capabilities that can be achieved using this transform was taken from work done by Chadha et al. This technique was primarily applied to

the lane marking identification system. The knowledge of this transform was then combined with the documentation describing the JPEG encoding process to develop an efficient image analysis algorithm which did not require multiple stacked frequency transforms.

The specifications for the timer system was then as closely aligned as possible to the IAAF rules [1] on automatic timing systems, given the requirements set in the project and budget. The size of the athletics track was used during the design of the gantry and the decision on which camera lenses to use. The dimensions also limited the type of communication protocol that can be used to communicate with cameras based on the maximum cable lengths. The application notes provided by STMicroelectronics were instrumental is deciding which communication system to use.

Cameras employing some sort of image compression before transmission was selected to decrease the time taken by transferring images between the camera and microcontrollers. This also necessitated the design of a decoder algorithm, which was done from first principals using the official documentation and some reverse engineering of test images.

Much work has been done through the years by many research teams on human and facial detection. This work did not directly correlate with the problem being discussed in this document, but many methods were transferable. Most of the image analysis tools written by the researchers mentioned executed on desktop computers with spare resources. The problem in this document required image analysis on a microcontroller which introduced many exiting challenges.

2. Approach

The project started with designing a camera module capable of taking 100 images per second. Important specifications that were taken into consideration included frame rate communication protocol, communication protocol speed, resolution, available lenses and configurability. The single board cameras outputting analogue video at a fixed frame rate were rejected as a result of the extra overhead introduced in the analogue to digital conversion process. The choice was then narrowed down by selecting all the cameras with a protocol that is theoretically capable of transmitting half the required images per second at a minimum resolution of 160 x 120. One high baud rate serial camera exceeded the requirement and was selected. The cameras achieved this by encoding the image using a JPEG algorithm on digital signal processor (DSP) chip included with the camera. This significantly reduced the size of the image to transmit.

The second stage involved the selection of a micro-controller that is responsible for handling all the data acquisition and processing functions over each lane. The initial concept split the data acquisition and data processing modules. This however proved unnecessary as a result of high performance micro-controllers from STMicroelectronics. The micro-controller specifications that were taken into consideration included supported communication peripherals, maximum clock speed, programming memory, flash memory, development tool-chain and power draw. The STM32F7 line of controllers was selected as a result of their clock speed of 216 MHz (the second best of their product line) and serial interfaces fast enough to communicate with the cameras without causing bottlenecks.

Work then started on designing JPEG decoding, lane identification and thorax finding algorithms in Python. A web-cam in conjunction with test images downloaded from the internet were used to develop these algorithms. A algorithm capable of fully decoding arbitrary JPEG images was eventually finished. Many different ways of detecting athletics lanes were tried including frequency transforms and template detection. Template detection proved to be the most efficient technique. Both template detection and colour matching was used to develop the thorax detection algorithm.

The next problem that needed solving was where to store the photos once they were taken. The chosen micro-controller included integrated flash memory but that memory is also used by the firmware in some cases. It is also not nearly enough for a few seconds of photos at the required frame rate. The solution involved sourcing memory chips and a memory controller that can be accessed over one of the controllers communication ports. The SPI bus was chosen as it supported the required data rate. Many flash memory chips were also available using that protocol that matched the required size and speed of the application.

The design then moved to how the system will detect when a race has been started. Races are traditionally started using a type of starting pistol. If a traditional starting pistol firing blanks is used it usually contains a small microphone that picks up the sound. This sound then gets amplified and transmitted over radio or cable to a timing system that starts a timer once it has been detected. It was decided that one of the camera modules, or a separate module if preferred, would detect this sound, arm all the other modules and start the timer.

A software system running on a laptop needed to communicate with the camera modules

over each lane. Multiple communication protocols were investigated including serial, USB and Ethernet. Ethernet proved to be the most extendible protocol allowing for many devices to share the same network. Commercial desktop network switches can be used to facilitate communication between any two modules on the network. In addition built in futures like broadcasting allowed one component to communicate with all the other components at the same time.

All the mentioned hardware components now had to be integrated into a printed circuit board (PCB). Electronic switches that control the power delivery to the cameras were added to save power when the cameras were not in use and to reset the cameras if they don't respond to new commands. A seven segment display was added for user feedback when pairing the device to a computer and to display error codes. A power conditioning system and heatsink was installed to supply the board with power. The micro-controller, debugger and Ethernet module was already built into the selected development board. This development board included a helpful board-to-board interconnect that was used to connect the designed PCB with it. The memory cards were designed on separate PCB's to allow the user to directly read their content from a computer. It also enables the memory to be upgraded or replaced in the future.

The project then moved to a low level software development phase. This phase started with designing a framework where all the other software components of the system can be fitted into. This was designed in the form of a state machine, where the system can be put into a predefined series of states based on commands from a central controller. This section of the firmware included a place where the board will initialise all its peripheral functions and test all the components connected to it. It then enters a loop where it waits new commands. The network communication software was then added to accept new commands, pass them to the state machine and respond. Camera firmware was then written to allow the state machine to control the cameras. Custom firmware also had to be developed for the memory configuration developed earlier in the project. Smaller libraries where written to control the seven segment display, starting gun interrupt and the real time clock (RTC).

After a workable system framework was implemented, work started on transferring all the Python image analysis tools to micro-controllers. The completed Python firmware which included the JPEG decoding algorithm, thorax detection system and lane detection algorithm was ported to C. Glue firmware was then written to make everything work together. This phase was mainly defined by a lot of debugging.

Software development then shifted from the micro-controller to the management computer that was responsible for controlling all the camera modules. This section involved designing a athletics management system with a graphical user interface. This system needed to allow the user to specify the number of camera modules they needed to add and control their functionality from a distance. This section also handled each athletes timer and showed their current time on the screen. In addition it allows the user to enter the athlete's information into the system and view their final photo.

All the electronic components were placed into a plastic housing that allowed them to handle external environmental conditions. A gantry was then built capable of spanning over a athletics field and the system was tested.

3. Design and implementation

3.1 Theoretical analysis and modelling

A diagram showing the top view of an athletics track finish line with the system installed in the three camera configuration is shown in figure (1) below. The diagram also shows the power delivery system (red), communication system (blue) and management computer (white). Names defined here will be used throughout the report to indicate the relevant components. The report will be divided into sections detailing how each subsystem was designed. Special sections will be included, where applicable, to give details with regards to PCB layout and other design considerations.

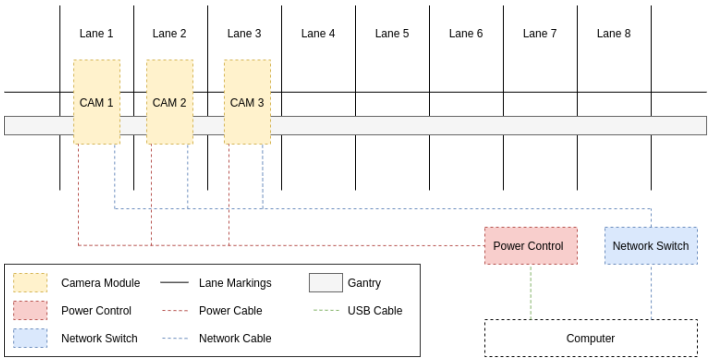


Figure 1.
Diagram representing the finished system and report naming scheme

3.1.1 Camera array

The design of the camera module denoted in diagram (1) as "CAM X" (yellow) required a camera array capable of taking 100 images per second. The frame rate of a standard digital camera is determined by the maximum rate at which the sensor can take a picture and the accompanying camera control, encoding and transmission electronics can process it. The image sensor is one of the core components that determines a camera's performance and thus the analysis will start there. Image sensors can be divided into complimentary metal oxide semiconductor (CMOS) and charge coupled device (CCD) sensors, where CMOS is the fastest as a result of a direct logic connection to each pixel. CCD sensors operate more like a shift register where data needs to be shifted out. Cameras with CMOS sensors were thus the logical choice for this array where fast access time is critical. The next factor that can play a role is the time the camera takes to process an image which occurs in cases where JPEG encoded images are the output. These images would first need to undergo a relatively complex mathematical and computationally expensive encoding process. Cameras with hardware encoders were thus selected where possible. The third and final bottleneck can be the maximum speed at which the camera can transmit the captured data to a computer. JPEG encoding helped a lot here by

significantly decreasing the size of the image to be transmitted. The downside is that it makes the length of the image and thus the time it takes to transmit variable. The length of a JPEG image is mainly a function of the detail it captured and the encoding quality. The minimum required data-rate can be calculated by estimating the average size of a JPEG encoded image. Average JPEG image sizes at a resolution of 160 x 120 are tabulated in table (1) below. These images contained a minimum to medium amount of detail as is expected of a frame that will mostly contain an empty athletics's field, with one person moving through it. The diminishing returns by decreasing the quality motioned in section (1) is very evident here.

Quality	Percentage	Average Size (bytes)
High	99%	1638
Good	66%	1394
Low	33%	1377

Table 1.
Average length of 160 x 120 JPEG encoded images at different quality settings with low to medium detail

The ideal number of cameras is the lowest number of cameras, in terms of cost and the complexity of the required camera control firmware. Subsequent calculations were thus conducted for two cameras where each required a frame rate of 50 frames per second. The symbol rate in serial communications, which is a common interface for single board cameras, is equal to the bit rate. The length of the highest quality image from table (1) was selected and a 20% error margin was added. The minimum bit rate required was calculated using equation (3.1). The constant of eight is used to convert from bytes to bits where "n" represents the number of images that needs to be taken, "s" is the size of the average image and "t" is the time frame available for transmission. This equation helped to find cameras that might be suitable for the application. The final frame rate will still depend on the speed of the sensor as will be shown shortly.

$$\begin{aligned} Bd &= \frac{8(n \times s)}{t} \\ Bd &= \frac{8(50 \times 1966)}{1} \\ Bd &= 786400 \end{aligned} \tag{3.1}$$

The SC20MPB [13] high baud rate serial camera met the desired bit rate specifications with a maximum bit rate of 921600 BAUD. The camera is equipped with a SC2235 CMOS image sensor which is capable of capturing one image every 0.02 seconds. The on-board JPEG encoder is implemented in the hardware of the Hi3518 DSP chip and should have a negligible delay on the system. Secondary serial commands that returns the length of the image, sets the resolution or the quality is short and thus negligible for the timing calculations. The expected timing diagram is given in figure (2). The label "start" indicates when a image capture was initiated by the micro-controller, while the label "stop" indicates when the controlling device received all the requested data. The blue arrows indicates the time the camera takes to capture

an image while the red indicates the time it takes the camera to transmit it back to the host.

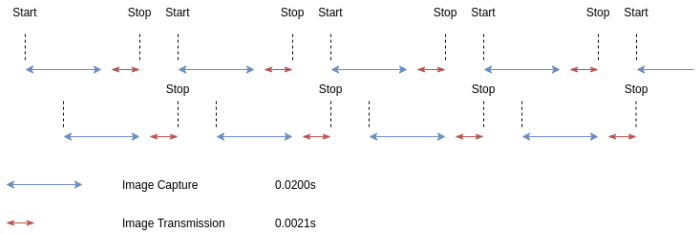


Figure 2.
Image capture timing diagram based on datasheet values for the SC20MPB high baud rate camera

Figure (2) indicates that the maximum frame rate of each camera when transmission and hardware requirements are taken into account totals to 49.9475 frames per second. The camera control firmware offsets the image capture command of the second camera as indicated to give a theoretical maximum frame rate of 99.8950 frames per second. A camera with a slightly faster sensor would be able to match the required 100 frames per second perfectly, but the error is almost negligible and sensor speeds are not available on a continuous spectrum.

The camera lenses were selected to give each camera a view of only the lane over which it has been installed. The camera height was selected at approximately 2.5 meters above the ground. This required a lens with a focal length of at least 5mm if the two cameras are placed behind each other. A 2.1mm lens was selected to capture one lane in it's entirety while also capturing sections of the adjacent lanes. The standard infra-red cut-off filter of 650nm was selected to prevent infra-red interference on a sunny day.

The firmware of the SC20MPB high baud rate serial camera was not perfectly documented. Upon reception of the camera a serial analysis tool showed that the test software supplied with the camera used a series of undocumented commands. One of the more useful extra command allows the camera to take six successive images at a predetermined frame rate and store them to local memory. This was later incorporated into the camera control firmware. Subsequent plans to decrease the amount of data to transmit by only requesting the data from the camera that contains the encoded image data was not successful due to limitations in the firmware. This technique would have saved the redundant transmission of 664 bytes of data for each image. This redundant data includes the standard and unchanging quantization and Huffman tables described earlier. More details are given in the JPEG decoding section. This as well as the undocumented command would later proof useful when the assumptions made of the size of the images and the encoding speed of the camera proofed optimistic. Communication was opened with the manufacturer about sharing the firmware for modification but they were reluctant and the attempt was ultimately unsuccessful. This may still be possible if the project is to be released commercially.

The power supply to the cameras will be documented in section detailing the layout of the cameras module's PCB.

3.1.2 Micro-controller Selection

The micro-controller and development board was selected based on adherence to the specifications listed below.

1. At least two serial interfaces with a minimum speed of 921600 BAUD for the cameras.
2. At least two high speed (48MB/s) SPI busses for external memory.
3. At least one high speed (400kB/s) I2C bus for configuration memory.
4. A built in Ethernet controller with speeds of at least 100MB/s.
5. A 32-bit central processing unit (CPU).
6. An included floating point unit (FPU).
7. An integrated debugger.

STMicroelectronics is the current industry leader in designing and manufacturing high performance 32-bit micro-controllers. Most of their development boards meet all the above requirements. Further differentiation was achieved by determining what clock speeds were required to meet the specifications. This was roughly calculated using the requirement that all processing should be done within a minute of the last athlete crossing the finish line, and that there would be 500 images to analyse if the cameras were to run for five seconds. It was known from the start that JPEG to pixel decoding would be necessary. It was also known that some type of looping algorithm will be implemented to analyse the image. The JPEG algorithm is projected to loop through the image data at least 5 times while performing the decoding function. The clock rate for this operation can then be estimated assuming image lengths of 1966 bytes and a required completion time of sixty seconds. The calculation is given in equation (3.2). The symbol n_i gives the number of images to analyse, n_l gives the number of loops through the data, d_l gives the data length of each image, i_{ml} gives the clock cycles for a given instruction and i_{pb} gives the number of ARM instructions to execute on each value.

$$\begin{aligned} f &= \frac{n_i(n_l \times d_l \times i_{ml} \times i_{pb})}{t} \\ f &= \frac{500(5 \times 1966 \times 5 \times 64)}{60} \\ f &= 26213333.33Hz \\ f &= 26MHz \end{aligned} \tag{3.2}$$

Analysing the pixels to determine whether an athlete has crossed the finish line will require looping through at most all the pixels in 500 images. The calculation was done assuming a minimum of two loops with 64 instructions per value in the loop. The resolution used will be

160×120 as determined in the previous section for maximum frame rate. The required clock speed is calculated in equation (3.3).

$$\begin{aligned} f &= \frac{n_i(n_l \times d_l \times i_{ml} \times i_{pb})}{t} \\ f &= \frac{500(2 \times 19200 \times 5 \times 64)}{60} \\ f &= 102400000 \\ f &= 102.4MHz \end{aligned} \tag{3.3}$$

The projected required clock speed based on equation (3.2) and (3.3) is 128.4MHz. These two algorithms will however only be the two main users of CPU time. Many other smaller algorithms will need to run constantly. A 50% safety factor is applied under the knowledge that the calculated clock speed is only a very rough estimation and that the system would need to run a few extra less computationally intensive algorithms as well. A micro-controller with a clock speed of at least 192.6MHz was required. The STM32F7 range of micro-controllers was a perfect fit, with a maximum clock speed of 216MHz. A development board in this series based on the STM32F767ZI micro-controller was selected for its easy to use input and output connectors. These will later be exploited to securely connect the cameras and all other peripheral components.

3.1.3 Memory Design

The average image sizes were given in table (1), where the length of the highest quality image was used with a safety factor of 20%. The same length of 1966 bytes was increased to a round 1024 bytes for all subsequent calculations. A conservative image capture time of five seconds was used based on the absolute maximum time an athlete will be in the view of the camera. The images were stored in their JPEG format to save on memory requirements and processing while capturing at the full frame rate. Storing the images in RAW pixel format would have required 19200 bytes (1875% more). Each camera was equipped with its own memory and controller to minimise the memory synchronisation that would have been needed if both cameras tried to write or read data at the same time. The required memory in bits was calculated using equitation (3.4). The symbol "n" represents the number of images that needs to be stored, while the symbol "s" represents the size of an image. The constant value of eight was used to convert from bytes to bits.

$$\begin{aligned} b &= n \times s \times 8 \\ b &= 250 \times 2048 \times 8 \\ b &= 4096000 \end{aligned} \tag{3.4}$$

Each camera thus requires approximately 4M-Bit of memory to store the low resolution images of the finish line. More memory was still required to properly index the stored images for later retrieval. There is no use in storing all the images if they can’t be easily retrieved. A

very basic version of a file system had thus to be created. This was done by setting aside a section of the memory to store the start address of the image, the end address of the image as well as the time at which it was taken. The indexing data is visualised in figure (3). The extra data required by the indexing totals another 96 bytes or 768 bits. The total required memory in bits is given in equation (3.5).

$$\begin{aligned} b &= 4096000 + 768(250) \\ b &= 42888000 \end{aligned} \tag{3.5}$$

While secure digital (SD) cards are available as an fast removable storage medium, it still requires the use of a full computer file system with all the software overhead that entails to use properly. It was thus decided to design a memory system with only a very basic file system. Most IC flash memory is only available as various surface mount components. Surface mount components will however needlessly complicate the manufacturing process. Through hole components were thus selected as an additional core requirement. A memory chip from Winbond was selected with the following specifications.

- Model: Winbond W25Q32JV
- Size: 32M-Bit (4194304 bytes, 16384 pages @ 256 bytes each)
- Interface: SPI
- Speed: 66MB/s

The maximum data-rate will however be limited by the micro-controller where the maximum rate is 48MB/s. This is however still more than enough to store the required images at the required speed. The memory requirement was however multiplied by four to allow the system to save and store images from multiple events before overwriting is required. This also proved useful later on when the image size was significantly increased.

