Localising Honeybees with Deep-Learning Based Methods

Oliver Thompson

Localising Honeybees with Deep-Learning Based Methods

Using Deep Learning Object Detection to Identify Bees

in Low Resolution Images

Oliver Jack Alexander Thompson

A thesis presented for the degree of

Master of Computer Science Computer Science

Durham University

2025

Masters Theses

Oliver Thompson

31st May 2025

Bio-hybrid systems are a field of study that combines biology, engineering and computer science to create a new, more effective solution to a variety of problems. The RoboRoyale project seeks to combine honeybee colonies and modern digital technology to help these bees thrive in their environment. Man-made changes to this environment, such as global-warming and pests introduced from other areas of the world have had a large impact on the ability of the European Honeybee's (*Apis Melifera*) ability to survive without human intervention. This study explores how deep-learning based object detection models can be trained to detect honeybees in low-light environments and how these methods compare to traditional ones. Additionally the deep-learning models are trained using simple images tagged by these traditional methods, in order to prevent manual tagging of data. This was done using images collected for the RoboRoyale project for the purposes of investigating the thermotactic aggregation of honeybees, particularly when a group of bees includes a queen.

Contents

List of Figures

List of Tables

0.1 Copyright

0.2 Declaration

The images used for analysis in this thesis were collected by a team at the University of Graz and not by the author.

1 Introduction

1.1 Background

1.1.1 The Importance of Honeybees

The European Honeybee (*Apis mellifera*) is an animal that plays a crucial role in many natural ecosystems and also for modern human society. Pollen limitation affects a plant species ability to reproduce and can lead to a reduced population or even extinction if it can not adapt quickly enough [1]. Animals are estimated to pollinate 87.5% of flowering plant species [2], and among these honeybees are the most frequent floral visitor in natural habitats worldwide [3]. The importance of honeybees is also likely to grow as many of other pollinators have declining populations due to environmental factors [4].

The honeybees are dominant in the role of agricultural pollinators [5], being also the largest pollinator of flowering agricultural plant species [6]. Their role is essential for the growing of many agricultural crops [7] and without them many foods, especially fruits, would be less available and more expensive to the consumer. Honeybees, due to the social structure of their society, have a unique importance among pollinators [8]. A single colony can contain many thousands of worker bees who are actively foraging, so even a small number of colonies can have a large impact on the pollination of an area. Honeybees have been shown to forage more than 9.5km from their hive [9], allowing a single hive to pollinate a large area. It is therefore important that the population of honeybees are maintained for the agricultural sector [10].

The products directly produced by honeybees are also important to us, with the first records of humans harvesting from bee colonies being over 8000 years ago [11].

Honey is the food-store of the bees, and can be collected and used as a foodstuff, high demand and the high demand to attain it has caused it to become the third most adulterated food product [12]. Beeswax has a long history of uses, such as in cosmetic products, producing candles and in art [13]. The pollen collected by honeybees can be harvested and used as a dietary supplement and has antioxidant, anti-inflammatory and antibacterial properties that can enhance a consumers health [14].

1.1.2 Honeybee Threats

Despite their importance to us and the natural environment, honeybee populations are under threat from a variety of factors [4].

One of the largest threats to the European honeybee currently is the varroa mite (*Varroa destructor*), which used to only affect the Asian Honeybee (*Apis cerena*), but was passed on and infected most European honeybee populations over the second half of the 20^{th} Century [16]. The mite affects the health of a bee greatly, can cause

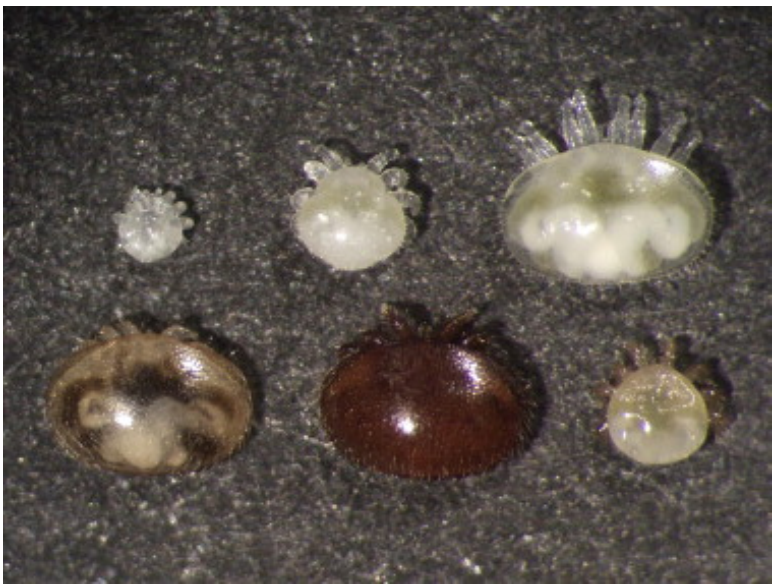

Figure 1: The varroa mite (*Varroa destructor*) across its life stages, from [15].

the spread of disease and can eventually lead to the loss of a colony if left unmanaged. It is the disease or pest with the largest impact on honeybees [17]. Nosema is an infection caused by two species of fungus (*Nosema apis* and *Nosema ceranae*) which majorly impacts honeybee colonies it affects [18].

Colony collapse disorder (CCD) is the sudden loss of a colony due to a rapid decline in worker bees in a colony [19]. The causes of this are not fully understood, and it is thought that a number of stressors affecting the bees may contribute [20]. European foulbrood is a disease caused by the bacterium *Melissocccus plutonius* which causes the brood of a colony to die in its cells [21]. American foulbrood, caused by *Paenibacillus larvae*, is a similar disease which will destroy a colony if infected [22].

The development cycle of honeybees can be affected by climate change [23] as they adapt their behaviour to the local climate, and unexpected weather patterns can cause their behaviours to have unintended harmful consequences. For example, once the weather warms in the spring, honeybee colonies consume their food reserves to grow their population quickly. If a period of cold weather occurs after this, due to an unstable climate, the colony will be no longer able to sustain itself off the forage available, and will either suffer a severe population decline or be destroyed. Drought and heavy rainfall can also adversely impact the bees ability to forage [24]. As well as impacting their food, climate change also can cause pests and diseases that affect the bees to be more impactful [25].

The use of pesticides in agriculture can have a significant negative impact on honeybee colonies [26]. The affects of the pesticides on an individual bee can be either lethal [27] or sub-lethal [28], and the negative effect on a colony as a whole can lead to its loss.

The increased difficulty in honeybee cultivation can be seen by the decline of honeybee populations in specific areas. Between 1960 and 2007, the number of honeybee colonies in Europe declined by 26.5%, and in North America by 49.5% [29].

1.1.3 RoboRoyale

RoboRoyale [30] is a project aiming to prevent a decline in bee populations by developing biological, machine learning and micro-robotic technologies. This will be done by replacing the court bees that normally surround the queen of a colony which regulate her behaviours such as egg-laying and pheromone production. The health of the queen is essential to the success of a colony, as she is responsible for the creation of all new members of the colony. Therefore, an unhealthy queen will result in a fewer number of bees being present in a colony, restricting its growth for the foraging season.

A micro-robotic system will replace the court bees which normally surround the queen and will control the queens egg-laying, pheromone production and feeding. Furthermore, the locations in the colony that the queen lays eggs can be controlled, preventing the laying of eggs in cells which are infected with parasites. Finally the number of eggs that the queen lays can be influenced by the system, preventing large numbers of eggs being laid too early in the year, which is becoming more common with climate instability due to climate change. A short period of warmer weather in early spring can cause this behaviour. When the cold weather returns the bees are unable to sufficiently feed there larger population, leading to a weaker colony due to a loss of resources.

Machine learning techniques will be used in order to optimise the system based

off the behaviour of the colony, and local conditions. This allows the system to adapt to a specific colony as well as evolve with the changing environment.

Through the development of this system, and the studies associated with it, we will gain a greater understanding of the behaviour and biology of honeybees. This is invaluable both for further work in this and related areas, and for apiarists seeking to improve the health and their understanding of their bees. Bees are also a vital part of many ecosystems and this project provides an opportunity to investigate how influencing one organism with a bio-hybrid system affects the surrounding ecosystem.

1.1.4 Object detection

Recent interest and research into object detection systems has been driven by the wide variety of applications these systems have [31]. These systems combine classification and localisation, outputting both the class of an object, and its position. State-of-the-art methods typically utilise deep-learning techniques.

An important use of object detection is for autonomous driving for cars, which is critical for how these systems perceive their environment [32]. These systems combine data from various systems, such as a camera and LiDAR, to predict what an object is. It is important that theses systems are robust and can deal with the effect of noise on images. This is because vehicles operate in a variety of weather and light conditions and need to be able to reliably detect objects even if it is raining or snowing [33]. It is important that autonomous vehicles are able to correctly read road signs [34] in order to safely travel a road network. For this the object detector needs to be capable of optical character recognition (OCR), translating the characters in the image into machine-encoded text [35].

Medicine is also an area where object detection has many uses. Detectors can be used to localise abnormalities when using medical imaging such as endoscopy [36], allowing automatic detection and diagnosis. During surgical procedures, especially those done robotically or with computer assistance, object detection can reduce errors and monitor patient activity [37]. Another application is dental care [38], where it can be used with radiographs, near-infrared light transillumination images, and clinical features.

A major used of object detection is in facial recognition, both in identify a person, and their emotions. When students perform online exams, as is becoming more often the case, and object detection can perform automated proctoring of these exams [39]. It also have uses in security, both in terms of tracking a person of interest in a crowd and controlling access to a computer [40].

Deep-learning based object detectors require a large number of pre-tagged images to train. This limits the use of such systems which can not use the already existing

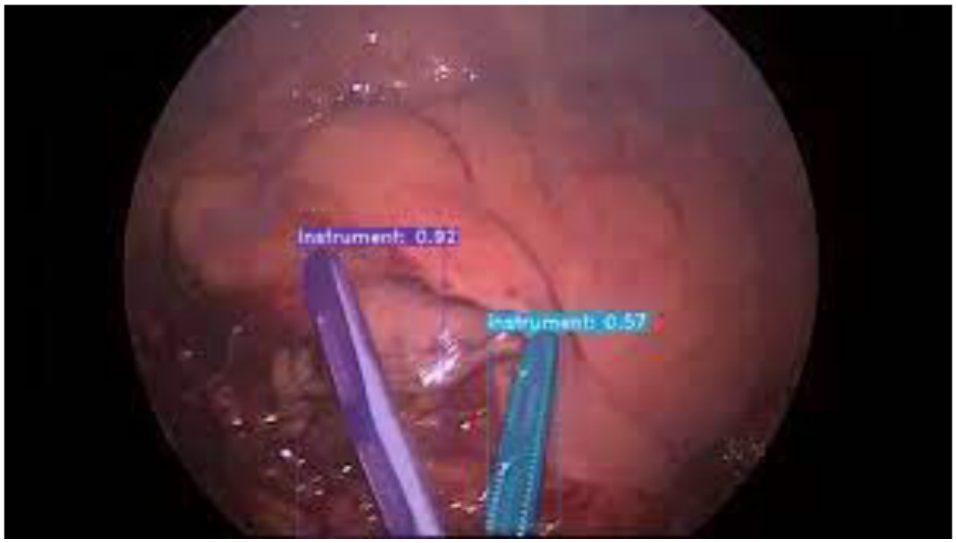

Figure 2: An object detection model localising medical instruments during surgery, from [37].

corpus of tagged images in a dataset like COCO [41], due to the length of time tagging the images required takes.

However, deep learning models are particularly suited for this application because they can be trained to a specific task without complex feature extraction, as would be the case with traditional models. This would be difficult in this case due to the general blurriness and lack of identifiable features in the images. It would also be difficult for traditional methods to determine the difference between multiple bees in close proximity.

1.2 Contributions

A contribution of this paper is to process experimental data created for the RoboRoyale project, investigating the thermotaxis of a honeybee queen and if the presence of a queen affects the thermotactic aggregation of a swarm of bees. This gives the project valuable insights into the behaviour of queen bees and their attendant "court bees", which is important due to their focus of using a robotic system to influence the behaviour of the queen of a colony, replacing the court bees. Heat could be used to control the movement and behaviour of the queen, in addition to other methods, allowing finer and more reliable influence of her movement and actions.

Another contribution is exploring how a deep-learning model for object detection can be tailored for a very specific real-world task. Research typically focuses on the general application of these models and their ability to perform on a number of sample datasets. This means some aspects of their use have not been fully explored, such as their use for a specific task using non-standard data or their ease of use for the average user. These factors are important for the adoption and use of technology and for it to fully realise its potential benefits for society.

Finally this paper investigates how using a traditional object detection can be used to generate tagged images to use for training a deep-learning based detector. Using a method such as this allows for many more images to be tagged for a specific task by massively reducing the man-hours required, allowing these detectors to be used for applications that were unfeasible previously.

1.3 Objectives

This investigation seeks to explore how deep-learning object detection methods can be utilised for specific tasks in the research environment. It compares different methods to find whether these state-of-the-art methods are outperformed by more traditional methods for these tasks, or whether they offer important advantages.

Hence, the objectives of this investigation are the following:

1. To provide accurate position estimates for honeybees for the RoboRoyale project from the provided images.

2. To investigate how traditional methods can be used to provide these position estimates.

3. To train deep-learning based object detection models for use on this task.

4. To compare the traditional and deep-learning based methods and find which provides higher accuracy.

5. To investigate the limits of these models in their current form for this task.

2 Related Work

2.1 Bee Behaviour

2.1.1 Thermotactic Aggregation

Precise thermoregulation occurs in two types of honeybee aggregations, winter clusters and reproductive swarms [42]. The temperature at the centre of a cluster is much warmer than the mantle or edge, with the centre of a winter cluster being between $18°C$ and $32°C$, and a swarm being $35°C$ and their mantles being between $9°C$ and $15°C$, and $15°C$ respectively. Bees move outwards if they are too warm, and inwards if they are too cold, and their behaviour is thought to only been in response to their local conditions, however this leads to a much more complex overall behaviour displayed by the superorganism.

Freshly hatched honeybees cannot produce heat endothermically, and therefore seek an area close to their preferred temperature of $36°C$ [43]. Individual bees do not tend to find this optimal temperature, with only collective thermotaxis seeming to occur. This is an act of collective decision making by the bees, and is an example of swarm intelligence [44]. An example of bees displaying this behaviour is shown in Figure 3.

2.1.2 BEECLUST

BEECLUST [45] proposed by Schimckl et al. is a swarm robotics algorithm for aggregation inspired by the thermotactic aggregation of honeybees. A robot proceeds forwards until it collides with another robot, where it then waits for a variable amount of time. The time the robot waits for is decided by the following formula:

Figure 3: An image showing honeybee drones aggregating around a heat source, from [45].

$$w = w_{Max}\frac{s^2}{s^2 + \theta}$$

where s is the temperature of the robots location, w_{Max} is the robot's maximum wait time and θ is the decay rate. The robot then turns a random angle ϕ between $[-180°, 180°]$ and then continues forward in a straight line again.

This mimics the way newly hatched honeybee drones aggregate in a warm area of a hive, and such behaviours can be useful for a number of different tasks. These include uses in agriculture, and search and rescue. For example nuclear material was lost in Australia [46], and a robotic swarm could have been used to search for the material, using the radiation as its temperature variable.

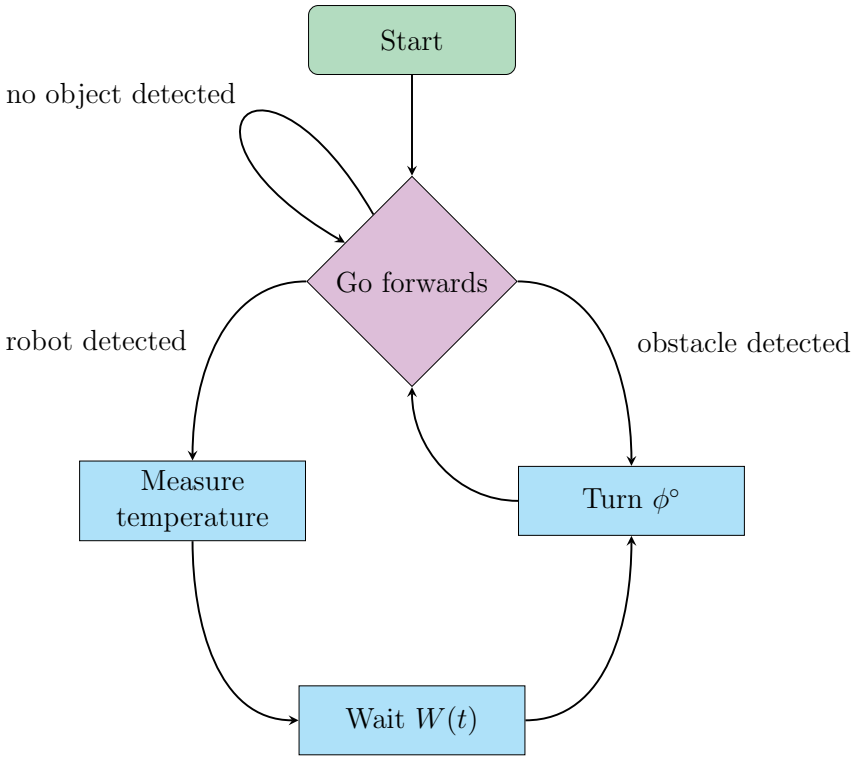

Figure 4: A flowchart showing the basic structure of the BEECLUST algorithm for aggregation.

