Invasion of the Chinese Mitten Crabs: development of a low power monitoring sensor

Joost Fonteyne*, Maarten Weyn*, Jonas Schoelynck†, and Michiel Aernouts*
*Faculty of Applied Engineering - Department of Electronics-ICT
University of Antwerp, Antwerp, Belgium
†Department of Biology
University of Antwerp, Antwerp, Belgium

***Abstract*—The Chinese Mitten crab is an invasive species in Flemish rivers and canals, causing ecological and economic problems due to its migratory and burrowing behavior. To address this issue, a monitoring sensor is being developed. We will contribute by building upon a previous study that described a possible implementation. In the previous study, a Raspberry Pi 4B with a camera was utilized alongside a YOLOV3-tiny recognition model, which was validated trough simulations using videos. This paper aims to replicate the previous study's results while minimizing power consumption and evaluating the setup on live crabs. To achieve lower power consumption, our approach involves running a small, low-power object detection model on an ESP32 microcontroller. Additionally, this paper briefly introduces an alternative solution that does not require a microcontroller and employs no machine vision software. Although the accuracy of our final setup may not match that of the previous implementation, our approach significantly reduces power consumption, leading to an increased battery life. Moreover, our setup is less expensive compared to the one in the previous study. We also investigate potential enhancements to improve the accuracy of our setup in future iterations. By minimizing power consumption and exploring alternative solutions, we aim to achieve a balance between accuracy, energy efficiency, and affordability.**

I. Introduction

The Chinese mitten crab is an invasive species, their migratory and burrowing behavior causes ecological and economical damage. The first step in countering them is knowing what routes they take and how may of them there are at a given location. There is currently no simple and easy way of monitoring the movement of these crabs. Today they need to be manually caught using traps or catching nets [1]. Then they need to be manually counted. This is not ideal and the purpose of this paper is to automate this process.

Some work on this topic has already been done and a theoretical solution has already been proposed [2]. However, this solution fails to address the fact that some rivers can be remote and are therefore not as easy to service for example every day. That is why this research will use the same methodology but try to alter the design so that it uses less power. The ultimate goal of this paper is to make a system that can achieve acceptable results but can be left to work on its own for as long as possible. We will try to accomplish this goal by lowering the overall power consumption of the system. While working towards this goal we did not forget the original requirements. The requirements displayed in Table I are taken from the paper by Ruben Joosen and Satish Singh [2]. We endeavor to maintain all requirements in mind but the most important ones at this stage are: cost, mobility, multi-deployability and power consumption. We also add good accuracy to this list. Although it was not mentioned in the original table of requirements it is fair to assume that it was implied.

TABLE I
Original requirements for sensor by Ruben Joosen and Satish Singh [2]

Nr.	Aspect	Description
1	Multi-deployability	It must be possible to install the device in a myriad of rivers in Flanders.
2	Cost	The cost of the device must be kept to a minimum with an absolute maximum of €300
3	Power consumption	The power consumption should be as small as possible.
4	Mobility	The device must be easy to install and remove.
5	Operation time	The device must be operative 24/7. With more data, relationships between migration patterns and day schemes or weather conditions can be investigated.
6	Connection	The device must have a transmitter to send data to a back-end server.
7	Turbidity	The captured footage must be usable even in high turbid environments.
8	Visible area	The width of the visible area must be at least 50 cm.
9	User-friendly	The device must have a self-explanatory user interface and must be easy to adjust and maintain.
10	Power supply	The device must have a rechargeable power supply e.g. a battery with water a turbine.

We will introduce a system based on the approach of the previous paper. Our system will limit power consumption while trying to maintain good accuracy. We do this by cleverly selecting a platform, model and tracker. We also propose a different method of frame capturing that further reduces power consumption. Our setup will be tested in a real setup with real crabs. In Section II we discuss the results of the previous

Master's thesis to obtain the diploma Master of Science in: Electronics and ICT Engineering Technology. Academic year 2022-2023.

paper on this topic. A method that is in use today will also be briefly discussed in this section. In section III we introduce a possible alternative solution, building on the already existing infrastructure. In section IV we enumerate the hardware and software of a new system we designed with the important requirements in mind. Additionally, we justify our choices. Section V contains a brief overview of the full system setup we used in our final experiments. In section VI we evaluate our system on accuracy and power consumption. Section VII discusses future work, which, in our opinion, has the potential to improve the system. The conclusion can be found in section VIII.