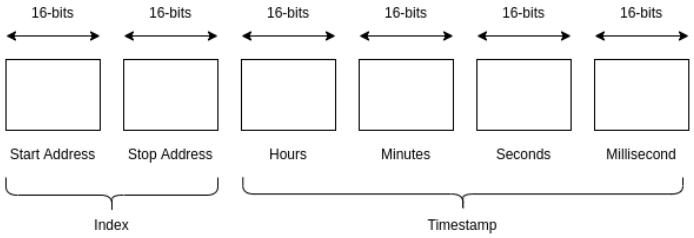


Figure 3.
A diagram representing how the image data is indexed in the memory

On a standard SPI bus with multiple connected devices, two pins are required for bidirectional data transfer (MISO & MOSI), one pin is required for the shared clock signal (CLK) and one pin is used to indicate which device is selected (CS). Four memory elements will thus require four device select wires. This can however be reduced to only two by using a decoder

IC. The decoder will also ensure that no two select pins can be activated at the same time. Pull-down resistors were connected to the chip select pins to ensure no element will remain active after it has been deselected. Bypass capacitors of $0.1\mu F$ were added to all memory elements as well as the decoder to protect against instantaneous changes in current flow as well as high frequency noise introduced by the SPI bus. The memory elements as well as the decoder operates with voltage levels between 3.0V and 3.6V. The camera module however operates at 5.0V. A 3.3V regulator was thus installed. The maximum current drawn by each circuit element is tabulated in table (2) below.

Component	Current (mA)	Amount	Total current (mA)
Memory Element	25	4	100
Decoder	1	1	1
Total			101

Table 2.
Maximum current draw of individual components of the memory card

This current value can now be used to find a suitable voltage regulator. Where possible the need for heat-sinks should be avoided. The maximum power to dissipate is given in equation (3.6), where the voltage driving the board is $V_I = 5.0V$. The current is taken as $I = 101mA$.

$$\begin{aligned} P &= (V_O - V_I) \times I \\ P &= (5.0 - 3.3) \times 0.101 \\ P &= 171.7mW \end{aligned} \tag{3.6}$$

The regulator should thus be capable of dissipating 171.7mW of power to the environment through only its package. The minimum required junction to ambient thermal resistance can then be calculated using equation (3.7). An ambient temperature of 30° C is assumed based on a hot day on an athletics track. The maximum junction temperature of package is assumed to be no higher than $125°C$.

$$\begin{aligned} R_\theta &= \frac{T_J - T_A}{P} \\ R_\theta &= \frac{125 - 30}{0.171} \\ R_\theta &= 409.357°C/W \end{aligned} \tag{3.7}$$

The L78L from STMicroelectrincs provides a low cost voltage regulator that meets the required current and heat dissipating specifications. The relevant parameters are listed below.

- Model: ST-L78L
- Output Current 100mA

- $R_\theta = 200°C/W$

The maximum allowable current does not perfectly match, but in no normal operational instance will a condition exist where all four memory elements draw their maximum power at the same time. The schematic of the memory circuit can now be shown in figure (4). A JST-1X7 connector was included to connect the memory board to the main camera board in the camera module.

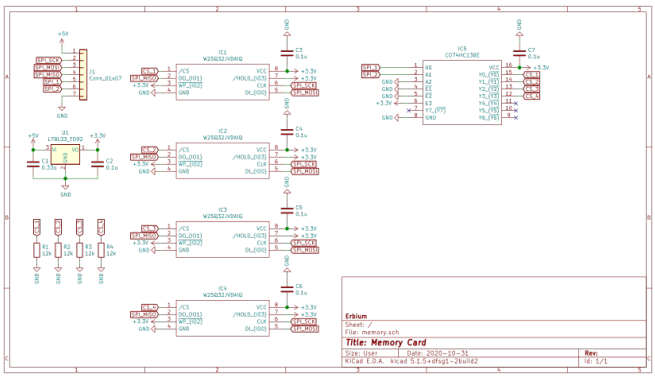


Figure 4.
Memory card schematic

The next step was to transfer the schematic of the memory card to a PCB. This step was conducted using KiCad. The top copper layer of the PCB was set to the 3.3V node, while the bottom copper layer was set to the ground node. This was done to ease the routing process as well as to decrease the inductance present in the traces. All bypass capacitors were placed close to the power pins of their respective components. A trace width of 0.5mm was used to increase the conductivity as far as the size constraints of the board allowed. A final 3D render of the PCB is given in figure (5).

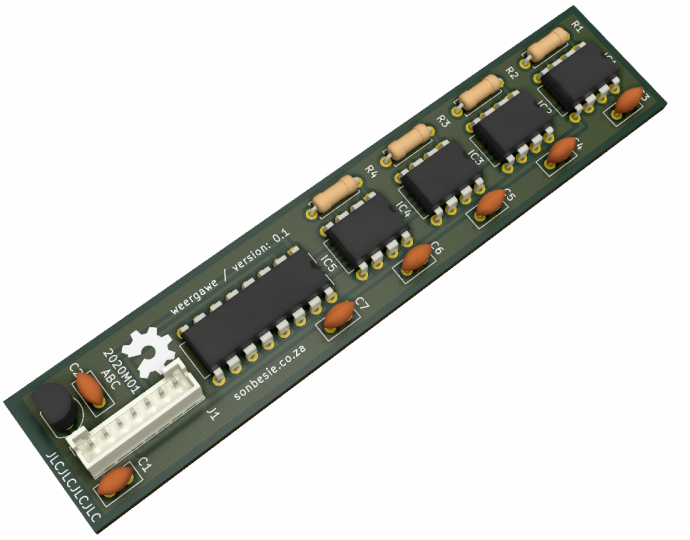


Figure 5.
Three dimensional render of the memory printed circuit board

The final bill of materials required to assemble the memory card is tabulated in table (3).

Component	Cost (R)	Amount	Total cost (R)
W25Q32JVDAIQ (Memory IC)	13.27	4	53.08
SN74HC138AN (Decoder IC)	7.35	1	7.35
B7B-PH-K-S (JST Connector)	6.67	1	6.67
ST-L78L (voltage regulator)	4.63	1	4.63
Resistors	2.457	4	9.828
Capacitors	3.298	7	23.086
Printed circuit board	31.18	1	31.18
Total			135.924

Table 3.
Memory card bill of materials

Two of these are needed per camera board bringing the final memory cost to R271.848. The memory element may also only need to be partially filled if less extra storage space is desired. Given the low cost of the memory it is probably wise to have a bit too much than too little.

3.1.4 Starting Pistol

This section describes the design of the circuit which determines whether the starting pistol has been fired. Starting guns can be classified into two distinct groups. The first is the traditional starting pistols that fires a blank to start a race. These contain a microphone positioned above the end of the barrel. Once the sound has been picked up it is transmitted to an amplifier and is then used in some circumstances to drive a timing system or loudspeakers.

The loudspeakers, if placed correctly, will decrease the delay between the gun being fired and the athletes/spectators hearing it brought about by the finite speed of sound. The second type of starting pistol is a fully electronic system. When the trigger gets pulled, the pistol emits a simulated sound that gets amplified and broadcast through loudspeakers to athletes and spectators. No blank is ever fired. This pulse can also be used to start a timing system. The fully electronic system was designed because the traditional staring gun got banned in some countries and regions, most notably the European Union. This was done because these pistols could in some cases be modified to fire real bullets. The timing system would however need to work with both of these systems. The input voltage and duration of the input pulse is thus unknown, given the many different starting systems and amplification levels available. The system should thus be configurable to work with all of them. A conceptual diagram of the starting system is given in figure (6).

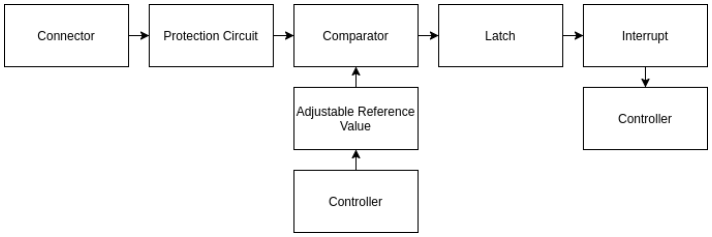


Figure 6.
Conceptual starting system diagram

A standard 3.5mm headphone jack was selected as the input for the trigger system. This was chosen as it is one of the standard audio cable connectors, with many converters available. A tree-pole 3.5mm jack was selected, where the right channel was left unconnected. The signal pulse in the left channel has an unknown voltage. To protect the other electronics on the board a Zener diode was added between the left channel and ground. The breakdown voltage of the diode was selected based on the maximum voltage the pins of the comparator circuit could withstand or the maximum reference voltage the board was capable of producing. The reference voltage requirement was selected because a higher comparator input voltage without a suitable voltage to compare it to would be redundant. The next problem to solve involved generating the reference signal for the compactor. This signal was used to set the sensitivity of the trigger circuit. In essence the input signal from the starting gun would need to be larger than this value to trigger the comparator. The digital to analogue converter (DAC) on the micro-controller has an theoretical maximum output voltage of 3.3V, based on it's internal logic level. The camera board would however be operating on 5.0V. A non inverting amplifier, given in figure (7), was then designed using an operational amplifier to amplify the voltage from the DAC to a 0 to 5V range.

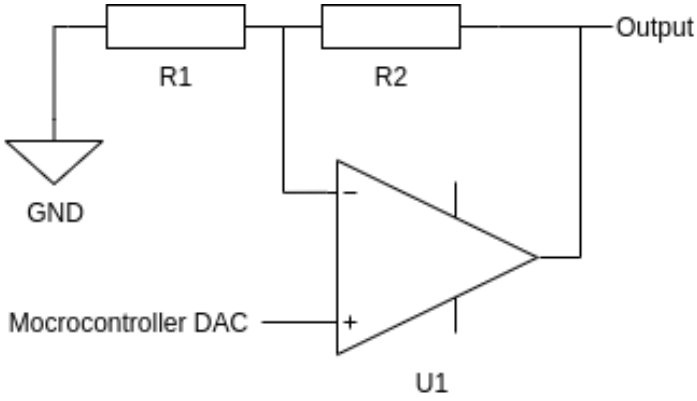


Figure 7.
Voltage amplifier example circuit

The values were selected to ensure a 5.0V output for a 3.3V input and a 0V output for a 0V input. All intermediate values should be linear. The standard non-inverting amplifier design gave values of $R_1 = 10K\Omega$ and $R_2 = 5.1K\Omega$. Bandwidth is not of great concern for the amplifier as the reference value will be set and left. The power consumption, 5V maximum output and rail to rail operation was however of great importance. The chosen component was thus the MCP6004 from Microchip. The IC featured the following important specifications.

- Model: Microchip MCP6004
- Rail-to-rail input and output
- Maximum supply voltage: 6V
- Low quiescent current of $100\mu A$
- Multiple package options including through hole

The next step was the selection of the comparator. A high speed compactor is required that introduces negligible delay to the timing system. The system needs to determine the athletes time when they cross the finish line with an accuracy of 10ms. The trigger system should thus aim to not introduce more than a 1% error over the error of the finish line to keep the finish line error dominant. The comparator should thus have a propagation delay of no more than $100\mu s$, but preferably less. The output of the comparator was then used as the input to a digital latching circuit. The micro-controller needed to be able to control the set reset lines of the lath. This will enable the micro-controller to arm the latch when the race is started and reset it after the athletes have started. Although the interrupt pins of the micro-controller are 5V tolerant, it will still be more elegant if the lath outputted a logic 3.3V instead of the 5V from which it is powered. A component that integrated both the comparator and latch was found that met the required specifications. The important specifications of the AD790 from Analog is given below.

- Model: Analog Devices AD790

- 45ns Maximum propagation delay
- Maximum single supply voltage 5V
- Quiescent current of 12mA
- Built in latch with selectable output voltage
- Multiple package options

The only drawback with this IC is the relatively large quiescent current of 12mA, when compared to the reference voltage amplifier. Pull-down resistors were installed at the 3.5mm input jacks left channel, the latch output and the set reset pin of the latch. Both the amplifier and the comparator needed an individual bypass capacitor of $0.1\mu F$. The board layout of the trigger circuit will be detailed in the next section with the camera switches and power supply.

3.1.5 Main board and plastic housing

The main board of the camera module needs to connect to the micro-controller module. The micro-controller development board provides a female board-to-board interconnect in the form of four banks of vertical header connectors with a pin pitch of 2.54mm. The main camera board was thus designed to directly interface with these connectors. The diagram in figure (8) gives a representation of where the board-to-board interconnect was located.

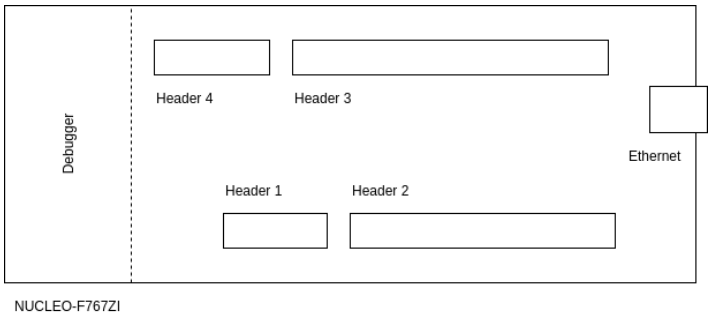


Figure 8.
Nucleo development board connection layout

The main camera board was selected to be the same size as the development board to enable a more space efficient and simple plastic housing. The first circuit that needed to be added was the electronic power switches of the cameras. Switches are needed to reset the cameras in case of an error and to conserve power when they are not in use. Electronically controlled switches are usually designed using transistors (BJT / MOSFET) or mechanical relays in combination with transistors. The maximum power consumption of a single camera is 170mA, and is thus the current that needs to be switched. Relays with coils switched by low power MOSFET's were chosen for the following reasons.

- Very low on resistance (compared to transistors)

- Audible sound to indicate camera is in the switched on state.
- Isolates the camera power pin from the rest of the board in the off position.

The relays had to have a low coil current and be able to switch the required current of 170mA at a voltage of 5V. The cheapest relay that matched the requirements were the OMRON G5LE-1 relay. Its specifications are listed below.

- Model: OMRON G5LE-1
- Coil voltage: 5V
- Coil current: 80mA
- Must operate voltage: 75%
- Must release voltage: 10%

An appropriate MOSFET can now be chosen to control the relay. The gate of the relay was driven using 3.3V as it is the logic level of the micro-controller. The BS170 MOSFET was chosen from ON Semiconductor with the following specifications.

- Model: BS170
- Maximum drain source voltage: 60V
- Maximum drain current 500mA
- Package T0-92

A status light emitting diode (LED) with a current limiting resistor of 270Ω was added. The maximum current through the LED is thus selected at about 20mA. The next components that were added to the main board was a user status indicator. This was added in addition to the camera status LED's to give feedback to the user regarding it's current internal state and possible error codes. The status indicator was chosen to be a simple seven segment display with a decoder driver. The decoders four inputs represented sixteen distinct states on the screen. The decoder and the screen operated on 5.0V with the decoder being sensitive to 3.3V logic level inputs from the micro-controller.

The external trigger circuit was next placed on the board. The trigger output logic level was connected to the reference logic level of the micro-controller.

The main board required a way to store important configuration settings. These configuration settings included the network address of the device, and all previously applied settings. The 24LC02 eeprom IC from Microchip was used for this purpose. The I^2C interface required only two data lines on the PCB. It was also very power efficient requiring only 1mA when active. The IC was placed in a socket to allow configuration swaps between devices. As per the I^2C standard $2k\Omega$ pull-up resistors were added to achieve an communication frequency of

400kHz.

The final board schematic is given in figure (9). All discussed circuits were connected to one of four header pins that plugged into the board-to-board interconnect provided by the micro-controller development board. All external connections to cameras and memory were interfaced using JST-1X4 and JST-1X7 headers.

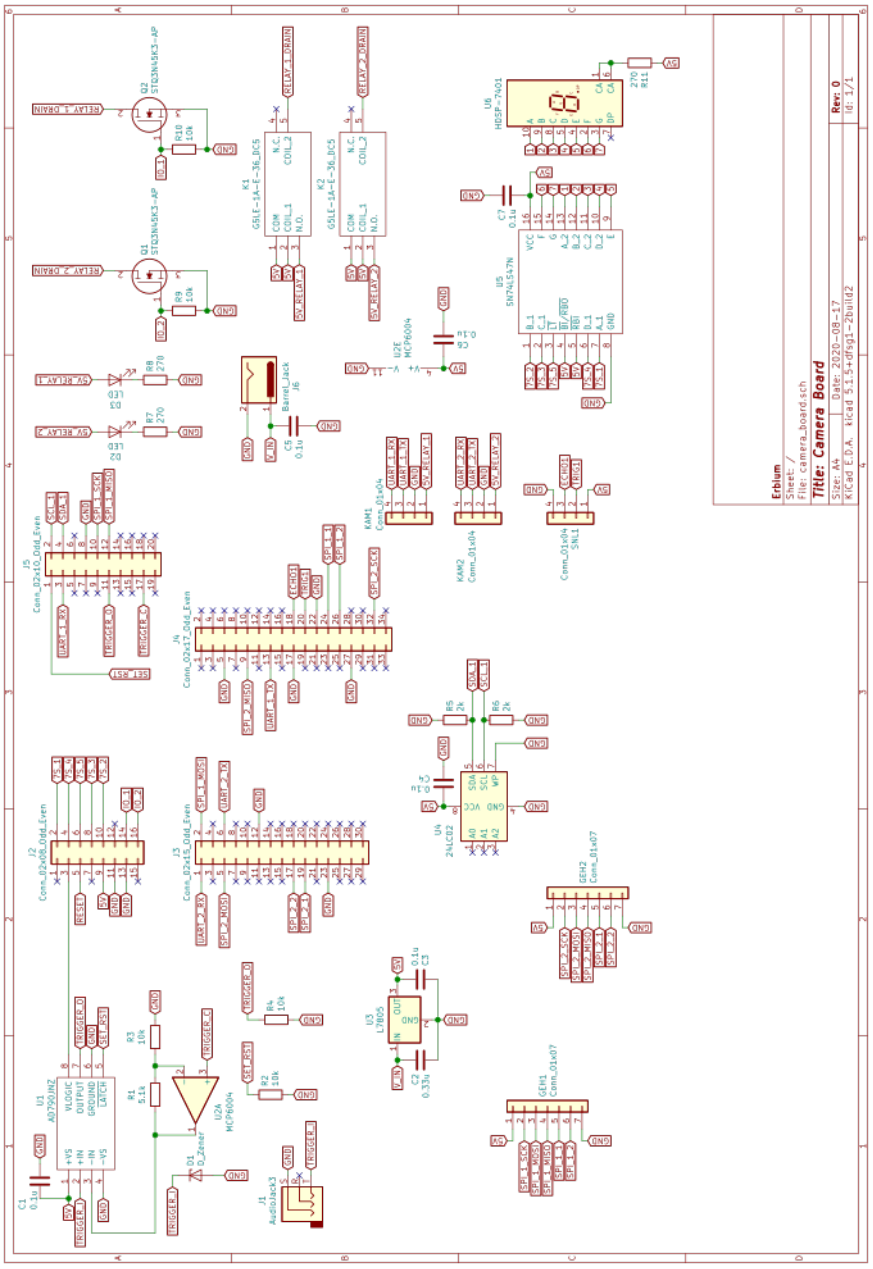


Figure 9.
Main board schematic

Standard barrel jacks were used to connect the external power adaptor to the board. A 5V voltage regulator was installed to condition the power powering the main camera board as well as the development board through the board-to-board interconnect. An variable input voltage between five and nine volts is desired. This allows a variety of wall power adaptors to power the camera module. The current required by all the major components is tabulated in the table (4) below.