2.1.3 Bee Colony Optimisation

Another algorithm inspired by research into honeybees is the Bee Colony Optimisation Metaheuristic (BCO) [47]. Honeybee foraging techniques are adapted to be used with combinatorial optimisation problems, where a food source represents a solution.

A predetermined number of "worker bees" inspect potential solutions and then return to the proverbial hive. Bees then continue exploring, as well as returning to the neighbourhood of previously found strong solutions, and exploring their local area for a more optimal result.

2.2 Bio-hybrid Systems

2.2.1 Overview

Bio-hybrid systems combine both biological and non-biological elements to create a more effective solution to a problem. Within this field sit animal-robot interaction systems (ARIS) and organism-robot interaction systems (ORIS) [48]. A wide range of different species can be used in these systems, on the microscopic, mesoscopic and macroscopic levels. Developing such systems produces a number of complications not inherent in traditional solutions, but can produce a system that is adaptive, multi-functional and resilient.

2.2.2 Robocoenosis

One use of bio-hybrid systems is as biosensors [49], where a living organism is used as a sensor to provide data to a computational system, rather than a standard electronic sensor. The Robocoenosis project [50] uses five different modules containing different organisms to measure different variables in an aquatic environment. These biosensors provide less precise data than traditional methods, but can be used to sense a variety of stressors to the organisms in the water, and calibration can be used to determine which is an issue for the organism [51]. This signals there is a problem in that environment an that further action should be taken.

One of the modules developed for this project uses *Daphnia*, a form of zooplankton, as its living organism [52]. These organisms change there phototaxis behaviours dependant on a variety of stressors in their environment, and this can be observed automatically using a camera. Factors in the environment that can be measured this way include: microplastics, pesticides, salinity, temperature and oxygen levels.

2.2.3 ASSISIBF

The ASSISIBF [53] project aimed to show how two different species could communicate at great distance using a robotic society that would communicate with animal societies. The two species used were European honeybees and zebra-fish (*Danio rerio*). They are two species which are social and exhibit collective behaviours [54].

The fish were placed in a glass tank of water, under which a wheeled robot was placed. This robot controlled a fish lure in the tank. This lure had been shown to be socially accepted by the fish and could be used to influence their movement [55] [56]. The lure would influence the fish to circle the tank in a specific direction, giving a behaviour which could be recorded.

The bees were placed onto a sheet of pressed beeswax, thermal control surfaces were then inserted through the wax. Juvenile bees have been shown to aggregate around a temperature source, and the robot can switch the thermal control surfaces on or off to influence the behaviour of the bees.

2.2.4 RoboRoyale

The RoboRoyale project [30] seeks to use *'ecosystem hacking'* [57] in order to perform conservation without being overly invasive to a honeybee colony. This is done by interacting with the queen of the colony, in order to influence the entire superorganism. The court bees that surround the queen are replicated using robots, allowing her behaviour to influenced, in order to make her perform actions more beneficial for the colony.

A number of robotic biomimetic agents surround the queen, attached to a larger robot, which controls their movement. A sheet of thin glass restricts the movement of the bees and agents to a 2D space inside the colony. A high resolution camera

on the main robot allows the queen to be tracked and followed by the system. The smaller agents clean and provide food to the queen, replacing the court bees that would perform these functions in a standard colony. Theses agents can be used to provide the queen with higher quality food than would be obtainable by normal bees, as well as guiding the queen on where and when to lay brood. The colony will follow a more optimal brood pattern than before, allowing it to expand more quickly when needed, and expansion to be prevented when it would not benefit the colony.

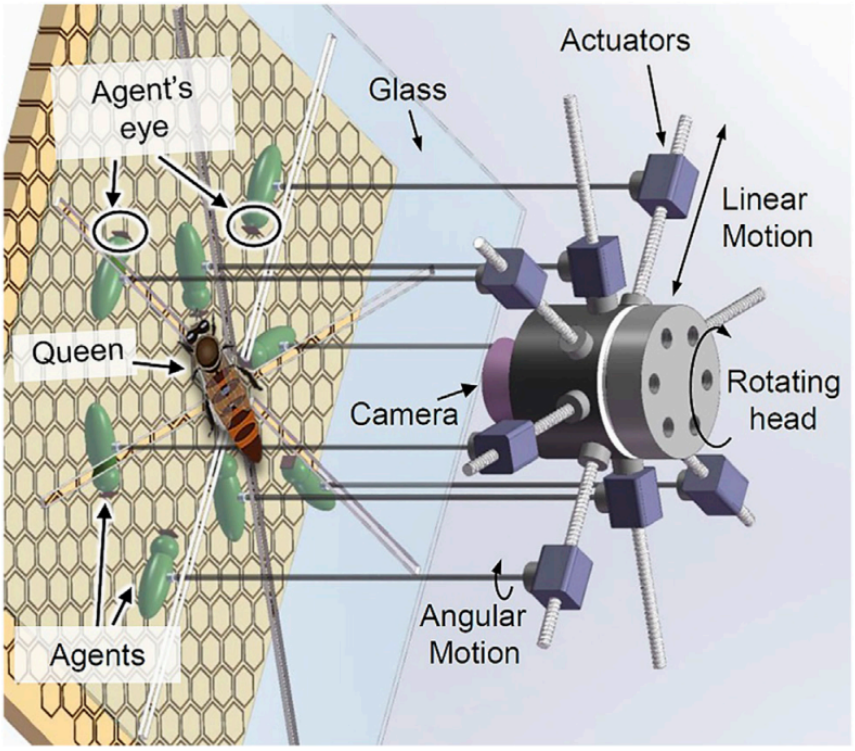

Figure 5: The robotic system developed for RoboRoyale, showing the main actuator unit and eight smaller biomimetic agents, from [30].

2.3 Object Localisation

2.3.1 Overview

For any mobile robotic system position estimation is an important issue to be solved [58]. It is crucial that these systems are precise and reliable, and it is often important that high-frequency position updates are provided. Many of these localisation systems do not scale well when a large number of objects are involved, limiting their use for certain applications, such as swarm robotics. There are two main types of object localisation methods: internal and external.

Internal localisation is when the robot calculates its own position, and these are the methods that are mainly explored. These methods are most effective when there is a ready-made map of the environment, or when this map can be created alongside the localisation method. The main drawback is that the robots need to be able to define their position both precisely and accurately compared to an external ground truth.

External localisation methods are when an external system produces the position estimates, the most famous example of which is GPS. Another example are fiducial markers, of which the main types used in swarm and mobile robotics are WhyCon [58], WhyCode [59], ArUco [60] and Apriltag [61]. They can be divided into two main groups active and passive methods.

2.3.2 WhyCon

WhyCon [58] is a vision-based external object localisation system designed to be used with multiple robots. It uses black-and-white roundels as fiducial markers to identify objects in the environment. The system is designed to be able to process hundreds of images per second while tracking hundreds of objects, making the system suitable

for applications that require either a fast update time such as a mobile robotics application, or the need to track a large number of robots, for example in swarm robotics. It is also computationally efficient enough to be used with standard off the shelf components.

The stages of the algorithm are the following:

1. The pattern detector searches for a black segment.

2. Initial tests are performed on the segment

3. A white segment attached to the black segment is searched for, starting at the expected centre point of the marker.

4. The dimensions and coordinates of the marker are identified.

5. The coordinates are transformed to a coordinate frame defined by the user.

When detecting the first segment, the algorithm searches randomly searches across the image. However, when the detector runs over the next image it first searches in the area it found these segments previously. This reduces the computation required massively as long as the objects are not moving too fast as only areas with known objects are searched over, making the model performance independent of image size.

WhyCon has been extended into WhyCode [59]. This method replaces the black hollow circle of WhyCon with a similar circle with a Manchester-encoded necklace [62] on the inside, as shown in Figure 6. This allows the orientation of the object to be found as well as its position.

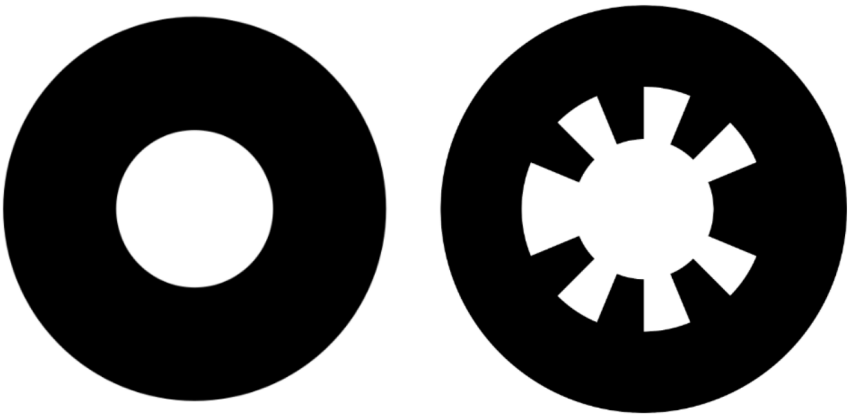

Figure 6: The fiducial markers used in WhyCon (left) and WhyCode (right), with the difference being the necklace encoding on the inside. From [59].

2.4 Object Detection

2.4.1 Overview

Object detection is a computer vision technique where an object is identified and located within an image or video. It combines classification (what an object is) with localisation (where an object is). An object detection model first defines a bounding box which is an area of the image where an object is detected. The bounding box is then assigned a class label from a predefined list of categories.

2.4.2 Viola-Jones

The Viola-Jones algorithm [63] uses traditional methods for object detection. It used an *integral image* to represent an image, this allows Haar-like features at any scale or location to be computed in constant time. The integral image $ii(x, y)$ at a location (x, y) for an image $i(x, y)$, is the sum of all pixels left and up from that location inclusive, or:

$$ii(x, y) = \sum_{x' \leq x, y' \leq y} i(x', y')$$

This constant time computation gives the detector much better computational efficiency to previous similar methods.

Another innovation was the use of Adaboost [64] to select a small number of important features to send to the classifier. This is done by constraining the weak classifier to only use a single feature, allowing the boosting process which selects which weak classifiers to act as a feature selection process. By limiting the number of features used, this method allows for much faster detection of objects.

The final major contribution was a method of combining complex classifiers using a degenerate decision tree, or cascade structure. As many sub-windows are rejected as early as possible, as almost all sub-windows will be negative. This helps the model to focus onto areas of interest, wasting less time searching areas not containing an object allowing for faster performance.

2.4.3 HOG

HOG (**H**istogram of **O**riented **G**radients) [65] is a traditional object detection model first proposed in 2005. The method uses well-normalised local histograms of image gradient orientations in a dense grid known as HOG features. Before extracting the features, analysis is conducted on the image using data from the training set in order to identify blocks of interest. Computational time is then saved by only extracting from blocks highlighted by this analysis. After feature extraction is performed, Linear Support Vector Machine (SVM) is used as the baseline classifier, which then decides if the the detection is an object or not.

2.4.4 Deep-Learning Based Object Detectors

State-of-the-art object techniques now utilise deep-learning methods for object detection. There are three main types of these detectors with distinct advantages and disadvantages.

1. **Two-stage detectors:** these models pass over an image twice. The first pass proposes an area of interest that an object could be in, and the second pass classifies that area to an object class. These models often are very accurate, but are computationally expensive.

2. **Single-stage detectors:** these predict the bounding box and object pass in one combined pass. This gives a faster performance, and makes these models well suited for real-time applications.

3. **Transformers:** these use attention mechanisms to simplify the detection pipeline and offer good performance on complex datasets. An example is DETR (**DE**tection **TR**ansformer) [66].

2.4.5 Faster R-CNN

Faster R-CNN (**Faster R**egion-based **C**onvolutional **N**eural **N**etwork) [67] is a two-stage deep-learning based object detection framework. It is a development of R-CNN [68] and Fast R-CNN [69], which were held back by time inefficient region proposals. This bottleneck was overcome by using a deep convolutional neural network to compute proposals, called Region Proposal Networks (RPNs), this structure can be seen in Figure 7. These share convolutional layers with the detection networks, making the cost for computing proposals much smaller.

The RPN slides a small network over the feature map output of the last shared

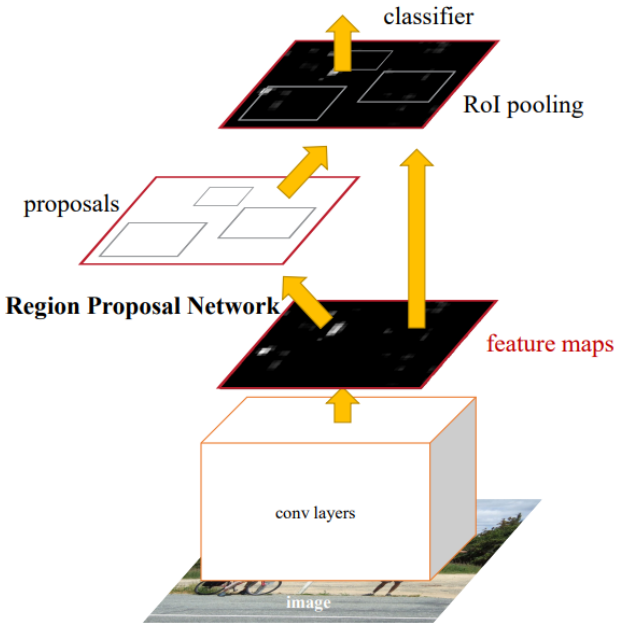

Figure 7: The architecture of Faster R-CNN, from [67].

convolutional layer. This network consists of a convolutional layer of size $n \times n$ followed by two networks of size 1×1, which are for box-regression (*reg*) and box-classification (*cls*). This architecture can be seen in Figure 8. For each location many region proposals are produced, each with a objectness score, which measures the likelihood of that detection belonging to a specific object class.

It is important this method is translation invariant, so that a translated object the proposal should also translate and the same function can be used for the proposal. Multiple scales were catered for by implementing a novel method called Pyramid of Filters. Multiple reference anchor boxes with different scales and aspect ratios are to for the regression and classification for bounding boxes. This allows objects of different scales to be detected with minimal extra cost.

When training the RPNs each anchor box is given a binary label for whether it represents an object or not. This label is true if an anchor has the highest IoU

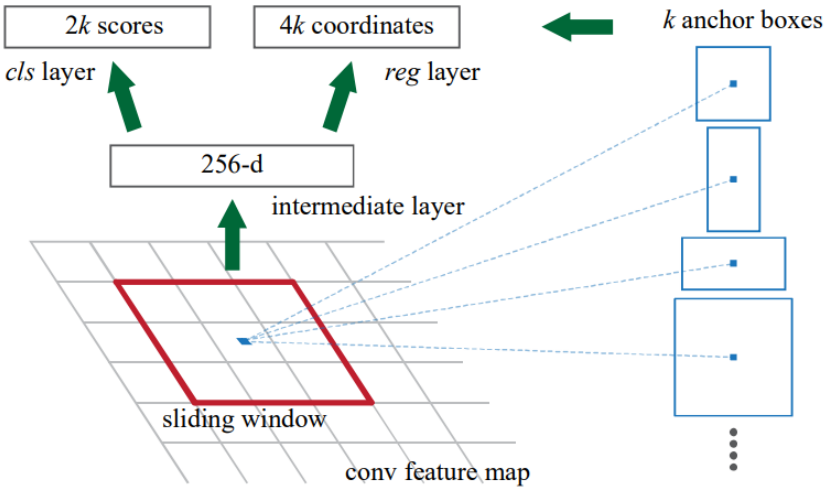

Figure 8: The architecture of the RPNs in Faster R-CNN, from [67].

overlap or has an overlap of greater than 0.7 with a ground-truth box. This label is false if the overlap is less than 0.3, all other boxes are not used for training. The RPN is then trained using end-to-end back propagation and stochastic gradient descent (SGD) [70]. The loss function is optimised against 256 randomly chosen anchors from amongst a sample of ratio of 1:1 for positive and negative anchors, rather than the complete set, in order not to give a bias to negative anchors.

Faster R-CNN was faster than the models that came before it, but is too slow to be used for real-time applications, only running at 7 frames per second [71]. More modern single-stage detectors are more suitable for these applications, as they are able to process more images in the same amount of time.

2.4.6 SSD

SSD (**S**ingle **S**hot multibox **D**etector) [71] is a deep learning-based object detection algorithm introduced by Liu et al.. This model sought to massively reduce the time required to infer from an image without sacrificing accuracy. It was the first deep-

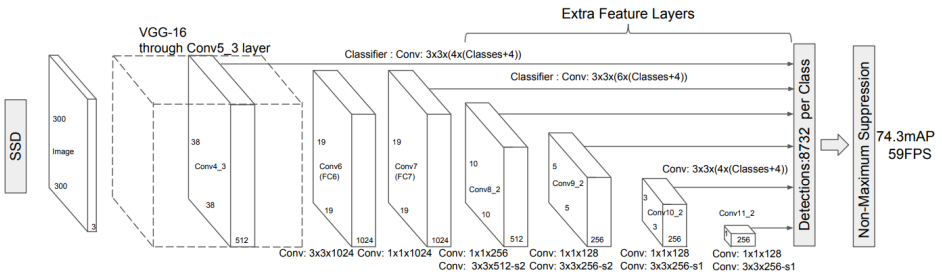

Figure 9: The architecture of SSD, from [71].

learning based object detector to be single-stage and maintain detection accuracy, predicting class scores and bounding boxes in a single pass, by not resampling features or pixels for bounding box predictions. This gave the model much better speed performance than contemporary models, such as Faster-CNN.

The first stage of the model is a feed-forward convolutional network whose output is bounding boxes for objects, as well as a score for each box for the presence of each object class in that box. A standard image classification architecture, such as VGG-16 [72], is used for the early network layers, or the base network. Alongside this is an auxiliary network, with multiple additional features. After this non-maximal suppression is applied to the boxes to generate the output.

This model also introduced default boxes, which are similar to the anchor boxes used in Faster R-CNN, the difference being these are applied to multiple feature maps with different resolutions. These boxes convolutionally tile a feature map, and then are used to generate offsets, and an object class score for each feature map cell.

Multi-scale feature maps were used to allow the model to detect objects of different size, this was done by adding a feature layer to the end of the truncated base base network. Convolutional filters were added to the feature layers to produce a fixed set of detection predictions. Kernels of size $3 \times 3 \times p$, where p is the number of channels in the layer, are applied to a number of locations, where it produces either

an offset to the default box coordinates or a category score.

SSD does not used region proposals for training, but relies on ground truth information assigned to specific outputs for the fixed set of detector outputs. Back propagation and the loss function can be applied end to end once an assignment is determined, with Smooth l1 loss [69] used for the localisation loss.

During training, default boxes, which vary in aspect ration, location and scale, need to be matched to ground truth detections. This is done by selecting default boxes with a jaccard overlap [73] of greater than 0.5. This allows the network to produce scores for multiple boxes, and not just the one with the highest overlap score. In order to handle different object scales, predictions from default boxes with different scales and aspect rations are combined in order to get a variety of predictions for different object shapes and sizes.

2.4.7 YOLO

YOLO (**Y**ou **O**nly **L**ook **O**nce) [74] is another single-pass deep-learning object detector. Image preprocessing for YOLO consists of normalising the image and resizing it to a fixed size the model has been trained on. This allows for the input to the model to be consistent with the images that the model has been trained on.

In order to combine all the steps of object detection in a single pass, the image is cut into $S \times S$ grid cells. A grid cell is responsible for detecting an object if the centre of that object inside that grid cell. For each grid cell, bounding boxes, B, are predicted with each bounding box having multiple values assigned to it:

- Pc: the models confidence that the boundary box contains an object, and that how accurate thinks the predicted box is. This is defined as $Pr(Object) * IOU_{pred}^{truth}$, the intersection over union between the predicted box and the

ground truth. If there is no object in that cell the score is zero.

- (x, y): the coordinates of the centre of the boundary box relative to the grid cell the box is in.
- (h, w): the height and width of the box relative to the entire image.
- C class confidence are also generated for each class in the model. Defined as $Pr(Class_i|Object)$, the probability of that class being present provided the grid cell contains an object. Only one set of confidence scores are generated per grid cell, irrespective of the number of boxes in that cell.

The output of the model is therefore a tensor of size $S \times S \times (B * 5 + C)$. Non-maximal suppression can then be performed on this output to remove duplicates.

There have been multiple iterative versions of the YOLO algorithm, each improving both the performance and accuracy of the last, utilising various changes and additions:

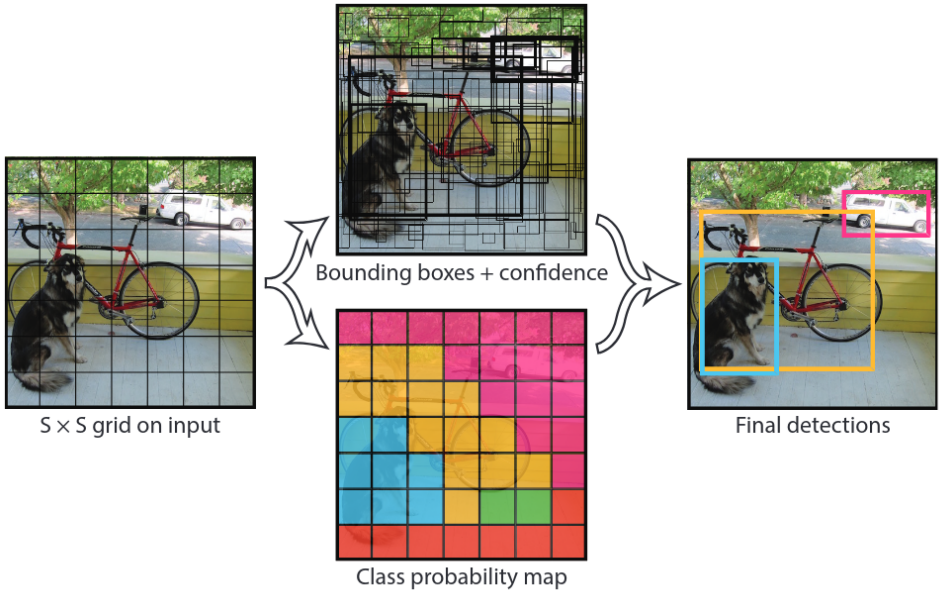

Figure 10: The system model of YOLO, from [74].

YOLOv1: Introduced by Redmon [74], YOLO used a novel single pass over the image using regression. Its architecture was inspired by GoogLeNet [75]. 24 convolutional layers were used to extract features from the image, followed by two fully connected layers which predicted the coordinates and probabilities of objects. For training a loss function consisting of multiple sum squared errors was used.

Because of its simple architecture and single pass, YOLO was much faster than other methods that existed at the time, allowing it to be used for real-time applications [76]. However, it had larger localisation errors compared to other contemporary methods. It also had difficulties with small objects and objects with different aspect ratios to those found in the training data. It also could only detect two objects of the same class in a grid cell, meaning it could not detect many objects close together, as well as only being able to learn from coarse features of objects.

YOLOv2: This model [77] was introduced by Redmon and Farhadi as an evolution of the first version, improving its accuracy without compromising the method's real-time performance.

All layers of the network were made convolutional, and batch normalisation to reduce over-fitting and improve convergence was used on all layers. Anchor boxes, with a predefined shape that match objects, were used to predict boundary boxes, with multiple anchor boxes generated per cell. These anchor boxes are extracted using KMEANS [78] clustering rather than being pre-defined.

The object layers were no longer fully connected as well, allowing multi-scale training, giving the model much more flexibility as it no longer requires all

images to be the same resolution. Additionally batch normalisation was used on all hidden-layers of the network to give a more consistent distribution of weight matrices.

YOLOv3: This model [79] improved the ability of YOLO to detect small objects by using multi-scale training, by generating feature-maps at three different scales, 13×13, 26×26 and 52×52. This allowed for the detection of different sized objects using a specific map.

The backbone was made deeper to 53 convolutional layers and utilised Darknet-53. The architecture of the model was also changed, being inspired by Feature Pyramid Network (FPN) [80].

YOLOv4: This version [81] used CSPDarknet53 as it's backbone, but with only 29 convolutional layers. However, cross-stage partial connections were used to improve performance without compromising computational efficiency.

PANet [82] was used in an unusual way, concatenating feature maps on a specific axis across different layers, in order to produce a feature map with for multiple layers, helping to increase accuracy. Spatial Pyramid Pooling [83] was used to allow for various input dimensions without having to resize or reshape images, by placing it between convolutional blocks where it could map any sized input to a fixed output.

YOLOv5: YOLOv5 [84] made the important change of moving the backbone to PyTorch [85], allowing the model to be much more easy to use. A layer of Spacial Pyramid Pooling Fast (SPPF) pooled features of different scale into a single feature-map, improving computation time. A number of other smaller improvements were made to many aspects of the model.

YOLOv6: In September 2022 YOLOv6 [86] [87] was introduced, utilising a more efficient backbone, EfficentRep based on RepVGG, which used more parallelism to increase computational performance. A quantisation scheme using RepOptimiser and channel-wide distillation was introduced, making the detector faster without compromising accuracy.

YOLOv7: YOLOv7 [88] used a concatenation based architecture, using a novel approach for model scaling which ensured the depth and width of the block were scaled rather than just one or the other, preventing distortions. It also utilised a new version of the Effient Layer Aggregation Network (ELAN) [89], E-ELAN, which controlled the shortest longest gradient path, by merging and shuffling cardinality features.

trainable-bag-of-freebies method

YOLOv8: YOLOv8's [90] backbone was similar to that of YOLOv5's, both having been created by Ultralytics. A semantic segmentation model, YOLOv8-Seg, extended the capabilities of the model, allowing for semantic segmentation of images.

2.4.8 RTMDet

RTMDet (**R**eal-**T**ime **M**odels for object **Det**ection) [91] is a novel real-time object detector proposed in 2022 by Lyu, Zhang, et. al. The model is a single-stage detector and was designed to be both efficient and versatile, also being able to be used for instance segmentation and rotated object detection.

Like most single-stage detectors, RTMDet divides its macro architecture into three stages, as shown in Figure 11. Those stages are:

1. **The Backbone:** the initial section of the model. It consists of multiple stages, each stage then consists of multiple building blocks.

2. **The Neck:** uses bottom-up and top-down feature propagation using the same building blocks as the backbone on the multi-scale feature pyramid produced by it to improve the feature map.

3. **The Head:** predicts the objects in the image using the feature map created by the neck.

The backbone architecture used for RTMDet is CSPDarkNet [81].

RTMDet uses 5×5 depth-wise convolutions inside the backbone architecture in order to attain a large effective receptive field, without inordinately increasing computational complexity. This improves the accuracy of the model significantly by enabling the context modelling for the image to better capture and model the image context [92]. This is a difference compared to other models which use re-parametrised 3×3 convolutions [93] instead. The advantages of the 5×5 depth-wise convolutions are that; it requires reduced memory for training, is faster to train, and after the the model is quantised it has a reduced error gap.

In order to enhance multi-scale features without having a heavier neck or using more parameters, both of which have a large effect on the memory and computational

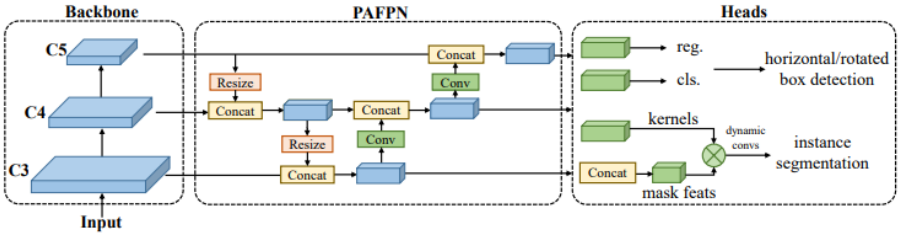

Figure 11: The macro architecture of RTMDet, from [91].

resources required to run the model, the expansion ration of basic blocks in the neck was increased. This helps the model better detect objects at various scales without sacrificing performance. Additionally, the width of blocks in the backbone was slightly, increased while the number of blocks was decreased. This was because there were additional point-wise convolutions needed after the depth-wise convolutions, and these changes allowed for more parallelisation so inference speed would not be adversely affected, allowing the model to still be used real-time.

A dynamic soft label assignment strategy was proposed alongside the model, inspired by generalised focal loss [94]. This reduced some of the issues with using binary labels, which could cause unstable and noisy matching. Improved versions of MixUp [95] and CutMix [96] were implemented, by adding a caching mechanism. This reduced the time used to mix images into the training pipeline. RTMDet also uses two-stage training similar to that explored in YOLOX [97].

A specialised model for rotated object detection RTMDet-R was created, using three additional steps. These were:

1. A 1×1 convolution layer was added at the regression branch to predict the rotation angle of the object.

2. The bounding box coder was modified to allow for rotated bounding boxes.

3. The loss function was replaced with one which supports rotated objects.

3 Methodology

3.1 RoboRoyale Experiments

3.1.1 Arena Design

The RoboRoyale [30] project aims to influence the behaviour of European honeybee queens using biomimetic robot agents to replace the court bees that normally feed, clean and direct them. In order to achieve this aim it is important for as much of the behaviour of the queens and their attendant bees to be understood. It is useful to understand as many methods of controlling a queen as possible, as these methods all have distinct advantages and disadvantages and can be used together to influence her behaviour more reliably.

One of the stimuli that could be used for this influence is heat. The thermotactic behaviour of honeybees means that heating and cooling areas causes the bees to move, giving a way to control them. While this behaviour is understood for young drones [98] and workers [43] how this manifests with the queen either with or without her attendant bees is not fully understood. In order to understand this behaviour a team performed experiments at the University of Graz in Austria.

These experiments were performed using the robotic system ASSISI (**A**nimal and robot **S**ocieties **S**elf-organise and **I**ntegrate by **S**ocial **I**nteraction) [99]. An array of static devices are built into the floor of an arena. These devices are mounted on a multi-rail system, and can be combined together in different formats due to their modular design. Each device is known as a CASU (Collective Actuator Sensor Unit), with the formation as a whole being termed a CASU-matrix. These devices can change their state based on an input from the outside. The CASUs are capable of producing multiple types of stimuli, such as: temperature control, vibrations,

Figure 12: A picture of the ASSISI arena, with bees present. Only a small part of each CASU unit is visible above the arena, containing its components to create a stimulus. Also not the wax sheets covering the units.

electromagnetic fields and different light frequencies generated by RGB diodes.

For these experiments a CASU-matrix of 64 units was assembled. This large matrix can be subdivided into multiple smaller arenas for different experiments if the require different formations of CASUs. The floor of the arena is covered in wax sheets (as shown in Figure 12), in order to prevent the presence of materials unfamiliar to the bees affecting the experiments. These sheets were replaced between experiments so that the smells or hormones from previous bees did not affect the behaviour of bees in following experiments. Additionally, honeybees display phototropic behaviour [100], and it was therefore necessary for experiments to be performed in a dark room, so the presence of light did not affect their movement. This meant an infrared camera located above the arena was used for filming the experiments, as it was able to detect the bees even in a dark environment. However this did mean the experiments were

recorded with a much lower resolution than could have otherwise been expected and in black and white, due to the limitations of infrared cameras compared to their visible light counterparts.

For these experiments, only the temperature stimulus function of the CASUs was needed. These were used to create temperature gradients across the arena, ranging from higher temperatures in some area, to lower temperatures in others. The temperature of the CASUs can be set within a range of $10°C$ to $40°C$. This is a range of temperatures at which the bees are comfortable, and does not overly stress them.

