A Real-Time Edge Computing System for Monitoring Bees at Flowers

Chapter · August 2024

DOI: 10.1007/978-3-031-67447-1_27

9 authors, including:

Gabriel Santiago Plaza
University of Puerto Rico at Río Piedras
2 PUBLICATIONS **3** CITATIONS
SEE PROFILE

Luke Meyers
University of Puerto Rico at Río Piedras
3 PUBLICATIONS **3** CITATIONS
SEE PROFILE

Noel Fanfan
University of Puerto Rico at Río Piedras
2 PUBLICATIONS **3** CITATIONS
SEE PROFILE

Jose Luis Agosto
University of Puerto Rico at Río Piedras
57 PUBLICATIONS **4,186** CITATIONS
SEE PROFILE

A Real-Time Edge Computing System for Monitoring Bees at Flowers

Josué A. Rodríguez-Cordero[1], