# Laser Pointer Turret Based Mosquito Air Defence System

Final Report

## A. Hartman
20475323

Submitted as partial fulfilment of the requirements of Project EPR402

in the Department of Electrical, Electronic and Computer Engineering

University of Pretoria

November 2023

Study leader: Prof. P. de Villiers

# Part 1. Preamble

This report describes work that I did *<to be completed>*.

*Project proposal and technical documentation*
This main report contains an unaltered copy of the approved Project Proposal (as Part 2 of the report).

Technical documentation appears in Part 4 (Appendix).

All the code that I developed appears as a separate submission on the AMS.

*Project history*
This project makes extensive use of existing algorithms on ... Some of the algorithms I used were adapted from ... 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 ⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯.

⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯          ⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯⎯
*Language editor signature*                              *Date*

*Declaration*

I, <u>A. Hartman</u> 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.

_____          _____

A. Hartman                                                              Date

# TABLE OF CONTENTS

# LIST OF ABBREVIATIONS

| | |
|---|---|
| **CAD** | computer aided design |
| **CCL** | connected components labelling |
| **CPU** | central processing unit |
| **CSI** | camera serial interface |
| **CUDA** | compute unified device architecture |
| **GPIO** | general purpose input/output |
| **GPU** | graphics processing unit |
| **GUI** | graphical user interface |
| **LED** | light emitting diode |
| **MIPI** | mobile industry processor interface |
| **PID** | proportional-integral-derivative |
| **PWM** | pulse width modulation |
| **RGB** | red, green, blue |
| **RPM** | revolutions per minute |
| **SORT** | simple online and real-time tracking |
| **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, unaltered from the final approved version that appears on the AMS.

# Part 3. Main Report

# 1. Literature study

Malaria is still one of the leading causes of death in low-income countries according to the World Health Organisation [1]. Mosquitoes that do not carry diseases are also a general nuisance in the everyday life of people living in mosquito-prone areas. Therefore, it is necessary to pursue improvement in our defence against mosquitoes.

To be able to design a laser pointer turret-based mosquito air defence system it is necessary to understand the principles of computer vision object detection and real-time tracking.

One approach towards tracking is to perform pattern matching. In general pattern matching is searching and checking images for the presence of other given images (patterns) to find and mark the patterns' locations (if any) within the given images. However, the study conducted by Hurtik et al. [2] presents results that indicate the best frame rate they achieved was 0.43 frames per second. This is far too slow to be used in a real-time tracking application.

Another approach is to perform particle filter-based tracking. This considers the proximity and behaviour of other targets. In the case of social insect tracking, it is known that two targets cannot occupy the same space, and targets will actively avoid collisions. Unfortunately, the joint particle tracker proposed in [3] suffers from exponential complexity.

A popular approach is to separate the detection and tracking functions. While numerous deep learning algorithms can detect objects based on appearance, it is worth noting that mosquitoes, particularly when not filmed up close, prove too minute to be reliably detected using appearance-based methods. A viable alternative is to detect objects by isolating the background and foreground of the image [4]. The foreground of the image contains the objects of interest. In [5] objects that are too close to one another are split into two and abnormally small objects are merged.

A proposed tracking method is the Simple Online Real-time Tracking (SORT) algorithm [6]. The algorithm is composed of an estimation model which makes use of a Kalman filter and a data association system that is solved optimally using the Hungarian algorithm.

In the proposed mosquito air defence system mosquito detection will be based on background and foreground isolation. This method is suitable because the system will operate in a known test environment where the background will change minimally. The online real-time nature of this system makes the case for pattern matching and particle filtering unfavourable because of the computationally intensive nature of these techniques. The methods in [4], [5], and [6] will be further investigated for the proposed system.

## 2. Approach

The aim of this project was to develop a system that would track and illuminate mosquitoes with a laser. The system will detect mosquitoes using a camera and laser will be controlled by a turret system. The system is implicitly required to operate in real-time in order to track and illuminate mosquitoes with a laser. This was taken into consideration in all the design aspects of the system. The project entails the first principles design of the following:

- A laser turret that can change the position of a laser beam.
- A control system that can control the laser turret.
- An object detection algorithm that can detect mosquitoes.
- An object detection algorithm that can detect the reflections of the laser beam.
- A multi-target tracking algorithm that can track multiple mosquitoes and predict their future locations.

The laser turret was inspired by two-axis laser scanners that use two mirrors to direct a laser beam in two dimensions as illustrated in Figure 7. The advantages of this approach compared to moving that laser diode itself is that the mirrors are rotated about their centre of mass which reduces the torque required to rotate the mirrors. This enables the laser turret to rapidly change the position of the mirrors and in tern the position of the laser beam. The drawback of this approach is that there will be additional scattering of the laser beam due to imperfections in the mirrors. This will not have significant impact on the system, therefore, this approach was chosen for the laser turret. A closed-loop proportional-integral-derivative (PID) control system will be used to control the laser turret. The laser position feedback for the control system will be provided by the laser detection system.

The object detection algorithm to detect mosquitoes can be implemented using a deep learning approach that will enable the identification and appearance matching of mosquitoes. This approach is computationally expensive and will require quality high-resolution images of mosquitoes. A less computational expensive approach is to use image segmentation techniques to separate the mosquitoes from the background of the image. An image segmentation approach greatly reduces the computational complexity at the expense of flexibility and robustness. Another trade-off to consider is that this approach will not be able to distinguish between mosquitoes and other objects similar in appearance. Image segmentation does not require the same quality and resolution images as a deep learning approach since it will not distinguish between objects of similar appearance. The benefits of an image segmentation approach outweigh the trade-offs compared to a deep learning approach for this project. Therefore, the image segmentation approach was chosen for this project.

The same considerations were made for the object detection algorithm to detect the reflections of the laser beam. An image segmentation approach was chosen for the same reasons as discussed above.

The multi-target tracking algorithm is required to track and predict the locations of mosquitoes. This can be efficiently achieved using a Kalman filter for one object. The Kalman filter is a recursive algorithm that can be used to estimate the state of a system that is subject to random noise. The Kalman filter is able to recursively predict the future state of a system,

which is required to predict the future locations of a mosquito. This can be extended to track multiple objects using the simple online and real-time tracking (SORT) algorithm that combines the Kalman filter and the Hungarian algorithm. The Hungarian algorithm is an optimal assignment algorithm that can be used to assign detections to tracks. The SORT algorithm is less computationally complex than other multi-target tracking algorithms while maintaining a comparable or higher accuracy [6].

A high-level overview of the system can be seen in Figure 1. The system was designed to operate in a controlled environment.
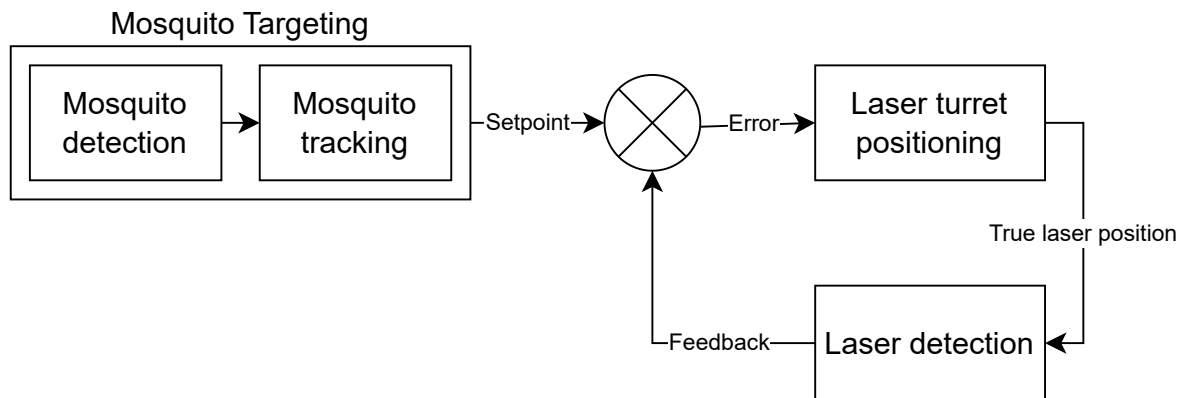


**Figure 1.**
**Overview of the system.**

# 3. Design and implementation

## 3.1   Design summary

This section summarises the project design tasks and how they were implemented (see Table 1).

| Deliverable or task | Implementation | Completion of deliverable or task, and section in the report |
|---|---|---|
| The laser turret and laser turret control system had to be designed and implemented by the student. | The laser turret and laser turret control system were designed and implemented from first principles. | Completed. |
| The mosquito and laser detection algorithm had to be designed and implemented on an embedded platform by the student. | The mosquito and laser detection algorithm was designed and implemented from first principles. | Completed. |
| The mosquito tracking and prediction algorithm had to be developed and implemented by the student. | The mosquito tracking algorithm and prediction was developed and implemented from first principles. | Completed. |
| An appropriate camera had to be selected by the student. | An appropriate camera was selected by the student. | Completed. |
| The various subsystems had to be integrated on a real-time embedded system. | The various subsystems were integrated on a real-time embedded system. | Completed. |
| Appropriate motors needed to be selected for the laser turret. | The stepper motors were selected based on the requirements of the laser turret. | Completed. |

**Table 1.**
**Design summary.**

## 3.2   Theoretical analysis and modelling

### 3.2.1   Mapping pixel co-ordinates to metric co-ordinates

To control the laser, its position must be known in the world co-ordinate frame. The laser position was measured using a camera, thus the camera's pixel co-ordinate frame must be mapped to the world co-ordinate frame. To perform this mapping a camera model is required. The forward imaging model of a camera is shown in Figure 2.
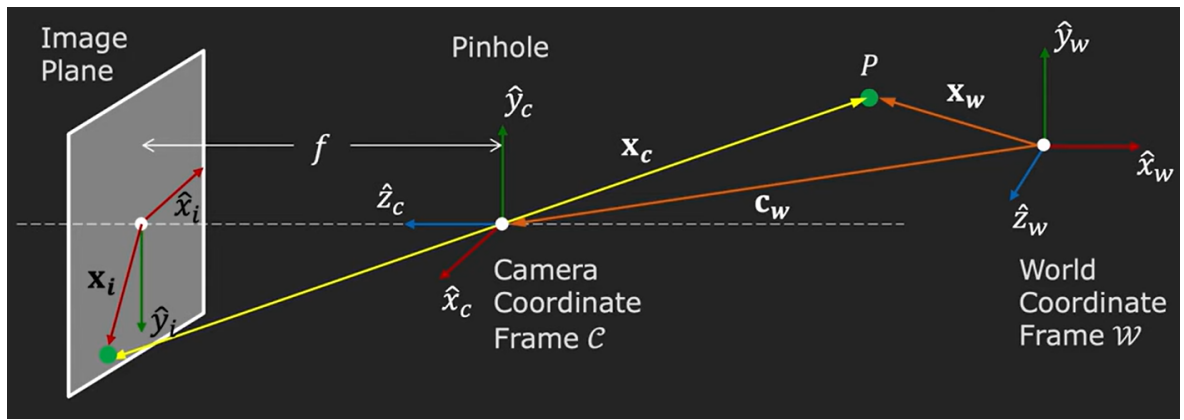


**Figure 2.**
**Forward imaging model of a camera. This figure was modified from [7].**

Using the forward imaging model the pixel distance was mapped to the metric distance for the *x*-axis with

$$X = Z \times \left( \frac{x - x_{ref}}{f_x} \right),$$
(1)

where $Z$ is the depth camera with respect to the world co-ordinate frame, $x$ is the pixel of interest, $x_{ref}$ is the reference pixel, and $f_x$ is the effective focal length of the camera. Similarly, the pixel distance was mapped to the metric distance for the y-axis. The effective focal length of the camera $f_x$ was determined through camera calibration. The mapping is performed for the *y*-axis.

### 3.2.2   Morphological operations

Erosion is a fundamental morphological operation used to remove small structures or details from a binary image. It is defined as the basic set operation of moving a structuring element (usually a smaller binary matrix) over the input binary image and finding the intersection of the structuring element with the image. This operation can be mathematically expressed as

$$(A \ominus B)(x,y) = \bigcap \{A(x+i, y+j) | (i,j) \in B\},$$
(2)

where

- *A* is the input binary image.
- *B* is the structuring element.
- ⊖ represents the erosion operation.
- $(x, y)$ are the pixel co-ordinates in the resulting image.

Dilation is another fundamental morphological operation, but it is used to enhance or grow the features in a binary image. Dilation can be defined as the set operation that moves the structuring element over the input image and computes the union of the element with the parts of the image where the structuring element "hits". The mathematical expression for dilation is as follows:

$$(A \oplus B)(x, y) = \bigcup \{A(x + i, y + j) | (i, j) \in B\} \tag{3}$$
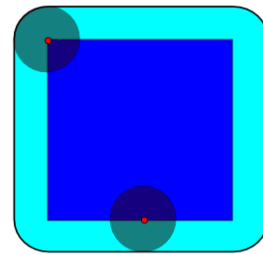
where

- *A* is the input binary image.
- *B* is the structuring element.
- ⊕ represents the dilation operation.
- $(x, y)$ are the pixel co-ordinates in the resulting image.

The erosion and dilation operations are illustrated in Figure 3.
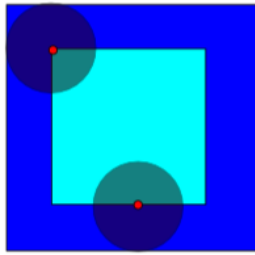


**(a) The erosion of the dark-blue square by a disk, resulting in the light-blue square.**

**(b) The dilation of a dark-blue square by a disk, resulting in the light-blue square with rounded corners.**

**Figure 3.**
**The figure shows an example of erosion and dilation. This figure was modified from Renatokeshet at the English Wikipedia (2008).**

Opening is a compound morphological operation that consists of an erosion followed by a dilation using the same structuring element. It is primarily used to remove noise and small objects from a binary image. The opening of an image *A* by a structuring element *B* is defined as

$$A \circ B = (A \ominus B) \oplus B, \tag{4}$$

where

- *A* is the input binary image.
- *B* is the structuring element.
- ∘ represents the opening operation.

Closing is another compound morphological operation that consists of dilation followed by

an erosion using the same structuring element. It is used to close small holes and gaps in objects and to connect nearby objects. The closing of an image $A$ by a structuring element $B$ is defined as

$$A \bullet B = (A \oplus B) \ominus B, \tag{5}$$

where

- $A$ is the input binary image.
- $B$ is the structuring element.
- $\bullet$ represents the closing operation.

Opening and closing are both idempotent operations, meaning that repeated openings or closing have no effect on the image. Opening and closing is illustrated in Figure 4.
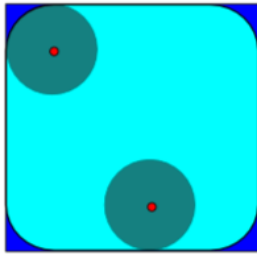


(a) The opening of the dark-blue square by a disk, resulting in the light-blue square with round corners.
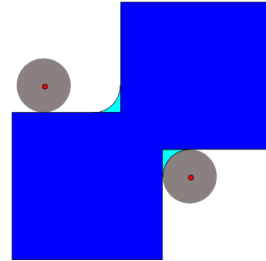


(b) The closing of the dark-blue shape (union of two squares) by a disk, resulting in the union of the dark-blue shape and the light-blue areas.

**Figure 4.**
**The figure shows and example of opening and closing. This figure was modified from Renatokeshet at the English Wikipedia (2008).**

### 3.2.3   Kalman filter

The Kalman filter is a recursive algorithm that estimates the state of a system from a series of noisy measurements. The Kalman filter is based on a linear dynamical system model, which is defined by the following equations:

$$\mathbf{x}_k = \mathbf{A}\mathbf{x}_{k-1} + \mathbf{B}\mathbf{u}_{k-1} + \mathbf{w}_{k-1}$$
$$\mathbf{z}_k = \mathbf{H}\mathbf{x}_k + \mathbf{v}_k \tag{6}$$

where

- $\mathbf{x}_k$ is the state vector at time $k$.
- $\mathbf{z}_k$ is the measurement vector at time $k$.
- $\mathbf{A}$ is the state transition matrix.
- $\mathbf{B}$ is the control matrix.
- $\mathbf{u}_k$ is the control vector at time $k$.

- $\mathbf{w}_k$ is the process noise vector at time $k$.
- $\mathbf{H}$ is the observation matrix.
- $\mathbf{v}_k$ is the measurement noise vector at time $k$.

### 3.2.4  SORT tracking

SORT (Simple Online and Realtime Tracking) is a popular algorithm for multi-object tracking in video sequences. The main goal of the SORT algorithm is to associate detection boxes across frames in a video sequence to form trajectories for each individual object. Here is a theoretical background on the SORT tracking algorithm: 1. Overview

SORT is designed to be both simple and efficient, achieving real-time performance. It is based on the tracking-by-detection paradigm, which means it relies on an external object detector to provide bounding boxes around objects of interest in each frame. The algorithm then associates these detections across frames to create tracks. 2. Detection

Before tracking, an object detection algorithm (such as Faster R-CNN, YOLO, or SSD) is applied to each frame to detect objects of interest. The output is a set of bounding boxes along with their associated confidence scores. 3. Association

The core of the SORT algorithm is the association step, where detections are linked across frames to form tracks. This is done based on the intersection over union (IoU) metric, which measures the overlap between two bounding boxes. The steps involved are:

Prediction: For each existing track, the Kalman filter is used to predict the new position of the object in the current frame. Matching: The predicted positions of existing tracks are matched with the current frame's detections based on the IoU metric. A bipartite graph matching algorithm (such as the Hungarian algorithm) is used to find the optimal assignment. Update: The Kalman filter for each matched track is updated with the current detection. Creation and Deletion: New tracks are created for detections that could not be matched to existing tracks. Tracks that have not been matched for a certain number of frames are deleted.

4. Kalman Filter

The Kalman filter is a critical component of the SORT algorithm, providing a way to predict and update the state of the tracks based on the detections. The state of a track is represented as $[x, y, \dot{x}, \dot{y}]$, where x,yx,y are the co-ordinates of the centre of the bounding box, ss is the scale/area of the bounding box, rr is the aspect ratio, and $\dot{x}, \dot{y}$ are the respective velocities. 5. Challenges and Limitations

Occlusions: SORT can struggle with objects that are occluded or interact closely with each other, as the IoU metric may not be sufficient for correct association. Identity Switches: In cases where objects cross paths or are close together, SORT can suffer from identity switches. Dependence on Detection: The performance of SORT is highly dependent on the quality of the detections provided by the external object detector.

## 3.3   Hardware design

The hardware block diagram is presented in Figure 5. The hardware consists primarily of a laser turret, a camera, a mosquito enclosure, and an embedded processing platform. The laser turret is used to move the laser beam across the mosquito enclosure. The camera is used to detect the laser beam and the mosquitoes. A bracket is used to position the laser turret and the camera relative to the mosquito enclosure. The embedded processing platform is used to control the laser turret and perform the image processing required for the laser detection and mosquito detection.
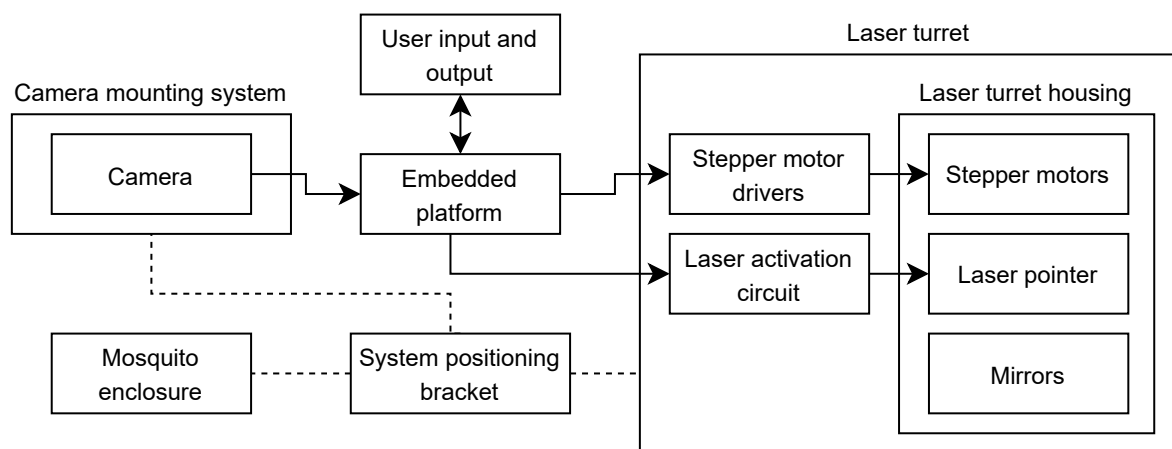
**Figure 5.**
**Hardware block diagram.**

### 3.3.1   *Embedded platform selection*

To select an embedded platform for the project the following requirements were considered:

**Processing power**
> The embedded platform must be able to perform the image processing required for the laser detection and mosquito detection in real-time. Image processing is computationally intensive, but can largely benefit from parallelisation. Therefore, an embedded platform with a graphics processing unit (GPU) will be preferred.

**Memory**
> The embedded platform must have sufficient memory to store the multiple images along with the required algorithms. A 1080p image with three 8-bit colour channels requires 6 MB of memory.

**Hardware interfaces**
> The embedded platform must be compatible with camera interfaces. It must also have

sufficient general purpose input/output (GPIO) pins to interface with the stepper motor drivers. The embedded platform must also have support for a display output and a keyboard and mouse input to provide a user interface to control the system.

**Availability and support**

It is import to select an embedded platform that is readily available. It is important to select an embedded platform with large body of knowledge available and sufficient scientific support community since this is an individual project without access to people with expertise with the embedded platform.

The Nvidia Jetson Nano was chosen as the embedded platform for the project. The Nvidia Jetson Nano is a single-board computer with a GPU and a quad-core central processing unit (CPU). The Nvidia Jetson Nano has a mobile industry processor interface (MIPI) camera serial interface (CSI)-2 camera interface and has 40 GPIO pins. The Jetson Nano has a quad-core ARM Cortex-A57 MPCore processor, an Nvidia Maxwell GPU with 128 compute unified device architecture (CUDA) cores, and 4 GB of LPDDR4 memory. It also features an HDMI display output and a universal serial bus (USB) 3.0 port. The Nvidia Jetson Nano is readily available and has a large body of knowledge available and sufficient scientific support community.

### 3.3.2 Camera selection

To minimise the computational complexity of the image processing required for the laser detection and mosquito detection a low video resolution will be used. The video resolution will not need to exceed 1080p. The camera must be able to capture images at a frame rate of at least 30 frames per second. The camera must also be compatible with the MIPI CSI-2 interface on the Nvidia Jetson Nano.

The Raspberry Pi Camera Module V2 was chosen as the camera for the project. The Raspberry Pi Camera Module V2 is an 8 MP camera that can capture images at a frame rate of 30 frames per second at 1080p. The Raspberry Pi Camera Module V2 is compatible with the MIPI CSI on the Nvidia Jetson Nano. The Raspberry Pi Camera Module V2 is readily available and has a large body of knowledge available and sufficient scientific support community.

### 3.3.3 Mosquito enclosure

The mosquito enclosure will be a rectangular prism as shown in Figure 6. The width of the enclosure must be a minimum length of 1 metre. The front facing surface will be transparent and the rest of the surfaces will be white. The enclosure will be fitted with internal lighting to ensure contrast between the background and the mosquitoes and to minimise the camera noise.
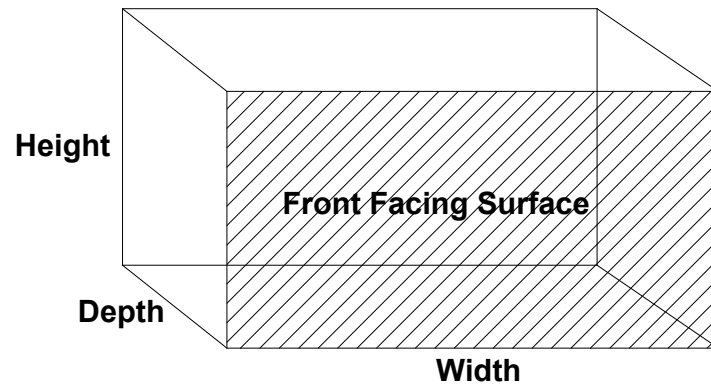
**Figure 6.**
**Mosquito enclosure dimensions.**

### 3.3.4  *Laser turret*

The laser turret design was inspired by commercial two-axis laser scanners. The position of the laser beam is controlled by adjusting the angles of two mirrors. This is illustrated in Figure 7. The mirrors are connected directly to the output shaft of the motors. The laser turret will be positioned orthogonally to the *xy*-plane of the mosquito enclosure. The point where the laser beam shines orthogonal to the *xy*-plane of the mosquito enclosure is considered the origin of the laser. This will occur when the two mirrors are at 45° relative to the laser beam.
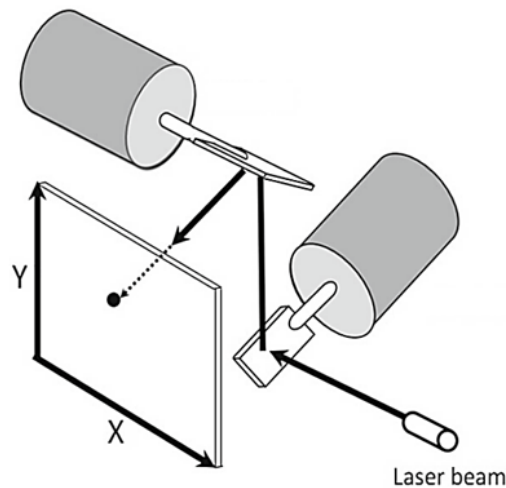


**Figure 7.**
**Two-axis laser scanner schematic. This figure was modified from [8].**

When a single axis of the laser turret is considered it can be seen that a right triangle is formed between the laser turret and the *xy*-plane mosquito enclosure as shown in Figure 8. Using the properties of a right triangle the laser beam angle $\theta$ required to shine the laser a distance *d*

relative to the origin of the laser can be calculated using

$$\theta = \arctan\left(\frac{d}{z}\right),\qquad(7)$$

where $z$ is the distance between the turret and the back wall of the mosquito enclosure.
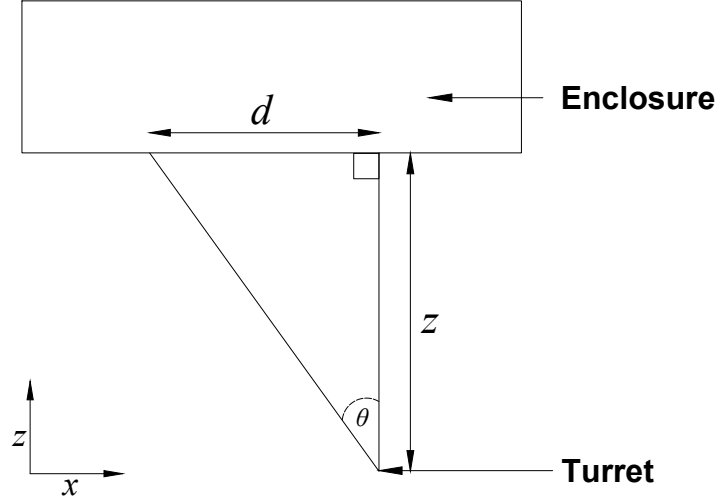


**Figure 8.**
**The right triangle formed between the laser turret and the mosquito enclosure.**

The lateral step size of the laser must be a maximum of $d_{min} = 2$ mm. This will ensure that the laser will be able to illuminate every position in the mosquito enclosure since the laser itself will be a few millimetres wide.

The required lateral speed is determined in accordance with the field specifications. The longest length of the mosquito enclosure $d_{max}$ will be 1 m. To ensure that the laser can illuminate any set point inside the mosquito enclosure within 2 s of receiving a step input, it was assumed that the laser must be able to move with a velocity $v_{laser}$ of least $1\,\mathrm{m\,s^{-1}}$ opposed to the $0.5\,\mathrm{m\,s^{-1}}$ required to move the 1 m length of the mosquito enclosure within 2 s. This was done to accommodate for the settling time of the laser turret control system.

## 3.4   Hardware implementation

### 3.4.1   *Mosquito enclosure*

The mosquito enclosure was constructed from a second hand glass fish tank with dimensions in the form *xyz* given by $90 \times 38 \times 32$ cm. All the surfaces other than front facing surface was retrofitted with a white lining. Internal light emitting diodes (LEDs) were added to the enclosure, shining from the top and bottom surfaces to minimise shadows on the back surface the enclosure, to ensure contrast between the background and the mosquitoes and to minimise the camera noise.

### 3.4.2   System positioning

The laser turret and the camera was placed outside the mosquito enclosure in known positions relative to the enclosure. The bracket, shown in Figure 9, was built to hold the laser turret and the camera in place relative to the mosquito enclosure.
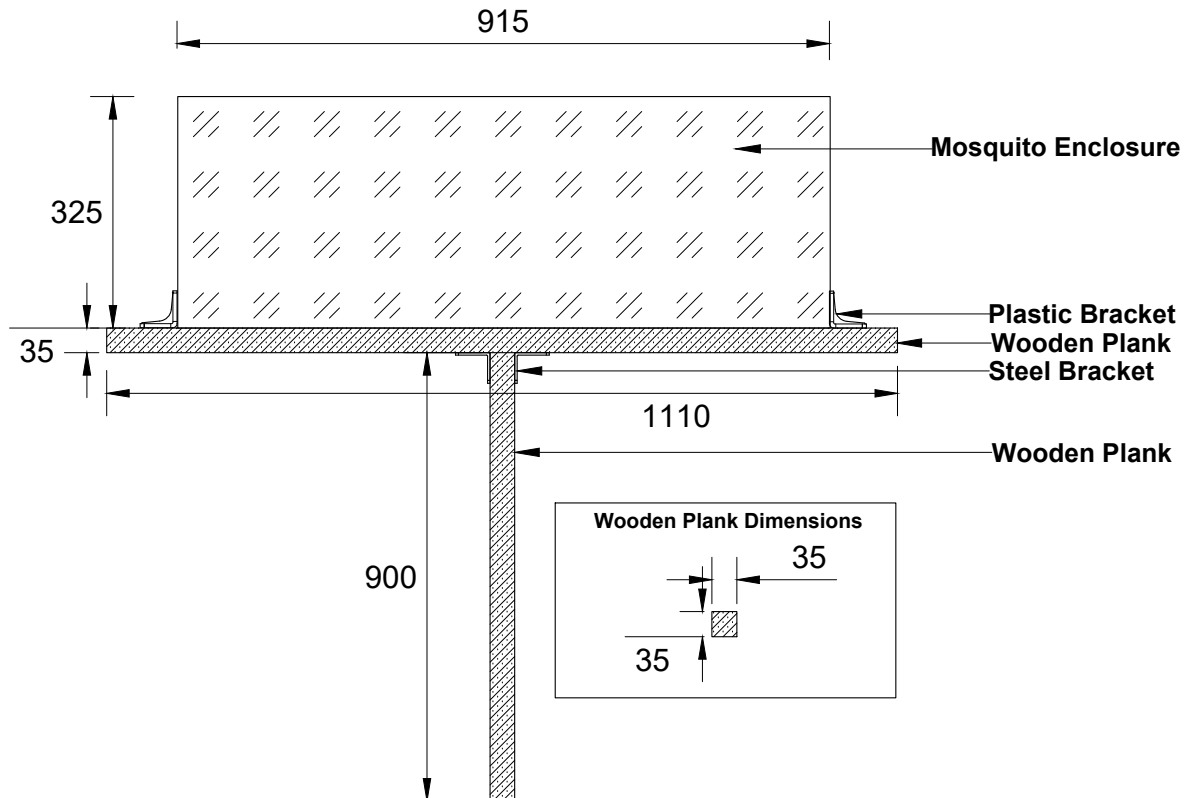


**Figure 9.**
**System positioning bracket.**

### 3.4.3   Camera mounting

The camera was mounted on the bracket in Figure 9 using a 3D printed camera mount. The camera mount was designed to be adjustable to ensure that the camera can be positioned in the centre of the mosquito enclosure. The camera was design in computer aided design (CAD). The camera mount is shown in Figure 10.

### 3.4.4   Laser turret

The specific geometry of the laser turret was designed with the goal to practically obtain a sufficiently small lateral step size of the laser on the back wall of the mosquito enclosure,

(a) Camera mount.                          (b) Camera base clamp.

**Figure 10.**
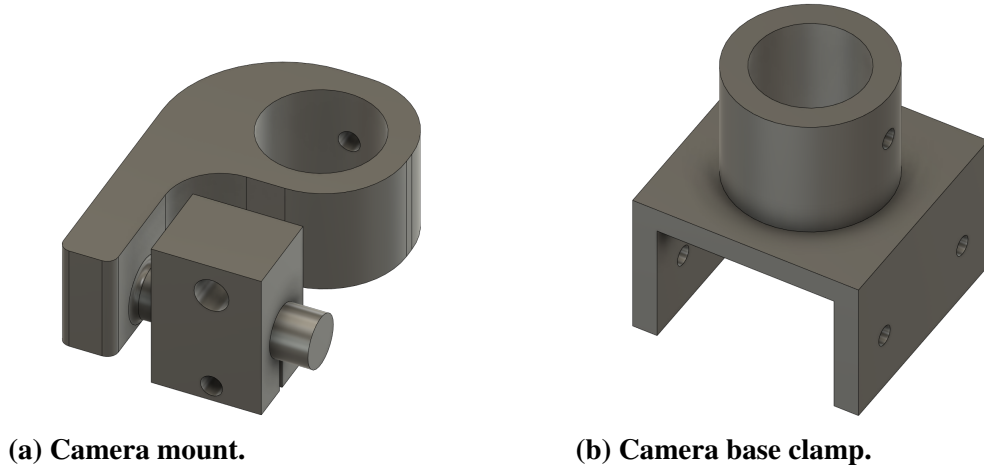**The camera mount and camera base clamp.**

while maintaining sufficient speed. The dimensions of the mirrors were chosen for practicality as $30 \times 15 \times 3$ mm.

### a.   Angular step size

The turret axis on which the laser beam is first incident on will be referred to as the first axis and the other axis will be referred to as the second axis for the remainder of this discussion. The rotation range of the first axis is bounded by the geometry of the turret since the laser beam reflected from the first axis must be incident on the second axis. With the chosen mirror dimensions, the maximum angle through which the first axis can rotate is $40°$, which rotates the laser beam through $80°$. This was determined geometrically in the to scale drawing in Figure 11. To accommodate for the mounting and alignment of the mirrors as well as other imperfections it was decided to consider a maximum rotation of $20°$, resulting in $\theta_{laser}^{max} = 40°$. The rotation range of the second axis is not bounded by the geometry of the laser turret since the laser beam reflected from the second axis is incident on the mosquito enclosure. Therefore, the laser turret will be oriented such that the first axis moves the laser beam parallel to the shorter height of the mosquito enclosure and the second axis moves the laser beam parallel to the longer width of the mosquito enclosure.

The minimum distance $z_{min}$ between the laser turret and the mosquito enclosure can be determined by rearranging Equation 7 to give

$$z_{min} = \frac{d}{\tan\left(\theta_{laser}^{max}\right)} = \frac{38 \text{ cm}}{\tan\left(40°\right)} = 45.3 \text{ cm}, \tag{8}$$

where $d = 38$ cm is the height of the mosquito enclosure and $\theta_{laser}^{max} = 40°$ is the maximum angle through which the first axis of the laser can rotate. The maximum step resolution $\theta_{step}^{max}$ of the motors can be calculated by substituting $z_{min}$ and the required lateral step size $d_{min}$ into an equation derived from Equation 7 given by

$$\theta_{step}^{max} = \frac{1}{2}\arctan\left(\frac{d_{min}}{z_{min}}\right) = \frac{1}{2}\arctan\left(\frac{2\text{mm}}{45.3\text{cm}}\right) = \frac{0.253°}{2} = 0.126°. \tag{9}$$
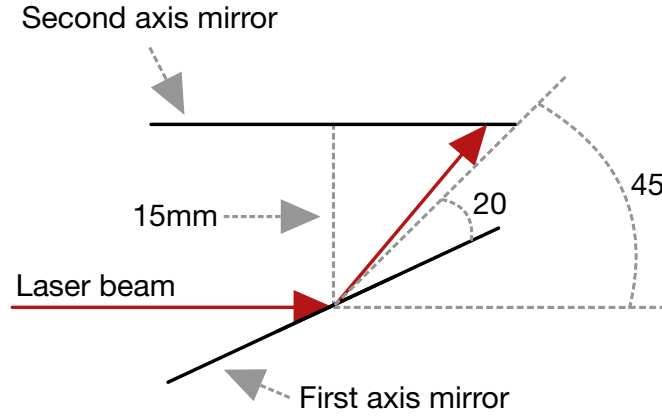
**Figure 11.**
**Rotation range of first axis with chosen mirror dimensions.**

The division by two in Equation 9 is done to compensate for the fact the laser beam angle is double the mirror angle.

### b. Angular speed

The maximum angular speed $\omega_{max}$ required by the motors can be determined with $d_{max} = 1$ m, $v_{laser} = 1\,\mathrm{m\,s^{-1}}$, and $z_{min} = 45.3$ cm. The maximum angle through which the turret must rotate is

$$\theta_{motor}^{max} = \frac{1}{2}\arctan\left(\frac{d_{max}}{z_{min}}\right) = \frac{1}{2}\arctan\left(\frac{1\,\mathrm{m}}{45.3\,\mathrm{cm}}\right) = \frac{65.6°}{2} = 32.8°. \tag{10}$$

The maximum angular speed required to move the laser beam at the lateral speed $v_{laser} = 1\,\mathrm{m\,s^{-1}}$ is

$$\omega_{max} = v_{laser} \times \theta_{motor}^{max} = 32.8°\,\mathrm{s^{-1}} = 0.572\,\mathrm{rad\,s^{-1}}. \tag{11}$$

The results in a required motor revolutions per minute (RPM) of

$$\omega_{max} = 0.572\,\mathrm{rad\,s^{-1}} \times \frac{60}{2\pi} = 5.5\,\mathrm{RPM}. \tag{12}$$

### c. Torque

The required torque $\tau$ was calculated using

$$\tau = I\alpha, \tag{13}$$

where $I$ is the moment of inertia of the load, which is the mirror, and $\alpha$ is the required angular acceleration.

The moment of inertia of the mirror was calculated using the equation for the moment of inertia of a rectangular prism with rotation about its $x$-axis as seen in Figure 12 given by

$$I = \frac{1}{12}M\left(b^2 + c^2\right), \tag{14}$$

where $M$ is the mass of the rectangular prism and $b$ and $c$ are the dimensions of the rectangular prism. The mass of the rectangular prism $M$ will be calculated using the density of glass $\rho = 2500\,\mathrm{kg\,m^{-3}}$ and the volume of the rectangular prism $V$ given by
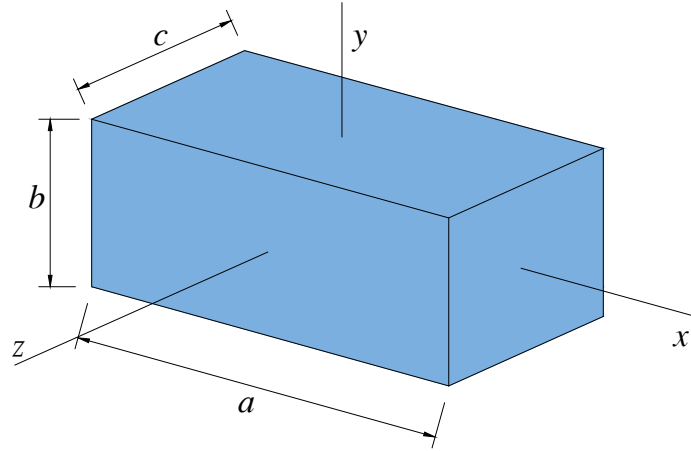
$$V = abc, \tag{15}$$

**Figure 12.**
**Axes and dimensions of a rectangular prism.**

where $a$, $b$, and $c$ are the dimensions of the rectangular prism. The dimensions of the rectangular prism are given by the chosen mirror dimensions as $a = 30\,\text{mm}$, $b = 15\,\text{mm}$, and $c = 3\,\text{mm}$. The mass of the mirror $M$ is

$$M = \rho V = 2500\,\text{kg}\,\text{m}^{-3} \times (30\,\text{mm})(15\,\text{mm})(3\,\text{mm}) = 3.38\,\text{g}. \tag{16}$$

Thus, the moment of inertia of the mirror is

$$I = \frac{1}{12}M\left(b^2 + c^2\right) = \frac{1}{12}(3.38\,\text{g})\left((15\,\text{mm})^2 + (3\,\text{mm})^2\right) = 6.59 \times 10^{-8}\,\text{kg}\,\text{m}^{-2}. \tag{17}$$

The required angular acceleration of the motor $\alpha$ is calculated with

$$\alpha = \frac{\Delta\omega}{\Delta t}, \tag{18}$$

where $\Delta\omega = \omega_{max} - 0$ is the change in angular velocity and $\Delta t$ is the change in time. The change in time $\Delta t$ was determined by assuming the laser must be able to accelerate from a stand still to $v_{laser} = 1\,\text{ms}^{-1}$ extremely rapidly to ensure that the motors could respond to the irregular flight pattern of a mosquito. It was assumed that this acceleration must occur within 10 ms. Thus, the required angular acceleration $\alpha$ is

$$\alpha = \frac{\Delta\omega}{\Delta t} = \frac{0.572\,\text{rad}\,\text{s}^{-1}}{0.01\,\text{s}} = 57.2\,\text{rad}\,\text{s}^{-2}. \tag{19}$$

Given the calculated moment of inertia and angular acceleration the required torque $\tau$ is

$$\tau = I\alpha = 6.59 \times 10^{-8}\,\text{kg}\,\text{m}^{-2} \times 57.2\,\text{rad}\,\text{s}^{-2} = 3.77 \times 10^{-6}\,\text{N}\,\text{m}. \tag{20}$$

**d.   Motor and driver selection**
Typical stepper motors have a basic step angle of 1.8°. This does not meet the required step resolution $\theta_{step}^{max} = 0.253°$, however the step angle can be reduced by using microstepping. Microstepping works by interpolating the pulse width modulation (PWM) signal to the stepper motor driver to produce intermediate step positions between the basic step positions. Stepper motor drivers that support up to 16 microsteps are readily available. Drivers that support up to 256 microsteps are also available. The step angle $\theta_{step}$ of a stepper motor using 16 microsteps

is

$$\theta_{step} = \frac{1.8°}{16} = 0.1125°, \tag{21}$$

which is sufficient to meet the required maximum step resolution $\theta_{step}^{max} = 0.127°$. Therefore, a typical stepper motor with a basic step angle of 1.8° operated using at least 16 microsteps will be sufficient.

A drawback of using microstepping is that the effective torque of the stepper motor is reduced with each increase in microstepping. The torque reduction due to microstepping can be calculated with

$$\tau_{eff} = \tau_{rated} \sin\left(\frac{\pi S}{2}\right), \tag{22}$$

where

- $\tau_{eff}$ is the effective torque of the stepper motor for a specific amount of microstepping.
- $\tau_{rated}$ is the rated torque of the stepper motor.
- $S$ is the reciprocal of the number of microsteps.

The effective torque of a stepper motor for the full range of microsteps available is tabulated in Table 2. It can be seen that microstepping has a drastic impact of the effective torque of stepper motors. Thus, the required torque will need to be calculated for the chosen microstepping operation.

| Microsteps | Effective torque |
|:----------:|:----------------:|
| 1 | 100% |
| 2 | 70.71% |
| 4 | 38.27% |
| 8 | 19.51% |
| 16 | 9.8% |
| 32 | 4.91% |
| 64 | 2.45% |
| 128 | 1.23% |
| 256 | 0.61% |

**Table 2.**
**Effective torque of stepper motor for full range of microsteps.**

The DRV8825 stepper motor driver was chosen since it is readily available and supports up to 32 microsteps. The DRV8825 stepper motor driver has a maximum current rating of 2.5 A and a maximum voltage rating of 45 V. The maximum microstepping mode will be utilised for increased resolution in the control of the laser. The required torque with 32 microsteps is

$$\tau_{required} = 3.77 \times 10^{-6}\,\mathrm{Nm} \times \frac{100\%}{4.91\%} = 7.68 \times 10^{-5}\,\mathrm{Nm}. \tag{23}$$

A bipolar NEMA8 stepper motor with specifications shown in Table 3 and a pull-out torque curve at half step shown in Figure 13 was chosen as the motors for the laser turret. From the pull-out torque curve in Figure 13, the maximum rated speed of the motor is given as 13 000

| Basic step angle | 1.8° |
|---|---|
| Holding torque | $180\,\mathrm{g\,cm} = 1.77 \times 10^{-2}\,\mathrm{Nm}$ |
| Rated voltage | 3.9 V |
| Rated current | 0.6 A |

**Table 3.**
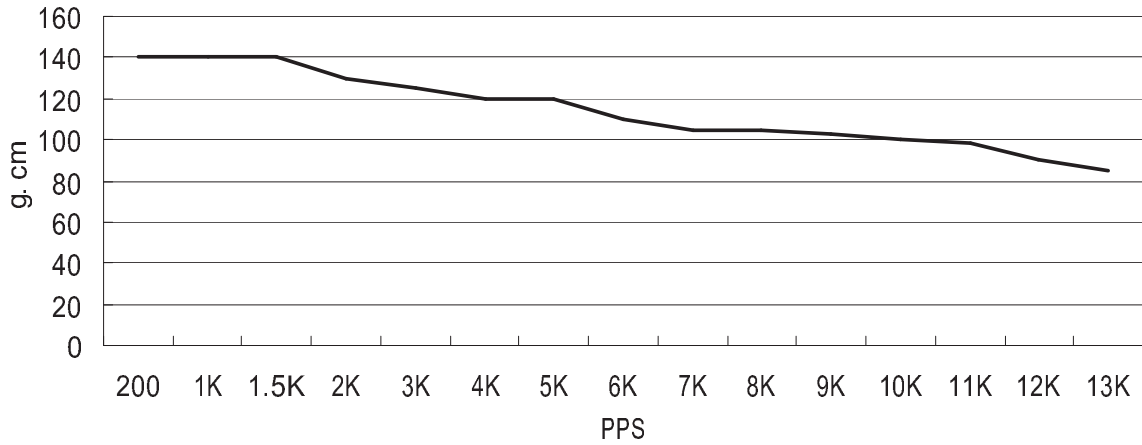**Specifications of the NEMA8 stepper motor.**



**Figure 13.**
**Pull-out torque curve of the stepper motor at half step with 24 V and 0.6 A.**

points per second at half step $0.9°$ which is

$$13\,000 \div \frac{360°}{0.9°} = 32.5\,\mathrm{RPM}, \tag{24}$$

with a pull-out torque greater than $100\,\mathrm{g\,cm} = 9.8 \times 10^{-3}\,\mathrm{Nm}$. The effective torque of the stepper motor at maximum speed with 32 microsteps is

$$\tau_{eff} = 9.8 \times 10^{-3}\,\mathrm{Nm} \times 4.91\% = 4.81 \times 10^{-4}\,\mathrm{Nm}. \tag{25}$$

The effective torque of the stepper motor at maximum speed is greater than the required torque for the laser turret, therefore the NEMA8 stepper motor is suitable for the laser turret and was also chosen for its compactness which can be seen in Figure 14.

### e.  Laser selection and activation circuit

A low-powered 5 mW red laser with adjustable focus was chosen for the laser turret. The laser has an operating voltage range of $2.8 - 5.2$ V. A low-powered laser was chosen for safety. The adjustable focus enabled the laser beam to be focused to a small area on the back wall of the mosquito enclosure. The laser must be red since the image processing will be done using only the red channel of the red, green, blue (RGB) video frame.