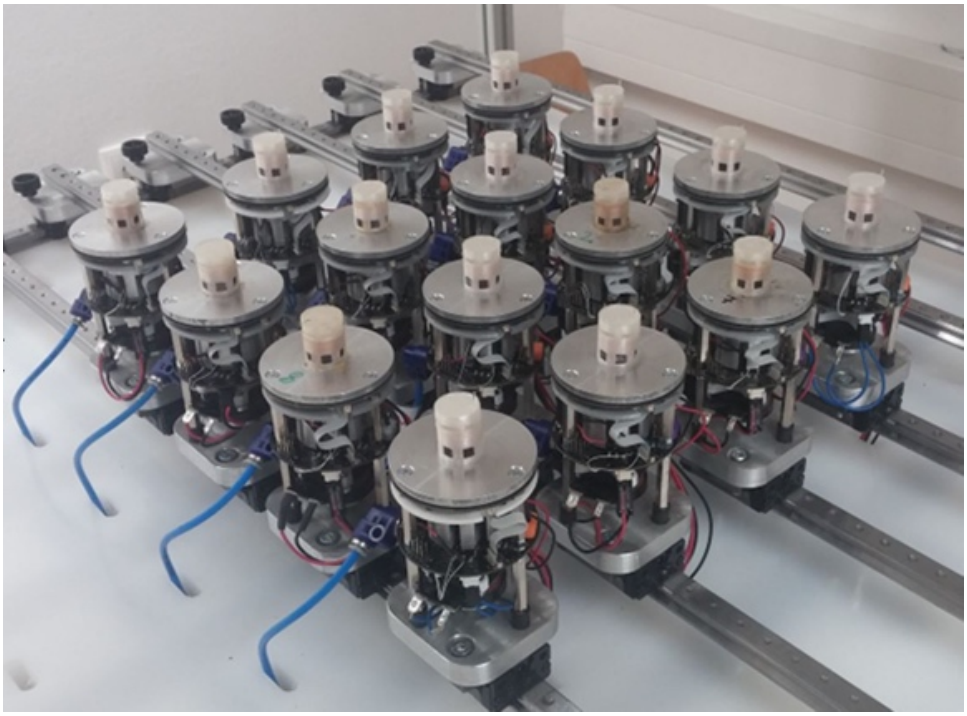

Figure 13: An array of CASUs, as used in the ASSISI arena, mounted on rails without any wax sheets covering them, from [99].

A thermoelectric Peltier module is used to provide a temperature stimulus, allowing fast temperature changes due to their fast response time. These modules can cause both a heating and cooling affect depending on how they are used. Heatsinks

are included in the unit to dissipate heat quickly once the temperature is changed.

A pre-programmed workstation can control the entire experimental platform, with a script running on this machine able to change the states of all CASUs in a matrix. This workstation is attached to the rest of the system using an industrial control terminal, and an Ethernet network is used to send data to each individual unit.

For these experiments honeybee workers were left unmarked, while the queens were given a drop of paint on their back. This was done so that the queens could be more easily identified from the rest of the bees.

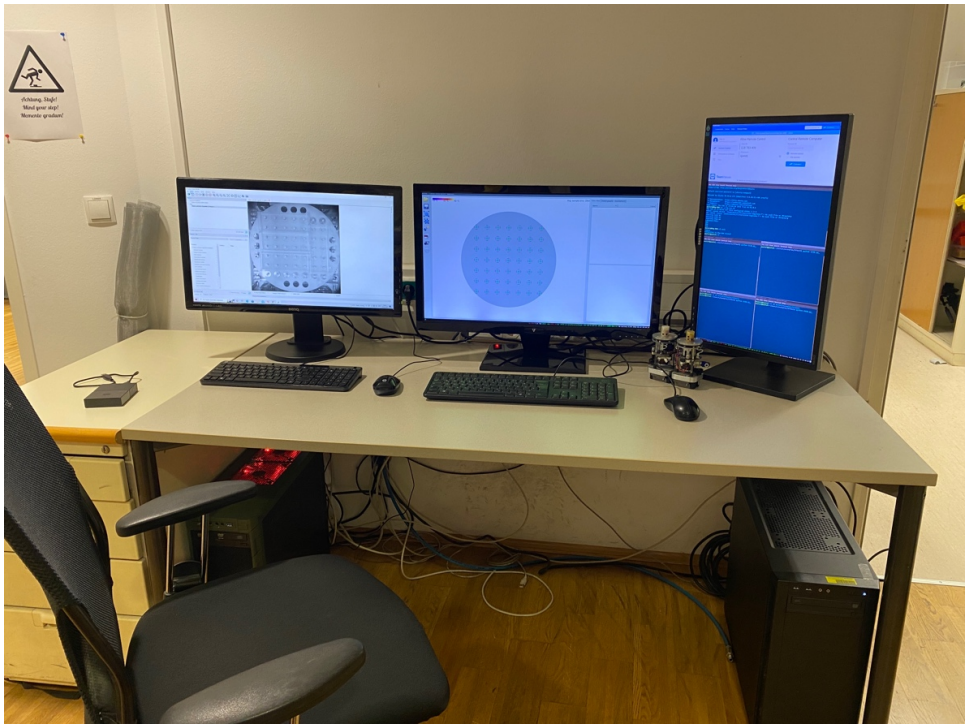

Figure 14: The workstation that runs the scripts to control the ASSISI arena.

3.1.2 Small Arena Experiments

The first set of experiments performed using the ASSISI arena for investigating the thermotactic behaviour of honeybee queens were done with a small arena size. A

CASU-matrix of just 2×1 was used for each arena. This allowed ten arenas to be fitted onto the main CASU-matrix, once sufficient space between arenas had been left to prevent bees from different experiments affecting each other.

Each experiment lasted for 120 minutes, with a picture being taken every second. At any one time one CASU of the two in an arena was heated. This was switched every X minutes during an experiment, in order to investigate how the bees would react to a change in thermal stimulus.

Overall four experiments were performed, two with just a single queen in each

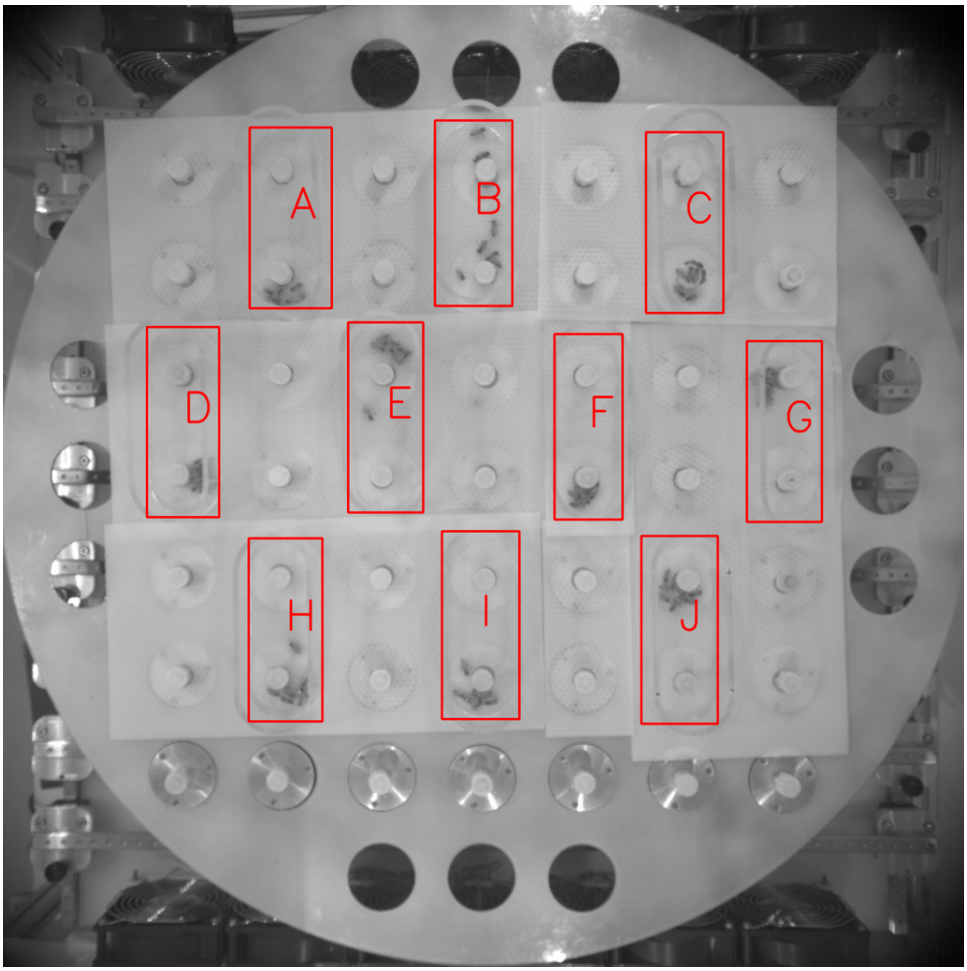

Figure 15: The layout of the ASSISI arena for experiments on the small arena. Each arena contains a 2×1 CASU-matrix, and is kept divided from the other arenas.

arena, and two with a queen and six attendant bees in each. Each of these experiments consisted of ten sub-experiments, one for each of the smaller arenas on the matrix. This meant that 40 experiments worth of data was produced, 20 for each experimental setting. From this $288,000$ images were produced by these experiments that would then require processing in order to extract the location data of the bees, so their thermotactic behaviour could be studied.

Table 1: The settings and dates for the experiments performed using the small arena setup.

	Date	Bees	Queen	Duration(mins)
1	28/06/2022	0	1	120
2	28/06/2022	0	1	120
3	29/06/2022	6	1	120
4	30/06/2022	6	1	120

3.1.3 Large Arena Experiments

The next set of experiments were performed using a larger arena size of 5×2 CASUs. In this configuration the entire matrix could only fit four arenas at one time. The CASU units were controlled in 1×2 pairs, with both units in one column being either heated or unheated at the same time.

A much larger range of experimental settings were used with this arena size. Three different parameters could be adjusted between experiments:

1. N_B, the number of non-queen bees in the arena

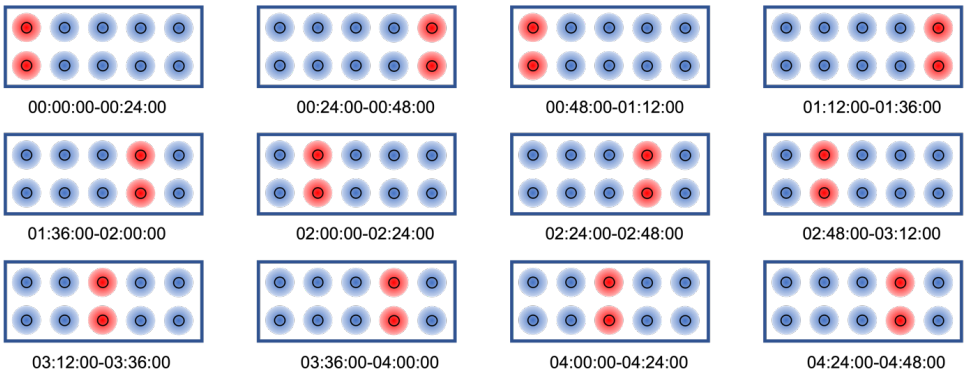

Figure 16: An example of timings for a script controlling the CASUs for large arena experiments, corresponding to pattern LETTER.

2. N_Q, the number of queen bees in the arena

3. S, which of the four timing scripts available was used

The number of non-queen bees (N_B) could be $\{0, 1, 6, 30, 60\}$, and the number of queens (N_Q) could be $\{0, 1\}$. The four timing scripts (S) were given the values of $\{a, b, c, d\}$. The full list of experiments performed is shown in Table 2.

In all four timing scripts the heated pair of CASUs was changed every 24 minutes. This gave twelve changes over the course of one experiment. An exemplar pattern is show in Figure 16.

These experiments lasted for 288 minutes, also having a picture taken once a second. This meant $17,280$ images were produced per experiment. Across all settings, 30 groups of experiments were performed, each containing four experiments with identical settings. This gave 120 total experiments, creating $2,073,600$ images requiring processing.

Table 2: The full list of experiments and their settings that were performed using the larger arena size.

	Date	Bees	Queen	Timing Script
1	18/06/2022	60	1	a
2	18/06/2022	60	1	a
3	19/06/2022	60	1	b
4	19/06/2022	60	1	b
5	20/06/2022	60	1	c
6	21/06/2022	60	1	d
7	22/06/2022	0	1	b
8	23/06/2022	0	1	a
9	24/06/2022	0	1	b
10	01/07/2022	6	1	a
11	07/07/2022	6	1	b
12	08/07/2022	6	1	b
13	12/07/2022	6	0	a
14	13/07/2022	6	0	c
15	14/07/2022	6	0	c
16	15/07/2022	6	0	d
17	19/07/2022	1	0	d
18	20/07/2022	1	0	c
19	21/07/2022	1	0	b
20	22/07/2022	1	0	a
21	26/07/2022	1	0	a
22	27/07/2022	1	0	b
23	28/07/2022	1	0	c
24	29/07/2022	1	0	d
25	02/08/2022	6	1	c
26	03/08/2022	6	1	d
27	04/08/2022	30	1	a
28	05/08/2022	30	1	b
29	09/08/2022	30	1	c
30	10/08/2022	30	1	d

3.2 Object Detection

3.2.1 Object Detection Pipeline

In order for the data collected to be useful, the honeybees present in the experiments would have to be localised. Over two million images had been produced across all 70 experiments that were performed. Manual processing of even a subset of this data would be far too time expensive in order to perform, especially as some images contain as many as 61 objects requiring tagging. Hence, it was required to find a method to automatically process these images, providing the positions of all honeybees in an image, with a high degree of accuracy and reliability.

Initially a traditional method for similar problems was trialled, blob detection. This method involved running a thresholding operation on an image to differentiate the bees from the background and then running a blob detection algorithm on the thresholded image in order to determine their positions.

However, these methods proved to be insufficient in this scenario. Due to the large populations of bees in some experiments, bees were often in very close proximity to each other. This led to the blobs produced by the thresholding operation merging

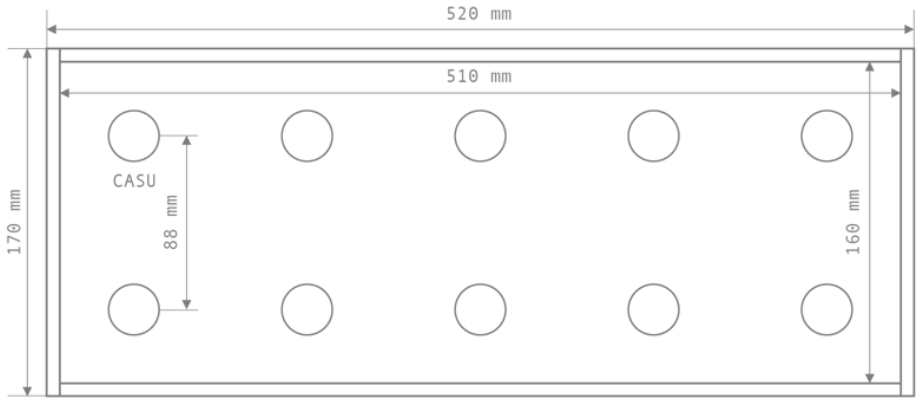

Figure 17: The size of a 2×5 arena.

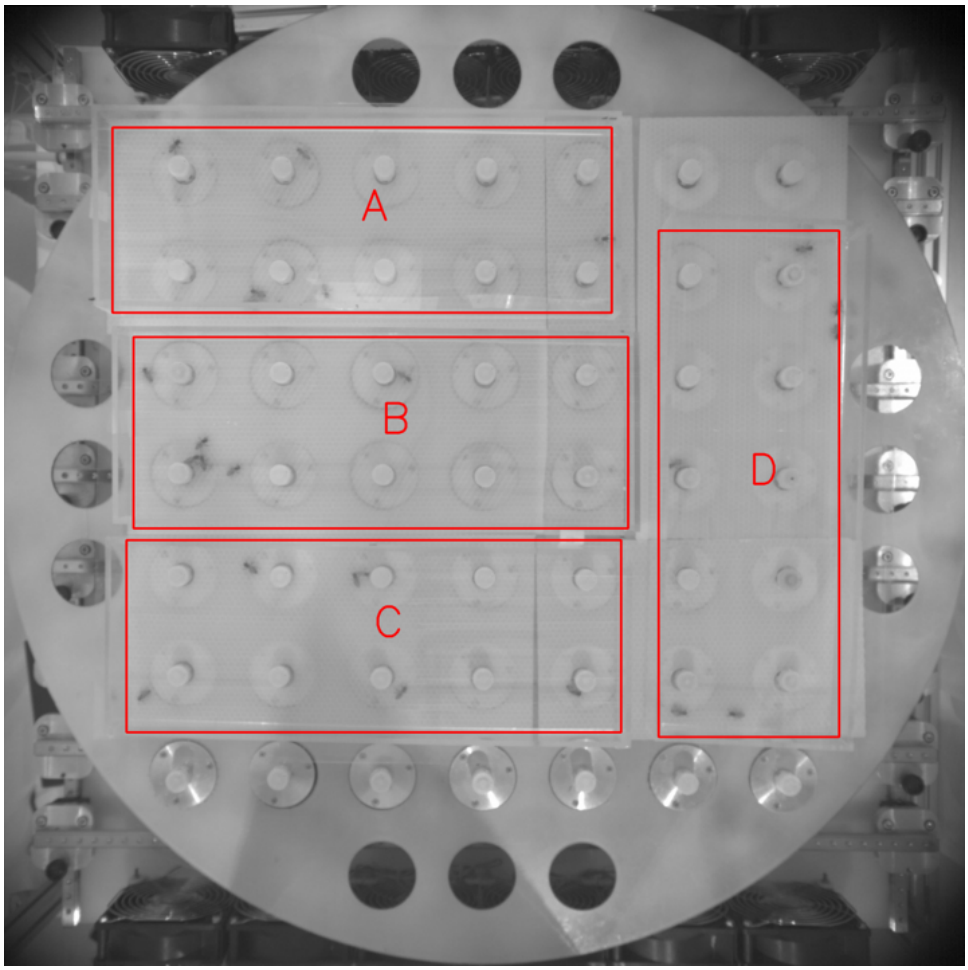

Figure 18: The layout of the ASSISI arena for experiments on the large arena. Each arena contains a 5×2 CASU-matrix.

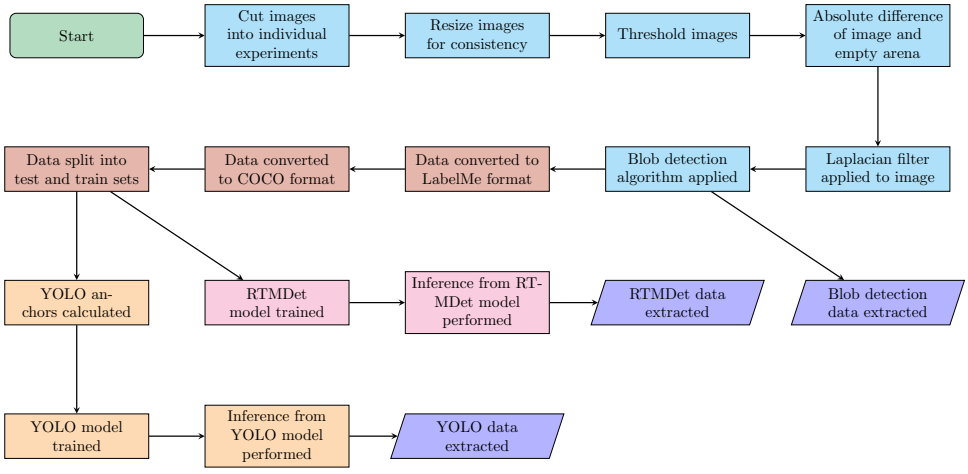

Figure 19: The object detection pipeline performed in this study.

together, becoming one large blob to be detected by the algorithm. This led to multiple bees being counted as one, or being ignored depending on the filtering parameters chosen for the algorithm.

In order to produce meaningful data from the images created by the experiments, it was necessary to accurately count the number of bees in specific areas of the arena, the areas that were either hot or cold. Therefore a method of localising the bees that was more robust against the issue of crowding was required. Given the recent advances in deep-learning based object detectors, it was decided to use utilise these methods to solve this problem. Two different deep-learning object detectors were chosen to be tested on the data to see if they could be used to localise the bees reliably. These models would have to be trained on pre-tagged experimental data in order to be used for this task.

When training a bespoke deep-learning based object detection model, a large amount of labelled data is normally needed in order to train the model to detect the objects necessary for that task. These images are normally labelled through manual tagging. This is a very time intensive process, as a human has to manually draw a bounding box around every single object in every image. This greatly increases the time and cost required to use these models for a custom task that requires bespoke training. It was therefore decided to investigate the feasibility of using data tagged by the traditional blob detection method to train the deep-learning models. These more complex models could then be used to localise the honeybees in the more crowded images, as well as other images the blob detector failed to correctly position the bees in.

The automatic tagging of these images would allow the deep-learning models to be trained without the manual tagging of any images, meaning the entire process of

tagging, training and localising would be automatic. Once the deep-learning models had been trained, they could be compared to the original blob-detection method and against manually tagged data in order to compare their performance and to find out if the deep-learning models provided a significant increase in accuracy and robustness.

3.2.2 Preprocessing

The first step required in order to process the images was to crop the original images. With multiple experiments in one image, as shown in Figure 20, each experiment

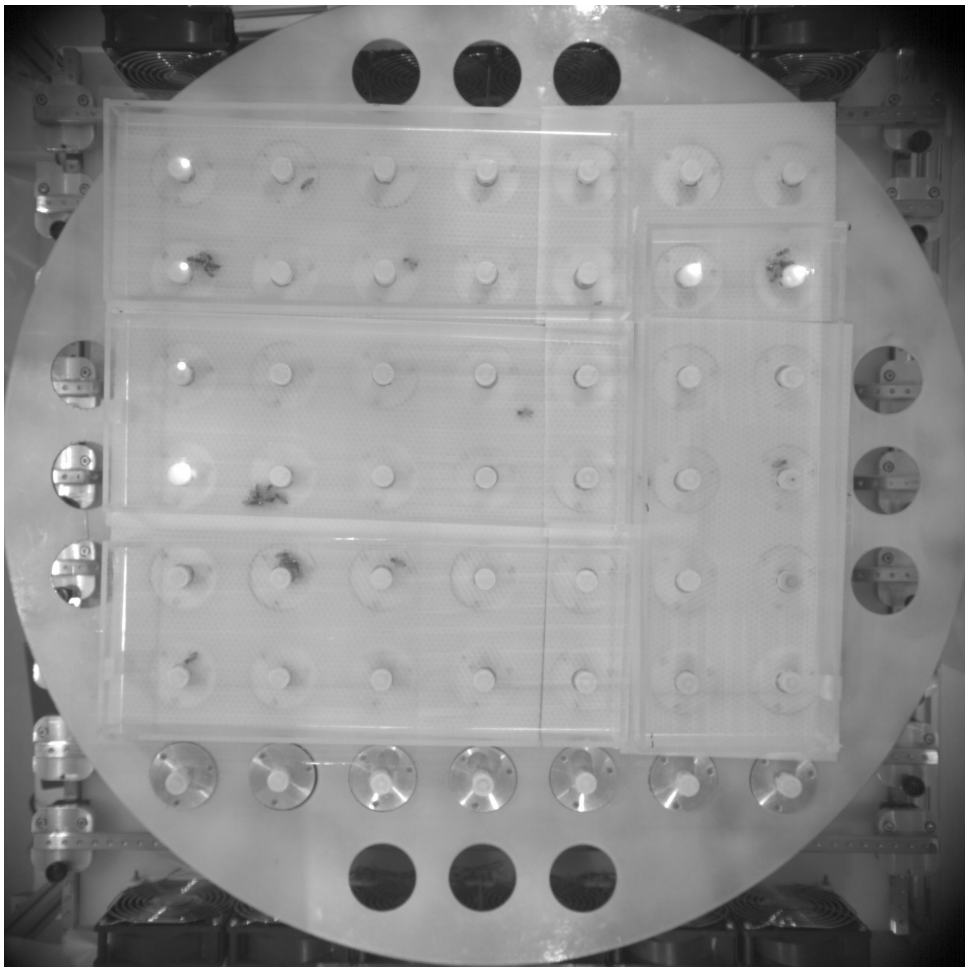

Figure 20: The initial image as taken by the camera, containing three different experiments

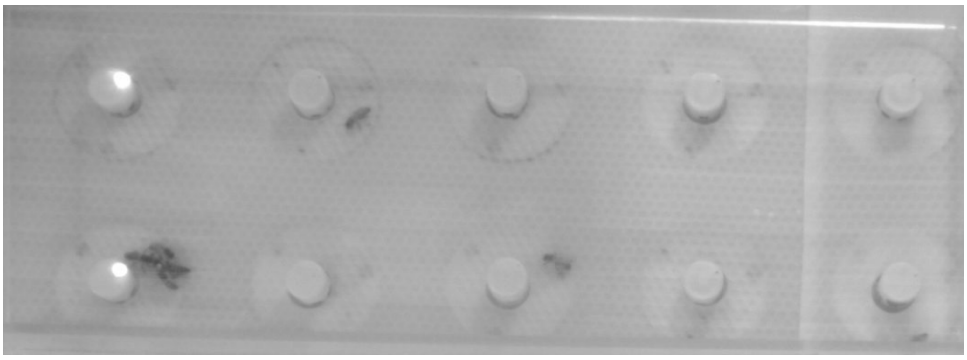

Figure 21: The image for a single experiment after being removed from the larger image and resized

was cut out from the main image and saved separately. These were then resized and reoriented for consistency. This produced a set of images that could be processed and then compared against each other. This produced images such as Figure 21.

3.2.3 Blob Detection

The blob detection algorithm used is from the OpenCV [101] library. It works by linking together large areas of similar colours or shades into one contiguous object, checking that object against a selection of filtering criteria in order to determine if that object is a blob that it is searching for, or not. In the case of these experiments the background of the wax sheets appears white, while the bees themselves are black, allowing the blob detection algorithm to differentiate them from their surroundings.

In order for the blob detection algorithm to be as effective as possible some preprocessing operations were first performed. The first of these was thresholding, this was performed using the following formula:

$$thresh(x,y) = \begin{pmatrix} 255 & if\ img(x,y) > 140 \\ \\ 0 & otherwise \end{pmatrix}$$

All bees are picked out in black, alongside some other details, and everything else

reduced to white.

Figure 22: The image after the performing thresholding the first time, with all areas above a value of 140 being set to white, and all others to black

Next the absolute difference of the thresholded image and the raw background image is found (Figure 24).

This was done to remove detail and noise from the image, both of these could confuse the blob detection algorithm.

This removed the CASU units from the image, to prevent them being detected and tagged as bees. This left only the thorax and abdomen of a bee visible in the image, as shown in Figure 25, however this would give more reliable object detection.

Laplacian filtering [102] is then applied to the image (Figure **??**). This is an

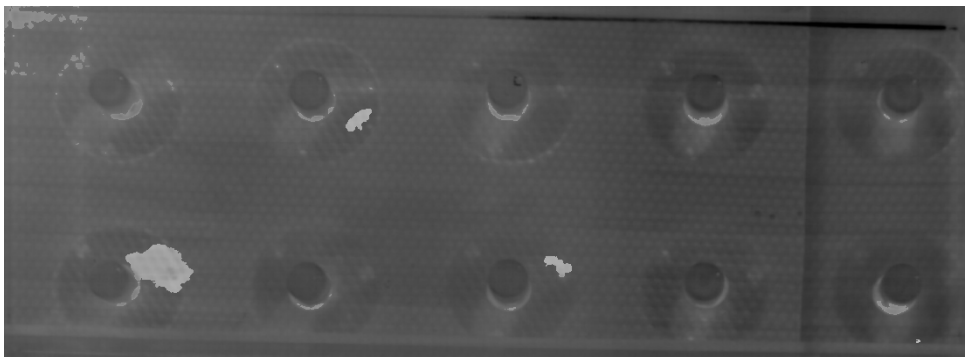

Figure 23: The image after the absolute difference function has been performed. The bees are picked out distinctly as white areas

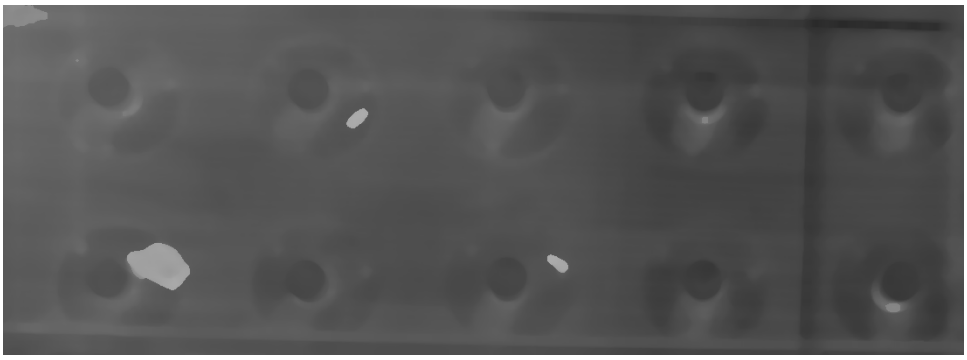

Figure 24: The image after the median blur is applied

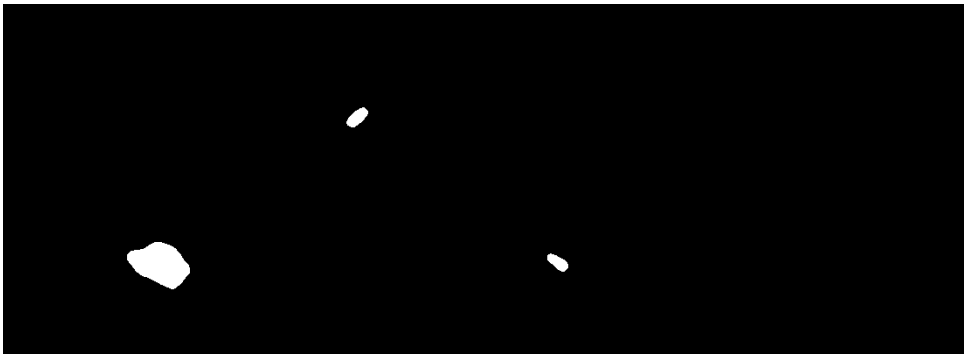

Figure 25: The image after the second set of thresholding

operation that represented by the formula:

$$L(x, y) = \frac{\delta^2 I}{\delta x^2} + \frac{\delta^2 I}{\delta y^2}$$

where $L(x, y)$ is the Laplacian of a pixel, and $I(x, y)$ is the intensity of a pixel. This increases the definition of the edges of objects in the image, making the detector more reliably able to detect the blobs.

The blob detection algorithm could then be applied to the Laplacian of the image. This algorithm first performs a variety of thresholding operations on an image to create a number of binary images. Connected components of similar shades could then be found by detecting contours in the image. The centres of these components is calculated, and if the centres of components in several binary images are close together, a blob is found in that location. The position of the centre of the blob, its

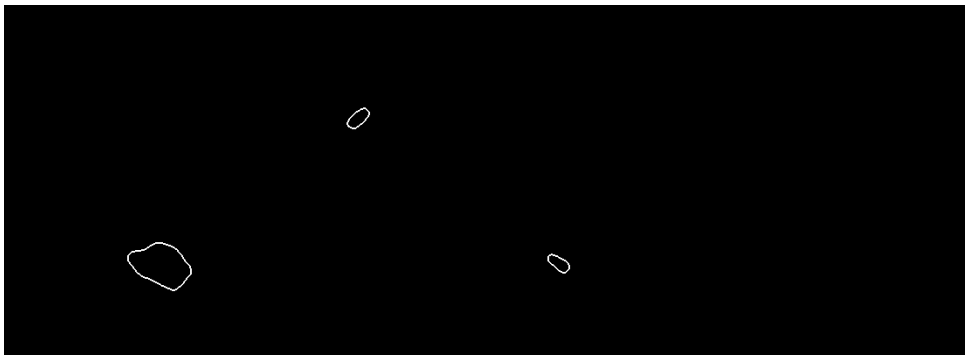

Figure 26: The image after the laplcian filtering

keypoints and size are then calculated.

A number of filters are then applied to the blobs. The filters used here were:

- minThreshold = 0
- maxThreshold = 2000
- filterByArea = False
- filterByConvexity = False
- filterByInertia = False

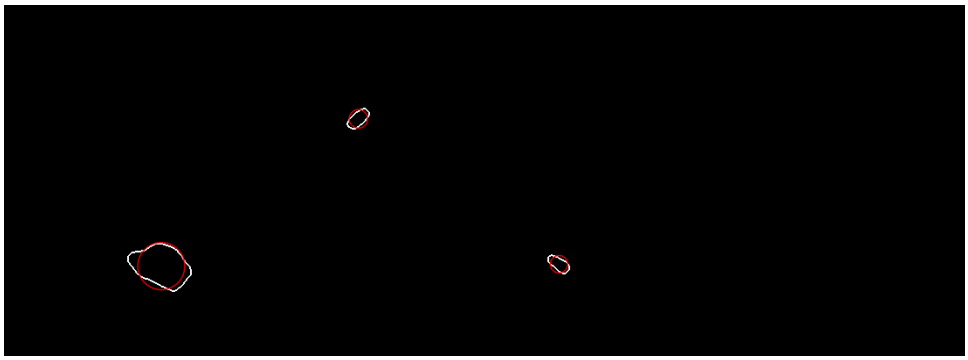

Figure 27: The image with the blobs found by the method marked

After the filtering is complete, the position of the blobs that relates to the location of the bees in the arena was then saved to a CSV file. Each entry contained the

name of the image, the x any y coordinates of the blob that had been found, and the size of the blob.

The number of bees in a blob is also estimated from the size of the blob,

Table 3: The format of the data that was created by the blob detection method

Image	x	y	Size
...	...	...	...