Element	Current (mA)	Amount	Total current (mA)
Memory	101	2	202
Camera	170	2	340
Processor	500	1	500
Relay coil	80	2	80
LED's	20	2	40
Total			1242

Table 4.
Total camera module current draw

A 5V low drop-out regulator was thus required that could output a constant 1.25A at an input voltage of between 5V and 9V. The L7805CV from STMicroelectronics was chosen with the following specifications.

- Model: L7805CV
- Output Current: 1.5A
- Output Voltage: 5V

If the system is powered at its upper voltage input of $9V$ a heatsink will be required with a thermal resistance lower than given in equation (3.8). The heatsink should dissipate approximately 4.968W of power to the environment. The ambient temperature is assumed as before to be $30°C$ for the same reason as stated in previous calculations.

$$\begin{aligned} R_{th-SA} &= \frac{(T_J - T_A) - (P)(R_{th-JC} + R_{th-CS})}{P} \\ R_{th-SA} &= \frac{(95) - (4.968)(5+1)}{4.968} \\ R_{th-SA} &= 13.122°C/W \end{aligned} \tag{3.8}$$

Given that the thermal resistance requirement is not too strict, most heat sinks with reasonable thermal mass would be able to dissipate the required energy. A low thermal resistance between the regulator and the heatsink would help lower the requirement even further. This was done by coating the regulators back in thermal conductive paste and using a nut and bolt to ensure the regulator and heatsink is thoroughly pressed against each other.

The final three dimensional render of the completed printed circuit board is given in figure (10). The layout was done in KiCad with the top copper region being connected to the 5V voltage node and the bottom copper region being connected to the ground node. The JST connectors as well as the board-to-board interconnects are visible on the left and right hand side of the board.

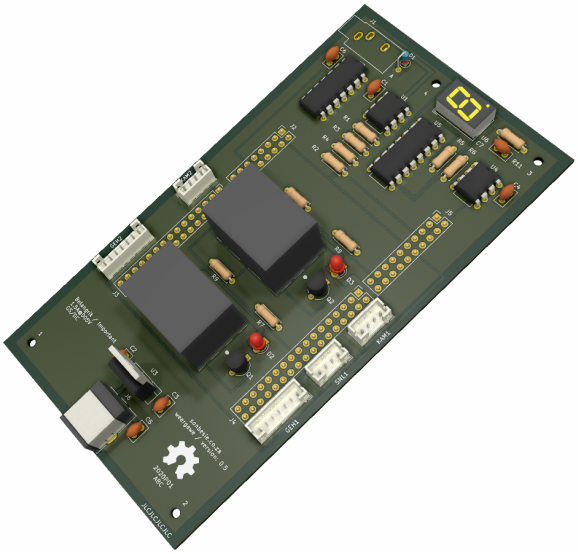


Figure 10.
Three dimensional render of the camera printed circuit board

The camera board was connected to the development board in inserted into a custom designed plastic housing. The final assembly of the hardware is given in figure (11).

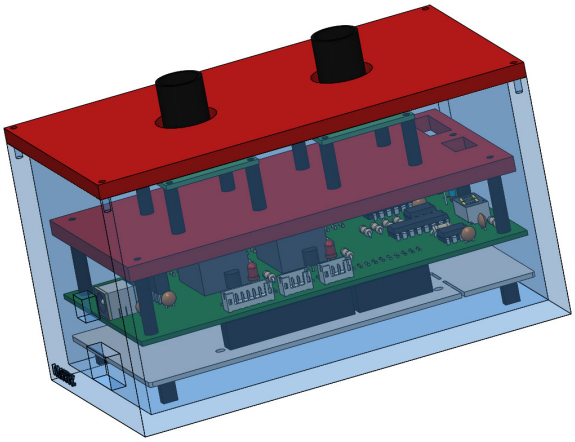


Figure 11.
Final hardware assembly

The plastic box was set to be 50% transparent for this visualisation. The bottom board is the

micro-controller development board. The board on top of it is the previously designed main camera module board. A plastic spacer was then inserted for the cameras to rest upon. A hole was left in this spacer to ensure the seven segment display is still visible. Cables between the main board and the cameras / memory are routed in the space between the spacer and the edges of the box. A 5mm clearance was left between all internal components and the inside edge of the box. A lid with two holes for the camera lenses was then added and held closed with four screws. All internal supports were glued. The final bill of materials of the camera module is given in table (5) below.

Component	Cost (R)	Amount	Total cost (R)
B7B-PH-K-S (JST Connector)	6.67	2	13.34
B4B-PH-K-S (JST Connector)	4.10	3	12.30
67996-130HLF (Header 1)	24.97	1	24.97
67996-134HLF (Header 2)	29.58	1	29.58
67996-116HLF (Header 3)	17.10	1	17.10
67996-220HLF (Header 4)	14.19	1	14.19
SJ1-3533NG (3.5mm Jack)	20.01	1	20.01
HDSP-7401 (Seven segment display)	57.11	1	57.11
L7805CV (Voltage regulator)	8.55	1	8.55
BS170 (Mosfets)	11.46	2	22.92
OMRON G5LE-1 (Relays)	27.30	2	54.6
Red Led's	8.02	2	16.04
PJ-037A (Barrel jack)	9.92	1	9.92
SN74LS47N (Decoder)	12.42	1	12.42
AD790 (Fast comparator)	320.18	1	320.18
MCP6004 (Voltage amplifier)	7.00	1	7.00
Resistors	2.46	11	27.06
Capacitors	3.30	7	23.10
Printed circuit board	31.18	1	31.18
Total			721.57

Table 5.
Camera module bill of materials

3.1.6 Firmware development

Custom firmware was developed to control all aspects of the camera board over a wired network connection. This section of the report will present a flow diagram in figure (12) of the full firmware layout and then go into deeper detail for the more interesting sections.

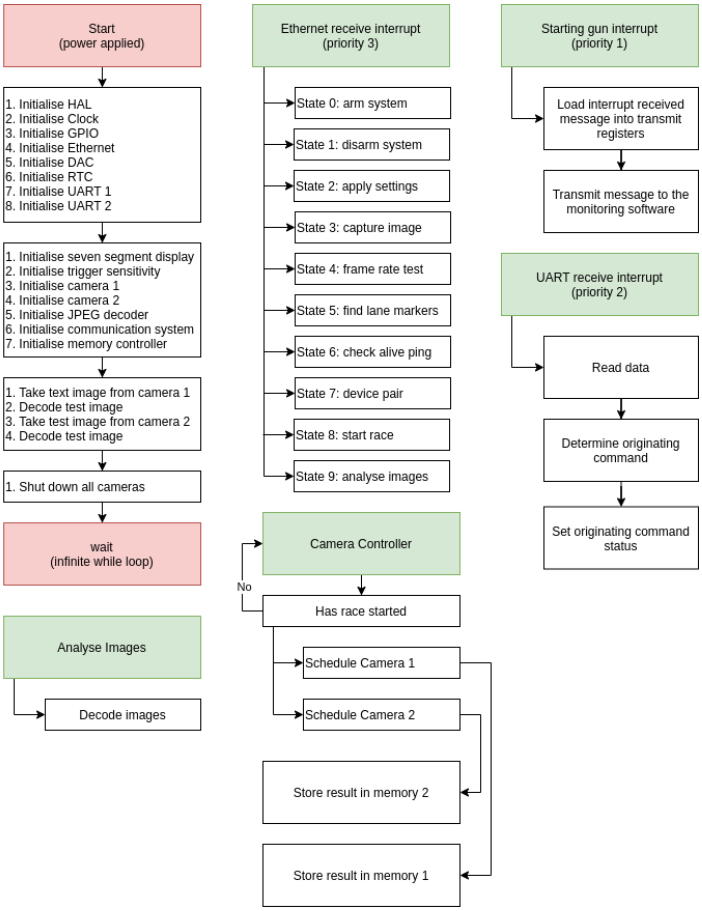


Figure 12.
Basic firmware design

As soon as the system receives power it enters it's initialisation section (red). Here the system initialises the manufacturer provided hardware application layer (HAL) drivers. This section was for the most part automatically generated by the integrated development environment with only slight alterations required. The system then initialises all the custom drivers for the self designed components. It starts by initialising the output of the seven segment display. It then sets the trigger sensitivity to the default value of 10/10 (5V reference voltage). Thereafter it turns the power on for the cameras and wait for their power up routine. The next section to be initialised is the JPEG decoder which decodes the data returned from the cameras to pixel values. The communication system is then initialised with a broadcast destination network address of FF:FF:FF:FF:FF:FF. The first time the device transmits a message it will communicate with every device on the network. The management computers network address will only be set later. The memory controllers are initialised last. A command is then transmitted to both cameras to capture an image where-after it is decoded as part of a self test. The system won't boot further if an error is encountered during the decoding section. The memory is also initialised in this step. The last step is to turn the cameras off again to save power.

The firmware for the communication system is based around a state machine implemented

in C language. The incoming and outgoing data takes the form given in figure (13). Data transmitted from the management computer to the camera module can be up to 64 bytes long. Data transmitted from the camera board to the management computer can have any length.

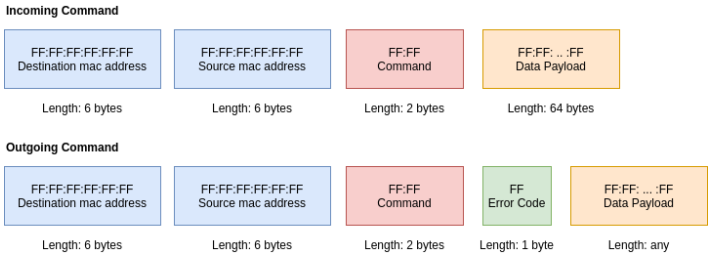


Figure 13.
Network protocol

The first six bytes in the incoming connection to the camera module is always the unique media access control (MAC) address of the destination device, which specifies exactly with which board the management computer is communicating. The next six bytes is the MAC address of the source of the transmission, which is usually the address of the management computer. This is followed by a two byte command that once decoded places the system into one of the ten defined states given in figure (12). The seven segment display installed on the camera board shows the number corresponding to the currently selected state. Once the state has finished executing the relevant instructions the outgoing command is generated by flipping the source and destination MAC addresses and echoing the command. Added to this is an error byte which contains an error code if the state did not execute successfully. If the command generated data it is appended after the error code. The MAC address of the management computer is saved when the camera board first receives the pair command in state seven. Thereafter the board uses the MAC address of the controller as destination address and it is considered paired. Added to the protocol is a keep alive ping that gets transmitted to the the camera module from the management computer at set time interval. The camera module answers this command by echoing the command back to the controller. This is done to ensure the connection is not lost without the user realising the problem.

The starting gun trigger firmware was implemented by resetting the latch on the comparator during initialisation. The default trigger sensitivity set during initialisation can be changed by the management computer though the sending of an update settings command. As soon as the command is received to arm the system the latch was removed from the reset state. A positive comparison on the comparator will then pull the latch output high, triggering the interrupt on the microcontroller. A command is then loaded into the transmission registers for transmission back to the management computer stating that the starting gun has been fired. Upon reception the management computer issues commands to all connected devices to turn on their cameras and detect athletes through the start race state. The management computer also has the option of disarming the system through state one, which resets all the actions taken in the arm state.

The management computer can place the system into state three where a picture can be taken and transmitted back. This can be done to ensure the cameras are correctly positioned over

the finish line. The system can also be placed into a frame rate test, where the built in RTC is used to constantly take pictures over a specified time frame. The number of successful images taken is then divided by the time frame and transmitted back to the management computer.

The previously mentioned settings state is where the camera module receives settings from the management computer and applies them to the relevant components. This state updates the camera resolution, baud rate, image quality, trigger sensitivity and the time and date on the RTC.

A fully custom network protocol was designed as described in figure (13) to save a significant amount of system resources when compared to implementing a full TCP/IP stack from a library. Most of the features associated with TCP/IP would not have been required even if it was implemented as the system did not need to connect to the internet. Only OSI level two switching on network switches was required.

3.1.7 JPEG decoding algorithm

The process that was used to decode the images produced by the SC20MPB cameras will be described here. At certain points explanations will be given on how the data was encoded to make the process more clear. Data in a JPEG image is encoded by the camera firmware using a DCT to transform the individual pixel values to their frequency components. The data is then passed through a quantisation algorithm where unneeded frequency information is discarded. The DCT is given in equation (3.9) and (3.13).

$$F(i,j) = \frac{1}{\sqrt{2N}} \sum_{x=0}^{N-1} \sum_{y=0}^{N-1} C(i)C(j)f(x,y) \cos\frac{(2x+1)i\pi}{2N} \cos\frac{(2y+1)j\pi}{2N} \tag{3.9}$$

$$C(u) = \begin{cases} \frac{1}{\sqrt{2}} & \text{if u = 0} \\ 1 & \text{if u > 0} \end{cases} \tag{3.10}$$

F(i, j) represents the DCT at the index of i and j. f(x, y) represents the pixel value at the index of x and y. The DCT was conducted on blocks of data with equal height and width, where N represents the size of the block.

The basic data structure of JPEG image is given in figure (14). Two byte long identifiers, which are indicated as yellow blocks, were encoded with the image data to identify what the next number of bytes represent. These were used by the decoder algorithm to determine where in the image file is the relevant data stored.

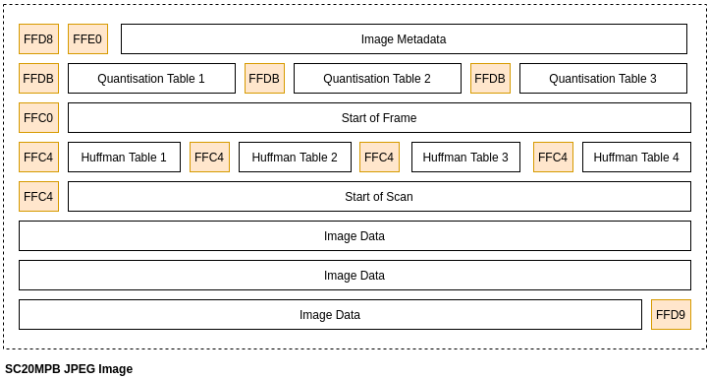


Figure 14.
JPEG Image data structure

The byte array starting with the hex characters FFEO indicates the start of the metadata section of the image. The next two bytes indicates the length of this section in bytes. The section then encodes the name of the encoding format, the version of the encoding format, the aspect ratio of the image and the size of the thumbnail. This data is just read out sequentially and stored in arrays. This information was mainly used to determine that the correct camera was connected for the decoding algorithm to work.

The next byte array starting with the hex field marker FFDB indicates the start of a quantization table of which there are three. These tables are used to set the quality of a image. During the encoding process groups of 8×8 pixels were processed by the DCT in equation (3.9). The cosine transform was performed separately on the luminance (Y) and two chrominance (Cb and Cr) tables of the images pixels. The conversion from RGB to luminance and chrominance values are given in equation (3.11).

$$\begin{aligned} Y &= 0.299R + 0.587G + 0.114B \\ C_b &= -0.1687R - 0.3313G + 0.5B + 128 \\ C_r &= 0.5R - 0.4187G - 0.0813B + 128 \end{aligned} \tag{3.11}$$

The result is a 8×8 grid with luminance/chrominance data as given in table (6). These values now represent the frequency components of the image, with the lowest frequency being in the top left corner and the highest frequency being in the bottom right corner. The photo can still be decoded with lossless quality from this table using the inverse discrete cosine transform (IDCT).

152.7	25.7	39.2	14.9	16.2	44.9	23.7	24.1
12.2	11.9	54.1	52.3	28.5	31.2	15.6	12.7
37.1	93.2	44.1	21.4	65.4	43.1	24.7	14.1
15.1	47.2	56.9	62.7	89.1	20.3	12.2	15.9
27.3	44.1	99.3	43.8	32.5	53.2	66.5	22.5
44.2	69.7	32.1	64.2	87.3	63.9	23.9	37.9
22.7	32.2	76.5	97.2	22.1	94.9	88.2	21.4
63.1	82.4	82.6	32.1	74.7	53.6	94.5	65.6

Table 6.
Frequency data pre-quantisation

The values in table (6) is then divided by the values in the corresponding cell in the quantization table given as table (7). When the management computer selects a different quality setting from the camera firmware this is the table that changes.

50	34	31	50	75	125	159	190
37	37	44	59	81	181	187	172
44	41	50	75	125	178	215	175
44	53	69	90	159	255	250	193
D56	69	115	175	212	255	255	240
75	109	172	200	253	255	255	255
153	200	243	255	255	255	255	255
225	255	255	255	255	255	255	255

Table 7.
Quantisation table

Depending on the algorithm the result is usually truncated to force an integer. The higher the value in the quantization table, the higher the chance is that the resulting value that will be stored is a zero. The data in that cell can then be regarded as lost. The tables are usually generated with higher values in the bottom right to reduce the higher frequency content in the image. The first quantization table applies to the luminance values of the image. The following two quantization tables applies to the chrominance values of the image.