The 3.3 V GPIO of the Nvidia Jetson Nano cannot supply enough current to power the laser. A switching transistor circuit was designed to enable the laser to be turned on and off from the Nvidia Jetson Nano GPIO. The schematic for the switching transistor circuit can be seen in Figure 15. When the input signal is high the transistor is switched on and the laser is powered.
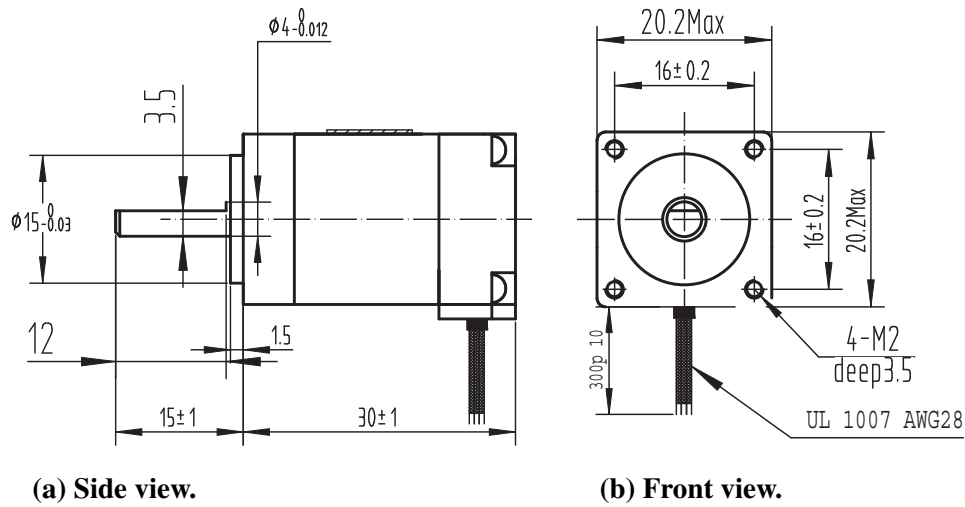
(a) Side view.

(b) Front view.

**Figure 14.**
**Dimensions of the NEMA8 stepper motor.**

When the input signal is low the transistor is switched off and the laser is not powered. In Figure 15, $R1$ is a pull-down resistor to ensure that the input signal is low when the Nvidia Jetson Nano is not driving the input signal. $R1$ was chosen as $56\,k\Omega$ to limit the current drawn from the Nvidia Jetson Nano GPIO. With $R1 = 56\,k\Omega$ the current drawn from the GPIO is $\frac{3.3\,V}{56\,k\Omega} = 59\,\mu A$. $R2$ and $D1$ represent the laser.



**Figure 15.**
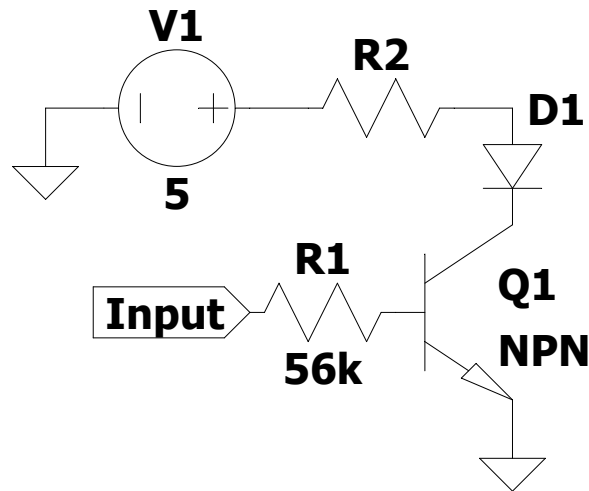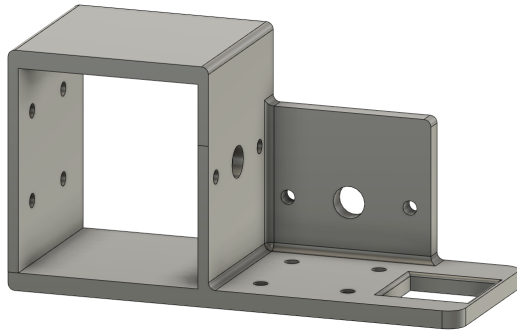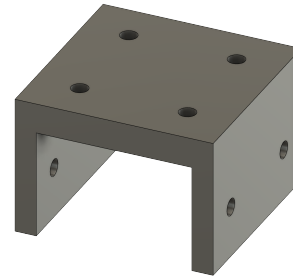**Switching transistor circuit.**

**f. Laser turret CAD** The laser turret was designed in CAD using Autodesk Fusion 360. The laser turret was designed to be mounted on the bracket shown in Figure 9. The main component and subcomponents of the laser turret can be seen in Figure 16 and Figure 17 respectively.
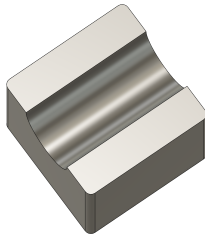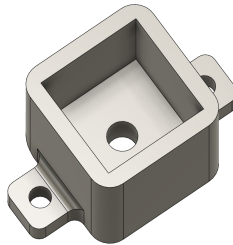
(a) Laser turret main housing.



(b) Laser turret clamp.

**Figure 16.**
**Laser turret main components.**



(a) Laser mount.



(b) Motor mount.
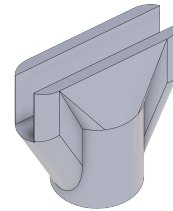


(c) Mirror holder.

**Figure 17.**
**Laser turret subcomponents.**

## 3.5 Software design

### 3.5.1 Mosquito and laser detection

It was assumed that the mosquitoes would be the only dark blobs in the enclosure and that the laser's reflections would be the only bright blobs in the enclosure. The video frame will be cropped such that only the white back surface of the enclosure is visible. To reduce the computational complexity of the image processing only the red channel of the RGB frame will be used, hence, the necessity of the red laser.

The mosquito detection was designed to detect dark blobs in the enclosure and would not be able to distinguish between mosquitoes and other dark blobs. The laser detection was designed to detect bright blobs in the enclosure and would not be able to distinguish between the laser's reflections and other bright blobs. The mosquito and laser detection processes are similar in nature. The basic detection process flow is shown in Figure 18. The red channel of

**Figure 18.**
**Detection process flow.**

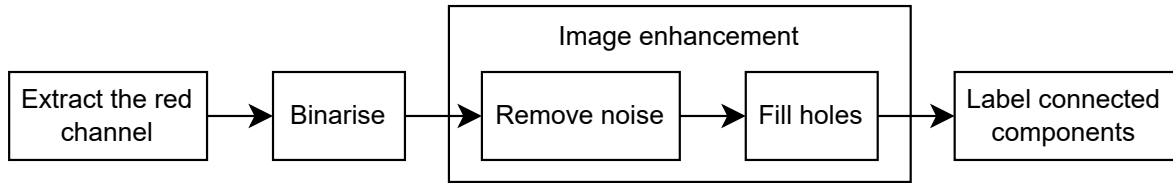the RGB frame will be extracted, which will be referred to as the frame for the rest of this discussion. The difference between the mosquito and laser detection processes occur in the binarisation step. A copy of the frame is made before binarising so that both the mosquito and laser detection can be performed since the frame will be overwritten in the image processing steps in pursuit of efficiency.

### a.    Binarisation
The frame will be binarised using a threshold value. The exposure of the camera will be fixed and the lighting in the mosquito enclosure will be controlled, thus the threshold value will be fixed. To detect mosquitoes the frame will be binarised with a less than threshold since the mosquitoes will be darker than the white background. To detect the laser's reflections the frame will be binarised with a greater than threshold since the laser and its reflections will be brighter than the white background. The ideal result of the binarisation is an image where the mosquitoes and the laser's reflections are the only foreground pixels in their respective copies of the frame. This will not be the case since the mosquitoes and the laser's reflections can not be perfectly isolated using a threshold value since the frame will contain noise from the camera sensor.

### b.    Image enhancement
The binarised frames are subject to noise, holes, and erroneously joint or disjointed sections. This can be resolved using morphological operations. The goal of the morphological operations is to ensure that all the foreground pixels that correspond to a detection are connected, that there are no foreground pixels that do not correspond to a detection, and that there are no erroneously joint or disjoint foreground pixels. There must be a one to one correspondence between the groups of connected foreground pixels and the objects that must be detected. The connectivity of pixels will be explained in paragraph c. Morphological operations work by passing a structuring element over the pixels in an image and performing a logical operation on the pixels in the image that are covered by the structuring element.

The noise and erroneously joint sections are removed using the morphological opening operation. Opening has been formally defined in subsubsection 3.2.2. The opening of $A$ by $B$ is the union of all translations of $B$ that are completely contained in $A$ as illustrated in Figure 4a. The holes and erroneously disjointed sections are removed using the morphological closing operation. Closing has been formally defined subsubsection 3.2.2. The closing of $A$ by $B$ is the complement of the union of all translations of $B$ that do not overlap $A$ as illustrated in Figure 4b.

The structuring element must be identical independent of orientation in a 2D image since

the orientation of the mosquitoes will be unknown. The structuring element will therefore be a disc. The size of the disc will be adjusted independently for the opening and closing operations for both the mosquito and laser detection. The structuring elements' size will be adjustable at runtime such that they can be tuned to provide optimal performance.

**c.    Connected components labelling**

From the enhanced binary frames the co-ordinates of the connected foreground pixels must be obtained. This is done using connected components labelling (CCL). CCL is the process of assigning a label to each foreground pixel in a binary image such that connected pixels have the same label. The connectivity of pixels are defined using either four or eight connectivity. Four connectivity defines the neighbourhood of a pixel as the pixels to the left, right, top, and bottom of the pixel. Eight connectivity defines the neighbourhood of a pixel as the pixels to the left, right, top, bottom, top left, top right, bottom left, and bottom right of the pixel. The CCL connectivity is shown in Figure 19.
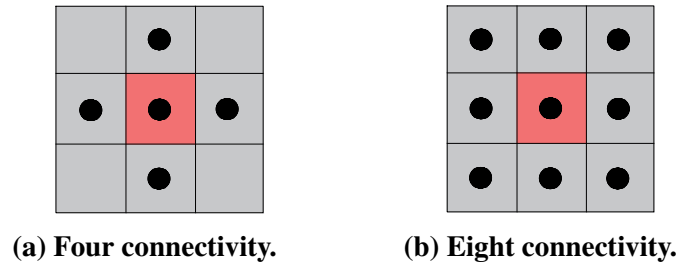


(a) **Four connectivity.**          (b) **Eight connectivity.**

**Figure 19.**
**Connected components labelling connectivity.**

Various algorithms exist to perform CCL. Two of the most commonly used algorithms are the two-pass algorithm and the breadth-first single-pass algorithm. The two-pass algorithm is generally faster for images with large connected components and the breadth-first single-pass algorithm is generally faster for images with small connected components. The nature of the mosquito and laser detection is such that the connected components will be small, thus the breadth-first single-pass algorithm was chosen. The breadth-first single-pass algorithm is presented in Algorithm 1.

### 3.5.2   Distinguishing between the laser's reflections

The laser must shine through the front glass of the mosquito enclosure. This results in various reflections of the laser beam. There are three predominate reflections that can not be distinguished using a brightness threshold. These reflections are:

- The reflection of the laser beam off the front glass at the origin of the laser turret, resulting from scattered light off the turret mirrors.
- The reflection of the laser beam incident on the front glass.
- The reflection of interest, which is the reflection of the laser beam incident on a mosquito or the back surface of the enclosure.

---

**Algorithm 1** Breadth-First Single-Pass Connected Component Labelling

---

1: **procedure** $\mathrm{S\,I\,N\,G\,L\,E\,P\,A\,S\,S\,C\,C\,L}(image)$
2:      $rows, cols \leftarrow$ dimensions of $image$
3:      $label \leftarrow 1$
4:      $labels \leftarrow$ 2D array of zeros of size $rows \times cols$
5:      **for** $y \leftarrow 0$ to $rows - 1$ **do**
6:          **for** $x \leftarrow 0$ to $cols - 1$ **do**
7:              **if** $image[y][x] = 1$ and $labels[y][x] = 0$ **then**
8:                  ExploreBlob$(image, labels, x, y, label)$
9:                  $label \leftarrow label + 1$
10:             **end if**
11:         **end for**
12:     **end for**
13: **end procedure**
14:
15: **procedure** $\mathrm{E\,X\,P\,L\,O\,R\,E\,B\,L\,O\,B}(image, labels, x, y, label)$
16:     $queue \leftarrow$ empty queue
17:     Enqueue$(queue, (x, y))$
18:     **while** not empty$(queue)$ **do**
19:         $(x, y) \leftarrow$ Dequeue$(queue)$
20:         **if** $x < 0$ or $x \geq cols$ or $y < 0$ or $y \geq rows$ or $image[y][x] = 0$ or $labels[y][x] > 0$
   **then**
21:             continue
22:         **end if**
23:         $labels[y][x] \leftarrow label$
24:         **for** each neighbor $(nx, ny)$ of $(x, y)$ **do**
25:             Enqueue$(queue, (nx, ny))$
26:         **end for**
27:     **end while**
28: **end procedure**

---

The reflection off the front glass at the origin of the laser turret remains stationary since the laser turret itself is stationary with only the angles of the mirrors changing. Thus, this reflection can be manually distinguished by drawing a bounding box around its position in the frame. The remaining two reflections are distinguished using the geometric properties of the system positioning. The $yz$-plane of the system is shown in Figure 20 showing the potential laser beam paths for various turret angles. The laser turret is slightly below the camera which is exaggerated in Figure 20 for illustration purposes. It can be seen in Figure 20 that if both detections are at or below the camera origin the reflection of interest is closer to the camera origin. If both reflections are at or above the camera origin then the reflection of interest is further from the camera origin. If the reflections are on either side of the camera origin the reflection of interest is the reflection above the camera origin. This logic is easily extended to three dimensions by considering the $yz$-plane and $xz$-plane independently.
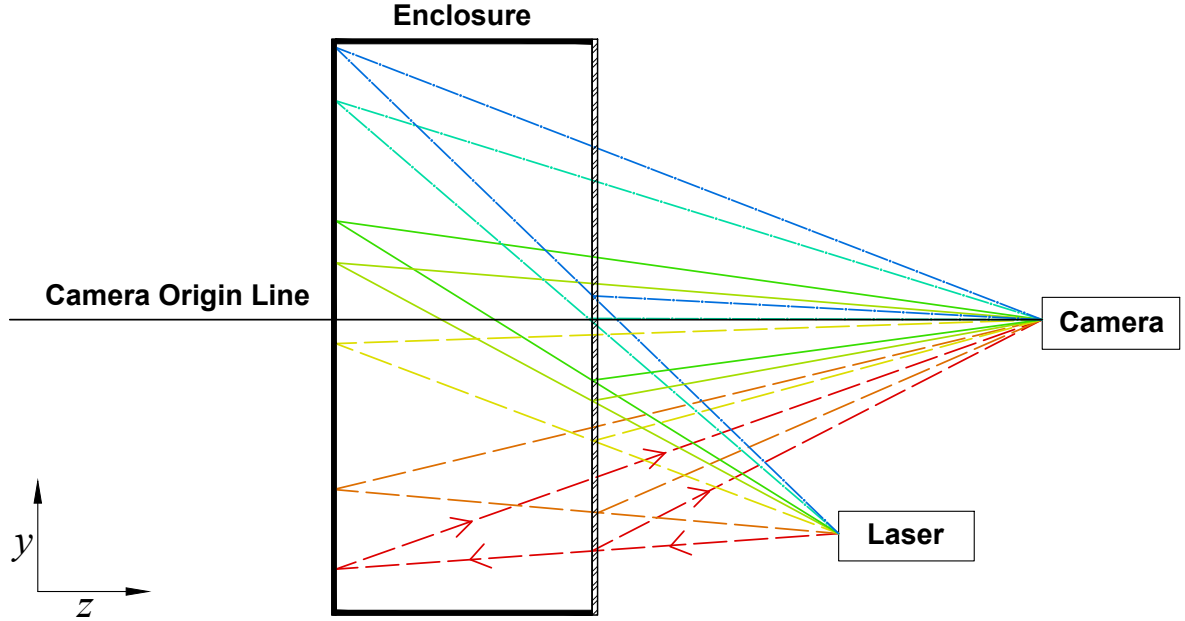
**Figure 20.**
**Distinguishing between the laser's reflections.**

### 3.5.3   *Mosquito tracking*

The mosquito tracking was performed using the SORT algorithm. The SORT algorithm is based on the tracking-by-detection paradigm, which means it relies on an external object detector. The object detector used was the mosquito detection algorithm described in sub-subsection 3.5.1. The SORT algorithm associates the detections across frames to create tracks.