3.2.4 Automatic Tagging

This data produced by the blob detection algorithm could then be used as the basis for the labelled images used to train the deep-learning based models. The coordinates and size of the blobs found could be used to create bounding boxes for the bees in the images. It is however important that the bounding boxes used to train the models were correct, and the data is clean, otherwise the models would be improperly trained.

Initially, in order to achieve this, due to the blob detection algorithm almost always correctly localising the bees in their images, only images from experiments where one bee was in an arena at a time was used, or experiments were the following expression held true:

$$((N_B = 1) \cap (N_Q = 0)) \cup ((N_B = 0) \cap (N_Q = 1))$$

These images produced a very high successful detection rate, and were not affected by the issue of crowding which affected all other images due to having multiple bees present. This therefore provided clean training data of correctly labelled images for the deep-learning models.

3.2.5 Deep Learning

Two different state-of-the-art deep-learning based object detection models were implemented and trained on these images and then their accuracy at localising the bees was compared. The first model used was YOLOv6 [87] and the second used was RTMDet [91]. Comparing this models would provide insights into how important the choice of model is when using a deep-learning based object detector for this kind of task. Both of these models were implemented using the MMYOLO library [103], which is based off the popular MMDetection framework [104], which itself uses PyTorch.

For each arena size 5,000 images containing a single bees were collated from the image labelled by the blob detection algorithm. The bounding boxes were then calculated for each image from the coordinates of the bee, and the size of the blob that was found.

The MMYOLO library uses the COCO dataset [41] to train models and can only accept labelled images in this format. However, the COCO format is very complex, for this reason it was found to be easier to transform the data to the much simpler LabelMe format [105] as an intermediate step. From this format the dataset could be more easily converted into the COCO format which is standard for many object detection models.

The dataset was then split into test and train sets at a $4:1$ ratio. Anchors then had to be calculated for the YOLOv6 model, in order to ensure optimal training. Once this was completed both models could then be trained on each dataset. This gave four fully trained models, two for each arena type.

It was found that the models produced by this training would not perform on images with more than one bee well, so one hundred manually tagged images with 7

bees were added to the training data to supplement the automatically tagged data.

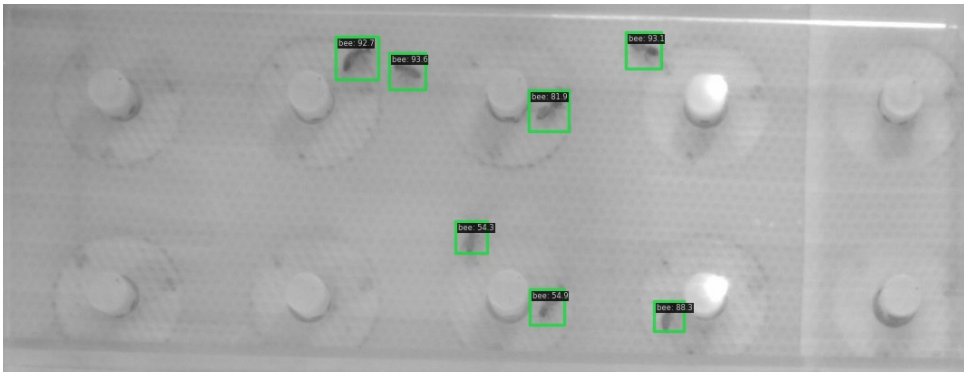

Figure 28: This image shows the output on this image produced by the RTMDet model.

3.3 Metrics

This gave three different models to be compared, the traditional blob detection method, and the two deep-learning based models, YOLOv6 and RTMDet. It was decided to compare these images by measuring their accuracy against manually labelled data to ensure fairness.

Table 4: The hardware details and the parameters used for training the deep learning models

	YOLOv6	RTMDet
GPU	NVIDIA RTX Titan	
Batch Size	32	32
Epochs	200	200
Learning Rate	0.0025	0.004

For small arenas, 40 images were manually labelled for experimental settings of $\{N_B = 0, N_Q = 1\}$ and $\{N_B = 6, N_Q = 1\}$. For large arenas, the same number of images were manually labelled for settings: $\{N_B = 0, N_Q = 1\}$, $\{N_B = 6, N_Q = 1\}$, $\{N_B = 30, N_Q = 1\}$ and $\{N_B = 60, N_Q = 1\}$. Bees localised by the three models could then be compared to these correctly labelled images.

The first metric used to compare the models was the number of successfully localised bees, or recall, N_{BC}. This is the sum across all 40 images for an experimental of the number of successfully localised bees for each model. If n_{BC} is the set of bees successfully localised in an image N_{BC} is given by the formula:

$$N_{BC} = \sum_{30}^{0} n_{BC}$$

The second metric used is the number of images where all bees were successfully localised, N_{IC}. Any image where a bee was not successfully localised, or a bee was localised where no bee actually existed was not counted for all bees having been successfully localised.

Both metrics assess different aspects of the performance of the models on these images. N_{BC} gives an assessment of the reliability of the models to detect any singular bee in any of the images. This gives the general performance of the model in most situations. The reason for also using N_{IC} is to test if the model struggles to detect a few bees in all images, or whether it will fail to detect a large number of bees in a couple of very difficult image, while correctly localising all bees in most images.

3.4 Statistical Analysis

3.4.1 Chi-squared Tests

Chi-squared (χ^2) tests [106] can be used to determine if an observed frequency differs significantly from an expected frequency for categorical data. This test is used here for both comparing both N_{BC} and N_{IC} results between models. It can be used to determine if the frequency of successful localisations differs significantly different depending on the object detection model, and can be used across all models, or between just two to compare if the difference between that pair is significant.

3.4.2 ANOVA Tests

Further statistical analysis is then performed using two-way ANOVA (**AN**alysis **O**f **VA**riance) [106] tests. The F-value calculated using this test determines whether the change of value of an independent variable is significant or not. This value can be calculated by using the following formula:

$$F = \frac{\text{variation among the sample means}}{\text{variation among individuals in the same sample}}$$

A larger F-value means that changes to that variable have a greater impact than could be expected from randomness in the data, and is therefore more significant.

The other value calculated using the ANOVA test is the p-value. This is calculated from the F-value of the variable and can be used to determine if a given independent variable is significant or not. For the purposes of this investigation a value of $p > 0.05$ can be considered to be significant.

4 Results

4.1 Small Arena

4.1.1 Data

The raw data for the experiments performed on the small arena can be found in Table 5. It can be seen that the results are better for the deep-learning models than the blob detection one. This is supported by the graphs in Figure 29.

4.1.2 Chi-squared Tests

The results from the χ^2 tests also do not suggest a significant difference in performance between the different models. Table 6 shows a small difference was detected between the different methods even for populations of only one bee. For the difference between the deep-learning models, the difference between the models was not large enough to be displayed to three significant figures in most cases.

Table 5: 2-way table for the experiments on the small arena

N_{Bee}	Model	N_{IC}	N_{BC}
1	Blob Detection	36	36
	YOLOv6	39	39
	RDM-DET	39	39
7	Blob Detection	1	146
	YOLOv6	8	197
	RDM-DET	8	205

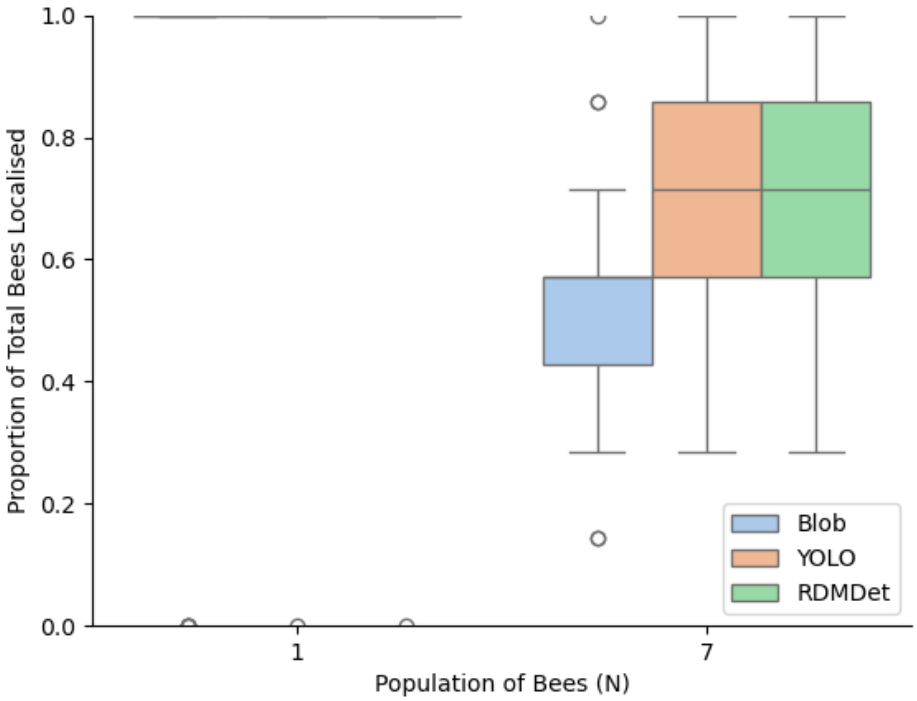

Figure 29: Graph showing N_{BC} for against different models and populations for the small arena experiments.

4.1.3 Analysis

The population of bees $\{N_Q + N_B\}$ never was high enough to present significant difficulties for any of the methods for the experiments performed on the small arena.

Table 6: χ^2 test results all methods

Arena Type	Variable Tested	1	7	31
Small	N_{BC}	0.960	0.029	
	N_{IC}	0.956	0.069	
Large	N_{BC}	0.983	0.026	0.00
	N_{IC}	0.983	0.073	N/A

It is therefore hard to gain meaningful results on their performance for this scenario.

4.2 Large Arena

4.2.1 Data

The results in Table 8 show a much greater difference in performance in models for the experiments using a large arenas, compared to what was seen for the experiments for small arenas. The performance of the blob detection method decreases very rapidly at higher populations of bees. This decline in performance is also notable in the deep-learning models, although it is not as dramatic.

It can be observed in Figure 31 that the variation of bees detected at higher populations is much higher than at lower populations. This suggests that large groups of bees form in some images and lead to a large number of bees in a single image being failed to be detected. At smaller populations this is either less likely or does not happen.

Table 7: χ^2 test results comparing deep-learning based models

Arena Type	Variable Tested	1	7	31
Small	N_{BC}	1.000	0.811	
	N_{IC}	1.000	1.000	
Large	N_{BC}	1.000	0.993	0.435
	N_{IC}	1.000	1.000	N/A

4.2.2 Chi-squared Tests

The data in Table 6 shows that the significance of the model chosen massively increases at higher populations. At a population of 7 the difference between different methods is still relatively minor, with not much variation. However at a population of 31 the difference is incredibly significant. This holds for both N_{BC} and N_{IC}.

4.2.3 ANOVA Tests

The ANOVA tests show a similar story to the χ^2 tests. For the comparisons between models in Table 9, the results show that choosing between RTMDet and YOLOv6

Table 8: 2-way table for the experiments on the large arena

N_{Bee}	Model	N_{IC}	N_{BC}
1	Blob Detection	38	38
	YOLOv6	40	40
	RDMDet	40	40
7	Blob Detection	2	156
	YOLOv6	8	214
	RDMDet	8	216
31	Blob Detection	0	305
	YOLOv6	0	545
	RDMDet	0	578

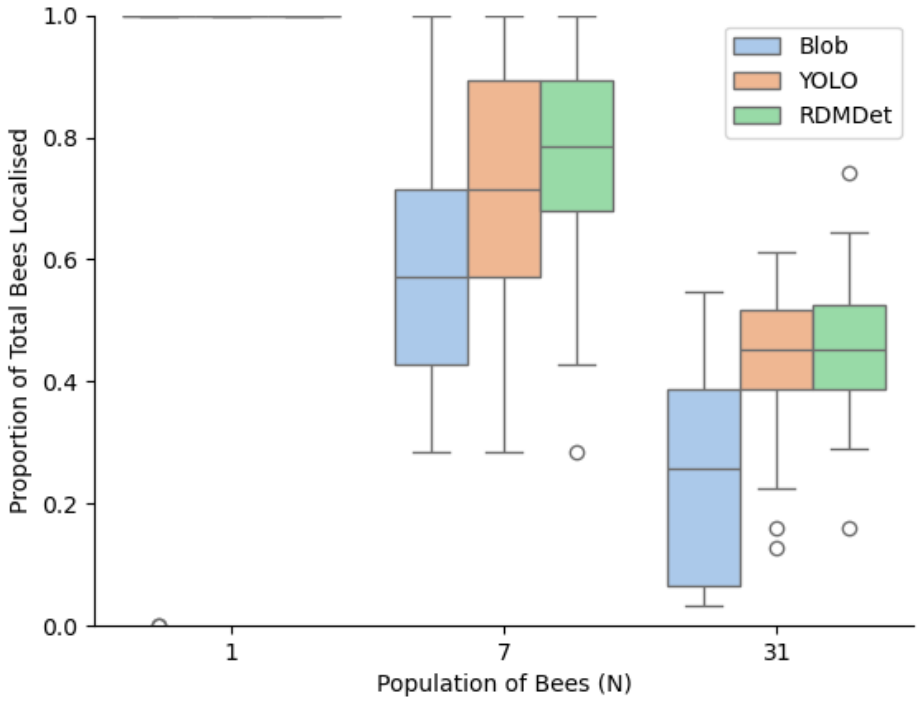

Figure 30: Graph showing the proportion of bees localised in each image for the large arena experiments

is not significant at lower populations although it does have an affect at higher populations, while the significance between the blob detection and deep-learning methods is very significant.

4.2.4 Analysis

It is clear that as the number of objects increases the proportion of images where all objects have been successfully drops substantially.

It is also clear that the proportion of objects successfully detected also drops. As bees tend to aggregate around the temperature source, they get very close together, including sometimes atop one another. This makes it very hard for the models to successfully detect these bees. The more the population of the bees increases, the

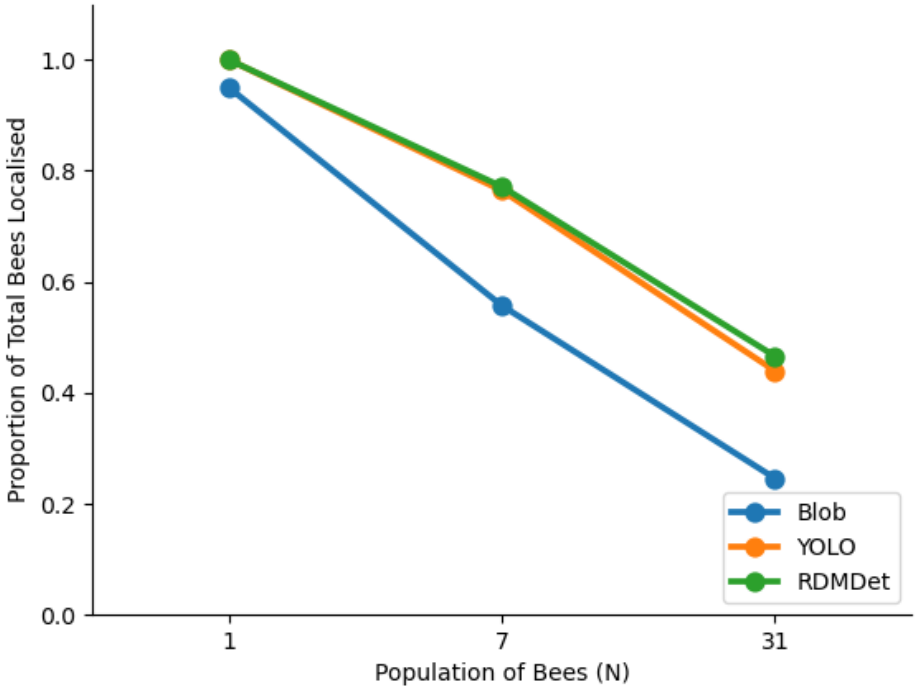

Figure 31: Graph showing N_{BC} for against different models and populations for the large arena experiments.

more this effect increases because there are more bee in the same sized area.

This effect is more pronounced with the blob detection method because if the blobs created by the bees overlaps the detector will no longer detect them as separate bees, but a new larger blob. Thus a dense region of bees can merge into one using this method, making a large number of bees unable to be detected at all. This explains why it was so unlikely that only a small number of bees were not localised correctly.