The algorithm then moved to the marker representing the start of frame section. This section contained information relating to the resolution of the image. This information was however not needed, given that the decoder already knows the resolution as it was stored when the resolution of the camera was set through the management computer.

The decoder then had to construct four Huffman tables based on the information given after the FFC4 marker. The data provided first gives the number of codes of a specific length. It is then followed by the data the code represents. The length of the codes range from one to sixteen. A text example of the data is given below.

```
0 code/s of 1 bit/s:
1 code/s of 2 bit/s: 00
5 code/s of 3 bit/s: 01 02 03 04 05
1 code/s of 4 bit/s: 06
```

This information was then decoded to the Huffman tree given in figure (15). The codes shown in the circles denote the decoded data while the binary sequence leading to that circle denotes the encoded data.

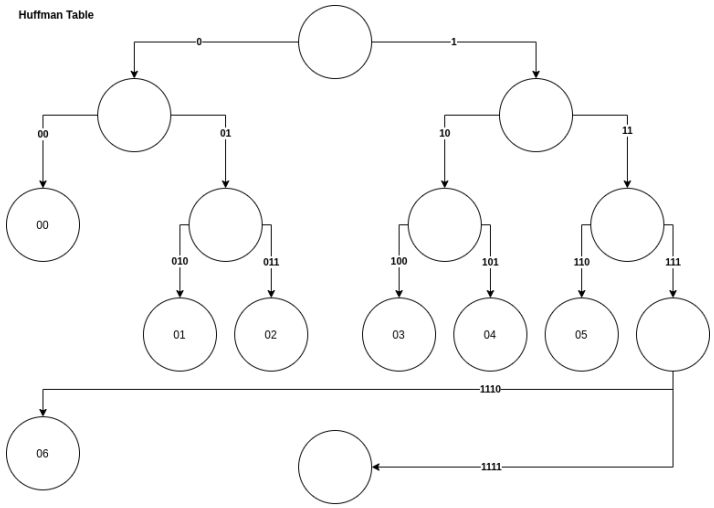


Figure 15.
Representation of a Huffman tree generated from the JPEG data

The algorithm starts by determining the number of available branches on each level of the Huffman tree that is left after the placement of a code stops that branch from growing. In the case of figure (15) level 1 has two branches. There are zero codes of length one bit, which means both are still available. Level two has four branches. There is however one code with a length of two bits, which occupies one of the branches, leaving only three remaining. Level three should have eight branches, but two branches are removed as a result of the code in level two which stops the branch from growing, leaving six left. There are five codes of three bits, leaving only one branch left on the third level to continue the tree. Level four should have sixteen branches. Fourteen of those branches should however be subtracted as a result of the previous six symbols, leaving only two. There is one code of length four bits leaving one unused code. This algorithm generates two arrays. The encoded array contains the binary value with which the value at the corresponding index in the decoded array has been encoded. When searching through the encoded data, a binary two in the encoded array would correspond to a hex value of 01 in the decoded array.

```
encoded = [0, 2, 3, 4, 5, 6, 14]
decoded = ['00', '01', '02', '03', '04', '05', '06']
```

The four available Huffman trees are used to encode/decode different sections of the image. Huffman table one is used to encode/decode only the zero frequency element of the luminance

data passed through the DCT in table (7). Huffman table two is used to encode/decode all the other frequency components of the luminance data. Huffman table three is used to encode/decode only the zero frequency data of the both chrominance tables. Huffman table four is used to encode/decode the rest of the frequency components present in both chrominance tables.

The start of scan is the next important section denoted by the FFC4 marker. This section provides the data regarding how many colour components the image contains. This information is assumed to be constant and ignored. The encoded data starts immediately after the start of scan field. The first Huffman table is selected, given that the first value will be the zero frequency component of the luminance table. The data is then read bit by bit and compared with the data in the encoded array presented earlier. The corresponding decoded value is read as soon as a match is found. This value gives the length of data to read that will represent the first zero frequency value of the decoded luminance table. Huffman table two is then selected and the process is repeated for the rest of the frequency components in the luminance table. An diagrammatic representation of the process is given in figure (16).

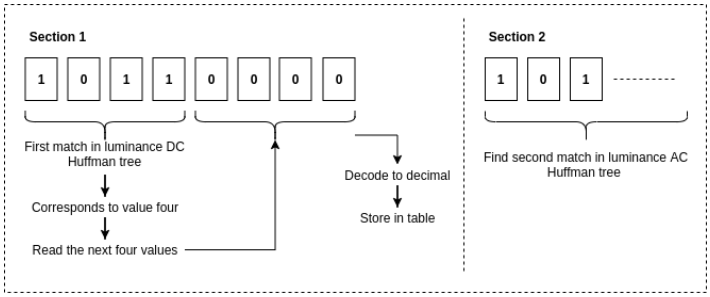


Figure 16.
JPEG decoding example with the first zero frequency value

This decoding process is stopped as soon as the last last field marker of FFD8 is located. The internal memory of the micro-controller now contains multiple 8×8 tables of frequency data representing the luminance of pixels. These tables were then multiplied by the quantization table given in table (7). The approximate frequency values have now been recovered. The next step in the decoding process is to recover the original pixel values by running the IDCT on every table. The IDCT at a row and column position of x and y respectively is given by equation (3.12).

$$f(x,y) = \frac{2}{N} \sum_{m=0}^{N-1} \sum_{n=0}^{N-1} C(m)C(n)F(m,n) \cos \frac{(2x+1)m\pi}{2N} \cos \frac{(2y+1)n\pi}{2N} \tag{3.12}$$

The variable N indicates the number of rows and columns that needs to be transformed. The function C(u) is given by equation (3.13). The function F(m, n) returns the value in the decoded frequency matrix at the given row and column position. The equation ends with two cosine terms which are a function of the provided output position, the current position in the summation and the total number of summations to be completed.

$$C(u) = \begin{cases} \frac{1}{\sqrt{2}} & \text{if u = 0} \\ 1 & \text{if u > 0} \end{cases} \tag{3.13}$$

This equation can easily be ported to C code. This will however be unwise from a performance point of view as it will require four nested for loops each with a length of eight. The more computationally elegant approach was to convert equation (3.9) to matrix form. The matrix form is given in equation (3.14).

$$T_{ij} = \begin{cases} \frac{1}{\sqrt{N}} & \text{if i = 0} \\ \sqrt{\frac{2}{N}} \cos \frac{(2j+1)i\pi}{2N} & \text{if i > 0} \end{cases} \tag{3.14}$$

Equation (3.14) was then used to generate a matrix that allows the IDCT to be calculated. The resulting matrix is given in table (8).

0.3536	0.3536	0.3536	0.3536	0.3536	0.3536	0.3536	0.3536
0.4904	0.4157	0.2778	0.0975	-0.0975	-0.2778	-0.4157	-0.4904
0.4619	0.1913	-0.1913	-0.4619	-0.4619	-0.1913	0.1913	0.4619
0.4157	-0.0975	-0.4904	-0.2778	0.2778	0.4904	0.0975	-0.4157
0.3536	-0.3536	-0.3536	0.3536	0.3536	-0.3536	-0.3536	0.3536
0.2278	-0.4904	0.0975	0.4157	-0.4157	-0.0975	0.4904	-0.2778
0.1913	-0.4619	0.4619	-0.1913	-0.1913	0.4619	-0.4619	0.1913
0.0975	-0.2778	0.4157	-0.4904	0.4904	-0.4157	0.2778	-0.0975

Table 8.
Matrix representation of the discrete cosine transform

An inverse representation of matrix (8) is then required to account for both dimensions in the IDCT. The inverse is given in table (9).

0.3585	0.4973	0.4685	0.4216	0.3585	0.2817	0.1940	0.0989
0.3447	0.4035	0.1799	-0.1079	-0.3623	-0.4973	-0.4668	-0.2802
0.3553	0.2802	-0.1891	-0.4884	-0.3518	0.0989	0.4629	0.4162
0.3610	0.1079	-0.4523	-0.2691	0.3610	0.4216	-0.1873	-0.4884
0.3461	-0.1079	-0.4717	0.2691	0.3461	-0.4216	-0.1954	0.4884
0.3518	-0.2802	-0.1936	0.4884	-0.3553	-0.0989	0.4611	-0.4162
0.3623	-0.4035	0.2028	0.1079	-0.3447	0.4973	-0.4572	0.2802
0.3485	-0.4973	0.4555	-0.4216	0.3485	-0.2817	0.1886	-0.0989

Table 9.
Inverse matrix representation of the discrete cosine transform

Both matrix (8) and (9) was stored in the internal memory of the micro-controller. When

the pixel values are required the matrix multiplication operation given in equation (3.15) is executed. Matrix T is given in table (8), while matrix T' is given in table (9). The matrix N represents the recovered pixel grid.

$$N = truncate(T' \times R \times T) + 128 \tag{3.15}$$

A truncate operation is performed on each element in the result array and a value of 128 is added to undo an optimisation done during encoding.

3.1.8 Timing system

The timing system was implemented by first broadcasting the current time from the management interface to all the camera modules connected to the network. The real time clocks in all the camera modules would thus be set to the same time. This time is saved when the starting pistol is fired. This is only done on the one module connected to the starting gun. The cameras are then started and instructed to start capturing the finish line, but to not yet save the data. The length of each JPEG image is a function of the detail it captured. More detail would result in a larger image. The lengths of these initial images are then stored in a 16 integer long shift register. After the register starts overflowing it's standard deviation is calculated. If the absolute difference between the newest length and the average length of the register is more than five standard deviations a motion detection event is triggered and the length is not added to the shift register. The system then takes 12 images at the maximum instantaneous frame rate using first camera one and then camera two. The time at which each image is taken is then saved with the image. This time is accurate to the millisecond. The difference between the time saved with the image selected by the thorax identification algorithm and the starting time is then used to calculate the total race time in the management interface. The on-board crystal oscillator driving the real time clock has a quality rating of less than 100ppm. For short races at most ten minutes will pass from the race being started until everyone has finished. This equates to about 600 seconds. At the given quality specifications the timer could lose up to 60ms seconds over that time period. Using a watch crystal (20ppm) and a carefully designed printed circuit board could reduce this time to less than 12ms. The time between the management interface and the camera modules should thus be updated before each race.

3.1.9 Line identification

One of the important requirements is for the system to accurately detect the lane boundaries of an athletics track. This is needed to ensure the system is able to determine when a person has crossed the finish line. A significant computational optimisation was found where the table of frequency values generated halfway through the JPEG decoding algorithm can be used to search for a line without having to fully decode it to pixel values. During the initial DCT, the image is divided into blocks where each block represents a grid of 8×8 pixels. The brightness of each block is encoded in the zero frequency component in the upper left hand corner with the coordinates (0, 0). The brightness of each block can then be compared to the one next to it. Large differences in brightness as caused by white lines on mono-colour

backgrounds can easily be spotted. A simulation was run on a photo taken from one of the two cameras of the system. The focus of the camera was set to keep the top third of the image in focus, which was necessary to identify an athlete's upper body. The grass and the finish line is therefore not fully in focus. The image is given in figure (17).

Figure 17.
Image of an athletics track with the finish line on the left hand edge

The image was then decoded using the Python implementation of the above mentioned image decoding algorithm. The zero frequency values were then extracted starting in the top left corner of the image running in a horizontal line to the right hand edge. This was repeated for each available row. The resulting zero frequency values were plotted using Matplotlib and is given in figure (18).

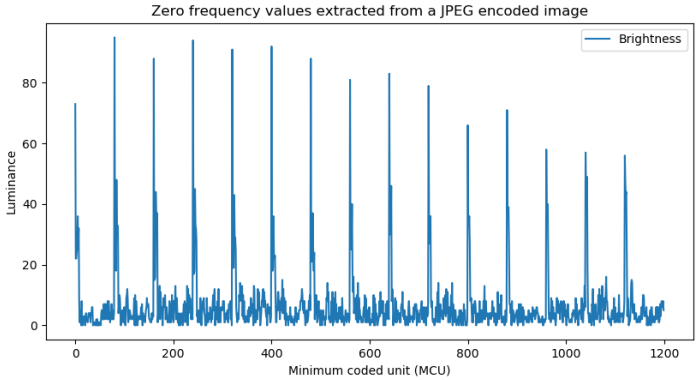


Figure 18.
Zero frequency values extracted from a JPEG encoded image

Figure (18) shows a large increase in brightness at the start of each new horizontal line. A Python program was then written to determine the location, with a x and y coordinate, of each of the points in the graph with a significantly higher brightness value than the previous point. The result of the algorithm is plotted in figure (19) over the raw brightness data given in figure (18).

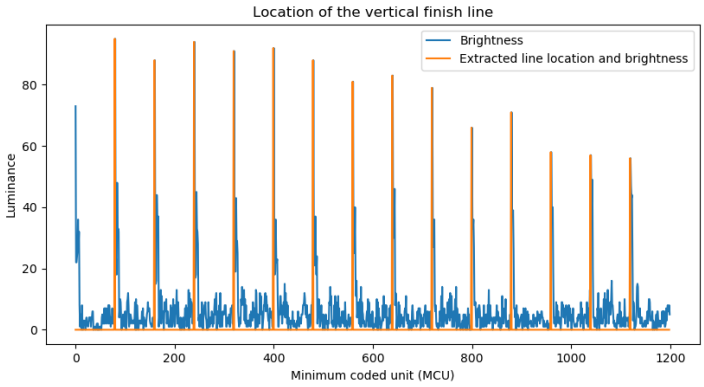


Figure 19.
Location of the vertical finish line

This algorithm was however dependant on the resolution of the JPEG image. A lower resolution image would contain less 8×8 encoding blocks, which would lower the accuracy of the system. A decision was thus made to move the system to a template matching algorithm. In this type of algorithm the system searches through an image to find the best match for an external template. This template is usually much smaller than the original image. The result is a grid, with the same resolution as the image, containing integers corresponding to the difference between the template and the image at each position. A low value indicates a better match (smaller difference) and a higher value indicating a worse match (larger difference). The mathematical expression that describes the template matching algorithm is given in equation (3.16). The variables a and b signifies the number of columns and rows in the template respectively. The variables x and y signifies the number of columns and rows in the main image respectively. The function $T(a,b)$ returns the pixel values of the template at the given coordinates, while $I(x,y)$ does the same for the main image. The function $R(x,y)$ stores the result at the provided coordinates.

$$R(x,y) = \Sigma_{a,b}(T(a,b) - I(x+a,y+b))^2 \tag{3.16}$$

The equation thus represents an operation where a loop goes through all the pixels in the main image. The equation then enters a second loop where it loops through all the pixels in the template. While looping through the template it takes the squared difference between each pixel in the main image, offset by the row and column of the template, and the template. In essence all the squared differences in the second loop is added and stored at the location defined by the function $R(x,y)$. To initially experiment with this function it's Python implementation was used as implemented in the OpenCV library. A web-cam was fixed at a height of 2.5m above a simulated athletics track. The simulated athletics track was constructed using lane boundaries with a width of 0.2m. The centres of the two lane boundaries were 1.27m apart. A slightly wider lane boundary was used to help with the initial development of the algorithm. Two squares consisting of a smaller red square pasted on top of a larger blue square was used as the template. It would have a light boundary and a dark center in the Cb colour space while the opposite will be true in the Cr colour space. The template in rgb and both of the

chromanance components are given in figure (20). The left image gives the RGB template, the center image the Cb template while the right gives the Cr template.

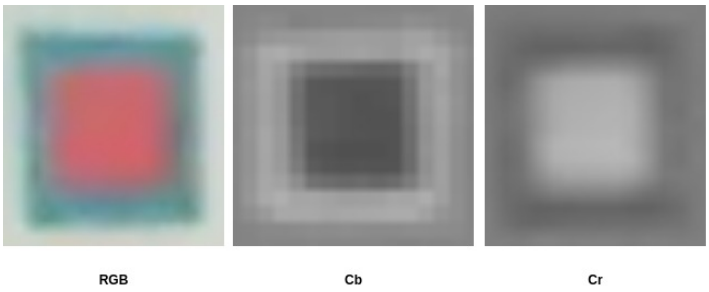


Figure 20.
Template in RGB, Cb and Cr colour spaces

A video taken of the test athletics track was input to the algorithm test Python script. Both the Cb and Cr templates were independently matched to the Cb and Cr components of the video frame. The resulting two dimensional array was normalised to values between 0 and 1. A detection threshold was set at 0.3 and all coordinates with values less than the threshold were saved. The coordinates were then passed through an algorithm which determined the length between each of the saved coordinates. If two coordinates were less than ten pixels away from each other the one with the higher value was discarded and the lower value saved. The result was a set of unique coordinates that corresponded with where the templates were located. The equation used to determine the difference between coordinates is given in equation (3.17).

$$d = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2} \tag{3.17}$$

The templates are placed on the finish line as shown in figure (21). One is placed on each lane boundary's intersection with the finish line. The second template is then placed one meter away from the finish line on the lane boundary.