II. Related work

A. Theoretical solution

A paper by Ruben Joosen and Satish Singh proposed a theoretical setup for monitoring the Chinese mitten crab [2]. They utilized a machine vision setup, employing a trained yoloV3-tiny object detection model on a Raspberry Pi 4B. For enhanced visibility in low-light conditions, they utilized a fish-eye camera with IR lighting. To track and count the detected crabs, they employed a minimum output sum of squared error (MOSSE) tracker. Their trained model achieved a 100% recognition rate on images with a single crab and an 80% recognition rate on images with multiple crabs. Additionally, they presented a physical setup to mitigate external factors. However, apart from their tracker, no power profiling was conducted. They did mention that their Raspberry Pi consumed 3.4 watts when idle. It is important to note that all their experiments were conducted in a software environment, and their results have not yet been verified in real-world scenarios.

B. Current solution

The traps currently in use for capturing the Chinese mitten crab rely on the crab's ability to climb vertically on a piece of mesh and navigate both water and air. As the crabs walk along the river floor, they cross a concrete ramp and inadvertently fall into a narrow opening. The only means of escape from the opening is by following a tube that extends from the water and leads to a collection box [3]. This trapping method has been successful over the years, with over one million crabs caught in just two years, while only capturing two toads [1]. Although not every crab is captured, the system has demonstrated its effectiveness.

III. Alternative solution

The existing traps can be easily modified to automate the counting of the crabs. This can be achieved by incorporating a light source on one side and a light sensor on the other side of the crab's trajectory as it falls into the collection box. When the sensor detects that the light has been interrupted, it can send a signal to a low-power microcontroller, which keeps track of the crab count. In an automated counting system, the collection box can be eliminated, eliminating the need for manual emptying. Instead, a LoRaWAN or similar connection can be installed to periodically transmit the crab count to a centralized dashboard, consolidating counts from all traps. Such a system has the potential to operate autonomously for extended periods without human intervention. This approach shows promise.

However, implementing such a trap modification is neither inexpensive nor easy. During installation, it is often necessary to temporarily lower the water level to facilitate the process, which may not be feasible in all locations. Additionally, the weight and size of the required components can present challenges in certain cases [1]. Once installed, relocating the trap becomes a cumbersome task. Consequently, this solution falls short in terms of cost-effectiveness and mobility requirements.

IV. Approach

A more portable and cheaper solution is the use of a camera and a platform running a machine learning algorithm much like the paper by Ruben Joosen and Satish Singh [2]. We will reevaluate the hardware and software, focusing on the low power constraint while also trying to provide the best possible accuracy.

A. Hardware

1) Power Measurement: All power measurements in this paper were conducted using the Power Profiler Kit II (PPK2) from Nordic Semiconductors.

2) Platform: While the previous setup utilized a Raspberry Pi 4B, we considered alternative options to optimize power consumption. After weighing the trade-off between consumption and performance, we decided to employ the ESP32-CAM kit. This kit includes an ESP32 microcontroller, an SD card reader, and an OV2640 camera. The ESP32 is a low-power microcontroller (MCU). Its suitability for machine learning (ML) applications is well-documented by various examples available on the internet, including support from the Edge Impulse platform. With a current draw of only 39 mA when running a simple blink program, the ESP32 surpasses the Raspberry Pi 4B, which was reported to have an idle consumption of 3.4 watts or 680 mA in the original paper [2]. Additionally, the ESP32 has 520 KB of RAM, providing sufficient capacity for a model that can achieve acceptable accuracy. Another notable feature is the ESP32's capability to enter a deep sleep state, drawing an average of only 6.5 mA. Although we do not utilize the wireless connectivity capabilities of the ESP32 in our current setup, they may prove beneficial for future iterations. The inclusion of an SD card reader simplifies the data collection process, facilitating the gathering of training data for our models, which will ultimately be deployed on the target platform.

3) Vision: One drawback of the ESP32-CAM kit is its lack of night vision capabilities with the provided camera. However, this limitation can be overcome by manually removing the camera's infrared (IR) filter and incorporating additional infrared LEDs into the project. We selected 850 nm wavelength LEDs over 940 nm LEDs, as IR cameras exhibit higher sensitivity to 850 nm, resulting in higher-quality images [4]. In our final setup we used eight IR LEDs placed in a square formation around the camera.

4) Movement Detection: To further reduce power consumption, we explored the inclusion of a motion sensor in the system. The idea was to trigger the capture of an image and the execution of the recognition model only when movement is detected in front of the camera. Considering that the crabs are cold-blooded, we opted for a microwave sensor instead of a passive infrared sensor (PIR). We utilized the Gravity Microwave Sensor V2 from DFrobot, which exhibited an average power consumption of 47 mA after removing unnecessary LEDs. However, subsequent tests revealed that this sensor did not effectively detect movement at the depths of the river where the crabs typically reside. As a result, we decided not to incorporate the microwave sensor in the final system.

B. Software

1) Model: To train our models, we utilized Edge Impulse, an online platform that facilitates data labeling and quick training of small models for edge deployment. Our model of choice was Faster Object More Objects (FOMO) by Edge Impulse. This model has demonstrated excellent performance on small microcontrollers (MCUs) and can achieve up to 30 times less processing power and memory usage compared to other similar dedicated algorithms [5]. FOMO is a customized version of MobileNetV2, which incorporates depth-wise separable convolution, a computationally efficient alternative to standard convolution [6] [7] [8]. The model employs the rectified linear unit six (ReLU6) as its activation function due to its robustness in low-precision computation [9]. Additionally, MobileNetV2 incorporates bottleneck layers to limit the number of channels in the network [7].

FOMO performs efficient object detection by functioning like a classifier while excluding certain convolution layers. Instead of generating a single-class classification for the entire image, it produces a per-region class probability map. FOMO employs a custom loss function that preserves locality in the final layer, resulting in a heat map indicating the object locations [10]. The selection of layer cutoff points is crucial in determining the resolution of the object map. In scenarios with numerous small objects in close proximity, careful consideration of this resolution is essential. However, given the size of the crabs in our case, this parameter is less critical, and we can maintain the standard setting, which reduces the resolution by eight to one compared to the original. By solely detecting centroids, FOMO simplifies object counting, as each activation on the heat map can be considered as an object [10].

C. Tracking and Counting

For tracking, we utilized a basic centroid tracking algorithm. This algorithm tracks objects across frames by calculating the Euclidean distance between old and new objects. It then assigns each old object to a new object, minimizing the total cost while ensuring that no object is used twice. This approach provides a straightforward implementation for tracking and serves as the foundation for more advanced algorithms. Since our model already outputs centroids, it was easy to integrate this tracking algorithm. We conducted tests on our setup, which achieved an FPS of around 1. From the results, we found that it works sufficiently well for determining whether a crab needs to be counted, as long as a precise mapping of the object's trajectory is not required. Considering our output image dimensions of 96 by 96, we draw an imaginary line at x = 43. Any old crab that cannot be assigned to a new detection by our tracker is counted if its x-coordinate exceeds this imaginary line.

1) Branch and Bound: To address our cost minimization problem, we need an algorithm commonly referred to as the "job assignment problem". The Hungarian algorithm is the established solution for this problem; however, implementing it on our esp32 microcontroller is impractical due to its complexity and size. [11] An alternative approach to ensure accurate problem-solving is brute-forcing, which involves calculating every possible combination. However, this method becomes computationally intense rapidly, with a time complexity of O(N!) [12]. Therefore, we have selected the "branch and bound" algorithm, which offers a more efficient variation of brute-forcing. With a time complexity of O(N*M) [12], where N represents the number of old crabs and M represents the number of new crabs, this algorithm provides a more reasonable solution.

The branch and bound algorithm starts from a root node and calculates the cost for all accessible nodes beneath the current node. It then moves to the node with the lowest cost and repeats these steps until there are no deeper layers to explore. The final layer represents the optimal distribution. The number of layers corresponds to the number of old crab detections in our problem. The cost associated with each node comprises the values of the previously assigned new crabs combined with the lowest combination of values from lower layers, given that the current new job is assigned to the current old crab. Each node represents a specific combination of an old crab and a new crab detection. Notably, a new crab detection that has already been assigned cannot be assigned to another old crab detection [13] [12]. The branch and bound approach will always arrive at the most optimal solution [14].

V. System description

Our system consists of an ESP32 connected to a camera and a transistor. The transistor's function is to enable the ESP32 to control the IR light, turning it on or off. The IR light is supplied by eight 850 nm LEDs arranged in a square formation around the camera lens. Ideally, these LEDs should draw a total of 640 mA when turned on. However, due to hidden resistances in the circuit, the actual value is slightly lower. The entire setup is housed in a transparent glass container, with the camera lens facing downwards. This container is positioned to break the waterline of a river or test setup with its bottom.

The ESP32 operates in a cycle where every 2 seconds, the LEDs are turned on, and a new picture is taken. Immediately after capturing the picture, the LEDs are turned off. It takes around 100 ms before the LEDs are fully on. The system then performs an inference on the image before entering a deep sleep state. The entire process of turning on the

LEDs, capturing the picture, and running the inference takes approximately one second. The remaining time is spent in deep sleep and the subsequent wake-up, which takes around 0.7 seconds.

If a crab is detected in a picture, this cycle is interrupted. After detecting a crab, the system promptly captures a new picture. If the second picture also contains one or more crabs, our tracking algorithm matches the locations of the old crabs with those of the new crabs in the picture. This matching is achieved by minimizing the total Euclidean distance between all the crabs using the branch and bound algorithm.

If our tracking algorithm observes that there are more old crabs than new ones, it determines which old crabs have disappeared. Crabs that do not appear in the optimal solution to the minimization problem are considered to have left the system's field of vision (FOV). If the last known location of a disappeared crab is beyond the x = 43 line, we consider it to have passed our threshold, and it is counted. The total number of counted crabs is stored in the ESP32's electrically erasable programmable read-only memory (EEPROM) to persist through deep sleep and reboots.

Once an image with no crabs is captured, the cycle ends. All the crabs from the previous image are evaluated against the x = 43 line to determine if they are eligible for counting. The system then returns to the starting cycle, where a picture is taken every 2 seconds. The inferencing is carried out by a FOMO MobileNetV2 0.35 model that was trained on 300 images taken with the setup. We used the Edge Impulse portal to label our images and train our model. Training images were converted to greyscale and resized to a square dimension of 96x96 pixels. We used squashing to resize the images, ensuring that crabs on the edges of the longer axis were not lost.

VI. System Evaluation

A. IR Vision

Good lighting is crucial for the success of our object detection system. Since the system needs to operate 24/7, including in dark river water and during nighttime, we use IR LEDs to provide independent illumination regardless of external conditions. However, there is a trade-off between lighting quality and power consumption, as increased light intensity requires more power. To address this, we conducted experiments to find an optimal lighting setup that offers sufficient illumination while minimizing current draw, with the goal of achieving clear pictures.

1) Four LEDs at 250 mA Total: We initially evaluated a lighting setup with four LEDs arranged in a square formation. However, the resulting images (Figure 1) had limited visibility, with only a small area in the middle being clearly visible. This was primarily due to insufficient light output from the LEDs and their focused nature, which created bright spots that hindered vision in the surrounding area.

2) Eight LEDs at 620 mA Total: In the new setup, we used eight LEDs placed in pairs, forming a square formation around the camera. All LEDs were connected in parallel and drew current through their own set of two 22-ohm resistors, totaling 44 ohms. Each LED operated at 1.5 V, and with a source voltage of 5 V, we expected a current draw of approximately 636.363... mA. However, the actual measured current was 620 mA.

Fig. 1. Visibility with four LEDs at 250 mA

Despite the increased light, we observed that the issue of the light being too focused persisted. To address this, we diffused the light of the LEDs. We found that diffusing the light resulted in clearer pictures with a larger illuminated area. A comparison between a picture taken with two sanded-off LEDs (left) and a picture taken with the same LEDs before sanding (right) is shown in Figure 2.

Fig. 2. Comparison of two LEDs: left - sanded off, right - normal

Since the crabs reside in bodies of water, it was important to ensure that our lighting setup also worked for submerged objects. We conducted a test by placing the camera and LEDs above the water, looking down at an object submerged in approximately 30 cm of water. In this test, we found that both a diffused light source and a non-diffused source produced very poor visibility. The non-diffused source suffered from the light being too focused, while the diffused source caused the image to be too hazy. In this experiment, we used an opaque container for light diffusion, and the reflection of light on the water surface reduced visibility in both setups. Figure 3 shows a picture taken with focused LED light (left) and diffused LED light (right).

To address these visibility issues, we placed the camera and LEDs at the bottom of a see-through container, ensuring that the container's bottom breaks the water surface. This solution improved visibility for both diffused and non-diffused LEDs. In this test, the LEDs were diffused using an opaque container. The setup with focused LEDs in a clear container resulted in decent visibility, with about half of the image being clearly visible and only some faint bright spots. The pictures taken with the LEDs and camera in an opaque container were almost completely visible. Figure 4 displays the setup with

Fig. 3. Comparison of LED lighting setups above waterline: left - clear container, right - opaque container

a clear container (left) and an opaque container (right), with the distance from the camera to the container's bottom being approximately 25 cm.

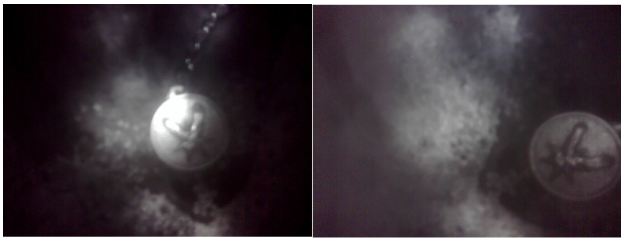

Fig. 4. Comparison of LED lighting setups breaking the waterline: left - clear container, right - opaque container

Placing the system slightly below the water level and breaking the waterline also proved beneficial in avoiding obstacles such as floating objects or masses that could obstruct vision. To further reduce haziness in the pictures, we used the eight diffused 850 nm IR LEDs in a glass container breaking the waterline. Figure 5 provides a comparison between a picture taken with focused LEDs in an opaque container (left) and a picture taken with the final configuration (right).

Fig. 5. Comparison of LED lighting setups breaking the waterline: left - opaque container with focused LEDs, right - glass container with diffused LEDs

B. Model accuracy

To evaluate the accuracy of our system we made a test setup with live crabs. We took an aquarium with 12 crabs and filled it with 40 cm of water. Half of the aquarium was filled with river gravel, the other half was not. We used a tarp to simulate different external lightning situations. We also used various random objects to test our model. We initially trained two final models. One model trained on images with different external lightning, different backgrounds and different image haziness. The other model was trained with images that had the same external lightning, background and haziness. After training the models we tested them on around 60 new images in the Edge impulse portal. Both models used 239 Kb of ram which fits comfortably in our esp32's 520 Kb while leaving space for other operations. Both models had a predicted inference time of 1.3 seconds. But later tests showed that the actual inference time on the esp32 was only 750 ms. Since our tracking and counting can make a counting mistake even if the detection stage only misses 1 crab in an image containing for example 10 crabs in total, we have to be stern with our accuracy. So in determining the overall accuracy of a model we do not consider an image in which 4 out of 5 crabs have been correctly detected as an accuracy of 80%. To determine our accuracy an image is either completely correct of completely wrong, there is no in between. With this rule we get an accuracy of 62% for the first model that was trained on images with mixed external factors. The second model has an accuracy of 72.6%. These results are not yet satisfactory. Even with the second model this would mean that every detection of our system there is a 27.4% Chance that we make a mistake on each image. Since crabs are only counted past the x = 43 line that gives us a Chance of 13.7% that one or more crabs get counted by mistake every new picture. For the final setup we chose the second model because of its higher accuracy. We do believe that it is fair to expect constant external factors. The paper by Ruben Joosen and Satish Singh already proposed the use of a square container around the setup to protect the setup from external factors [2]. Other river detection systems that have been proven to work use the same approach to shield from external influences and increase accuracy [15]. We believe the main reasons behind our low accuracy's to be insufficient lightning leading to pictures with badly visible crabs and the fact our current setup allows crabs to appear only partially in the FOV. In both models partial crabs were also labelled as crabs.

C. Power consumption

We will compare two modes. In the first mode the system stays continuously on and takes a picture every 960 ms. In the second mode our system weaves in 300 ms of deep sleep between cycles when no crabs are detected. Going into deep sleep means our system has to wake-up as well. This takes 730 ms. In the second mode we switch to mode one when a crab is detected. This is to get a higher FPS for our tracker at the cost of a higher current draw. When no crab is detected our system reverts back to mode two.

1) Mode one: In this mode, the system captures a picture every 960 ms. We can identify three distinct phases in our power profiler during this mode. The first phase involves turning on the LEDs, which takes 100 ms and draws a current of 630 mA. In the second phase, a picture is taken, averaging 110 ms with a current draw of 660 mA. It is important to note that the high current draw during this stage is not solely due to capturing the picture but is also attributed to the LEDs remaining on. After this step, the LEDs turn off, resulting in

a significant drop in current draw. In the third and final phase, the system performs inference on the captured picture, taking 750 ms with an average current draw of 120 mA. The system then begins a new cycle. On average, each cycle has a current draw of 234 mA, which serves as our baseline for comparing the improvements in mode two.