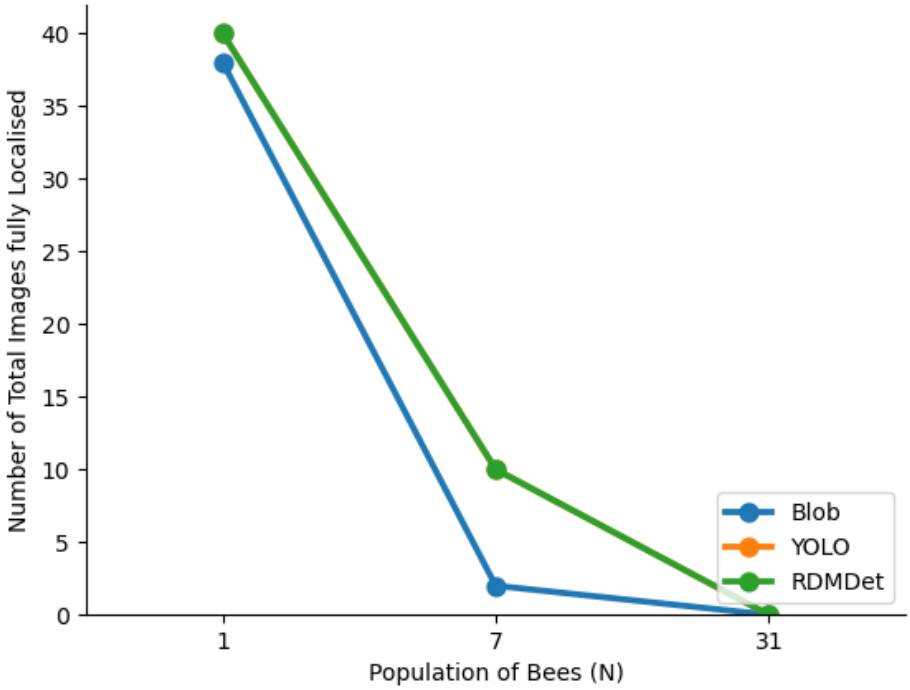

Figure 32: Graph showing N_{IC} for against different models and populations for the large arena experiments.

5 Evaluation

5.1 Review of Objectives

5.1.1 Providing Position Estimates

For the images produced by the experiments, localisation data was provided with a good degree of accuracy. A higher degree of accuracy would have been desirable, but the nature of the task provided difficult images for these operations to be performed on. Many of the objects were superimposed on top of each other, blocking each other from view, making 100% accuracy impossible. However, due to the outcomes wanted from the data, the models proved sufficient to the task. They correctly localise enough of the bees for the behaviour of the group as a whole to be determined.

5.1.2 Investigating Traditional Methods

This investigation showed clearly some of the drawbacks of traditional methods for providing the location estimates required for this experiment.

Differentiating separate objects in crowds was a clear issue, the blob-detection model having no ability to pick apart objects once they were close enough together because the thresholding had produced one large blob from a crowd of bees. The results included in this investigation represent those that produced the best results, making it difficult to improve this method as the parameters of the thresholding algorithm had already been modified to include as little of the bee as possible while still detecting them reliably.

Furthermore, producing the blobs requires the objects to be distinct enough from their environment for thresholding to distinguish them. This means this method

Table 9: Results of ANOVA tests performed on data from the large arena experiments comparing different models against each other

Model 1	Model2	1		7		31	
		F	P	F	P	F	P
Blob Detection	YOLOv6	2.05	0.156	23.0	0.00	37.8	0.00
Blob Detection	RTMDet	2.05	0.156	24.9	0.00	48.3	0.00
YOLOv6	RTMDet	N/A	N/A	0.029	0.866	1.18	0.281
Combined		2.05	0.156	16.2	0.00	33.4	0.00

would be incompatible with many experiments where this would not be the case.

Finally, there was no method by which this model could separate queens and drones. By transforming all bees to blobs, there was no way to distinguish the two categories of objects reliably, the white dots on the back of the queens having been lost in the transformation. There are examples of objects that this method could distinguish between, based on the size and shape of the objects, but this was not the case in this instance.

5.1.3 Training Deep-Learning Models

Two different deep-learning models were trained on this data, YOLOv6 and RTMDet. These models were able to detect the bees successfully, with a reasonable level of accuracy, considering the difficulties inherent with this investigation, namely the crowding of bees due to their aggregation.

Both models have methods to deal with some blur in the image, which is important for localising objects in these images because of their low resolution. In the case of YOLOv6 this is due to the anchor-free detection which reduces box-instability, helping to reduce the effects of the blur produced by the lower resolution camera. For RTMDet the factor that helps is the improved receptive field of the model, which helps the model deal with blur and occlusion, both of which are prominent in these images.

This was performed with minimal manual intervention, all data having been tagged by the traditional model, with the input images only being inspected to ensure they were accurately and reliably tagged. This produced a significant decrease in time required to tag these images, the training taking hours, and the tagging minutes, compared to the intractably long time manual tagging would have taken.

5.1.4 Comparing Traditional and Deep-Learning Models

Throughout the investigation the deep-learning models performed better than the traditional method. They overcome many of the shortfalls of that method, not being reliant on a single method of detecting. While still suffering from the issue of crowding and obscured objects, they did show improved performance with a more populated environment.

5.1.5 Investigating the Limits of These Methods

It was shown in this investigation that the deep-learning based models still struggled to perform accurately on a sufficiently crowded image. This shows that it best to design experiments were overcrowding is not an issue, as it will allow the models to perform at their best.

5.2 Chi-Squared Suitability

5.2.1 Requirements

In order to use a Chi-Squared test the data has to follow the following rules:

1. No category has a mean of less than 5
2. The categories should be as natural as possible.
3. The raw data should be kept.
4. No category should vastly exceed the others.

5.2.2 Analysis

The data collected for the experiments is suitable for Chi-Squared in most cases. For the experiments with blob detection where no images have been detected the mean

of that category is below zero. Inspection of the data in those cases shows the results for blob-detection are evidently different compared to the deep-learning methods, the blob-detection having fully tagged zero images correctly with the deep-learning models still proving somewhat effective. Otherwise the data is not transformed before the tests are used.

5.3 ANOVA Suitability

5.3.1 Requirements

There are three assumptions that the data needs to follow for ANOVA tests to be suitable [106]:

1. **Normally Distributed Data:** The residuals of the dependant variable should be normally distributed.

2. **Homogeneity of Variance:** the variance of each group should be equal.

3. **Independence of Observations:** The experiments should not affect one another.

ANOVA tests are relatively robust if these assumptions are violated, making them very useful for analysing data without too much worry about the assumptions being fully met.

5.3.2 Limitations

A major limitation of this study is the specialised task investigated. The task requires the model to locate a large number of objects, on a greyscale, low resolution image, so results may be different when used on clearer images and with colour that

provides a clearer object for the model to localise. This investigation also had to locate a very high number of objects, which is not typical for the use of these models.

Another limitation is the use of only one traditional method of object localisation. Other methods could provide different performance, especially with other tasks and image types. The deep-learning models available are constantly evolving, and the performance of the models used here could be very quickly surpassed by newer models. It is also the case that while it is fast to locate objects using these models, they take a lot of time to train due to being computationally complex.

6 Conclusion

6.1 Findings

The results show that the deep-learning models provide much more accurate position estimates for this problem than the traditional method used. There appears to be little difference between the two state-of-the-art models used however. This suggests that there is a place for training specific use deep-learning based object detection models in the research environment.

The deep-learning models provide good accuracy, especially on the less crowded images, and their data can be used for this task. However, if it is of critical importance that the data is completely accurate these models did make mistakes and cannot guarantee a flawless output in the same way as a human's position estimates could. They do provide much faster output though, and can be used on data that is so large it would be impossible to process manually.

Further, it was found that differentiating between two very similar objects, such as a between a honeybee drone and a honeybee queen is very difficult for these models.

6.2 Further Work

Currently it requires a lot of specific knowledge and hardware to tailor train a deep-learning model in this fashion. Both major frameworks for deep-learning based object detection require Nvidia GPUs specifically, with the frameworks not being compatible with other manufacturers. Additionally, the software required is complex to install, with very specific versions of software being compatible with each other, with these requirements being difficult to find online due to the available guides

being out of date and the install guides being out of date and therefore not leading to a working system. For these methods to be used across many research fields and to reach their full potential, they need to be made more accessible, and the ease of use needs to be given more thought.

This investigation only tested these models on one environment. The type of environment and experiment could have a large impact on the performance of these models. Testing these models for a wider variety of similar tasks would provide a better understanding of their efficiency for these tasks. It would also allow an investigation into how changing the environment or type of task affects the performance of these models, giving new insights into their use.

Only two different deep-learning models were tested in this investigation and other models could provide different advantages or disadvantages compared to these. In particular comparing a two-stage detector such as Faster R-CNN to these models could provide valuable insights into how a system using object detection for similar tasks as this one should be designed and which models they should utilise.

Additionally, it would be useful to compare these methods to a more complex conventional method such as HOG. However, after OpenCV removed support for training HOG models in 20xx it has become substantially more difficult to create a model for a specific task. This limits the feasibility of using it for a research task similar to the one here, given the number of other models available which allow for easier training.

The images taken as part of the thermotactic aggregation experiment were purely greyscale, having been taken with a thermal? camera. Colour could have a large impact on the performance of the models, giving another difference between an object and it's environment. Hence an investigation comparing greyscale and colour

images could provide a useful insight into how to design an experiment to properly utilise these methods, finding out whether it is important to ensure that images contain colour, or whether it would be unnecessary.

References

[1] T.-L. Ashman, T. M. Knight, J. A. Steets, P. Amarasekare, M. Burd, D. R. Campbell, M. R. Dudash, M. O. Johnston, S. J. Mazer, R. J. Mitchell, M. T. Morgan, and W. G. Wilson, "Pollen Limitation of Plant Reproduction: Ecological and Evolutionary Causes and Consequences," *Ecology*, vol. 85, no. 9, pp. 2408–2421, 2004.

[2] J. Ollerton, R. Winfree, and S. Tarrant, "How many flowering plants are pollinated by animals?" *Oikos*, vol. 120, no. 3, pp. 321–326, 2011.

[3] K.-L. J. Hung, J. M. Kingston, M. Albrecht, D. A. Holway, and J. R. Kohn, "The worldwide importance of honey bees as pollinators in natural habitats," *Proceedings of the Royal Society B: Biological Sciences*, vol. 285, no. 1870, p. 20172140, Jan. 2018.

[4] S. G. Potts, J. C. Biesmeijer, C. Kremen, P. Neumann, O. Schweiger, and W. E. Kunin, "Global pollinator declines: trends, impacts and drivers," *Trends in Ecology & Evolution*, vol. 25, no. 6, pp. 345–353, Jun. 2010. [Online]. Available: 4

[5] I. H. Williams, "The dependence of crop production within the European Union on pollination by honey bees," *Agricultural Science Reviews*, vol. 6, pp. 229–257, 1994.

[6] L. A. Garibaldi, I. Steffan-Dewenter, R. Winfree, M. A. Aizen, R. Bommarco, S. A. Cunningham, C. Kremen, L. G. Carvalheiro, L. D. Harder, O. Afik, I. Bartomeus, F. Benjamin, V. Boreux, D. Cariveau, N. P. Chacoff, J. H. Dudenhöffer, B. M. Freitas, J. Ghazoul, S. Greenleaf, J. Hipólito, A. Holzschuh,

B. Howlett, R. Isaacs, S. K. Javorek, C. M. Kennedy, K. M. Krewenka, S. Krishnan, Y. Mandelik, M. M. Mayfield, I. Motzke, T. Munyuli, B. A. Nault, M. Otieno, J. Petersen, G. Pisanty, S. G. Potts, R. Rader, T. H. Ricketts, M. Rundlöf, C. L. Seymour, C. Schüepp, H. Szentgyörgyi, H. Taki, T. Tscharntke, C. H. Vergara, B. F. Viana, T. C. Wanger, C. Westphal, N. Williams, and A. M. Klein, "Wild pollinators enhance fruit set of crops regardless of honey bee abundance," *Science (New York, N.Y.)*, vol. 339, no. 6127, pp. 1608–1611, Mar. 2013.

[7] N. W. Calderone, "Insect pollinated crops, insect pollinators and US agriculture: trend analysis of aggregate data for the period 1992-2009," *PloS One*, vol. 7, no. 5, p. e37235, 2012.

[8] *Status of Pollinators in North America*. Washington, D.C.: National Academies Press, Apr. 2007.

[9] M. Beekman and F. L. W. Ratnieks, "Long-range foraging by the honey-bee, Apis mellifera L." *Functional Ecology*, vol. 14, no. 4, pp. 490–496, 2000.

[10] N. Carreck and I. Williams, "The economic value of bees in the UK," *Bee World*, vol. 79, no. 3, pp. 115–123, Jan. 1998.

[11] A. Mizrahi and Y. Lensky, *Bee Products: Properties, Applications, and Apitherapy*. Springer Science & Business Media, Jun. 2013.

[12] N. L. García, "The Current Situation on the International Honey Market," *Bee World*, vol. 95, no. 3, pp. 89–94, Jul. 2018.

[13] S. Bogdanov, "Beeswax: Uses and Trade," Jan. 2009.

[14] B. Denisow and M. Denisow-Pietrzyk, "Biological and therapeutic properties of bee pollen: a review," *Journal of the Science of Food and Agriculture*, vol. 96, no. 13, pp. 4303–4309, 2016.

[15] P. Rosenkranz, P. Aumeier, and B. Ziegelmann, "Biology and control of *Varroa destructor*," *Journal of Invertebrate Pathology*, vol. 103, pp. S96–S119, Jan. 2010.

[16] E. Genersch, "Honey bee pathology: current threats to honey bees and beekeeping," *Applied Microbiology and Biotechnology*, vol. 87, no. 1, pp. 87–97, Jun. 2010.

[17] ——, "American Foulbrood in honeybees and its causative agent, *Paenibacillus larvae*," *Journal of Invertebrate Pathology*, vol. 103, pp. S10–S19, Jan. 2010.

[18] R. Galajda, A. Valenčáková, M. Sučik, and P. Kandráčová, "Nosema Disease of European Honey Bees," *Journal of Fungi*, vol. 7, no. 9, p. 714, Sep. 2021, publisher: Multidisciplinary Digital Publishing Institute.

[19] D. vanEngelsdorp, J. D. Evans, C. Saegerman, C. Mullin, E. Haubruge, B. K. Nguyen, M. Frazier, J. Frazier, D. Cox-Foster, Y. Chen, R. Underwood, D. R. Tarpy, and J. S. Pettis, "Colony Collapse Disorder: A Descriptive Study," *PLOS ONE*, vol. 4, no. 8, p. e6481, Aug. 2009.

[20] G. Singh and A. Rana, "Honeybees and colony collapse disorder: understanding key drivers and economic implications," *Proceedings of the Indian National Science Academy*, Feb. 2025.

[21] E. Forsgren, G. E. Budge, J.-D. Charrière, and M. A. Z. Hornitzky, "Standard methods for European foulbrood research," *Journal of Apicultural Research*, Jan. 2013.

[22] D. C. de Graaf, A. M. Alippi, K. Antúnez, K. A. Aronstein, G. Budge, D. De Koker, L. De Smet, D. W. Dingman, J. D. Evans, L. J. Foster, A. Fünfhaus, E. Garcia-Gonzalez, A. Gregore, H. Human, K. D. Murray, B. K. Nguyen, L. Poppinga, M. Spivak, D. van Engelsdorp, S. Wilkins, and E. Genersch, "Standard methods for American foulbrood research," *Journal of Apicultural Research*, vol. 52, no. 1, pp. 1–28, Jan. 2013.

[23] Y. L. Conte and M. Navajas, "Climate change: impact on honey bee populations and diseases."

[24] M. Ali, I. Abdellah, and M. Eletmany, "CLIMATE CHANGE IMPACTS ON HONEYBEE SPREAD AND ACTIVITY: A SCIENTIFIC REVIEW," *Chelonian Conservation and Biology*, vol. 18, pp. 531–554, Nov. 2023.

[25] B. W. Rowland, S. P. Rushton, M. D. F. Shirley, M. A. Brown, and G. E. Budge, "Identifying the climatic drivers of honey bee disease in England and Wales," *Scientific Reports*, vol. 11, no. 1, p. 21953, Nov. 2021.

[26] N. Simon-Delso, G. S. Martin, E. Bruneau, L.-A. Minsart, C. Mouret, and L. Hautier, "Honeybee Colony Disorder in Crop Areas: The Role of Pesticides and Viruses," *PLOS ONE*, vol. 9, no. 7, p. e103073, Jul. 2014.

[27] B. Nguyen, C. Saegerman, C. Pirard, J. Mignon, J. Widart, B. Thirionet, F. Verheggen, D. Berkvens, E. De Pauw, and E. Haubruge, "Does Imidacloprid

Seed-Treated Maize Have an Impact on Honey Bee Mortality?" *Journal of economic entomology*, vol. 102, pp. 616–23, May 2009.

[28] N. Desneux, A. Decourtye, and J.-M. Delpuech, "The Sublethal Effects of Pesticides on Beneficial Arthropods," *Annual Review of Entomology*, vol. 52, no. 1, p. 81, 2007.