**a.   Tracks**

The mosquito tracks are created using the Kalman filter. The theoretical background of Kalman filter is discussed in subsubsection 3.2.3. The kinematic equation representing the flight of a mosquito must be defined to determine the state space of the Kalman filter. The kinematic equation used to model the flight of a mosquito is

$$\mathbf{x}_k = \mathbf{x}_{k-1} + \dot{\mathbf{x}}_{k-1}\Delta t + \frac{1}{2}\ddot{\mathbf{x}}_{k-1}\Delta t^2 \, , \tag{26}$$

where $\mathbf{x}_k$ is the position vector defined by the 2D pixel co-ordinate of a mosquito $\begin{bmatrix} x & y \end{bmatrix}^{\mathrm{T}}$. A constant velocity model with acceleration as noise will be used since mosquitoes have an erratic flight pattern. Therefore, the set of differential equations describing the state space of the Kalman filter is

$$\mathbf{x}_k = \mathbf{A}\mathbf{x}_{k-1} + \mathbf{B}\mathbf{u}_k + \mathbf{w}_k$$

$$\begin{bmatrix} x_k \\ y_k \\ \dot{x}_k \\ \dot{y}_k \end{bmatrix} = \begin{bmatrix} 1 & 0 & \Delta t & 0 \\ 0 & 1 & 0 & \Delta t \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_{k-1} \\ y_{k-1} \\ \dot{x}_{k-1} \\ \dot{y}_{k-1} \end{bmatrix} , \tag{27}$$

where $\mathbf{u}_k$ is the input vector, which is the acceleration of the mosquito according to its kinematic flight model defined in Equation 26. However, this acceleration is unknown and will be compensated for by the process noise $\mathbf{w}_k$, thus $\mathbf{u}_k = \mathbf{0}$. The process noise $\mathbf{w}_k$ is assumed to be drawn from a zero mean multivariate normal distribution $\mathcal{N}$ with covariance $\mathbf{Q}_k$. Hence, $\mathbf{w}_k \sim \mathcal{N}\left(0, \mathbf{Q}_k\right)$. Thus, $\mathbb{E}[w_k] = 0$ and $\mathbb{E}[w_k w_k^{\mathrm{T}}] = \mathbf{Q}_k$. The process noise covariance matrix $\mathbf{Q}$ is given by

$$\mathbf{Q} = \begin{bmatrix} \sigma_x^2 & 0 & \sigma_x \sigma_{\dot{x}} & 0 \\ 0 & \sigma_y^2 & 0 & \sigma_y \sigma_{\dot{y}} \\ \sigma_{\dot{x}} \sigma_x & 0 & \sigma_{\dot{x}}^2 & 0 \\ 0 & \sigma_{\dot{y}} \sigma_y & 0 & \sigma_{\dot{y}}^2 \end{bmatrix}, \tag{28}$$

where $\sigma_x$ and $\sigma_{\dot{x}}$ are the standard deviations of the position and velocity, respectively. The standard deviation of the position is defined as the standard deviation of the acceleration $\sigma_{\ddot{x}}$ multiplied by $\frac{1}{2}\Delta t^2$ since this is the effect that the acceleration will have on the position as shown in Equation 26. Similarly, the standard deviation of the velocity is defined as $\Delta t \sigma_{\ddot{x}}$. Therefore, the process noise covariance can be written as

$$\mathbf{Q} = \sigma_{\ddot{x}}^2 \begin{bmatrix} \frac{\Delta t^4}{4} & 0 & \frac{\Delta t^3}{2} & 0 \\ 0 & \frac{\Delta t^4}{4} & 0 & \frac{\Delta t^3}{2} \\ \frac{\Delta t^3}{2} & 0 & \Delta t^2 & 0 \\ 0 & \frac{\Delta t^3}{2} & 0 & \Delta t^2 \end{bmatrix}. \tag{29}$$

The position of a mosquito is the only component of the state that is measured by the detection system. Thus, the observation matrix $\mathbf{H}$ is

$$\mathbf{H} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}. \tag{30}$$

The measurement noise covariance matrix $\mathbf{R}$ is

$$\mathbf{R} = \begin{bmatrix} \sigma_x^2 & 0 \\ 0 & \sigma_y^2 \end{bmatrix}, \tag{31}$$

where $\sigma_x^2$ and $\sigma_y^2$ are the variances of the position detected. The initial state covariance matrix $\mathbf{P}$ is given by

$$\mathbf{P} = \begin{bmatrix} \sigma_x^2 & 0 & 0 & 0 \\ 0 & \sigma_y^2 & 0 & 0 \\ 0 & 0 & \dot{x}_{max}^2 & 0 \\ 0 & 0 & 0 & \dot{y}_{max}^2 \end{bmatrix}. \tag{32}$$

The Kalman filter is initialised with the position of the mosquito as the initial state $\mathbf{x}_0$. The position of the mosquito in the next frame $\hat{\mathbf{x}}_{k|k-1}$ is predicted using the adapted predict equation of the Kalman filter $\hat{\mathbf{x}}_{k|k-1} = \mathbf{A}_k \mathbf{x}_{k-1|k-1}$. This prediction will be used in the association step before the update step. The update step of the Kalman filter is performed using the update equation of the Kalman filter $\mathbf{x}_{k|k} = \mathbf{x}_{k|k-1} + \mathbf{K}_k \left( \mathbf{z}_k - \mathbf{H}_k \mathbf{x}_{k|k-1} \right)$, where $\mathbf{K}_k$ is the Kalman gain and $\mathbf{z}_k$ is the measurement vector. The measurement vector $\mathbf{z}_k$ is the position of the mosquito detected in the current frame.

### b. Association

The association of tracks and detections will be done using the Hungarian algorithm, which is an optimised optimal assignment algorithm. The Hungarian algorithm has a time complexity

of $O(n!)$ compared to the time complexity naïve approach $O(n^3)$. The Hungarian algorithm works as follows:

**Generate a cost matrix.**

Represent the problem with a square $n \times n$ matrix $\mathbf{C}$ where the element $\mathbf{C}_{i,j}$ represents the cost of assigning track $i$ to detection $j$. The cost function is the euclidean distance between the predicted location and the detected location of the mosquito.

**Row and column reduction.**

Subtract the minimum value in each row from each element in the row. Perform the same operation for each column. This will produce a matrix with at least one zero in each row and column.

**Cover zeros and check for optimality.**

Cover all the zeros in the matrix using the minimum number of horizontal and vertical lines. If the minimum number of covering lines is $n$, an optimal assignment exists among the zeros, and the algorithm can proceed to the assignment phase. If not, you need to adjust the matrix and repeat from this step.

**Adjust the matrix.**

Find the smallest element that is not covered by any line. Subtract this element from all uncovered elements, and add it to all elements at the intersections of the covering lines. Return to *cover zeros and check for optimality*.

**Assignment.**

Once an optimal assignment is found, read the assignment from the zeros in the matrix. ~~Each row will have exactly one zero, and each column may have one zero. If a column has no zeros, the track corresponding to that column is unassigned. If a column has more than one zero, the track corresponding to that column is assigned to the detection corresponding to the zero that is covered by a star. If a row has more than one zero, the track corresponding to that row is assigned to the detection corresponding to the zero that is covered by a star.~~

### 3.5.4  *Laser turret control system*

The laser turret control system is a closed-loop PID controller. A PID controller is represented by the following equation

$$u(t) = K_p e(t) + K_i \int_0^t e(\tau)d\tau + K_d \frac{de(t)}{dt}, \tag{33}$$

where $u(t)$ is the control signal, $e(t)$ is the error signal, $K_p$ is the proportional gain, $K_i$ is the integral gain, and $K_d$ is the derivative gain. The goal of the laser turret control system is to position the laser beam at the target co-ordinates. The target co-ordinates is defined by the mosquito detection and tracking system in pixels. The stepper motors are controlled in steps, thus the target pixels must be mapped to the number of steps required to move the laser beam to the target co-ordinates. The steps are defined in reference to the origin of the laser defined in subsubsection 3.3.4, which is the position when the laser beam is perpendicular to the

*xy*-plane of the mosquito enclosure. The block diagram of the laser turret control system can be seen in Figure 21. The two axes of the laser turret are controlled independently each having their own PID controller. The PID controller is implemented in software and the control signal is sent to the stepper motor drivers using GPIO.
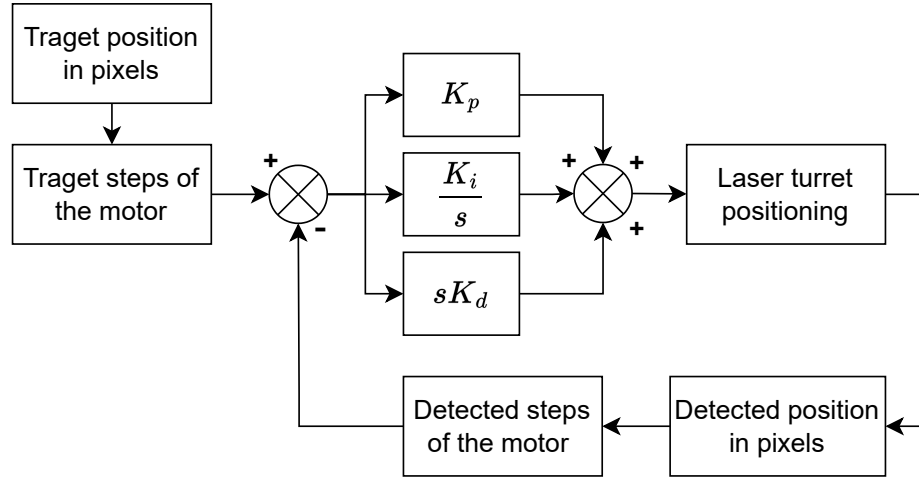


**Figure 21.**
**Laser turret control system.**

## 3.6   Software implementation and optimisation

### 3.6.1   Image processing

The resolution of the camera was set to $1280 \times 720$ and was cropped such that only the back wall of the mosquito enclosure is visible. This low resolution was selected to reduce the computational complexity of the image processing. The binarisation and image enhancement processes described in subsubsection 3.5.1 are suitable for parallelisation. Therefore, custom CUDA kernels were written to perform these operations on the Nvidia Jetson Nano GPU. The kernel for the morphological erosion operation is shown in Listing 1 to demonstrate the structure of a CUDA kernel. Utilising the GPU with kernels such as Listing 1 significantly improves the performance of the image processing system since the processing is performed concurrently across the 128 CUDA cores of the Nvidia Jetson Nano GPU. The image processing is optimised using three CUDA kernels, one for binarisation, one for morphological erosion, and one for morphological dilation. These kernels are used 10 times per frame to perform the mosquito and laser detection. Thus, this optimisation has a significant impact on the performance of the system.

```
1  __global__ void erosion(uint8_t* input,
2                          uint8_t* output,
3                          uint8_t* struct_elem,
4                          int struct_elem_radius) {
5    int x = blockIdx.x * blockDim.x + threadIdx.x;
6    int y = blockIdx.y * blockDim.y + threadIdx.y;
7
8    if (x < d_COLS && y < d_ROWS) {
9      int min_val = 255;
10     for (int i = -struct_elem_radius; i <= struct_elem_radius; i++) {
11       for (int j = -struct_elem_radius; j <= struct_elem_radius; j++) {
12         if (y + i >= 0 && y + i < d_ROWS && x + j >= 0 && x + j < d_COLS)
13         {
14           if (struct_elem[(i + struct_elem_radius) *
15                           (2 * struct_elem_radius + 1) +
16                           j + struct_elem_radius] == 1)
17           {
18             int idx = (y + i) * d_COLS + (x + j);
19             min_val = min(min_val, (int)input[idx]);
20           }
21         }
22       }
23     }
24     output[y * d_COLS + x] = min_val;
25   }
26 }
```

**Listing 1.**
**Erosion GPU kernel.**

### 3.6.2 *Mosquito tracking*

The matrix operations for the Kalman filter were implemented from first principles using loops.

### 3.6.3 *Laser turret control system*

The laser turret control system consists of various independent subcomponents that required meticulous attention to detail to successfully integrate. The integration will be discussed in subsection 3.7. The focus in this section will be on the subcomponents.

#### a.   Interfacing with stepper motor drivers
The stepper motors drivers were connected to the Nvidia Jetson Nano GPIO according to the diagram in Figure 22. The motors are enabled by setting the ENABLE PIN to a logical high. The direction of rotation is determined by the logic level of the DIRECTION PIN. The stepper motors are driven by pulsing the STEP PIN at the desired frequency for the desired number of steps.
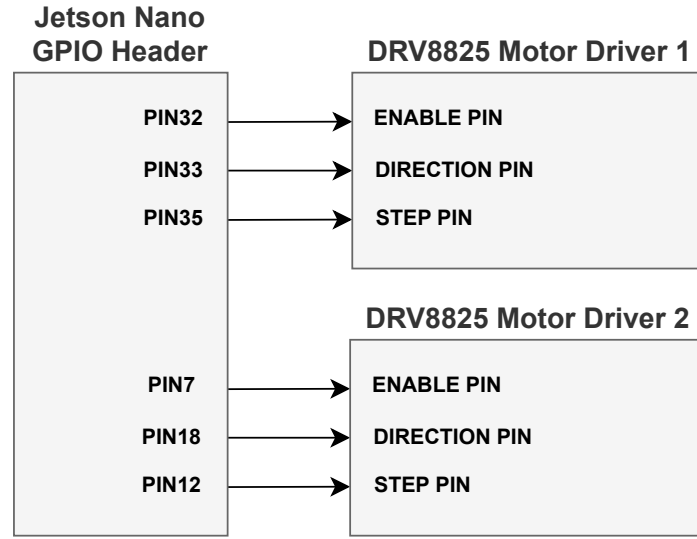
**Jetson Nano
GPIO Header**

**DRV8825 Motor Driver 1**

PIN32 ⟶ ENABLE PIN

PIN33 ⟶ DIRECTION PIN

PIN35 ⟶ STEP PIN

**DRV8825 Motor Driver 2**

PIN7 ⟶ ENABLE PIN

PIN18 ⟶ DIRECTION PIN

PIN12 ⟶ STEP PIN

**Figure 22.
Stepper motor driver connections.**

---

**Algorithm 2** Stepper Motor Control

---

1: *ENABLE PIN* ← logical high
2: *DIRECTION PIN* ← *direction*
3: **for** *step* ← 0 to *steps* − 1 **do**
4:      *STEP PIN* ← logical high
5:      delay(*period*)
6:      *STEP PIN* ← logical low
7:      delay(*period*)
8: **end for**

---

**b.   Converting pixels to steps**

The target position in pixels must be converted to the number of steps required to move the laser beam to the target position. This conversion is done using Equation 1, with the laser origin as the reference pixels. Hence, the motors steps are calculated in respect to the 45° origin position.

## 3.7   Final system integration

The system was written in C++ and integrated into a single executable system. The system is required to operate in real-time. Therefore, the mosquito tracking and control of the laser turret must occur simultaneously. To achieve this multi-threading was required. This can be seen in the software flow diagram shown in Figure 23. Thread 1 performs the laser detection, mosquito detection, and mosquito tracking. Threads 2 and 3 are used to control each axis of the laser turret. The communication between threads is performed using atomic variables. This system enables the laser turret to continue adjusting the position of each axis to reach the target position while the video frames are processed to provide feedback for the laser

turret control system and the updated position of the mosquito being targeted. The system was implemented using the `std::thread` library in C++.