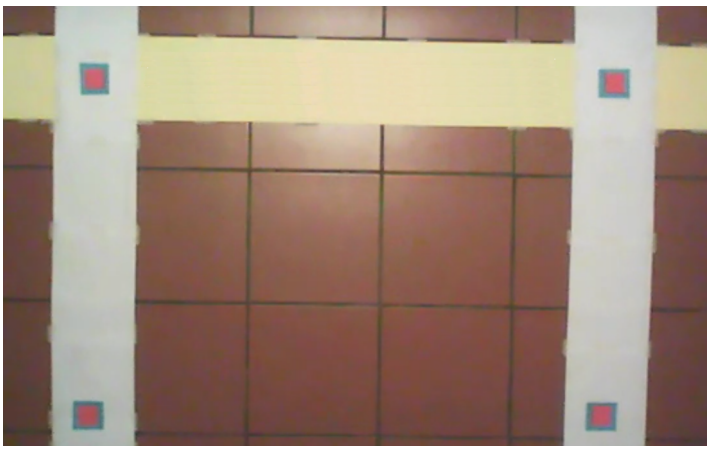


Figure 21.
Finish line with templates

The coordinates of the detected templates were then sorted into unique rows and columns. Two arrays were thus generated. The one containing unique x coordinates and the second containing unique y coordinates. This allowed the system to automatically draw a detection box where the thorax detection algorithm should be active. The top left coordinate of the box was given by the first column (n+0) and the first row (m+0). The bottom right coordinate is given by column (n+1) and row (m+1). Coordinate (n+1, m+0) would give the top right corner of the second lane while (n+2, m+1) would give the location of the bottom right corner of that lane. The size of the detection area was then tuned my multiplying it with overshoot constants (0.9-1.1) to increase or decrease the size of the detection area. A larger detection area may be incorrectly triggered by an athlete in the neighbouring lane. A smaller detection area may miss an athlete if they don't fully stay in their lane. The system then used these same coordinates to draw in the finish line. The line was drawn from column (n+0) and the first row (m+0) to the last column (n+x-1) and the first row (m+0), where x signifies the number of lanes within the camera's field of view. A finish line with the detection box as well as the finish line is given in figure (22).

Figure 22.
Finish line with graphics applied

The algorithm was then ported to C for implementation on the camera-module. The first step in this algorithm was to assemble the 8×8 blocks generated by the JPEG decoding algorithm into a final picture for the luminance and both chromanance components. The luminance component of the image was then passed to a custom implementation of a template matching algorithm used in the Python test. The algorithm consisted of two loops running inside another as described in equation (3.16). The algorithm was however changed to allow the system to attempt to match different templates instead of only one at a time. Multiple templates were formed from the same template by imaging it at different angles. This increased the reliability of the system significantly. The commented pseudo-code below explains the structure of the algorithm.

```
// loop through all the rows in the image
while (Row < Rows -> Row++)
    // loop through all the columns in the image
```

```
while (Col < Cols -> Col++)

    // loop through all the rows in the templates
    for (int a = 0; a < templateRows; a++)
        // loop through all the columns in the template
        for (int b = 0; b < templateCols; b++)

            // sum the squared differences
            // between the image and template
            sum1 += power((Templ1[a][b]-
             Image[Row+a][Col+b]),2);
            sum2 += power((Templ2[a][b]-
             Image[Row+a][Col+b]),2);
            .
            .

    // assign the smallest difference to the result
    Result[Row][Col] = smallest(sum1, sum2);
```

The decision was made to rather use the luminance component for the template matching algorithm. This decreased complexity and would not necessarily decrease detection effectiveness. The resolution of the luminance and chromanance components were also halved on the cameras in the camera module as a result of the 4:2 chroma sub-sampling ratio used by the JPEG encoding algorithm. The colour was thus sampled at half the resolution of the black and white component. The templates themselves were also changed to make them orientation agnostic. They were changed from squares to circles. Figure (23) gives the final templates.

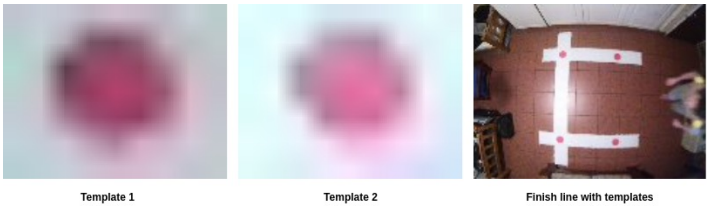


Figure 23.
Final templates

The result array was then, as in the Python implementation, passed to a function that determined local minimums. A local minimum is defined as the lowest location on a section of a two dimensional grid. The local minimum function identifies the the local minimum values as well as their coordinates. These values along with their coordinates are then sorted from low to high. The first coordinates and their corresponding values are then guaranteed to belong the best matches. The same algorithm that determines unique rows and columns, as used in the Python implementation, is then used to sort the coordinates. The algorithm then returns the unique rows and columns to the main program. To increase the reliability of the program the same operation is carried out five times using five different frames. The

results they produced is then compared and the values that are most often seen is selected and transmitted back to the management interface through the developed network protocol. The management interface then draws the required graphics and presents them to the user.

3.1.10 Athlete torso identification

The next major section of the project involved designing an algorithm capable of determining when an athlete's torso has crossed the finish line. The design was started using test images downloaded from photo sharing websites. Filters were added and transformations were done to these images while learning the OpenCV library. As with the line identification algorithm the real work started when a web-cam was placed over the same test athletics track. The previously described line identification algorithm was then added to the test Python program. The subsequent work in the test program was focussed on finding the athlete's shoulders in the green detection box. Initially the same template identification technique used in the lane identification algorithm was used. The templates would be fixed to the shoulders of athletes. This proved ineffective given the amount of possible orientations of the template when fixed to a moving shoulder. Figure (24) gives a frame from a test video where this technique was used.

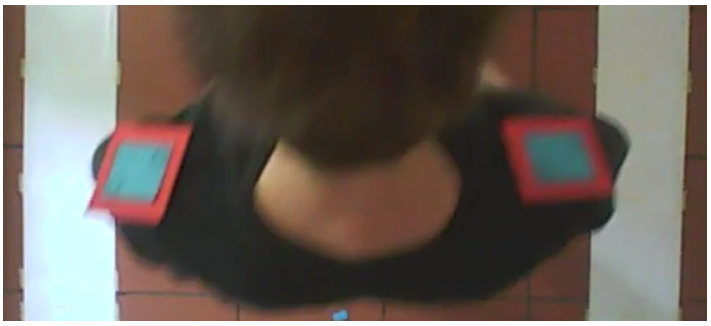


Figure 24.
Shoulder identification with template matching

This method was then replaced by a system where the two colour channels in the cameras were used to identify a specific colour. Here it was not used as in the line identification system to add redundancy. It was instead used to make the marker appear very bright in one colour channel and very dark in the second colour channel. Based on these specifications the colour blue was chosen. It will appear bright in the Cb colour space and dark in the Cr space given their make-up listed in equation (3.11). Thresholds were then added to the colour spaces. A binary threshold was added to the Cb colour space. All values over 150 was set to a 255, while all under was set to 0. A binary inverse threshold operation was applied to the Cr colour space where any value under 100 was set to 255, while any value over was set to 0. These two masks were then passed through a logical and operation. The bright areas in the resulting mask would thus correspond to the locations of a marker with this specific colour. These markers were attached using Velcro strips to the shoulders of athletes. Figure (25) gives the RGB image on the left hand side and the result of the mask on the right hand side.

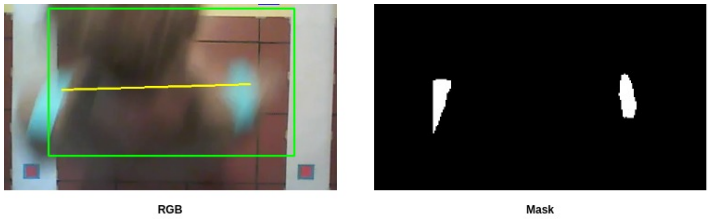


Figure 25.
Shoulder identification with colour matching

The location where the detection box caused a cut-off of the left side of the shoulder marker is visible on the mask in figure (25). This illustrates the trade-off that was made during the line identification algorithm. An algorithm then passes over the mask and checks if a pixel close to the current white pixel has been added to the result array. If no such pixel is found, it is added. If a coordinate already exists close to this coordinate the average between all the coordinates already added and this new coordinate is stored. The end result is an array containing the coordinates of the center of the two white blobs present in the right hand side of figure (25). A yellow line was then drawn between these two identified points as is visible on the RGB image. The position of the coordinates was then compared to the earlier coordinates of the finish line. The frame was then saved where this difference was the least.

Porting the algorithm to C for implementation on the micro-controller was started by first taking test images of different colours to determine if the work done using the web-cam will easily transfer to the new platform. The image as well as colour transformations to the Lu, Cr and Cb colour spaces are given in figure (26).

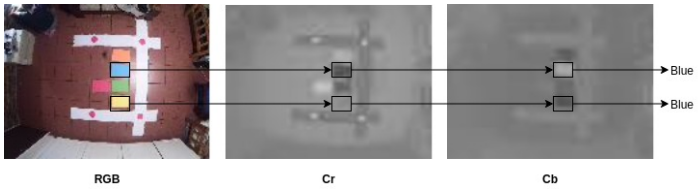


Figure 26.
Comparing colour in different JPEG colour channels

The blue and yellow markers are indicated in each image. Blue would still satisfy the requirement to implement the original algorithm developed in Python. Yellow was eventually chosen as a replacement given that it's exceptionally dark in the Cb colour space. This negated the need for the logical and operation and only required the cameras to decode one of the colour channels. The same function used in the line identification was again used to find local minimums in the images after they have been reconstructed from the 8×8 blocks returned from the JPEG decoding algorithm. The local minimums were then sorted from low to high. The differences between the coordinates of the two lowest values and the vertical line representing the finish line was then found. This value was saved to an array. After all the images where an athlete was present in was analysed, the system went through this array to

find the image where the markers were the closest to the finish line. This image along with it's time stamp was then transmitted back to the management interface.

3.1.11 Management interface

A management interface was designed for the camera module. The interface had to meet the following requirements.

1. A graphical user interface was needed for ease of use by non technical staff.
2. It should compile into a binary file for portability (use across different computers).
3. Allow athlete information to be loaded through the user interface or by opening a pre-prepared file.
4. Allow devices to be loaded through the user interface or by opening a pre-prepared file.
5. Change settings on all connected camera modules at the same time.
6. Interface with each camera module directly to conduct the qualification tests.
7. Administer a race by starting, stopping, resetting the time.
8. Provide an option to export time data to a .csv file.

The user interface was designed using the Qt framework. C++ was chosen as the language the management interface was written in to enable limited portability between it and code written for the micro-controller. It is also the default language for Qt development. A custom Ethernet frame based network could only be created on a Linux based system, and thus is the management interface limited to operating systems using the Linux kernel. All other commercial operating systems limit low level access to the network controllers. Compatibility could be extended if custom kernel level drivers could be written for Windows and MacOS. The main screen of the management interface is split into three sections. Section 1 is where a race is started, stopped and reset. It also provides the main race timer where the elapsed time since the start of the race is shown. Section two is where the athletes information is displayed in a row corresponding with their lane number. The number field next to each athlete's delete button will show the athletes final time once the race is terminated. The final image, taken where the athlete crosses the finish line, is shown in each athletes settings window. Section three is where the camera module for each lane is displayed in a row corresponding to the lane over which it is installed. A screen-shot of the management interface with athletes and devices loaded is given in figure (27). The important features are documented in the image.

Figure 27.
Screenshot of the management interface

The settings button positioned next to each devices gives access to tests and settings that will only affect the specific camera module. Global settings, which includes setting the time on the camera modules, are available from the tools menu. New athletes and devices can be added through the file menu. The windows for the athletes and devices are given in figure (28) and (29) respectively. Figure (28) provides a way to add new devices or edit existing devices on page one. Page two provides the winning image and advanced timing information. A way to save the winning image is also provided. Figure (29) provides a way to add new devices or edit existing devices. Page two allows the user to conduct a frame rate test, calibrate the system or take test images. Page three allows the user to download all taken images from the camera module.

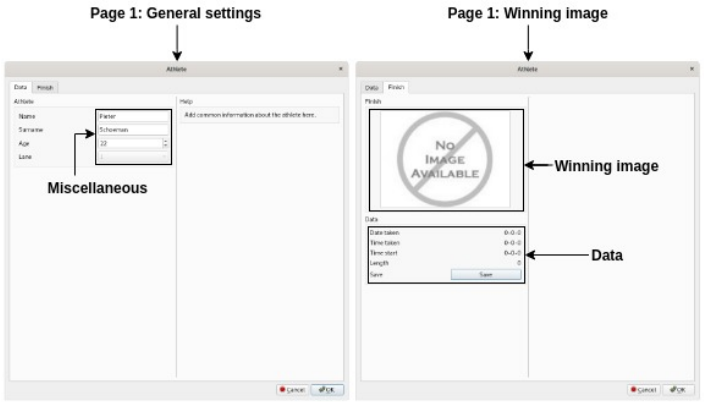


Figure 28.
Athlete settings dialog box

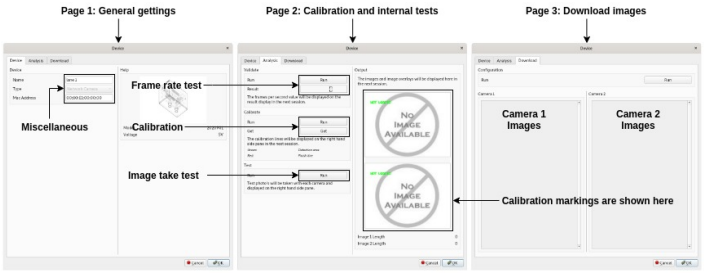


Figure 29.
Device settings dialog box

If devices or athletes need to be added in bulk .dev or .ath files can be used respectively. The .dev file is defined below. The new device is started by typing the device number. In the next line the device's human readable name is given. The next line gives the MAC address of the device and the final line gives the devices revision number. As many devices as required can then be added using this approach.

```
device 0
name lane 1
address 00:80:E1:00:00:00
type 1

device 1
name lane 2
address 00:80:E1:00:00:01
type 1
```

The .ath file is defined below. A new athlete is started by typing the athlete's number. The next lines gives the athlete's age, lane number, name and surname. This process is then repeated as many times as needed.

```
athlete 0
age 99
lane 1
name John
surname Doe

athlete 1
name Jane
surname Doe
age 99
lane 2
```

The network connection is implemented using the C++ raw socket application programming interface (API). The data handler is implemented using a state machine following the same methodology employed in the micro-controller firmware.

3.1.12 Gantry

The gantry needed to span the width of an entire athletics track. The required width was defined by the IAAF as 9.36m. A 10.5m span was selected to ensure an adequate gap between the outside edges of the outer lanes and the supports of the gantry. The gantry was designed to be 2.5m above the ground to ensure adequate space between tall athletes and the cameras. This would also make it the same height as the test set-up. The gantry also needed to be movable and deployable by one person. A low cost was the final requirement. This left the design with only two workable materials. The one being PVC piping and the other being aluminium piping. In the end PVC piping was chosen as a result of the many prefabricated fittings that made the assembly much quicker. Designing and manufacturing the required aluminium fittings would not have fitted into the time frame of the project. PVC pipes are however less rigid than aluminium and tended to bend. This was fixed by building a vertical support on both ends and fixing the middle of the span to both ends of the support. A diagram detailing the final design is given in figure (30).

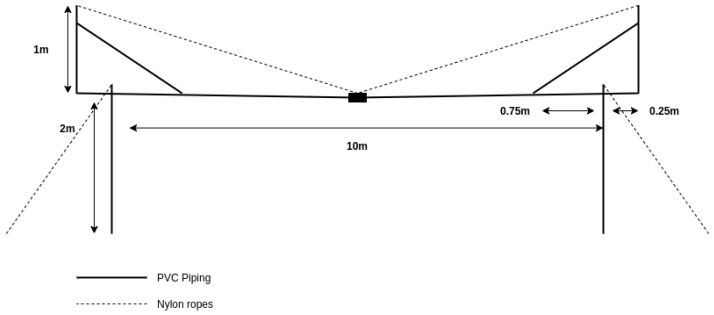


Figure 30.
Diagrammatic representation of the gantry

The black box in the middle of the span in figure (30) represents the connector that connects the two half's of the gantry together. The feet on both sides are also removable. Experimentation proved that metal feet were a material better suited to the strains involved in the gantry. Ropes attached to the feet were anchored in the ground to keep the gantry stable. Eight short horizontal piping stubs were added to the one side of the pipes spanning over the track. A suitable PVC pipe extender was then added to the one side of the camera module to connect to the pipe stub. This enabled the more expensive camera modules to be stored in a different place than the gantry.

The original gantry was damaged during transportation. The connection between the plastic horizontal pipe and the vertical metal uprights were broken. The design was thus switched to a more transportable and lighter gantry with a shorter span of 3m that can only be used to develop and demo one of the camera modules over a test athletics lane. Figure (31) gives an image of the original damaged gantry. Figure (32) gives an image of the new test gantry with a test athletics track drawn underneath it.

Figure 31.
Image of the gantry (version one)

Figure 32.
Image of the gantry (version two)

3.2 Design summary

This section provides a summary of the design tasks and how they were implemented in table (10). All tasks listed were completed by 26 January 2021.