[29] D. vanEngelsdorp and M. D. Meixner, "A historical review of managed honey bee populations in europe and the united states and the factors that may affect them," *Journal of Invertebrate Pathology*, vol. 103, pp. S80–S95, 2010. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0022201109001827

[30] M. Stefanec, D. N. Hofstadler, T. Krajník, A. E. Turgut, H. Alemdar, B. Lennox, E. Şahin, F. Arvin, and T. Schmickl, "A Minimally Invasive Approach Towards "Ecosystem Hacking" With Honeybees," *Frontiers in Robotics and AI*, vol. 9, Apr. 2022.

[31] A. R. Pathak, M. Pandey, and S. Rautaray, "Application of Deep Learning for Object Detection," *Procedia Computer Science*, vol. 132, pp. 1706–1717, Jan. 2018.

[32] D. Feng, A. Harakeh, S. L. Waslander, and K. Dietmayer, "A Review and Comparative Study on Probabilistic Object Detection in Autonomous Driving," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 8, pp. 9961–9980, Aug. 2022.

[33] C. Michaelis, B. Mitzkus, R. Geirhos, E. Rusak, O. Bringmann, A. S. Ecker, M. Bethge, and W. Brendel, "Benchmarking Robustness in Object Detection:

Autonomous Driving when Winter is Coming," Mar. 2020.

[34] J. Li and Z. Wang, "Real-Time Traffic Sign Recognition Based on Efficient CNNs in the Wild," *IEEE Transactions on Intelligent Transportation Systems*, vol. 20, no. 3, pp. 975–984, Mar. 2019.

[35] A. Vahab, M. S. Naik, and P. G. Raikar, "Applications of Object Detection System," vol. 06, no. 04, 2019.

[36] R. Yang and Y. Yu, "Artificial Convolutional Neural Network in Object Detection and Semantic Segmentation for Medical Imaging Analysis," *Frontiers in Oncology*, vol. 11, Mar. 2021, publisher: Frontiers.

[37] M. G. Ragab, S. J. Abdulkadir, A. Muneer, A. Alqushaibi, E. H. Sumiea, R. Qureshi, S. M. Al-Selwi, and H. Alhussian, "A Comprehensive Systematic Review of YOLO for Medical Object Detection (2018 to 2023)," *IEEE Access*, vol. 12, pp. 57 815–57 836, 2024.

[38] M. Tsuneki, "Deep learning models in medical image analysis," *Journal of Oral Biosciences*, vol. 64, no. 3, pp. 312–320, Sep. 2022.

[39] I. Ahmad, F. AlQurashi, E. Abozinadah, and R. Mehmood, "A Novel Deep Learning-based Online Proctoring System using Face Recognition, Eye Blinking, and Object Detection Techniques," *International Journal of Advanced Computer Science and Applications (IJACSA)*, vol. 12, no. 10, 2021.

[40] V. Ghenescu, R. E. Mihaescu, S.-V. Carata, M. T. Ghenescu, E. Barnoviciu, and M. Chindea, "Face Detection and Recognition Based on General Purpose DNN Object Detector," in *2018 International Symposium on Electronics and Telecommunications (ISETC)*, Nov. 2018, pp. 1–4.

[41] T.-Y. Lin, M. Maire, S. Belongie, L. Bourdev, R. Girshick, J. Hays, P. Perona, D. Ramanan, C. L. Zitnick, and P. Dollár, "Microsoft COCO: Common Objects in Context," Feb. 2015.

[42] J. Watmough and S. Camazine, "Self-Organized Thermoregulation of Honeybee Clusters," *Journal of Theoretical Biology*, vol. 176, no. 3, pp. 391–402, Oct. 1995.

[43] M. Szopek, T. Schmickl, R. Thenius, G. Radspieler, and K. Crailsheim, "Dynamics of Collective Decision Making of Honeybees in Complex Temperature Fields," *PLoS ONE*, vol. 8, no. 10, p. e76250, Oct. 2013.

[44] M. M. Millonas, "Swarms, Phase Transitions, and Collective Intelligence," Jun. 1993.

[45] T. Schmickl, R. Thenius, C. Möslinger, G. Radspieler, S. Kernbach, M. Szymanski, and K. Crailsheim, "Get in Touch-cooperative decision making based on robot-to-robot collisions," *Autonomous Agents and Multi-Agent Systems*, vol. 18, pp. 133–155, Feb. 2009.

[46] A. Radford, "Australia radioactive capsule: Missing material more common than you think," *BBC News*, Feb. 2023. [Online]. Available: https://www.bbc.co.uk/news/world-64512297

[47] D. Teodorovic, P. Lucic, G. Markovic, and M. D. Orco, "Bee Colony Optimization: Principles and Applications," in *2006 8th Seminar on Neural Network Applications in Electrical Engineering*, Sep. 2006, pp. 151–156.

[48] D. Romano, M. Porfiri, P. Zahadat, and T. Schmickl, "Animal–robot interaction—an emerging field at the intersection of biology and robotics," *Bioinspiration & Biomimetics*, vol. 19, no. 2, p. 020201, Mar. 2024.

[49] A. Update, L. C. J. Clark, R. B. Spokane, M. M. Homan, R. Sudan, and M. Miller, "Long-term Stability of Electroenzymatic Glucose Sensors Implanted in Mice," *ASAIO Journal*, vol. 34, no. 3, p. 259, Sep. 1988.

[50] W. Rajewicz, T. Schmickl, and R. Thenius, "The Use of Robots in Aquatic Biomonitoring with Special Focus on Biohybrid Entities," in *Advances in Service and Industrial Robotics*, A. Müller and M. Brandstötter, Eds. Cham: Springer International Publishing, 2022, pp. 521–527.

[51] W. Rajewicz, N. Helmer, T. Schmickl, and R. Thenius, "Living Organisms as Sensors for Biohybrid Monitoring Systems," in *Biomimetic and Biohybrid Systems*, F. Meder, A. Hunt, L. Margheri, A. Mura, and B. Mazzolai, Eds. Cham: Springer Nature Switzerland, 2023, pp. 348–362.

[52] W. Rajewicz, D. Romano, T. Schmickl, and R. Thenius, "Daphnia's phototaxis as an indicator in ecotoxicological studies: A review," *Aquatic Toxicology*, vol. 265, p. 106762, Dec. 2023.

[53] "EU FET project ASSISIbf – Artificial Life Laboratory." [Online]. Available: https://alife.uni-graz.at/projects/assisibf/

[54] F. Bonnet and F. Mondada, "Automated Setup to Conduct Experiments with Mixed Societies of Fish and Robots," in *Shoaling with Fish: Using Miniature Robotic Agents to Close the Interaction Loop with Groups of Zebrafish*

Danio rerio, F. Bonnet and F. Mondada, Eds. Cham: Springer International Publishing, 2019, pp. 67–74.

[55] F. Bonnet, J. Halloy, and F. Mondada, "Follow the dummy: Measuring the influence of a biomimetic robotic fish-lure on the collective decisions of a zebrafish shoal inside a circular corridor," in *2018 IEEE International Conference on Soft Robotics (RoboSoft)*. Livorno: IEEE, Apr. 2018, pp. 504–509.

[56] L. Cazenille, B. Collignon, Y. Chemtob, F. Bonnet, A. Gribovskiy, F. Mondada, N. Bredeche, and J. Halloy, "How mimetic should a robotic fish be to socially integrate into zebrafish groups?" *Bioinspiration & Biomimetics*, vol. 13, no. 2, p. 025001, Jan. 2018.

[57] M. Szopek, R. Thenius, M. Stefanec, D. Hofstadler, J. Varughese, M. Vogrin, G. Radspieler, and T. Schmickl, "Autonome Roboterschwärme als Stabilisatoren gefährdeter Ökosysteme," 2021.

[58] T. Krajník, M. Nitsche, J. Faigl, P. Vaněk, M. Saska, L. Přeučil, T. Duckett, and M. Mejail, "A practical multirobot localization system," *Journal of Intelligent & Robotic Systems*, 2014, publisher: Springer Netherlands. [Online]. Available: http://dx.doi.org/10.1007/s10846-014-0041-x

[59] J. Ulrich, J. Blaha, A. Alsayed, T. Rouček, F. Arvin, and T. Krajník, "Real Time Fiducial Marker Localisation System with Full 6 DOF Pose Estimation," *ACM SIGAPP Applied Computing Review*, vol. 23, no. 1, pp. 20–35, Mar. 2023.

[60] S. Garrido-Jurado, R. Muñoz-Salinas, F. J. Madrid-Cuevas, and M. J. Marín-Jiménez, "Automatic generation and detection of highly reliable fiducial mark-

ers under occlusion," *Pattern Recognition*, vol. 47, no. 6, pp. 2280–2292, Jun. 2014.

[61] E. Olson, "AprilTag: A robust and flexible visual fiducial system," in *2011 IEEE International Conference on Robotics and Automation*, May 2011, pp. 3400–3407.

[62] W. Y. Chen and J. D. Louck, "Necklaces, MSS sequences, and DNA sequences," *Advances in Applied Mathematics*, vol. 18, no. 1, pp. 18–32, 1997.

[63] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," in *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001*, vol. 1, Dec. 2001, pp. I–I.

[64] Y. Freund and R. E. Schapire, "A Decision-Theoretic Generalization of On-Line Learning and an Application to Boosting," *Journal of Computer and System Sciences*, vol. 55, no. 1, pp. 119–139, Aug. 1997.

[65] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, vol. 1, Jun. 2005, pp. 886–893 vol. 1.

[66] N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, and S. Zagoruyko, "End-to-End Object Detection with Transformers," May 2020.

[67] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks," Jan. 2016.

[68] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," Oct. 2014.

[69] R. Girshick, "Fast R-CNN," Sep. 2015.

[70] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel, "Backpropagation Applied to Handwritten Zip Code Recognition," *Neural Computation*, vol. 1, no. 4, pp. 541–551, Dec. 1989.

[71] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "SSD: Single Shot MultiBox Detector," 2016, vol. 9905, pp. 21–37.

[72] K. Simonyan and A. Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition," Apr. 2015.

[73] D. Erhan, C. Szegedy, A. Toshev, and D. Anguelov, "Scalable Object Detection using Deep Neural Networks," Dec. 2013.

[74] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You Only Look Once: Unified, Real-Time Object Detection," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2016, pp. 779–788.

[75] C. Szegedy, Wei Liu, Yangqing Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. Boston, MA, USA: IEEE, Jun. 2015, pp. 1–9.

[76] "A Comprehensive Review of YOLO Architectures in Computer Vision: From YOLOv1 to YOLOv8 and YOLO-NAS."

[77] J. Redmon and A. Farhadi, "YOLO9000: Better, Faster, Stronger," Dec. 2016.

[78] A. M. Ikotun, A. E. Ezugwu, L. Abualigah, B. Abuhaija, and J. Heming, "K-means clustering algorithms: A comprehensive review, variants analysis, and

advances in the era of big data," *Information Sciences*, vol. 622, pp. 178–210, Apr. 2023.

[79] J. Redmon and A. Farhadi, "YOLOv3: An Incremental Improvement," Apr. 2018.

[80] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature Pyramid Networks for Object Detection," Apr. 2017.

[81] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, "YOLOv4: Optimal Speed and Accuracy of Object Detection," Apr. 2020.

[82] S. Liu, L. Qi, H. Qin, J. Shi, and J. Jia, "Path Aggregation Network for Instance Segmentation," Sep. 2018.

[83] K. He, X. Zhang, S. Ren, and J. Sun, "Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition," 2014, vol. 8691, pp. 346–361.

[84] G. Jocher, A. Chaurasia, A. Stoken, J. Borovec, NanoCode012, Y. Kwon, K. Michael, TaoXie, J. Fang, imyhxy, Lorna, Z. Yifu, C. Wong, A. V, D. Montes, Z. Wang, C. Fati, J. Nadar, Laughing, UnglvKitDe, V. Sonck, tkianai, yxNONG, P. Skalski, A. Hogan, D. Nair, M. Strobel, and M. Jain, "ultralytics/yolov5: v7.0 - YOLOv5 SOTA Realtime Instance Segmentation," Nov. 2022.

[85] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer, "Automatic differentiation in PyTorch," in *NIPS-W*, 2017.

[86] C. Li, L. Li, H. Jiang, K. Weng, Y. Geng, L. Li, Z. Ke, Q. Li, M. Cheng, W. Nie, Y. Li, B. Zhang, Y. Liang, L. Zhou, X. Xu, X. Chu, X. Wei, and

X. Wei, "YOLOv6: A Single-Stage Object Detection Framework for Industrial Applications," Sep. 2022.

[87] C. Li, L. Li, Y. Geng, H. Jiang, M. Cheng, B. Zhang, Z. Ke, X. Xu, and X. Chu, "YOLOv6 v3.0: A Full-Scale Reloading," Jan. 2023.

[88] C.-Y. Wang, A. Bochkovskiy, and H.-Y. M. Liao, "YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors," Jul. 2022.

[89] C.-Y. Wang, H.-y. Liao, and I.-H. Yeh, *Designing Network Design Strategies Through Gradient Path Analysis*, Nov. 2022.

[90] R. Varghese and S. M., "YOLOv8: A Novel Object Detection Algorithm with Enhanced Performance and Robustness," in *2024 International Conference on Advances in Data Engineering and Intelligent Computing Systems (ADICS)*, Apr. 2024, pp. 1–6.

[91] C. Lyu, W. Zhang, H. Huang, Y. Zhou, Y. Wang, Y. Liu, S. Zhang, and K. Chen, "RTMDet: An Empirical Study of Designing Real-Time Object Detectors," Dec. 2022.

[92] W. Luo, Y. Li, R. Urtasun, and R. Zemel, "Understanding the Effective Receptive Field in Deep Convolutional Neural Networks," Jan. 2017.

[93] X. Ding, X. Zhang, N. Ma, J. Han, G. Ding, and J. Sun, "RepVGG: Making VGG-style ConvNets Great Again," Mar. 2021.

[94] X. Li, W. Wang, L. Wu, S. Chen, X. Hu, J. Li, J. Tang, and J. Yang, "Generalized Focal Loss: Learning Qualified and Distributed Bounding Boxes for Dense Object Detection," Jun. 2020.

[95] H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz, "mixup: Beyond Empirical Risk Minimization," Apr. 2018.

[96] S. Yun, D. Han, S. J. Oh, S. Chun, J. Choe, and Y. Yoo, "CutMix: Regularization Strategy to Train Strong Classifiers with Localizable Features," Aug. 2019.

[97] Z. Ge, S. Liu, F. Wang, Z. Li, and J. Sun, "YOLOX: Exceeding YOLO Series in 2021," Aug. 2021.

[98] K. Crailsheim, U. Eggenreich, R. Ressi, and M. J. Szolderits, "Temperature preference of honeybee drones (Hymenoptera: Apidae)," *Entomologia Generalis*, pp. 37–47, 1999.

[99] K. Griparić, T. Haus, D. Miklić, M. Polić, and S. Bogdan, "A robotic system for researching social integration in honeybees," *PLOS ONE*, vol. 12, no. 8, p. e0181977, Aug. 2017.

[100] D. E. Minnich, "The photic reactions of the honey-bee, Apis mellifera L." *Journal of Experimental Zoology*, vol. 29, no. 3, pp. 343–425, Nov. 1919.

[101] G. Bradski, "The OpenCV Library," *Dr. Dobb's Journal of Software Tools*, 2000.

[102] "Spatial Filters - Laplacian/Laplacian of Gaussian." [Online]. Available: https://homepages.inf.ed.ac.uk/rbf/HIPR2/log.htm

[103] M. Contributors, "MMYOLO: OpenMMLab YOLO series toolbox and benchmark," https://github.com/open-mmlab/mmyolo, 2022.

[104] K. Chen, J. Wang, J. Pang, Y. Cao, Y. Xiong, X. Li, S. Sun, W. Feng, Z. Liu, J. Xu, Z. Zhang, D. Cheng, C. Zhu, T. Cheng, Q. Zhao, B. Li,

X. Lu, R. Zhu, Y. Wu, J. Dai, J. Wang, J. Shi, W. Ouyang, C. C. Loy, and D. Lin, "MMDetection: Open mmlab detection toolbox and benchmark," *arXiv preprint arXiv:1906.07155*, 2019.

[105] B. C. Russell, A. Torralba, K. P. Murphy, and W. T. Freeman, "LabelMe: A Database and Web-Based Tool for Image Annotation," *International Journal of Computer Vision*, vol. 77, no. 1-3, pp. 157–173, May 2008.

[106] D. Moore, W. Notz, and M. Fligner, *The basic practice of statistics*. W. H. Freeman, 2015.