2) Operation Mode Two: The second mode is the polling mode, where a picture is taken approximately every two seconds. From a power consumption perspective, we can divide each cycle in this mode into five parts. The first part is deep sleep, during which the system remains in deep sleep for 300 ms with a current draw of 6.5 mA. The second part is when the system wakes up, taking 730 ms with an average current draw of 75.8 mA. The other three phases are the same as in mode one. In this mode, the average current draw per cycle depends on the number of times a crab is detected. To provide an idea of how this affects power consumption, we conducted measurements by gradually increasing the number of consecutive crab detections in each cycle. This increased the cycle length and the time the system spent in mode one. It's worth noting that, in this case, we define a cycle as the period between two deep sleep occurrences. We stopped at ten consecutive crab detections, as a clear pattern had emerged at that point. The results of these measurements can be found in Table II. The graph in Figure 6 demonstrates a clear trend. The use of mode two is always advantageous because even with infinite consecutive crab detections, our power consumption will converge to that of mode one. The graph also illustrates that mode two yields the most significant power savings when no crabs are present. Although it may not be as substantial an improvement as the initially planned movement sensor, it still reduces power consumption by almost half when no crabs are detected. To put this into perspective, a system running only mode one on a large 10000 mAh battery capable of full discharge would drain that battery in 41 hours. In mode two, the battery would last anywhere between 41 and 71 hours. It's important to note that these calculations do not account for any power harvesting or data transmission. For reference, the setup described in the paper by Ruben Joosen and Satish Singh, when idling with the camera and its IR spots turned off, would drain the same battery in just 14 hours.

TABLE II
RESULTS OF CURRENT MEASUREMENTS FOR POLLING MODE (MODE 2)

Number of consecutive crabs detected	**current draw in mA**
0	139.5
1	170.84
2	186.91
3	197.08
4	203.47
5	208.02
6	209.46
7	212.4
8	214.9
9	215.28

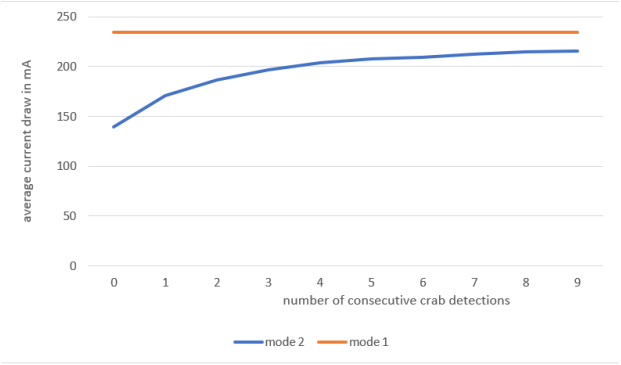

Fig. 6. Graph comparing average current draw of continuous mode (mode 1) to polling mode (mode 2)

VII. FUTURE WORK

While our study has primarily focused on power consumption, there are still outstanding requirements that require further investigation. In future research, it is important to address these remaining requirements comprehensively. Although the experiments conducted in this paper were performed on live crabs within controlled test environments, additional tests in real river environments should be conducted. Understanding the system's performance in natural river settings will provide valuable insights and help validate the effectiveness of the proposed approach.

Furthermore, we are of the opinion that exploring the effects of a container that shields the system from external factors is a worthwhile endeavor. Such a container has the potential to enhance system accuracy by mitigating the impact of environmental variables. Future experiments should be conducted to assess the effectiveness of implementing a protective container and quantify its impact on the system's overall performance.

Originally we aimed to implement a movement sensor into the system to save power. Unfortunately the sensor we selected could not penetrate water deep enough to pick up on the movement of the crabs. We maintain the belief that the deployment of such a sensor can serve as a potent tool in further mitigating power consumption within the system. Consequently, we advocate for the pursuit of experiments that investigate the use of a sensor capable of attaining the required water penetration depths.