Deliverable or task	Implementation	Completion

Build a camera array capable of taking 100 images per second. Implement the firmware to control the operation of the cameras.	Two single board cameras were sourced with image sensors capable of taking 50 images per second. These cameras were integrated into an array and firmware was successfully written from first principals to communicate and control the timing of the cameras. The array was only capable of a reduced frame rate.	Completed
Design a communication system capable of communicating with camera modules installed in up to eight lanes.	A custom Ethernet frame based network protocol was written from first principals. The network can theoretically communicate with as many devices as the physical network switching gear allows.	Completed
Design a system capable of determining whether the starting pistol has been fired. The system needed to be configurable to work with as many types of starting pistols and sound system as possible.	Of the shelf integrated circuits were chosen and assembled into a working detection circuit. Safety systems were added to prevent unsupported voltages from damaging the system. The design was done from first principals.	Completed
Design a PCB for the electronics on the main camera module.	A PCB was designed using the free and open source PCB development suite KiCad. This design included the layout and placement of the previously mentioned starting pistol detection system. The PCB also included the main voltage regulator, heat-sink, camera switches and external connections. The design was done from first principals.	Completed
Design a removable PCB for the systems memory. This is where the system stores all the captured photos.	The removable PCB containing the memory was designed from first principals. Of the shelf memory components and controllers were added to the board. Memory requirements were calculated from first principals.	Completed
Design an algorithm capable of decoding the captured images from their native JPEG format to their original pixel values.	The algorithm was designed from first principals using the official JPEG documentation and articles describing various aspects of the process.	Completed

Design an algorithm capable of determining the position of lane markers drawn on the athletics field.	An algorithm was successfully developed from first principals utilising encoded data from the JPEG decoding algorithm to determine the position of vertical lines. This algorithm was later switched to a template detection algorithm and ported from Python to C.	Completed
Design an algorithm capable of determining the time an athlete's thorax crosses the finish line.	A system capable of determining the position of coloured markers placed on an athlete's shoulders which indicated the position of the thorax was developed in Python and ported to C.	Completed
Design a management system executing on a personal computer to control the camera modules and administer a race.	A management system was written in the C++ programming language using QT as a visual framework. The computer side of the custom network protocol was implemented here.	Completed
Design a cheap gantry capable of spanning over an athletics track.	A gantry was designed with a span of 3 meters. The gantry was designed to be modular and can be broken in two to make it transportable. The gantry was only intended as a proof of concept and was designed to allow the testing of one camera module.	Completed
Design effective firmware to initialise and control the smaller elements of the design.	Firmware was written to control the initialisation and self test of the camera and decoder firmware components. Firmware was also written to indicate the systems internal status on external displays.	Completed
Design a plastic housing to meet the environmental conditions given in the project proposal and to make the system easier to handle and install.	The plastic housing was designed from first principals in Solidworks. A 3D printer was then used to manufacture all the designed components.	Completed

Table 10.
Design summary

4. Results

4.1 Summary of results achieved

Intended outcome	Actual outcome	Location in report
Core mission requirements and specifications		
The system should be capable of determining the time an athlete's thorax crosses the finish line with a accuracy of 10ms.	The system was able to determine the time an athlete's thorax crossed the finish line with an accuracy of 80.02ms.	Section 4.2.1 section 4.2.4 and section 4.2.5
A camera array will be created capable of filming the finish line at a speed of 100 frames per second.	The system achieved a maximum sustained frame rate of six frames per second. For the athlete detection a system was implemented to capture bursts of 12 images at an improved frame rate of 25 frames per second.	Section 4.2.4
The system should always correctly report the position of a white line painted on a grass field (finish line and lane markers).	The system was able to correctly determine the position of the finish line and lane boundaries. The positions were then shown to the user through an easy to use user interface. Different coloured lines were used to differentiate between the finish line and the lane markers.	Section 4.2.2
The reported position of the finish line should always be inside the dimensions of the 1.27 x 0.05-meter rectangle of the real finish line or within the 0.05-meter width of the lane boundaries.	The reported positions of the lane boundaries were always within a 1.27 x 0.05-meter rectangle of the real finish line or within the 0.05-meter width of the lane markers.	Section 4.2.2
The system should use the data from the line identification algorithm to determine if the vertical space above the top lane marker and the space below the bottom lane marker are even (+-10%).	The centring advice was communicated to the user through an intuitive user interface. The detection box and finish line shows whether the system has a good enough view of the athletics track to function.	Section 4.2.2

It should not take more than one minute from the last of eight athletes crossing the line to process the results.	The average processing time for the calibration algorithm is 26.04s. The average processing time for the thorax identification algorithm is 21.26s. The image processing algorithm will thus complete and present all the results of the race within approximately 21.26s of the last athlete finishing.	Section 4.2.3
A data processing module capable of processing the data at a rate of (500/60) XY pixels per second is required. At a resolution of 120×160 it is 160000 pixels per second. If only 12 images are taken this reduces to 3840 pixels per second.	The JPEG decoder achieved a processing rate of 11469.53 pixels per second. The line identification algorithm achieved a processing rate of 2901.97 pixels per second. The thorax identification algorithm achieved a processing rate of 167409.47 pixels per second.	Section 4.2.6

Table 11.
Result summary part one

Intended outcome	Actual outcome	Location in report
Field condition requirement and specifications		
The system will have to operate outside on an athletics field during an active event.	The calibration algorithm correctly detected the finish line and bottom lane boundary. It also correctly identified the top lane boundary but continuously overshot it.	Section 4.2.7
The system should work on any athlete irrespective of physical differences.	A diverse set of athletes had no effect on the accuracy of the system.	Section 4.2.8
The system should be capable of operating during temperature ranges common in South Africa	An ambient temperature of up to 35.0°C had no significant effect on the performance of the system.	Section 4.2.9

Table 12.
Result summary part 2

4.2 Qualification tests

The following section describes the tests that were carried out to determine whether or not the system met it's specifications. The video file used to simulate the starting pistol, the athlete test file and the device test file is provided in the technical documentation that accompanies this report.

4.2.1 Qualification test 1: Determining the system accuracy

This test was developed to determine the accuracy with which the system can determine whether an athlete has crossed the finish line. In this test a volunteer runs underneath the gantry on a simulated athletics track. The system then transmits the winning image, along with all the other images taken during the test to the management interface. A person then determines whether the system made the optimal choice.

The designed prototype of the camera module given in figure (11) was used as the device under test. The 26 January 2020 release of the device firmware was loaded. The TOTOLINK-A850R network switch and router was used to facilitate the connection between the device under test and the management computer. The switch was loaded with firmware version V1.1.4-B20170526.1150. Version 1.2 of the management interface released on 26 January 2020 was used to control the device and place it into test mode. A Lenovo G510 was used as the management computer. Two CAT5 Ethernet cables with a length of one meter each was used for the network connection between the device under test and the management computer. A 9V switching mode wall power supply was used to power the camera module. A Nokia 5.0 playing a video of a starting pistol being fired was used as a starting gun simulator. A audio cable with two male 3.5mm jacks was used to connect the phone to the device under test. A simulated lane of an athletics track was constructed by cutting pieces of white paper to the correct dimensions (0.2-meter width) and pasting them to tiled floor. The lane identification markers were then fixed to the simulated finish line. The device under test was placed 2.5m above this simulated athletics track.

The camera module was powered using a switching mode power supply capable of outputting 9V at 2A. One end of the first network cable was plugged into the network port of the camera module. The other end of the cable was plugged into port one of the network switch. The one end of the second network cable was connected to port two of the network switch. The other end was connected to the network port of the management computer. The mobile phone is connected through its headphone jack, using an audio cable, to the camera module's audio input.

The management interface was opened. The camera module with a MAC address of 00:80:E1:00:00:00 was loaded, along with test athletes. The camera settings were changed to automatic time: enabled, resolution: 160×120, baud-rate: 921600, quality: 60 and starting gun sensitivity: 2. The device settings of the module under test was then opened and the system was calibrated. The device was then armed. An athlete ran underneath the gantry

fifteen seconds after the simulated starting gun was fired. The winning image and time was then transmitted to the management interface, along with all the other images taken of the race.

The test was repeated five times with the results tabulated in table (13) through (17). Each table contains the winning image selected by the device, followed by the optimal image selected a volunteer. The last column gives the error between the frame selected by the algorithm and the frame selected by the volunteer. The time the system provided for the athlete is then compared to the result captured by a volunteer with a stopwatch. The maximum error introduced by the detection algorithm is then added to the maximum error introduced as a result of the frame rate and the propagation delay of the starting pistol. These factors were investigated in qualification tests 4.2.4 and 4.2.5. The final results are tabulated in table (18). The orientation change in the images are due to the orientation of the camera's changing as camera one hands off to camera two. One camera is used to capture the first six frames of the finish where after the second camera takes over for the remaining six images.

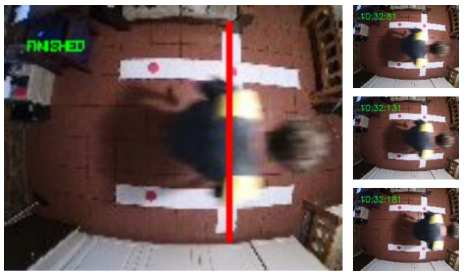
Image	Optimal	Finish
		1 Frame/s = 40ms
System Time	Human Time	Difference
16:135	16:440	00:305

Table 13.
Qualification test 1: Repeat 1

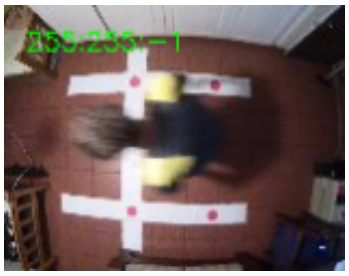
Image	Optimal	Finish
		0 Frame/s = 00ms
System Time	Human Time	Difference
16:145	17:220	01:075

Table 14.
Qualification test 1: Repeat 2

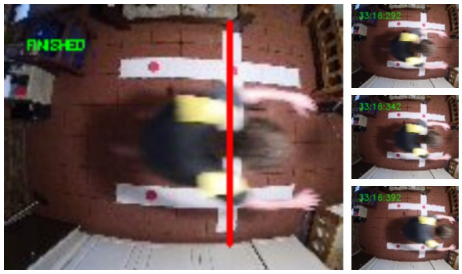
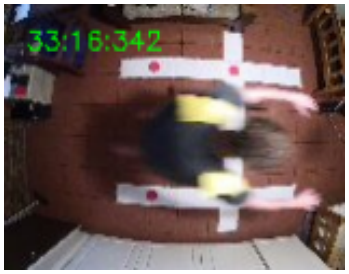
Image	Optimal	Finish
		0 Frame/s = 00ms
System Time	Human Time	Difference
14:949	16:650	1:701

Table 15.
Qualification test 1: Repeat 3

Image	Optimal	Finish
		0 Frame/s = 00ms
System Time	Human Time	Difference
16:108	16:360	0.252

Table 16.
Qualification test 1: Repeat 4

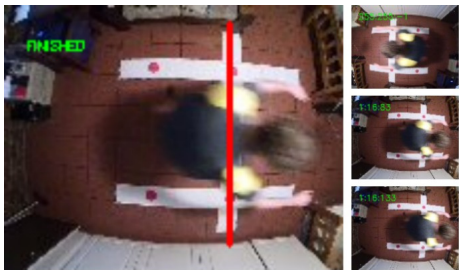
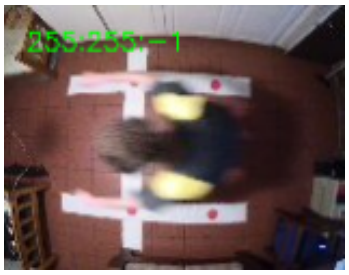
Image	Optimal	Finish
		1 Frame/s = 40ms
System Time	Human Time	Difference
16:116	16:520	0.404

Table 17.
Qualification test 1: Repeat 5

Detection Error	Frame Rate Error	Propagation Error	Total Error
40ms	40ms	$20\mu s$	80.02ms

Table 18.
Qualification test 1: Total error

The maximum error introduced by the system is 80.02ms, which does not meet the original specification of an accuracy of 10ms. The system is however more accurate than hand timing with the delta between the time provided by the system and the time provided by a volunteer with a stopwatch being larger than 0.7474ms.

4.2.2 Qualification test 2: Determining the position of the lane markers

This test was developed to determine the accuracy of the lane marking identification algorithm.

The designed prototype of the camera module was used as the device under test. The 26 January 2020 release of the device firmware was loaded. The TOTOLINK-A850R network switch and router was used to facilitate the connection between the device under test and the management computer. The switch was loaded with firmware version V1.1.4-B20170526.1150. Version 1.2 of the management interface released on 26 January 2020 was used to control the device and place it into test mode. A Lenovo G510 was used as the management computer. Two CAT5 Ethernet cables with a length of one meter each was used for the network connection between the device under test and the management computer. A 9V switching mode wall power supply was used to power the camera module. A simulated lane of an athletics track was constructed by cutting pieces of white paper to the correct dimensions (0.2-meter width) and pasting them to tiled floor. The lane identification markers were then fixed to the simulated finish line. The device under test was placed 2.5m above this simulated athletics track.

The camera module was powered using a switching mode power supply capable of outputting 9V at 2A. One end of the first network cable was plugged into the network port of the camera module. The other end of the cable was plugged into port one of the network switch. The one end of the second network cable was connected to port two of the network switch. The other end was connected to the network port of the management computer.

The management interface was opened. The camera module with a MAC address of 00:80:E1:00:00:00 was loaded, along with test athletes. The camera settings were changed to automatic time: enabled, resolution: 160×120, baud-rate: 921600, quality: 60 and starting gun sensitivity: 2. The device settings of the module under test was then opened and the system was calibrated. The user interface then showed where the system detected the lane markers and the finish line. The calibration was performed independently on both cameras.

The two calibration images taken before the graphics representing the lane boundaries were added are given in figure (33). The calibration test was conducted five times. The line offset from the center of the templates, placed on the lines to identify them, were measured in pixels for both images. The measurements were taken from the middle of the top and bottom detected lane boundaries in relation to the middle of the top left and bottom right template respectively. The horizontal distance between the middle of the detected finish line and the center of the top left template was also taken. The maximum offset error allowed to meet the specification is five pixels. The number of pixels containing the painted line above / below and to the left / right of the identified lines were also taken to access if the lines were placed in the middle of the much wider test boundary lines. The tables containing the calibration images with the graphics applied and the measurement data are given in table (19) to (23). The red lines indicate the position of the finish line. The green box represents the detection area where the thorax identification algorithm is active. The green box overshoots the finish line by design to allow for detection over the finish line. The top and bottom of the green box is however located inside the lane boundaries as per the original specifications.

Figure 33.
Base calibration image

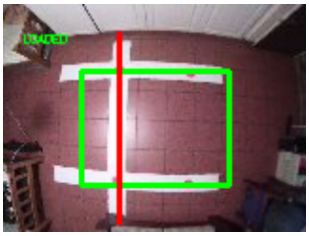
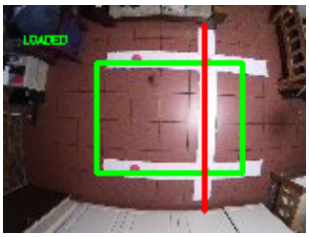
Image	Top	Bottom	Finish
	03 Pixels Error 07 Pixels Above 06 Pixels Below	05 Pixels Error 09 Pixels Above 04 Pixels Below	00 Pixels Error 04 Pixels Right 09 Pixels Left
	03 Pixels Error 09 Pixels Above 04 Pixels Below	05 Pixels Error 09 Pixels Above 04 Pixels Below	04 Pixels Error 09 Pixels Right 04 Pixels Left

Table 19.
Qualification test 2: Repeat 1

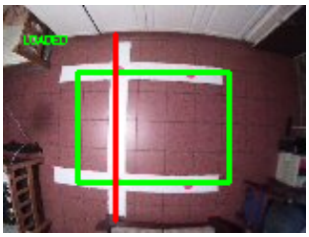
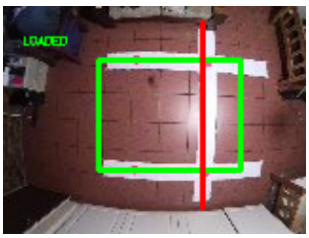
Image	Top	Bottom	Finish
	03 Pixels Error 07 Pixels Above 06 Pixels Below	04 Pixels Error 05 Pixels Above 08 Pixels Below	05 Pixels Error 09 Pixels Right 04 Pixels Left
	04 Pixels Error 04 Pixels Above 09 Pixels Below	03 Pixels Error 04 Pixels Above 09 Pixels Below	05 Pixels Error 11 Pixels Right 02 Pixels Left

Table 20.
Qualification test 2: Repeat 2

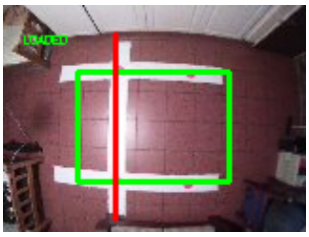
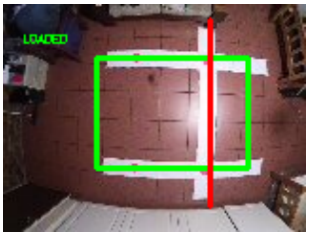
Image	Top	Bottom	Finish
	05 Pixels Error 06 Pixels Above 07 Pixels Below	04 Pixels Error 04 Pixels Above 09 Pixels Below	05 Pixels Error 09 Pixels Right 04 Pixels Left
	05 Pixels Error 05 Pixels Above 08 Pixels Below	02 Pixels Error 04 Pixels Above 09 Pixels Below	05 Pixels Error 04 Pixels Right 09 Pixels Left

Table 21.
Qualification test 2: Repeat 3

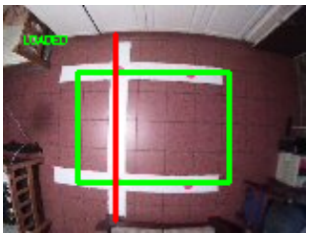
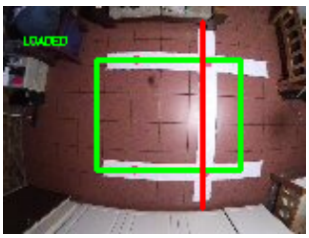
Image	Top	Bottom	Finish
	05 Pixels Error 06 Pixels Above 07 Pixels Below	04 Pixels Error 04 Pixels Above 09 Pixels Below	05 Pixels Error 09 Pixels Right 04 Pixels Left
	05 Pixels Error 06 Pixels Above 07 Pixels Below	02 Pixels Error 04 Pixels Above 09 Pixels Below	05 Pixels Error 11 Pixels Right 02 Pixels Left

Table 22.
Qualification test 2: Repeat 4

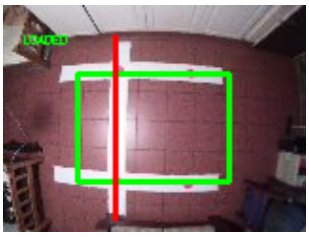
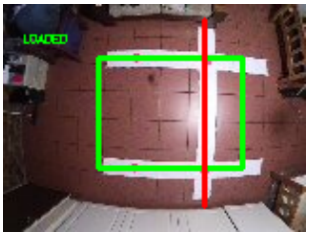
Image	Top	Bottom	Finish
	05 Pixels Error 09 Pixels Above 04 Pixels Below	04 Pixels Error 04 Pixels Above 09 Pixels Below	05 Pixels Error 08 Pixels Right 05 Pixels Left
	04 Pixels Error 06 Pixels Above 07 Pixels Below	03 Pixels Error 05 Pixels Above 08 Pixels Below	04 Pixels Error 10 Pixels Right 03 Pixels Left

Table 23.
Qualification test 2: Repeat 5

The calculated lane markers are inside the width of the simulated athletics track at all times. The specification was thus met. The green detection box successfully identified the relevant athletics lane and showed that the camera module was in fact correctly positioned. This specification was thus also met.