The laser turret control system in threads 2 and 3 continuously poll flags that indicate if there is new laser position feedback available and if there is a new target position available. If either is available the control system breaks out of the loop that provides the pulses to drive the stepper motors. The new data is used to update required control action determined by the PID controller. The system then resumes to drive the stepper motors until the control action is completed, or new data is available.
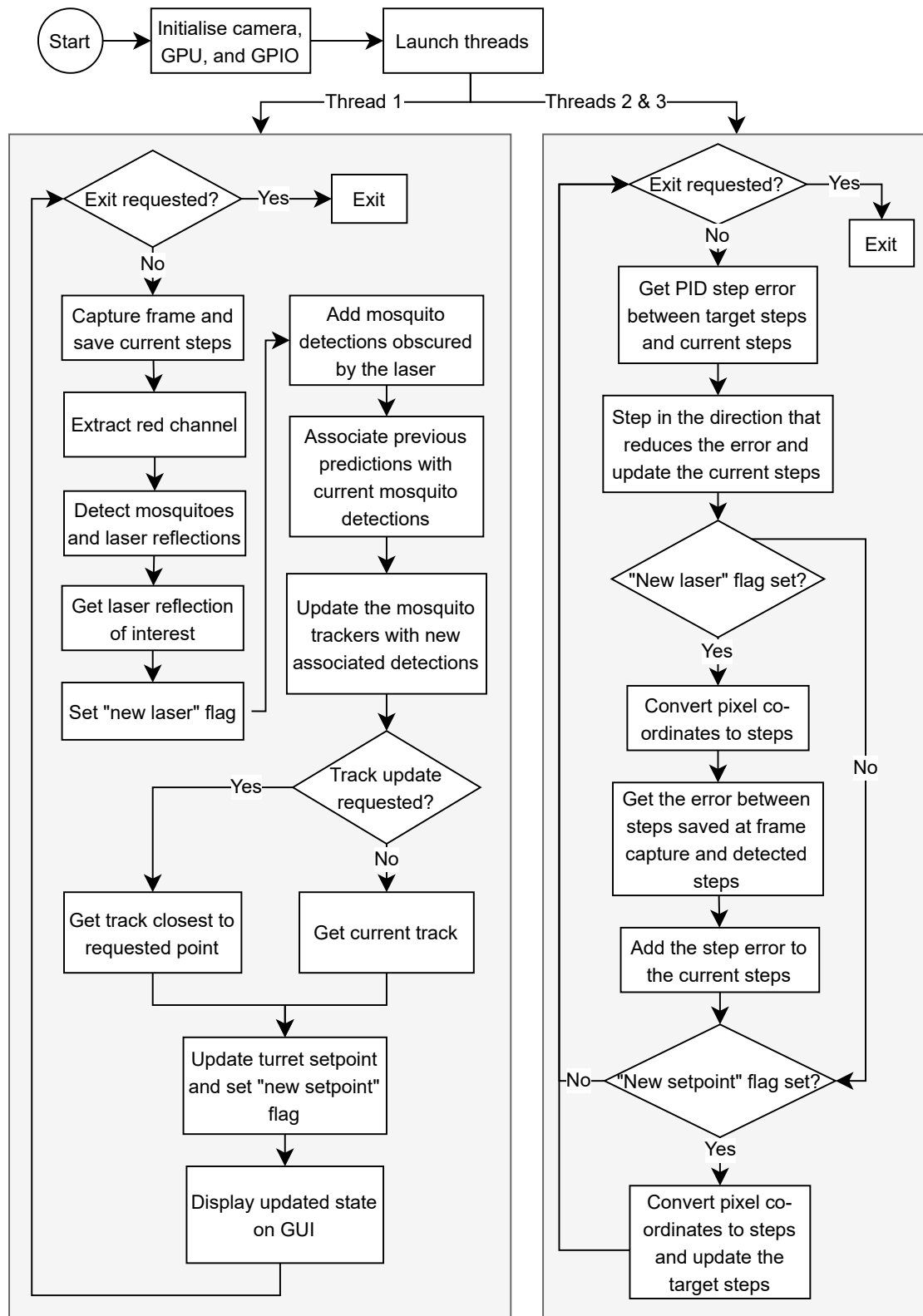
**Figure 23.**
**Software flow diagram.**

# 4. Results

## 4.1 Summary of results achieved

| Intended outcome | Actual outcome | Location in report |
|---|---|---|
| **Core mission requirements and specifications** | | |
| The system must track mosquitoes in a mosquito enclosure and illuminate a mosquito every 5 seconds. | The system tracks mosquitoes in the enclosure and illuminates a mosquito when it is stationary or moves predictably. | Section 4.2.1 |
| The laser must be able to illuminate a set point within 2 seconds accurate to within 1 millimetre. | The laser is able to illuminate set point within 2 seconds accurate to within 1 millimetre. | Section 4.2.2 |
| The system must be able to detect mosquitoes with a 90% accuracy and 5% false positive rate. The detection must be updated at least every 500 milliseconds. | The system is able to detect mosquitoes with 90% accuracy. The detection is updated every 500 milliseconds. | Section 4.2.3 |
| The system must be able to track mosquitoes with 90% accuracy of correct association between frames after 5 seconds. | The system is able to track mosquitoes with 90% accuracy or correct association between frames. | Section 4.2.4 |
| **Field condition requirements and specifications** | | |
| Mosquitoes should be in an enclosure with controlled lighting and white lining on all the sides except the front facing side. The enclosure should be at least 1 metre wide. | The enclosure has LEDs to control the lighting and white lining on all the sides except the front facing side. The enclosure is 0.9 metres wide. | sec |
| If mosquitoes cannot be obtained a suitable substitute will be used. The substitute will be a similar flying insect. | Mosquitoes and similar flying insects were obtained. Dead mosquitoes were also used. | sec |

**Table 4.**
**Summary of results achieved.**

## 4.2 Qualification tests

**Qualification test 1: Test of tracking and illuminating a mosquito**

*Objectives of the test or experiment*

The objective of this experiment is to determine whether the system can track and illuminate a mosquito in the mosquito enclosure. The requirement states that the system must track mosquitoes in the enclosure and illuminate a mosquito every 5 seconds.

*Equipment used*

The Nvidia Jetson Nano was used as the embedded platform to control the system. The Raspberry Pi Camera Module V2 was used to capture the video feed. The laser turret developed for this project was used to position the laser. The mosquito enclosure, laser turret, and camera was positioned as described in subsubsection 3.4.2. The Nvidia Jetson Nano was connected to user peripherals for user input and output. C++ code was developed specifically to capture the results for qualification test 1.

*Test setup and experimental parameters*

Mosquitoes were simulated with a small black dot attached to a white shafts. The shafts were inserted into the enclosure and moved by hand to mimic the motion of mosquitoes.

*Results or measurements*

The results from three different scenarios are presented in Figures 24 to 26. The first scenario is where the laser is set to target the detected location of the mosquito. The second scenario is where the laser is set to target the predicted location of the mosquito. In the third scenario two mosquitoes were inserted into the enclosure and the laser was set to target the predicted locations of the mosquitoes.
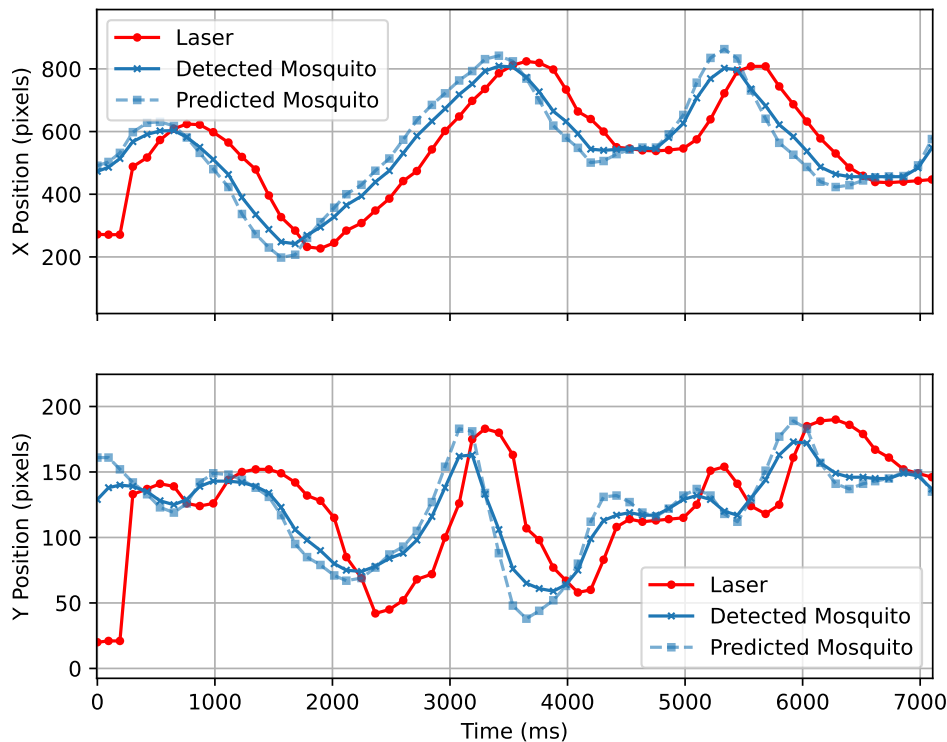


**Figure 24.**
**The position of the laser, detected mosquito, and predicted mosquito over time. The laser was set to target the detected location of the mosquito in this test.**
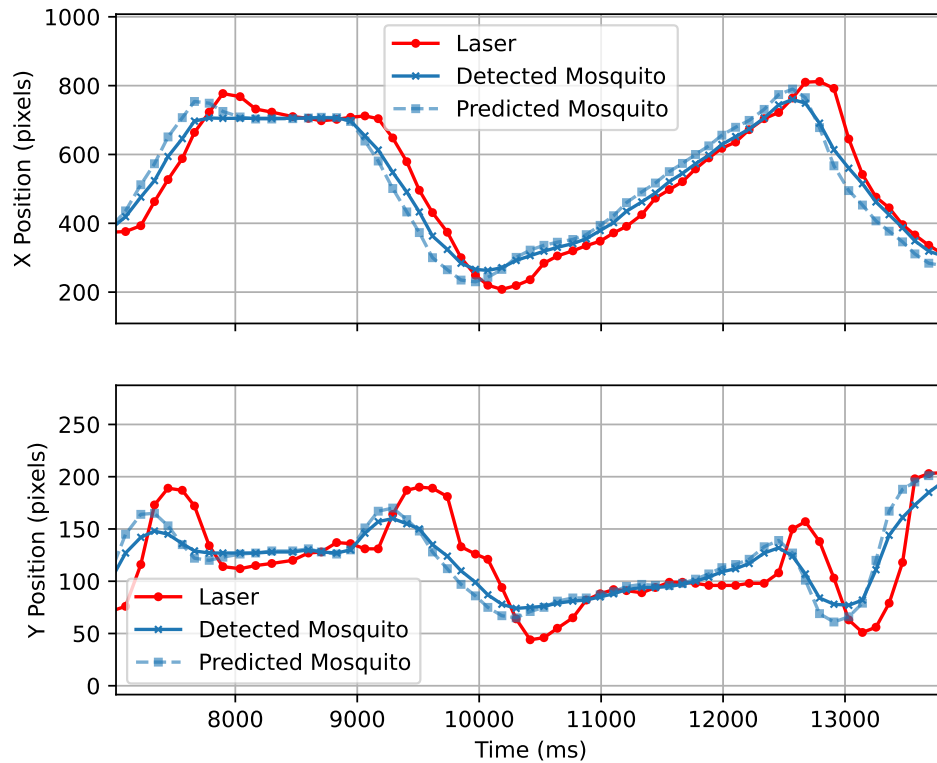
**Figure 25.**
**The position of the laser, detected mosquito, and predicted mosquito over time. The laser was set to target the predicted location of the mosquito in this test.**

*Observations*

In Figures 24 to 26, it can be seen that the system is actively adjusting the position of the laser to track the mosquitos. In Figure 24, the system was set to target the detected position of the mosquito. With this targeting technique it can be seen that the laser is trailing behind the actual position of the mosquito unless the mosquito is stationary. In Figure 25, the system was set to target the predicted position of the mosquito. With this targeting technique it can be seen that the laser reaches the actual position of the mosquito if the mosquito moves linearly for a sufficient amount of time. However, this targeting technique also results in significant overshoot when the mosquito changes direction. Figure 26 shows the system tracking two mosquitoes in the enclosure. It can be seen that the system is able to track both mosquitoes simultaneously. The system selects one of the mosquitoes to target with the laser and the other mosquito is ignored by the laser. The same tracking behaviour is observed as in Figure 25, where the laser is set to target the predicted location of the mosquito.
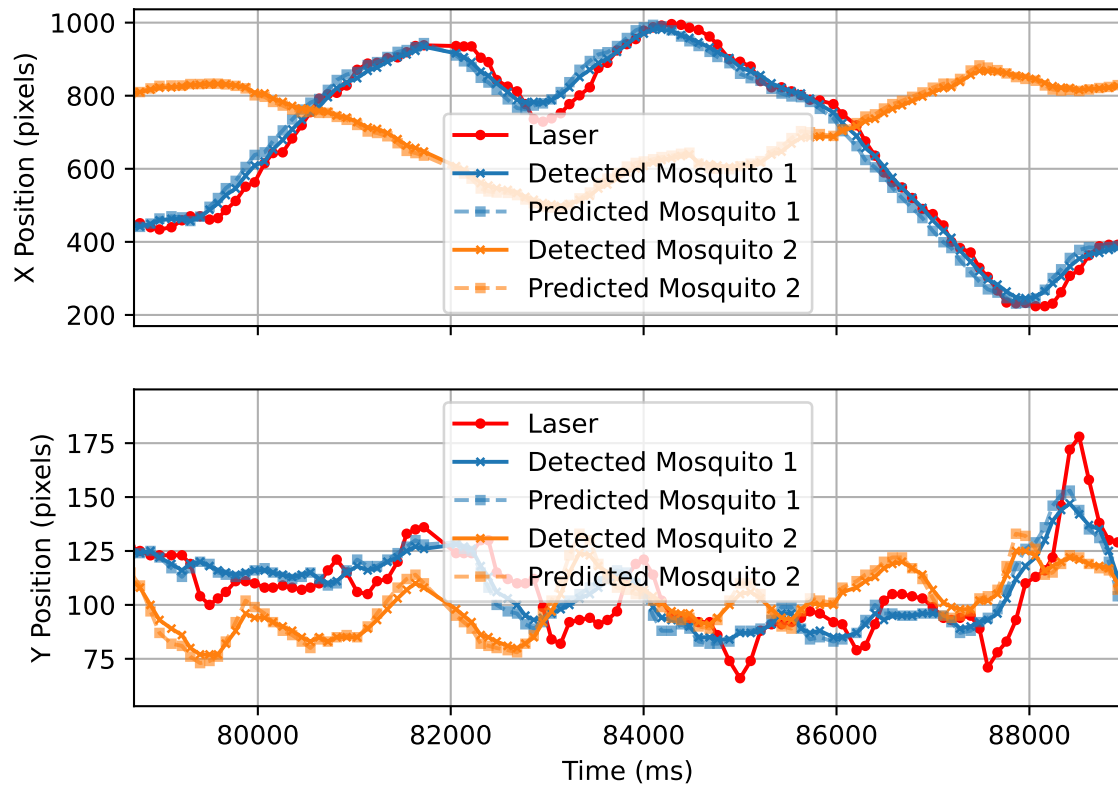
**Figure 26.**
**The position of the laser, detected mosquitoes, and predicted mosquitoes with two mosquitoes in the enclosure over time. The laser was set to target the predicted location of the mosquito in this test.**

**Qualification test 2: Measurement of the time taken for the laser to reach a set point**

*Objectives of the test or experiment*
The objective of this experiment is to measure the time it takes for the laser to reach a set point. The requirement states that the laser must be able to illuminate a set point within 2 seconds accurate to within 1 millimetre.