VIII. CONCLUSION

The proliferation of the Chinese mitten crab in Flemish waterways presents significant challenges, as there is currently a lack of easy, inexpensive, reliable, and low-labor-intensive methods to map their presence. In an effort to address this need, we developed a machine vision-based counting sensor inspired by a previous theoretical solution. Our aim was to extend the system's battery life while maintaining accuracy.

The results of our experiments indicate that the accuracy of our system is currently unsatisfactory and inferior to the original approach. However, these experiments have provided

valuable insights that can guide future improvements to enhance accuracy. Despite this limitation, we were successful in significantly reducing the power consumption of the system compared to the original setup. Furthermore, our solution boasts a lower cost, with an approximate price of €15, in contrast to the previous setup's cost of approximately €100. This reduction in cost by €85 offers a notable advantage.

In conclusion, while our current system's accuracy falls short of expectations, our research highlights the potential for further advancements in accurately detecting and monitoring the presence of Chinese mitten crabs. By leveraging the lessons learned from our experiments, future work can focus on refining the system's accuracy while maintaining its power efficiency. Such improvements will contribute to the development of a practical and efficient solution to combat the challenges posed by the spread of the Chinese mitten crab in Flemish waterways.

REFERENCES

[1] H. R. J. S. K. H. Schoelynck J, Van Loon P, "Design and testing of a trap removing chinese mitten crabs (eriocheir sinensis, h. milne edwards, 1853) from invaded river systems." *River Res Applic.*, 1–11. 2020.

[2] M. W. J. S. Ruben Joosen, Satish Singh, "Determination of migration patterns of crabs in a high turbid submerged environment using object detection and tracking."

[3] B. D'hondt, P. Van Loon, H. Keirsebelik, D. Slootmaekers, P. Boets, K. Van Roeyen, H. Verreycken, and J. Schoelynck, "The use of "postbox traps" in the control of chinese mitten crab in flanders (belgium)," Apr. 2022.

[4] "White paper: Infrared illumination for time-of-flight applications," Lumileds Holding B.V., Tech. Rep.

[5] C. Nicolas, B. Naila, and R.-C. Amar, "Tinyml smart sensor for energy saving in internet of things precision agriculture platform," in *2022 Thirteenth International Conference on Ubiquitous and Future Networks (ICUFN)*, 2022, pp. 256–259.

[6] Z. M. Chng, "Using depthwise separable convolutions in tensorflow," Deep Learning for Computer Vision, 2022. [Online]. Available: https://machinelearningmastery.com/using-depthwise-separable-convolutions-in-tensorflow/

[7] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "Mobilenetv2: Inverted residuals and linear bottlenecks," 2019.

[8] J. Aira, T. Olivares, F. M. Delicado, and D. Vezzani, "Mosquiot: A system based on iot and machine learning for the monitoring of aedes aegypti (diptera: Culicidae)," *IEEE Transactions on Instrumentation and Measurement*, vol. 72, pp. 1–13, 2023.

[9] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "Mobilenets: Efficient convolutional neural networks for mobile vision applications," 2017.

[10] L. Moreau and M. Kelcey, "Announcing fomo (faster objects, moreobjects)," Edge Impulse, 2022. [Online]. Available: https://www.edgeimpulse.com/blog/announcing-fomo-faster-objects-more-objects

[11] H. W. Kuhn, "The hungarian method for the assignment problem," *Naval Research Logistics Quarterly*, vol. 2, no. 1-2, pp. 83–97, 1955. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/nav.3800020109

[12] Job assignment problem using branch and bound. [Online]. Available: https://www.geeksforgeeks.org/job-assignment-problem-using-branch-and-bound/

[13] E. L. Lawler and D. E. Wood, "Branch-and-Bound Methods: A Survey," *Operations Research*, vol. 14, no. 4, pp. 699–719, August 1966. [Online]. Available: https://ideas.repec.org/a/inm/oropre/v14y1966i4p699-719.html

[14] D. R. Morrison, S. H. Jacobson, J. J. Sauppe, and E. C. Sewell, "Branch-and-bound algorithms: A survey of recent advances in searching, branching, and pruning," *Discrete Optimization*, vol. 19, pp. 79–102, 2016. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1572528616000062

[15] G. Boussarie, N. Teichert, R. Lagarde, and D. Ponton, "Bichicam, an underwater automated video tracking system for the study of migratory dynamics of benthic diadromous species in streams," *River Research and Applications*, vol. 32, no. 6, pp. 1392–1401, 2016. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/rra.2984