4.2.3 Qualification test 3: Determining the total processing time

This test was developed to measure the real world processing time of the device under test during calibration as well as in race mode. In this test a stopwatch was used to determine the time the system took from the point where an athlete crossed the finish line up to the point where a time result was displayed on the user interface. A stopwatch was also be used to determine how long the system took from where the calibration button was pressed up to when the calibration data was received by the user interface.

The designed prototype of the camera module was used as the device under test. The 26 January 2020 release of the device firmware was loaded. The TOTOLINK-A850R network switch and router was used to facilitate the connection between the device under test and the management computer. The switch was loaded with firmware version V1.1.4-B20170526.1150. Version 1.2 of the management interface released on 26 January 2020 was used to control the device and place it into test mode. A Lenovo G510 was used as the management computer. Two CAT5 Ethernet cables with a length of one meter each was used for the network connection between the device under test and the management computer. A 9V switching mode wall power supply was used to power the camera module. A Nokia 5.0 playing a video of a starting pistol being fired was used as a starting gun simulator. A audio cable with two male 3.5mm jacks was used to connect the phone to the device under test. A simulated lane of an athletics track was constructed by cutting pieces of white paper to the correct proportions.

The camera module was powered using a switching mode power supply capable of outputting 9V at 2A. One end of the first network cable was plugged into the network port of the camera module. The other end of the cable was plugged into port one of the network switch. The one end of the second network cable was connected to port two of the network switch. The other end was connected to the network port of the management computer. The mobile phone is connected through its headphone jack, using an audio cable, to the camera module's audio input.

The management interface was opened. The camera module with a MAC address of 00:80:E1:00:00:00 was loaded, along with test athletes. The camera settings were changed to automatic time: enabled, resolution: 160×120, baud-rate: 921600, quality: 60 and starting gun sensitivity: 2. The device settings of the module under test was then opened and the system was calibrated. The time the system took to calibrate was measured using a stopwatch and noted. The stopwatch was started when the calibrate button was pressed and stopped as soon as the calibration data was received. The system was then placed in race mode. The phone playing a video of starting pistol being fired was used to start the test race. Fifteen seconds after the start of the race a test athlete ran underneath the camera module over the simulated finish line. After the athlete passed over the finish line a stopwatch was started. This stopwatch was stopped as soon as the management interface showed the athlete's time.

Table (24) gives the calibration processing time. The calibration test was repeated five times to get an average processing time. Table (25) gives the processing time for determining the time in which an athlete has crossed the finish line. The test was also repeated five times to determine an average value.

Test No	Minutes	Seconds	Milliseconds
Test 1	0	26	320
Test 2	0	26	170
Test 3	0	26	190
Test 4	0	25	700
Test 5	0	25	820
Average			00:26:040

Table 24.
Qualification test 3: Calibration processing time

Test No	Minutes	Seconds	Milliseconds
Test 1	0	22	410
Test 2	0	21	440
Test 3	0	21	200
Test 4	0	20	880
Test 5	0	20	370
Average			00:21:260

Table 25.
Qualification test 3: race processing time

The time required to process the time in which an athlete has crossed the finish line was less than one minute and thus meeting the required specification.

4.2.4 Qualification test 4: Determining the maximum continuous and instantaneous frame rate

This test was developed to calculate the accuracy of the timing system. This test first determines the maximum sustained frame rate at which the camera array can film the finish line. The test then determines the maximum instantaneous frame rate.

The designed prototype of the camera module was used as the device under test. The 26 January 2020 release of the device firmware was loaded. The TOTOLINK-A850R network switch and router was used to facilitate the connection between the device under test and the management computer. The switch was loaded with firmware version V1.1.4-B20170526.1150. Version 1.2 of the management interface released on 26 January 2020 was used to control the device and place it into test mode. A Lenovo G510 was used as the management computer.

Two CAT5 Ethernet cables with a length of one meter each was used for the network connection between the device under test and the management computer. A 9V switching mode wall power supply was used to power the camera module.

The camera module was powered using a switching mode power supply capable of outputting 9V at 2A. One end of the first network cable was plugged into the network port of the camera module. The other end of the cable was plugged into port one of the network switch. The one end of the second network cable was connected to port two of the network switch. The other end was connected to the network port of the management computer.

The management interface was opened. The camera module with a MAC address of 00:80:E1:00:00:00 was loaded, along with test athletes. The camera settings were changed to automatic time: enabled, resolution: 160×120, baud-rate: 921600, quality: 60 and starting gun sensitivity: 2. The device settings of the module under test was then opened and the frame rate test was selected. The frame rate test takes images at the maximum instantaneous rate allowed by the cameras for a period of five seconds. The resulting image count is then divided by the time and the average frame rate is transmitted back to the management computer and displayed to the user. This is the continuous frame rate. The images are then transmitted to the management computer along with a time stamp indicating when they were taken. From here the instantaneous frame rate is calculated and displayed. The discrepancy is a result of a limitation on the camera's firmware where the maximum frame rate can only be sustained for six images (or 12 images when the cameras are used in a relay configuration).

An example of the continuous frame rate result produced by the management software is given in figure (34). The test was repeated five times with the results tabulated in table (26). The images taken during each test with their timestamps are provided in figure (35) to (39). Table (27) summarises the calculated instantaneous frame rate for each test.

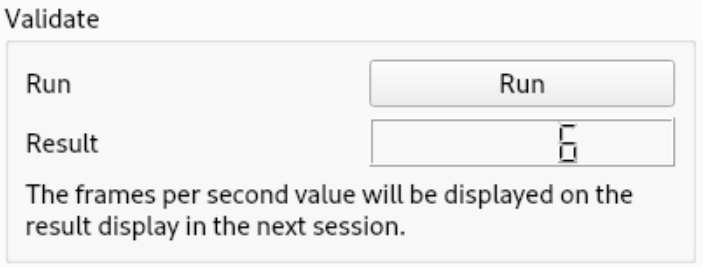


Figure 34.
A screen capture of the frame rate test displaying the continuous frame rate

Test No	Frame Rate (fps)
Test 1	06
Test 2	06
Test 3	07
Test 4	06
s Test 5	06
Average	6.2

Table 26.
Continuous frame rate test

Test No	Inter frame Time (ms)	Frame Rate (fps)
Test 1	40	25
Test 2	40	25
Test 3	40	25
Test 4	40	25
Test 5	40	25
Average	40	25

Table 27.
Instantaneous frame rate test

Figure 35.
Qualification test 4: Repeat 1

Figure 36.
Qualification test 4: Repeat 2

Figure 37.
Qualification test 4: Repeat 3

Figure 38.
Qualification test 4: Repeat 4

Figure 39.
Qualification test 4: Repeat 5

The camera module's continuous frame rate is consistent around six frames per second. The instantaneous frame rate is consistent around 25 frames per second. Both of these values are significantly less than the system was designed to perform. This is one of the largest factors determining the accuracy of the system as shown in qualification test 4.2.1.

4.2.5 Qualification test 5: Determining the starting pistol delay

This test was designed to measure the delay between the signal arriving at the camera module that the starting gun has been fired and the race timer being started. A external test pin is pulled high as soon as the race timer is started.

The designed prototype of the camera module was used as the device under test. The 26 January 2020 release of the device firmware was loaded. The TOTOLINK-A850R network switch and router was used to facilitate the connection between the device under test and the management computer. The switch was loaded with firmware version V1.1.4-B20170526.1150.

A Techtronics TBS1052 EDU oscilloscope was used to measure the propagation delay of the starting pistol. Version 1.2 of the management interface released on 26 January 2020 was used to control the device and place it into test mode. A Lenovo G510 was used as the management computer. Two CAT5 Ethernet cables with a length of one meter each was used for the network connection between the device under test and the management computer. A 9V switching mode wall power supply was used to power the camera module. A Nokia 5.0 playing a video of a starting pistol being fired was used as a starting gun simulator. A audio cable with two male 3.5mm jacks was used to connect the phone to the device under test. A simulated lane of an athletics track was constructed by cutting pieces of white paper to the correct proportions.

The camera module was powered using a switching mode power supply capable of outputting 9V at 2A. One end of the first network cable was plugged into the network port of the camera module. The other end of the cable was plugged into port one of the network switch. The one end of the second network cable was connected to port two of the network switch. The other end was connected to the network port of the management computer. The mobile phone is connected through it's headphone jack, using an audio cable, to the camera module's audio input. Test wire A leading from the camera board was connected to channel one of the oscilloscope. This wire is pulled high whenever the starting pistol detected interrupt is triggered. Test wire B leading from the input port of the starting pistol detection system is connected to channel two of the oscilloscope. The oscilloscope was set to trigger on channel one.

The management interface was opened. The camera module with a MAC address of 00:80:E1:00:00:00 was loaded, along with test athletes. The camera settings were changed to starting gun sensitivity: 2. The system was armed and the test video of the starting pistol was played.

The sound waveform was captured over the pin that is pulled high on detection of a fired starting pistol. The oscilloscope readout is given in figure (40). The blue signal is the audio signal while the yellow signal is the interrupt output.

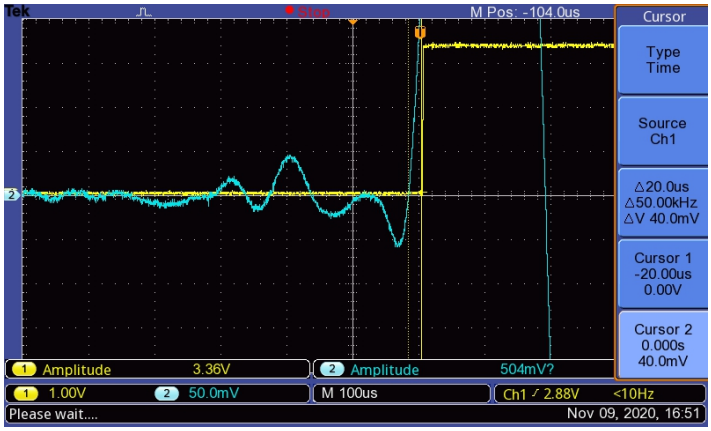


Figure 40.
Oscilloscope readout of the starting pistol propagation delay

The delay between the audio signal starting it's first rise and the interrupts first rise is $20\mu s$. The starting pistol propagated through the trigger circuit and triggered the interrupt within a period of $20\mu s$. This delay is almost negligible to the accuracy of the final system.

4.2.6 Qualification test 6: Determining algorithm performance

The objective of this experiment is to determine the exact time the system requires to process the images taken after the cameras have been started. The test is split into three sections. The first section measured the performance of the designed JPEG decoder. The second test measured the performance of the calibration algorithm. The third test measured the performance of the shoulder identification algorithm.

The designed prototype of the camera module was used as the device under test. The 26 January 2020 release of the device firmware was loaded. The TOTOLINK-A850R network switch and router was used to facilitate the connection between the device under test and the management computer. The switch was loaded with firmware version V1.1.4-B20170526.1150. Version 1.2 of the management interface released on 26 January 2020 was used to control the device and place it into test mode. A Lenovo G510 was used as the management computer. Two CAT5 Ethernet cables with a length of one meter each was used for the network connection between the device under test and the management computer. A 9V switching mode wall power supply was used to power the camera module. A Nokia 5.0 playing a video of a starting pistol being fired was used as a starting gun simulator. A audio cable with two male 3.5mm jacks was used to connect the phone to the device under test.

The camera module was powered using a switching mode power supply capable of outputting 9V at 2A. One end of the first network cable was plugged into the network port of the camera module. The other end of the cable was plugged into port one of the network switch. The one end of the second network cable was connected to port two of the network switch. The other end was connected to the network port of the management computer. A mini USB-B to USB-A cable was used to connect the debugger circuit connected to the camera module to the management computer. The mobile phone is connected through its headphone jack, using an audio cable, to the camera module's audio input.

The firmware was loaded on the camera board and the debugger was started. Breakpoints and software timers were used to determine the time a block of code was executing. The management interface was opened. The camera module with a MAC address of 00:80:E1:00:00:00 was loaded, along with test athletes. The camera settings were changed to automatic time: enabled, resolution: 160×120, baud-rate: 921600, quality: 60 and starting gun sensitivity: 2. The system was then calibrated and the execution time noted. The system was then placed into race mode. The simulated starting pistol was fired and a volunteer ran underneath the gantry. The execution time was then again noted.

The experiment was repeated five times and the results are listed in table (28) below. The second column gives the time the decoding process took to decode only the Huffman and quantization tables during the JPEG decoding phase. The third column gives the time the decoding process took to decode the frequency encoded data for all three colour channels and

transform the data to pixel values. The fourth column gives the time the calibration algorithm took to process the decoded data and perform it's calibration routine. Table (29) gives the same decoding time data with the shoulder decoding algorithm given in the fourth column.

Test No	Initial Tables (ms)	Frequency Data (ms)	Calibration (ms)
Test 1	4	1637	6621
Test 2	4	1640	6610
Test 3	4	1666	6621
Test 4	3	1702	6621
Test 5	4	1726	6608
Average	3.8	1674.2	6616.2

Table 28.
Decoding and calibration execution time

Test No	Initial Tables (ms)	Frequency Data (ms)	Calibration (ms)
Test 1	4	1637	72
Test 2	4	1640	72
Test 3	4	1666	71
Test 4	3	1702	72
Test 5	4	1726	72
Average	3.8	1674.2	71.8

Table 29.
Decoding and shoulder detection execution time

The performance, in pixels per second, of the JPEG decoding algorithm was then determined using (4.1). The symbol p_s represents the number of pixels that can be processed in a second, n_p represents the number of pixels per image and p_t represents the time to process one image.

$$\begin{aligned} p_s &= \frac{n_p}{p_t} \\ p_s &= \frac{19200}{1.6742} \\ p_s &= 11469.53 \end{aligned} \tag{4.1}$$

An average of 11469.53 pixels can thus be processed per second using this algorithm. The same symbols are used to calculate this value for the calibration algorithm. The calculation is given in equation (4.2).

$$\begin{aligned} p_s &= \frac{n_p}{p_t} \\ p_s &= \frac{19200}{6.6162} \\ p_s &= 2901.97 \end{aligned} \tag{4.2}$$

An average of 2901.97 pixels can thus be processed per second using this algorithm. The same symbols were again used to calculate the pixels per second value for the shoulder identification algorithm. The calculation is given in equation (4.3).

$$\begin{aligned} p_s &= \frac{n_p}{p_t} \\ p_s &= \frac{19200}{0.0718} \\ p_s &= 167409.47 \end{aligned} \tag{4.3}$$

This gives a processing speed of 167409.47 pixels per second.

4.2.7 Qualification test 7: Operating in field conditions

This test was designed to demonstrate the effectiveness of the system in real operating conditions in an uncontrolled environment. The calibration test conducted for qualification test 4.2.2 was repeated but this time the test was done using the gantry given in figure (32). Figure (41) gives the graphics drawn on the image after the calibration procedure was completed.

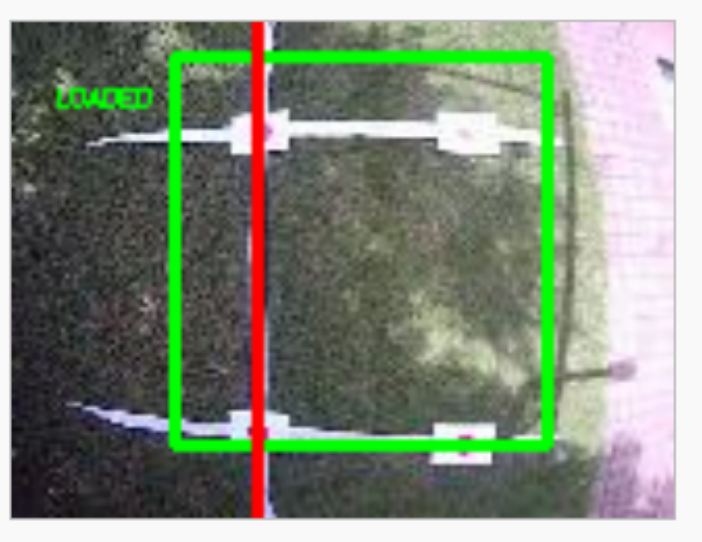


Figure 41.
Calibration algorithm in field conditions

Many of the calibration tests done exhibited the behaviour where the top lane boundary is overshot. The finish line and bottom lane boundary is however placed in the right location. More templates will be added in the future to combat this specific scenario.

4.2.8 Qualification test 8: Evaluating the effect of different athletes

This test was developed to determine the effect of a set of diverse athletes on the functioning of the torso detection algorithm. Three test athletes each of different heights, sexes and ages ran underneath the gantry while in race mode. The same maximum error as found in qualification test 4.2.1 is expected.

Exactly the same procedure as used in qualification test 4.2.1 was used. The results are given in table (30) through (32). Person 1 is a white 18 year old female with a height of 1.65m. Person 2 is a white 55 year old male with a height of 1.86m. Person 3 is a white 53 year old female with a height of 1.64m.

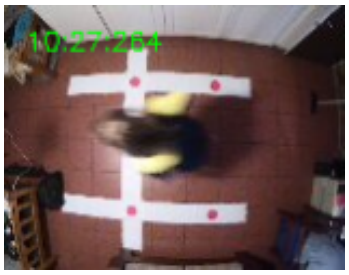
Image	Optimal	Finish
		1 Frame/s = 40ms

Table 30.
Qualification Test 8: Person 1

Image	Optimal	Finish
		0 Frame/s = 00ms

Table 31.
Qualification Test 8: Person 2

Image	Optimal	Finish
		0 Frame/s = 00ms

Table 32.
Qualification Test 8: Person 3

No difference in error between qualification test 4.2.1 and this test was observed.

4.2.9 Qualification test 9: Determining the effect of temperature on the system

This test was developed to determine the effect of temperature on the performance of the system. The execution time of the calibration procedure was used to compare the functioning of the system between temperatures. This function was selected because it is the most computationally expensive function and as a result a higher environmental temperature may negativity affect it's performance when the processor gets too hot.