*Equipment used*
The Nvidia Jetson Nano was used as the embedded platform to control the system. The Raspberry Pi Camera Module V2 was used to capture the video feed. The laser turret developed for this project was used to position the laser. The mosquito enclosure, laser turret, and camera was positioned as described in subsubsection 3.4.2. The Nvidia Jetson Nano was connected to user peripherals for user input and output. C++ code was developed specifically to capture the results for qualification test 2.

*Test setup and experimental parameters*

1. The set point was set to an arbitrary location in the mosquito enclosure.

2. The laser was manually positioned to an arbitrary location in the mosquito enclosure.

3.  The laser turret control system was activated.

4.  The time and position of the laser was recorded for each frame captured by the system until the laser reached a settling point.

5.  Steps 1 to 4 were repeated.

*Results or measurements*

Figure 27 shows the position of the laser over time for multiple runs with different starting points with a constant set point. Figure 28 shows the euclidean distance between the laser and the set point in pixels for multiple runs with different starting points and set points. Figure 28 has been restricted to the time range of 1200 to 2200 milliseconds such that the exact error between laser and set point can be observed. The radius of the laser was measured to a minimum of 5 pixels. The radius of a 2 millimetre disc was measured to be an average 2 pixels.
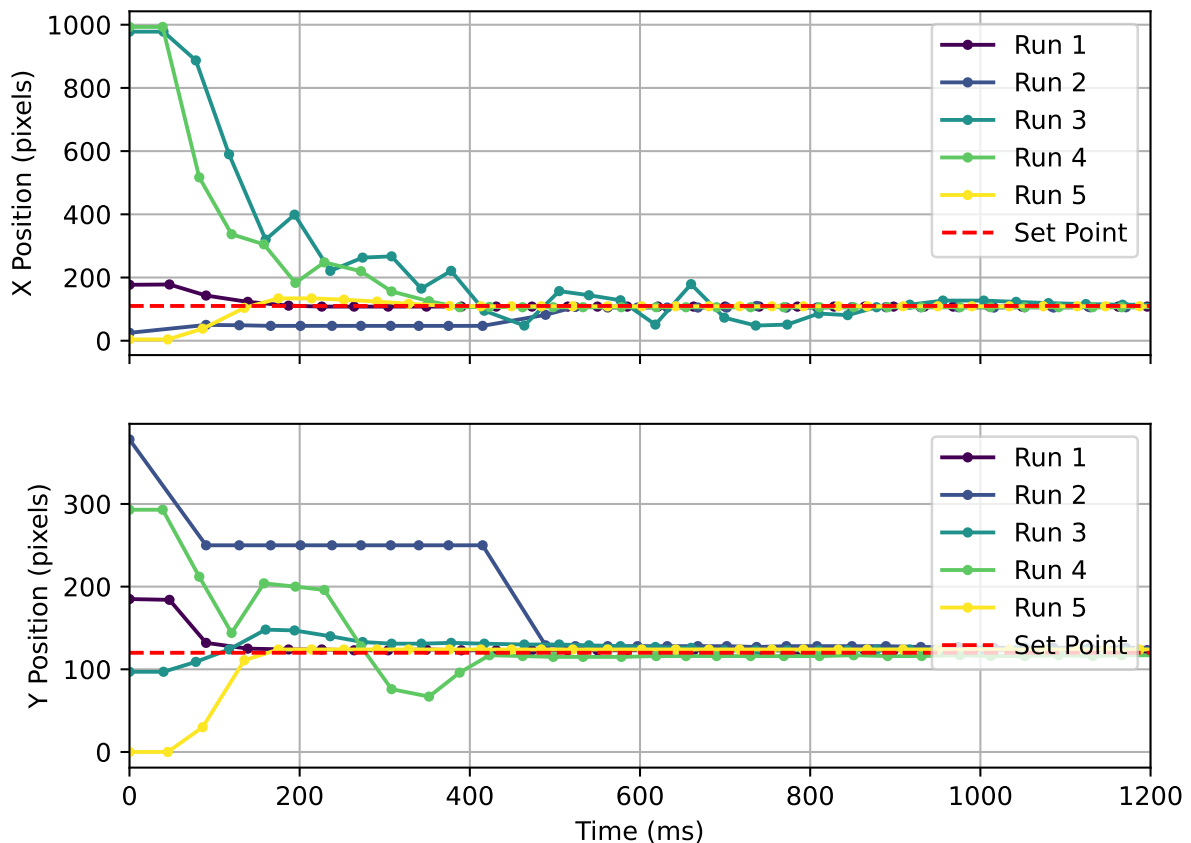


**Figure 27.**
**The position of the laser over time for multiple runs with different starting points with a constant set point.**

*Observations*

Figure 27 shows the position of the laser over time for multiple runs with different starting points with a constant set point. It can be seen that general behaviour of the control system adjusts the position of the laser to move towards the set point. Figure 28 shows the euclidean
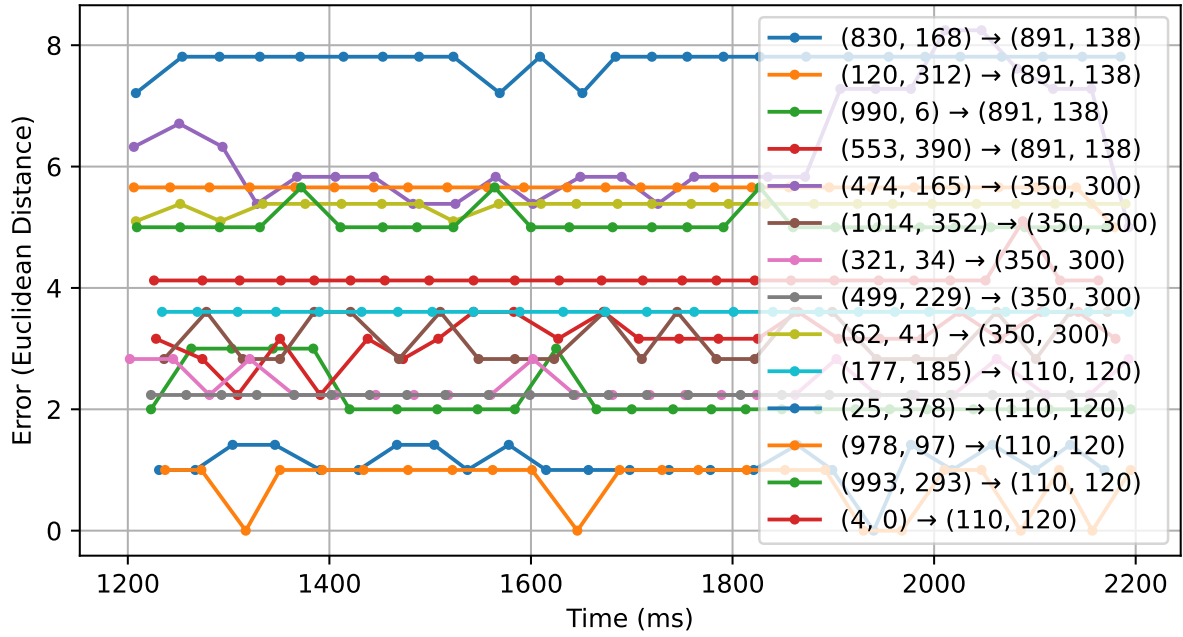
**Figure 28.**
**The euclidean distance between the laser and the set point in pixels for multiple runs with different starting points and set points.**

distance between the laser and the set point in pixels for multiple runs with different starting points and set points. The graph is restricted to the time range of 1200 to 2200 milliseconds which is after the position of the laser has stabilised. It can be seen from the various tests that the maximum euclidean distance between the laser and the set point is less the 8 pixels.

**Qualification test 3: Test of mosquito detection**

*Objectives of the test or experiment*
The objective of this experiment is to determine whether the system can detect mosquitoes in the mosquito enclosure. The requirement states that the system must be able to detect mosquitoes with a 90% accuracy and 5% false positive rate. The detection must be updated at least every 500 milliseconds.

*Equipment used*
The Nvidia Jetson Nano was used as the embedded platform to control the system. The Raspberry Pi Camera Module V2 was used to capture the video feed. The laser turret developed for this project was used to position the laser. The mosquito enclosure, laser turret, and camera was positioned as described in subsubsection 3.4.2. The Nvidia Jetson Nano was connected to user peripherals for user input and output. C++ code was developed specifically to capture the results for qualification test 3.

*Test setup and experimental parameters*
Mosquitoes and similar flying insect specimens were glued to white shafts as shown in Figure 29.

**Figure 29.**
**Dead mosquitoes glued to white shafts.**

The test procedure was as follows:

1. A known number of mosquitoes were inserted into the enclosure.

2. The detection system was activated for 100 frames.

3. The number of mosquitoes detected in each frame was recorded as well as the time between each frame.

4. The experiment was repeated for different amounts of mosquitoes in the enclosure.

*Results or measurements*
The detection accuracy and false positive rate was determined using

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}, \qquad (34)$$

and

$$\text{False positive rate (FPR)} = \frac{FP}{FP + TN}, \qquad (35)$$

where TP is the number of true positives, TN is the number of true negatives, FP is the number of false positives, and FN is the number of false negatives. The detection time requirement was determined by calculating the time between each frame.

The true positives, false positives, false negatives, and true negatives are shown in Figure 30. The detection update interval is shown in Figure 31. The detection results for qualification test 3 are summarised in Table 5

| Metric | Value |
|---|---|
| True Positives | 599 |
| False Positives | 0 |
| False Negatives | 1 |
| True Negatives | 100 |
| Accuracy | 0.998571 |
| False Positive Rate | 0.0 |

**Table 5.**
**A summary of the detection results.**
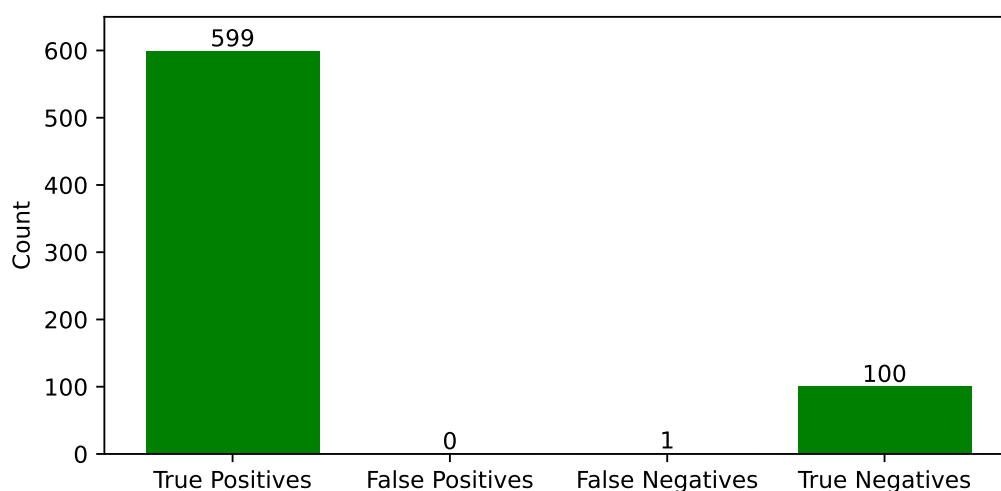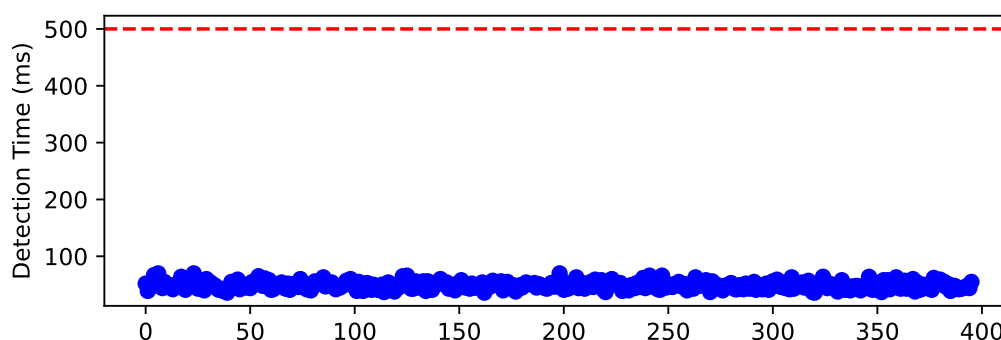
**Figure 30.**
**Mosquito detections.**



**Figure 31.**
**Mosquito detection update interval.**

*Observations*
The system performs detections within 100 milliseconds which is well within the 500 millisecond requirement. The system is able to detect mosquitoes with a 99% accuracy and 0% false positive rate.

**Qualification test 4: Test of mosquito tracking**
*Objectives of the test or experiment*
The objective of this experiment is to determine whether the system can track mosquitoes in the mosquito enclosure. The requirement states that the system must track mosquitoes in the enclosure with 90% correct association after 5 seconds.

*Equipment used*
The Nvidia Jetson Nano was used as the embedded platform to control the system. The Raspberry Pi Camera Module V2 was used to capture the video feed. The laser turret

developed for this project was used to position the laser. The mosquito enclosure, laser turret, and camera was positioned as described in subsubsection 3.4.2. The Nvidia Jetson Nano was connected to user peripherals for user input and output. C++ code was developed specifically to capture the results for qualification test 4.

*Test setup and experimental parameters*
Mosquitoes were simulated with a small black dot attached to a white shafts.

1. The mosquito shafts were inserted into the enclosure.

2. The data capturing procedure was started.

3. The mosquito moved by hand to mimic the motion of mosquitoes.

4. The data capturing procedure automatically stops after 100 frames have been recorded.

*Results or measurements*
Figure 32 shows the detected and predicted position of the tracked mosquitoes in the camera pixel co-ordinate system.
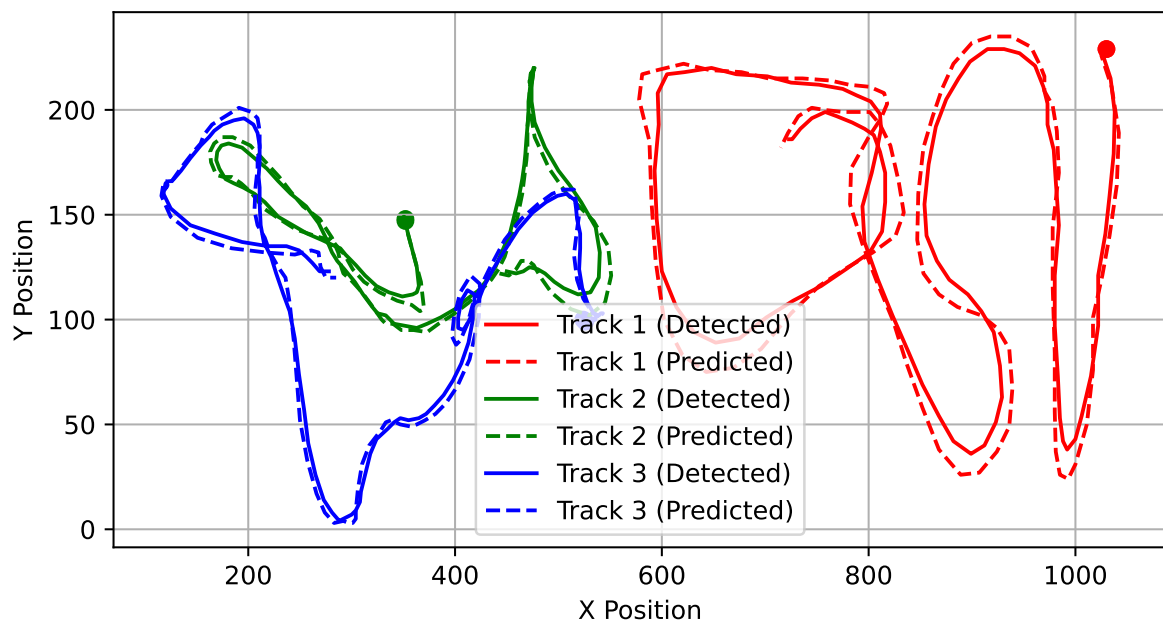


**Figure 32.**
**The detected and predicted position of the tracked mosquitoes in the camera pixel co-ordinate system.**

*Observations*
In Figure 32, it can be seen that the system is able correctly associate the mosquitoes between frames. The detected and predicted paths travelled by the mosquitoes are plotted. It can be seen that the predicted locations significantly deviates from the detected locations when there is a change in the direction of the path travelled by the mosquito. Three mosquitoes were tracked in this test.