The same set-up was used as was used during qualification test 4.2.3 to measure the calibration time. A Fluke 17B+ multimeter was used with an attached type-k thermocouple. The device was powered down for the internal temperature of the camera module to equalise with the environmental temperature. A reading of the environmental temperature was taken together with a reading of the temperature inside the camera module. Five successive calibration readings were then taken with the internal temperature being noted together with the execution time after each run. After each calibration measurement a continuous frame rate measurement was also taken to asses the performance of the cameras under temperature. After the five readings the camera module was left powered on and connected to the management interface for 30 minutes. The same calibration and frame rate test was then conducted and compared. This experiment was repeated two times. The first was conducted with an ambient temperature of around 25.0°C. The second was taken during a hot day with an ambient temperature of around 35.0°C. The results are given in table (33) and (34).

The ambient temperature at the start of the experiment was 23.6°C outside the camera module and 24.7°C inside the camera module. The outside ambient temperature at the end of the experiment was 23.7°C.

Test No	Execution Time (s)	Temperature (°C)	Frame Rate (fps)
Test 1	26.240	24.4	6
Test 2	27.580	25.7	6
Test 3	26.980	26.5	6
Test 4	26.580	27.7	6
Test 5	26.690	27.9	6
Test 6	26.150	32.8	6

Table 33.
Qualification Test 9: Repeat 1

The ambient temperature at the start of the experiment was 35.0°C outside the camera module and 40.1°C inside the camera module. The outside ambient temperature at the end of the experiment was 30.5°C.

Test No	Execution Time (s)	Temperature (°C)	Frame Rate (fps)
Test 1	27.930	35.3	5
Test 2	27.360	38.3	5
Test 3	27.520	36.9	5
Test 4	28.430	38.7	5
Test 5	26.000	39.5	5
Test 6	27.320	40.2	5

Table 34.
Qualification Test 9: Repeat 2

The outside temperature does not seem to have an significant effect on the performance of the system up to a temperature of 35.0°C. The average processing time increased slightly from 26.814s to 27.448s.

5. Discussion

5.1 Interpretation of results

Qualification test one, which determined the systems accuracy, found that in some instances a winning image was selected that was not the optimal choice. In essence there existed an image previous to the selected image where one of the athlete's shoulders were already touching the painted finish line. This was due to the system trying to minimise the distance between the detected finish line (red line in the test images) and the markers on the shoulders. The line is placed in the middle of the painted finish line which created the chance that an image existed where the shoulders were touching the painted line but not the detected line and a second image existed where the shoulders were directly over the detected line. This resulted in the system ignoring the optimal image. This however only resulted in a maximum of an one frame error. At a frame rate of 25 frames per second that can cause a maximum error of 40ms. Most errors would however be smaller than this. Different styles of running over the finish line were also used during this test, to add extra difficulty for the system, without finding any impact. The maximum error introduced by the reduced frame rate and the propagation delay of the starting pistol in starting the on-board timer was added to the detection error to give a maximum worst case scenario error of 80.02ms. This error was however still less than the error found where hand timing was used.

The second test quantified how accurately the system managed to detect the simulated lane boundaries. The specification required that the detected location of the lines always reside within the boundaries of the images lines. The test was repeated five times with each test calibrating both cameras. The results showed that the system was capable of meeting this requirement. The finish line and lane boundaries were correctly identified and shown to the user. A green box was displayed to the user that showed the detection area. A red line was shown showing the finish line. During all the tests that were conducted both the finish line and the top and bottom of the green box always stayed within the lane. The lines were not always placed in exactly the same area. The analysis of the number of pixels, of the imaged line, that were above / below or to the left / right of the drawn line showed that a variance of two to three pixels were common. It can thus be concluded that the template matching method of detecting lines met the required specification. The same system accurately showed when the system was correctly centred over the finish line.

The third qualification test was used to determine the total processing time. This was done to determine if the system met the specification that all processing had to complete within one minute of the last athlete crossing the finish line. It was thus assumed that the last athlete passed underneath the camera module used in this test. All other camera modules would thus get a head start and leave this one to finish last. The time from where the athlete passed over the finish line up to the point where the system returned a time was taken. Five repetitions of this experiment gave an average time of 21 seconds and 260 milliseconds. This system thus successfully met this specification of finishing all processing within one minute. The calibration time was also taken to determine how long the system would take to set up. This

test returned a slightly longer time of 26 seconds and 040 milliseconds.

The fourth qualification test measured the performance of the camera array. The original plan was to operate the two cameras in an interleaved mode thereby doubling the frame rate of the sensors from 50 to 100 frames per second. Limitations of the camera's JPEG encoding and processing hardware introduced delays limiting the continuous frame rate to 6 frames per second. An undocumented feature however allowed each camera to take 6 images at an increased frame rate of 25 frames per second. The images are then transmitted back to the controller in bulk. While this process is ongoing the second camera is instructed to take six more images. The two cameras are thus used in a relay mode. The instantaneous frame rate was determined by reviewing the timestamps added to the images by the timing algorithm during the initial capture algorithm. Although a rate of 25 frames per second is lower than the specification it was still able to capture the finish line with reasonable accuracy.

The fifth qualification test determined the delay introduced by the circuit waiting for the starting pistol. The delay between when a signal was received from the starting pistol to when the timer was started was measured using an oscilloscope. The delay proved to be an almost negligible $20\mu s$ but was used to calculate the accuracy of the system in qualification test one.

The sixth qualification test defined the performance of the designed algorithms. The race algorithm was required to process 160000 pixels per second to meet the processing requirements if 500 images were taken of the finish line at a resolution of 160×120. The required frame rate to complete the processing in one minute could however be reduced to 3840 pixels per second given that the frame rate was lower than initially expected and the time window where the cameras were active was reduced (thus less images to process). The processing rate for the decoding algorithm was added to the processing rate of the shoulder identification algorithm. This yielded a processing rate of 10996.56 pixels per second. This processing rate would not have been enough to match the specification if the original number of images needed to be taken. The efficiency of these algorithms should thus be revised when higher frame rate cameras are connected to the module. It was however enough to meet the specification if only 12 images had to be taken.

The seventh qualification test showed the calibration algorithm was also capable of operating outside. This unfortunately also introduced a detection area overshoot error. This result may be attributed to a slight difference in the height of the camera module or the effect of the sloped testing area where it was deployed.

The eighth qualification test repeated test one but with volunteers of different heights, sexes and ages. The results showed no discernible difference in system accuracy based on any of these criteria. This was expected as the system only searches for the shoulder markers and will work on anything if these markers are attached.

The ninth qualification test attempted to determine the effect of the ambient temperature on the functioning of the device. Two experiments were conducted at different ambient temperature values. Each experiment was replicated five times over a period of thirty minutes. No effect on system performance was found up to the highest tested temperature.

5.2 Critical evaluation of the design

Although fixing markers to painted lanes and athletes shoulders resulted in a working and fairly accurate system it is not the most elegant solution to the problem. Attaching the markers to the athletes adds an extra step to organising a school athletics race. Not all athletes would listen to the instructions to not wear yellow clothes requiring clothing changes before the race starts, and thus delaying the program. Small, one frame late, detection errors were also still encountered as a result of a not fully optimised detection algorithm. The reduced frame rate of the cameras were the largest problem. The processing delays on the camera's were not fully taken into account in the design phase. Firmware optimization work was done to fully utilise the cameras functions which made the prototype function but at a frame rate one quarter of what was required. A better solution would be to use the sensors, which were capable of meeting the design specification if used in a two camera array, in another camera design that gives the micro-controller direct access to the sensor. Integrating the power delivery to the camera module in the two unused wires of a standard Ethernet cable will reduce the number of cables required for the system to work. The installation of wider angle lenses in the cameras will enable the observation of more of the athletics track. There is also no technical reason why one camera module cannot be used for multiple lanes if the cameras lenses have a wide enough angle and the firmware is updated. Improvements in the starting pistol detection circuit may include the installation of more than one type of audio connector. A comparison system capable of comparing an incoming signal to a 0V to 12V signal instead of the current 0V to 5V reference signal would be advantageous. Usability improvement can also be made to the user interface. Currently tests are selected in a dialog and only executed when the dialog is closed. The returned data is then loaded when the dialog is opened back up again. Some user may find this confusing and would prefer the tests to be executed as soon as the button is pressed.

Strong points in the current design includes the reasonably low cost of the individual camera modules. This cost has the potential to further reduce as the device is optimised. The yellow shoulder markers were an easy and fast way to implement a thorax detection system. The template detection algorithm to identify markers placed on the athletics lanes was also a much simpler method of detecting the correct line geometry than using frequency transformation techniques. The design of the gantry also proved to be a low cost but efficient method of creating a 10.5m span, although it was damaged. The quick release connection constructed using PVC connectors to connect the cameras to the gantry also proved useful in transporting the system. The communication system that uses a custom Ethernet frame based protocol proved to be a very efficient method of linking different electronic devices together. It used much less memory and other system resources compared to a micro-controller based implementation of a full TCP/IP stack. The only disadvantages to this implementation is that it cannot easily connect to the internet, though some may see this as an advantage. The mechanical design and construction of the housing as well as the design and manufacturing of the PCB's proved to be a very effective and fast way to prototype ideas.

The system is expected to fail if both the athletes shoulders are outside the lane they have been assigned to. The result of this failure will be the incorrect detection of an athlete finishing,

or the complete lack of a detection to begin with. The athletes will also not be allowed to compete with yellow clothing, as that colour is used for the thorax detection algorithm. If yellow clothing is worn the system may incorrectly determine the position of the shoulders, which will result in an inaccurate time. The system is also expected to fail when more than one device is added with the same MAC address. If this occurs then both devices will respond to commands transmitted to just the one. Changing the baud rate of the cameras without restarting the entire system is also not currently possible. The result of attempting this action is simply that the cameras will stop responding and the user interface will give an error code. The management interface is also only currently compatible with a Linux based operating system. Rapid large scale changes in lighting may also temporally trigger the motion detection system while in race mode.

5.3 Design ergonomics

The gantry is the largest mechanical part to the design. It was therefore designed to be easily split in two sections of 5m. This enables the gantry to be easily transported on the roof of a standard utility vehicle. All the camera modules are connected to the gantry using quick release fittings held in by friction. The more expensive camera modules can thus be easily removed to be stored and transported in a safe manner. The user interface was also designed to be very easy to use. Athlete and device files can be prepared before race day to not waste time capturing all the athletes names and details before the race. Green orange and red indicators gives visual feedback to users regarding the state of the network connection or the state of each individual camera module. Red indicators indicates that the previous command was unsuccessful. Yellow indicators indicates that the previous command was partially successful and green indicators indicates a successfully executed command. The individual cameras as well as the memory cards are fully user replaceable if they were to break.

5.4 Health, safety and environmental impact

The design includes many plastic components. There are multiple companies in South Africa that specialises in recycling the PVC piping used for the construction of the gantry and quick release connectors. The PCB's were also manufactured using lead and assembled using solder which contained lead. The electronic components should thus be safely disposed of at the products end of life. A lot of research was done on recycling the lead embedded in the PCB as well as used in the solder. There are a few of these e-waste recycling companies located in South Africa.

5.5 Social and legal impact of the design

The system was not designed to be compliant with the full range of IAAF rules as it was only intended as a cheaper alternative to commercial or manual timing systems for schools. The system is not expected to have a very large social impact and will mainly allow teachers that did the timing manually to pay more attention to the athletes and spectators during events.

6. Conclusion

6.1 Summary of the work completed

This document describes work done to design a low cost simple integrated gantry-based timing system for school athletics. The main objectives of this system was to accurately determine the time at which a athlete crossed the finish line.

A literature review was conducted on techniques to locate objects in images. The review also looked into gesture recognition in videos. Information gained from this study was expanded to design the architecture of the gantry based timing system. A circuit was constructed to accurately start the timers when a starting pistol was fired. A communication protocol was also designed using Ethernet frames to facilitate a connection between the management computer and up to eight camera modules. A custom PCB was designed where the starting pistol detector circuit, electronic switches for the cameras, screen, memory and voltage regulators were installed. The PCB also contained interfaces for the external memory board and cameras. A removable memory board was also designed to hold the pictures taken by the cameras. A plastic 3D printed housing was developed and the hardware was assembled. Firmware to decode the JPEG images produced by the cameras was written from first principals and implemented first in Python and then in C on the micro-controller. Firmware was then written to use partially decoded JPEG data to determine lane markers on an athletics field. This feature was later replaced by a template matching algorithm that looks for templates fixed to the finish line and lane boundaries. This was also implemented from first principals, first tested in Python and then transferred to C. An algorithm was also implemented from first principals to find a specific colour pattern attached to the shoulders of athletes. This was also first developed in Python and ported to C. Firmware was written to use a real time clock to save the time when the race was started as well as to add time stamps to each taken image. A management program for the camera modules was written in C++ using QT as a user interface. This interface allows the user to load athletes from a file or add them manually. It also allows to add devices from a file or add them manually. Options are provided to change camera settings, set starting pistol sensitivity and change the time on the camera modules. Test images and frame rate tests can then be requested from each module individually. Races can also be started, stopped, reset and saved from the main window.

6.2 Summary of the observations and findings

The results indicated that it was not as simple to determine a high frame rate using single board cameras as initially thought. The need to use more parallel connectors to communicate with these cameras was understood. The algorithm developed for decoding the images did however meet the specifications and it was found that the first section of the decoding process can be ignored to save processing time (although not much). The cameras did not need to decode the quantization and Huffman tables for each image. That process could be done once on a test image before the race.

6.3 Contribution

The KiCad software suite for designing schematics and PCB's was learned from no initial knowledge. This included learning technical jargon that described the component footprints, copper layers etc. The process to manufacture these boards were also went through for the first time. The OpenCV image processing Python library was also learned from no initial knowledge. This started with a short online course and progressed from there. Knowledge of image processing concepts and colour terminology was gained. Reading into DCT's and IDCT's as well as how the full JPEG encoding/decoding concept works helped in the designing of the relevant software components. Knowledge of how filters can be applied to the world of photography provided insight to tackling image processing related problems. The capabilities of the range of STM32F7 micro-controllers offered by STMicroelectronics was learned. New tools to program these devices like CubeMX was mastered. This includes the firmware debug features which were very helpful in porting the Python code to C. Many previously unknown features of the C programming language was learned. This included techniques to properly manage a large firmware project in a language that is not object orientated or has an automatic garbage collector. The OSI model was learned while investigating the use of Ethernet as a viable communication protocol. A lot of previously unknown network terminology and concepts was also learned from this investigation. The process of designing plastic housings, and fitting all the required components before manufacturing, was also done for the first time. Solidworks was employed for this task. Qt creator was learned to create the graphical user interface and program the management software.

The only software modules in the final camera module that were the work of someone else was the HAL libraries taken from STMicroelectronics. These libraries control the low-level peripherals of the STM32F7 processor. Communication with the study leader was limited due to the COVID19 pandemic. It took the form of a couple of progress meetings through Google Meet and weekly progress reports submitted through Google Forms. A detailed email explaining the project progress was sent when the Google Form was not sufficient. A few in person meetings took place where the project was discussed direction was given.

6.4 Future work

Future work should start with widening the field of view of the cameras to enable them to capture more than a few lanes. Work should then be done on updating the firmware to enable the system to detect all the new lane markings. Better thorax identification techniques like polygon fitting could be employed to remove the need to wear identification marks on shoulders. The JPEG decoding algorithm as well as the camera control system could also be moved to a custom FPGA implementation. This will allow a more parallel and faster implementation than what was achieved on a micro-controller. This will keep some of the performance gains identified in the project, like the case where only one colour component is decoded, but will significantly increase it's performance. A power over Ethernet implementation can also be added to decrease the amount of cables to be routed to each camera module. A better camera implementation of the sensor found inside the SC20MPB

can also be investigated to improve both the continuous and instantaneous frame rate. This would involve a more parallel connector or a faster / lack of encoding algorithm. The circuit board should be redesigned and miniaturised once all specifications are sufficiently met. This will decrease cost by combining the development board with the custom designed main board and thus the removal of unnecessary components and complexity.

7. References

[1] "Competition and technical rules," International Association of Athletics Federations, Monaco, MC, Rules, Nov. 2019.

[2] "Photo finish guidelines," International Association of Athletics Federations, Monaco, MC, Rules, Jun. 2020.

[3] S. Koceski and N. Koceska, "Vision-based gesture recognition for human-computer interaction and mobile robot's freight ramp control," in *Proceedings of the ITI 2010, 32nd International Conference on Information Technology Interfaces*, 2010, pp. 289–294.

[4] Z. Lin, L. S. Davis, D. Doermann, and D. DeMenthon, "Hierarchical part-template matching for human detection and segmentation," in *2007 IEEE 11th International Conference on Computer Vision*, 2007, pp. 1–8.

[5] A. R. Chadha, P. P. Vaidya, and M. M. Roja, "Face recognition using discrete cosine transform for global and local features," in *2011 international conference on recent advancements in electrical, electronics and control engineering*, 2011, pp. 502–505.

[6] A. Shabaan Samra, S. El Taweel Gad Allah, and R. Mahmoud Ibrahim, "Face recognition using wavelet transform, fast fourier transform and discrete cosine transform," in *2003 46th Midwest Symposium on Circuits and Systems*, vol. 1, 2003, pp. 272–275 Vol. 1.

[7] "Information technology - digital compression and coding of continuous-tone still image requirements and guidelines," International Telecommunication Union, Geneva, CH, Standard, Sep. 1992.

[8] E. Hamilton, "Jpeg file interchange format," Milpitas, CA, US, Sep. 1992.

[9] K. Cabeen and P. Gent, "Image compression and the discrete cosine transform," College of the Redwoods, Eureka, CA, US, Tech. Rep., Feb. 1999.

[10] "Implementing an emulated uart on stm32f4 microcontrollers," STMicroelectronics, Geneva, CH, Tech. Rep., Aug. 2016.

[11] "Developing applications on stm32cube with lwip tcp/ip stack," STMicroelectronics, Geneva, CH, Tech. Rep., May 2015.

[12] L. Tian, L. Wu, Y. Wang, and G. Yang, "Binocular vision system design and its active object tracking," in *2011 Fourth International Symposium on Computational Intelligence and Design*, vol. 1, 2011, pp. 278–281.

[13] *2 Mega Pixels Serial JPEG Camera*, Spinelelectronics, 3 2020, rev. A.

The final report is supplemented with additional material that accompanies this report. Included in the additional material is technical documentation that provides more detail on the implementation, including complete software listings, circuit designs and calculations, a user guide and all data that were measured.