# 5. Discussion

## 5.1  Critical evaluation of the design

### 5.1.1  Interpretation of results

#### a.  Laser turret

It can be seen in Figure 27 that laser turret responds rapidly to a step input. The laser turret reaches a settling point within 600 ms of the step input for all runs in Figure 27 other than run 3, which reached a settling point within 1200 ms. The system specification requires the laser turret to reach a settling point within 2 seconds. The response time of the laser turret is well within this specification.

In Figure 28, it can be seen that the laser turret reaches the setpoint accurate to within 6 pixels for all tests with one exception. The system specification requires the laser turret to reach the setpoint accurate to within 1 millimetre. To determine whether this accuracy has been met the following measurements were made: the radius of the laser beam is a minimum of 5 pixels and the radius of a known 2-millimetre disc is on average 2 pixels. These two measurements indicate that the laser can be a maximum of 6 pixels away from the setpoint and still be within 1 millimetre of the setpoint. This indicates that all the test runs in Figure 28, except for one which has an error of 8 pixels, reach the setpoint accurate to within 1 millimetre. However, the radius of the laser beam and the know 2-millimetre disc do vary. The laser beam radius varies with specific lighting conditions and the incident angle with the mosquito enclosure. The radii of the laser beam and disc also vary depending on the location in the camera frame in terms of the depth which they are from the camera and the lateral position in the camera frame since the camera is subject to lens distortion. This means that pixel distance that equates to 1 millimetre accuracy is not constant. The measurements of laser beam and disc radii were made conservatively such that the accuracy of the laser turret is likely to be better than 1 millimetre.

In run 1 of Figure 27, it can be seen that the laser plateaus for about 300 ms before reaching the target position. This is likely due to a false negative laser detection. The laser position error is not updated for the period of false negative detections and thus no control action is taken since the control system is unaware of the error.

#### b.  Mosquito detection

The mosquito detection system performs with 99% accuracy and 0% false positive rate under the test conditions. This excellent performance that is well within the specifications of 90% accuracy and less than 5% false positive rate. However, the detection system was tested using a small set of specimens that were available. The specimens were dead mosquitoes and other similar flying insects that were glued to white shafts for testing purposes. This means that the robustness of the detection system cannot be accurately evaluated due to a lack of sufficient test data.

### c.   Mosquito tracking

In Figure 32, it can be seen that system is able to track multiple mosquitoes simultaneously. The tracking specification requires that mosquitoes are correctly associated between frames with 90% accuracy. It can be seen by inspecting the systems tracked paths of the mosquitoes in Figure 32 that the system has correctly associated the mosquitoes between frames with 100% accuracy. This was determined by analysing the different flight paths captured. It can be seen that there is no other logical association of the flight paths other than the one shown in Figure 32. This system is able to correctly associate mosquitoes between frames with flight paths that cross each other. These results indicate that the tracking system meets the system specification. However, the tracking results were obtained using a small set of specimens that were moved by hand to mimic the true flight of mosquitoes. This does not accurately represent the true flight and behaviour of mosquitoes. The true flight of mosquitoes is likely to be more erratic than the mimicked flight. Real mosquitoes are also capable crossing paths with trajectories that can lead to ambiguous identities that can result in incorrect association. However, for the performance of the overall system this should not be a concern since the ultimate goal of the system would be to kill all mosquitoes in the enclosure.

The tracking system does predict the future position of the mosquitoes, however it can be seen in Figure 32 that when movement of the mosquitoes are highly non-linear this prediction is not accurate. This is because the system does not accurately model the movement of the mosquitoes. The acceleration of the mosquitoes are modelled are system noise since the acceleration of the mosquitoes cannot be accurately model due to their erratic movement.

### d.   Overall system performance

Although the laser turret does meet the system specification it is not able to track a mosquito when it flies erratically.

### 5.1.2   Critical evaluation

The Nvidia Jetson Nano was ideally suited for this project. The detection system could be improved by performing CCL on the GPU rather than the CPU.

### 5.1.3   Unsolved problems

The system does not account for the slight difference in perspectives between the camera and the laser. This difference in perspective is inevitable due to the physical separation of the camera and the laser required to ensure that the laser and camera do not obstruct each other's point of view of the mosquito enclosure. The difference in perspective results two different physical locations of the laser beam corresponding to the same two-dimensional point in the camera's pixel co-ordinate frame. The two physical locations correspond to the point in the $xy$-plane in which the mosquito is and the point on the $xy$-plane of the back wall of the mosquito enclosure. The laser turret will always settle on the physical location that corresponds to the point on the back wall of the mosquito enclosure. This is not a problem when the difference in depth between the $xy$-plane of the back wall of the enclosure and the $xy$-plane in which the mosquito is is small. Through testing the maximum depth at which this

difference is not problematic was found to be approximately half the depth of the mosquito enclosure. To resolve this problem it is necessary to determine the depth of the mosquito which is being tracked. This requires additional hardware such as a depth camera or a second camera with a known separation from the first camera. The depth of the mosquito can then be determined using triangulation.

The feedback from the camera based laser detection system is the only mechanism that the system has to sense the position of the laser turret's motors. This means that if the laser beam is outside the video frame the system has no way of knowing the position of the laser turret. This is problematic when the target position of the laser turret is near the edge of the video frame. When the target position is near the edge of the video frame the laser beam is likely to go outside the video frame due to overshoot and system inaccuracies. When this happens the system is unable to determine the required control action to return the laser to within the video frame since the feedback required to determine this control action is unavailable. This problem can be resolved by adding position encoders to the motors of the laser turret. This will enable the system to determine the position of the laser turret even when the laser beam is outside the video frame, albeit with less accuracy.

### 5.1.4   Strong points of the design

The system is can operate in a variety of external lighting conditions. This is because of the internal lighting in the mosquito enclosure that is used to control the lighting conditions in the enclosure.

The lightweight design of the processing components of the system allow the system to operate at multiple frames per second. The higher the frame rate the better the performance of the system.

### 5.1.5   Expected failure conditions

The mosquito detection system will fail to detect a mosquito if the mosquito is not large enough to create sufficient contrast between itself and the background of the mosquito enclosure.

The system will fail when the mosquito is more than half the depth of the mosquito enclosure away from the back wall of the mosquito enclosure as discussed in subsubsection 5.1.3.

## 5.2   Considerations in the design

### 5.2.1   Ergonomics

The system has been designed to be modular to facilitate easy transportation of the system. The system is easy to assemble with the correct placement of components with the use of the system positioning bracket that can be seen in Figure 9. The system has user-friendly

graphical user interfaces (GUIs). The main GUI displays the mosquito tracking and laser targeting in real-time as well as critical system information. The system enables manual control of the laser turret via the terminal. The system has a host of parameters that can be adjusted at runtime to optimise the performance of the system for the field conditions. The system also has numerous display options with different levels of granularity to facilitate debugging and optimisation of the system. At any stage of operation the user can type '?' into the terminal to view the current commands available.

### 5.2.2   Health and safety

Direct eye contact with the laser beam can be harmful to the user. A low-powered laser diode was used to minimise the risk of eye damage. A laser activation circuit was also designed to mitigate the risk of accidental laser eye contact. The laser is powered off by default and must be activated by the user through the terminal. The laser can easily be deactivated through the terminal with the single key command '`l`'. The presence of live mosquitoes in the mosquito enclosure poses a health risk to the user. The mosquito enclosure is designed with mesh openings that can easily be opened and closed to allow access the enclosure with minimal risk of mosquitoes escaping.

### 5.2.3   Environmental impact

The system does not create any pollutants or harmful emissions. If the system reaches its end of life, the system can be disassembled and most of its components can be recycled. The embedded platform, the motors, and the laser can be repurposed for other applications.

### 5.2.4   Social and legal impact

The 5 mW laser used in the system does not require a permit or compliance with specific regulations to operate. If the system is improved to operate within residential areas the system could prevent the spread of malaria and other mosquito-borne diseases. The system could also be used to prevent the spread of other insect-borne diseases such as the Zika virus.

### 5.2.5   Ethics clearance

The system does not require ethics clearance.

# 6. Conclusion

## 6.1   Summary of the work completed

## 6.2   Summary of the observations and findings

## 6.3   Contribution

## 6.4   Future work

# 7. References

[1] W. H. Organisation. (2020, 12) The top 10 causes of death. [Online]. Available: https://www.who.int/news-room/fact-sheets/detail/the-top-10-causes-of-death

[2] P. Hurtik, D. Číž, O. Kaláb, D. Musiolek, P. Kočáek, and M. Tomis, "Software for visual insect tracking based on f-transform pattern matching," in IEEE Second International Conference on Data Stream Mining & Processing, Lviv, Ukraine, 2018.

[3] Z. Khan, T. Balch, and F. Dellaert, "Efficient particle filter-based tracking of multiple interacting targets using an mrf-based motion model," in Proceedings of the 2003 IEEE/RSJ Intl. Conference on Intelligent Robots and Systems, Las Vegas, Nevada, USA, 2003.

[4] W. Liang, H. Wang, and H. Krim, "A behaviour-based evalutation of product quality," in International Conference on Acoustics, Speech, and Signal Processing, 2016.

[5] Y. Bao and H. Krim, "Video tracking of insect flight path: Towards behavioral assessment," in IEEE, 2018.

[6] A. Bewley, Z. Ge, L. Ott, F. Ramos, and B. Upcroft, "Simple online realtime tracking," Queensland University of Technology, University of Sydney, Tech. Rep., 2017.

[7] S. Nayar, "First principles of computer vision." [Online]. Available: https://fpcv.cs.columbia.edu/

[8] "Polarization effects of galvos in SLA 3D printing," https://henryquach.org/galvopolarization.html, accessed: 2023-10-14.

# Part 4. Appendix: technical documentation

# HARDWARE part of the project

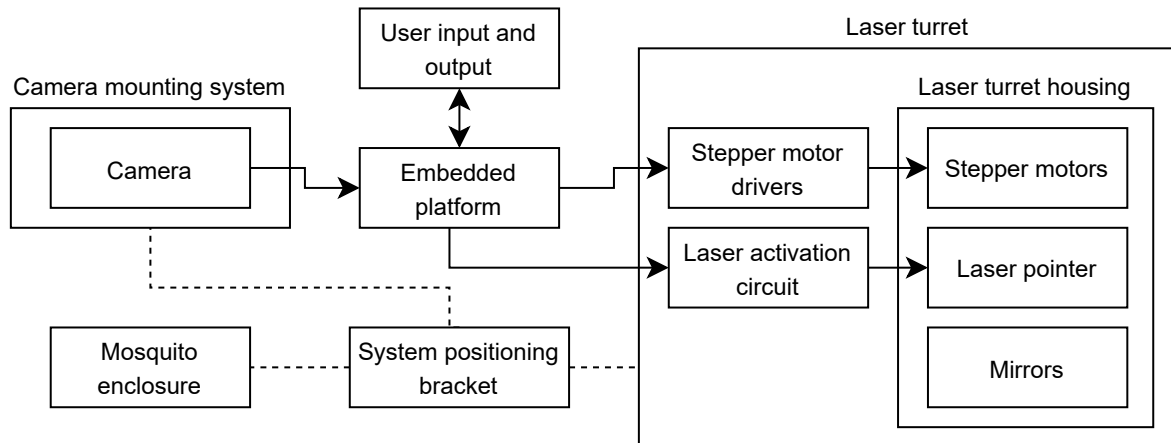## Record 1. System block diagram



**Figure 33.**
**System block diagram.**

## Record 2. Systems level description of the design

## Record 3. Complete circuit diagrams and description

## Record 4. Hardware acceptance test procedure

## Record 5. User guide

# SOFTWARE part of the project

## Record 6. Software process flow diagrams

## Record 7. Explanation of software modules

## Record 8. Complete source code

Complete code has been submitted separately on the AMS.

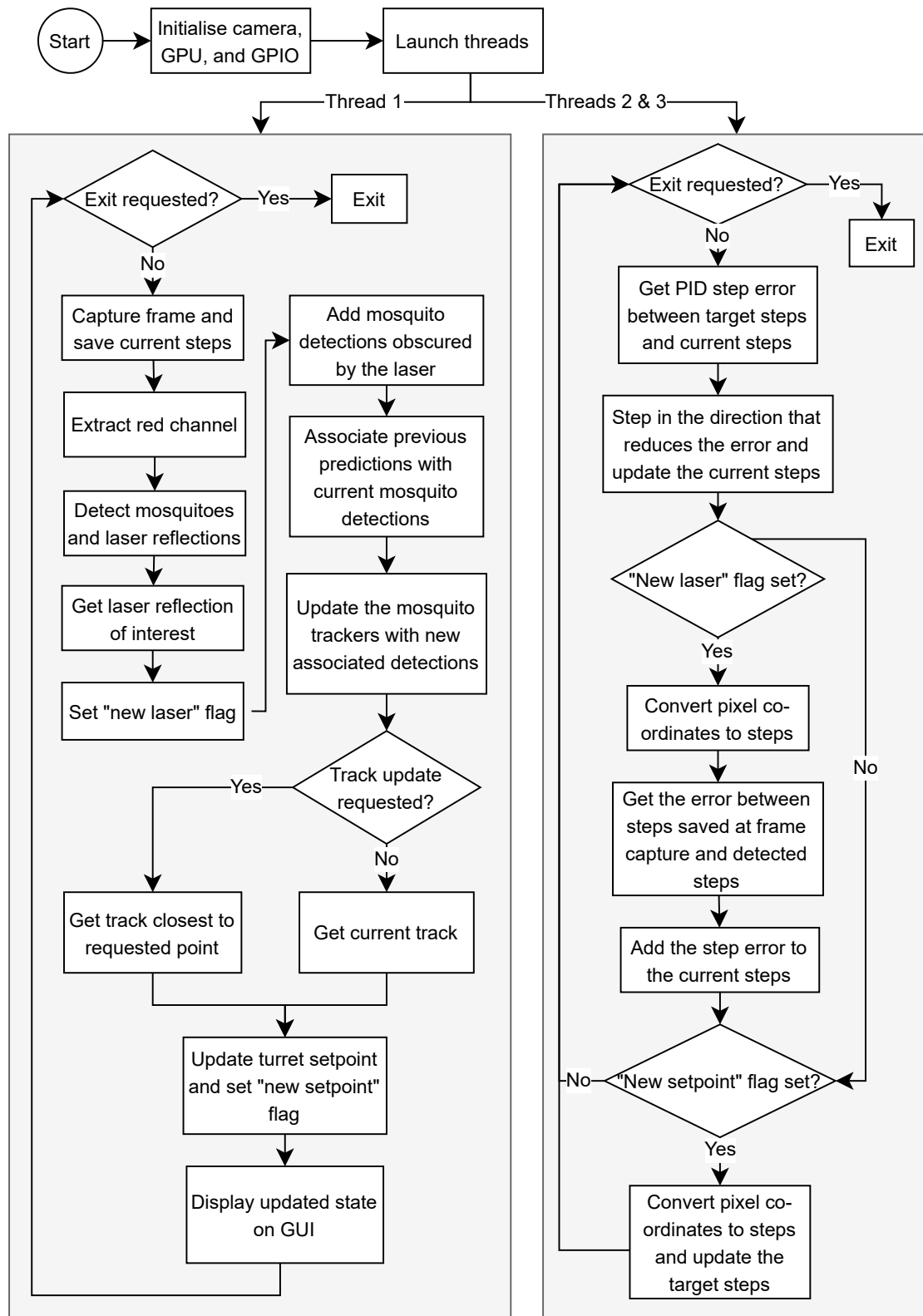## Record 9. Software acceptance test procedure

## Record 10. Software user guide

**Figure 34.**
**Software flow diagram.**

# EXPERIMENTAL DATA

## Record 11. Experimental data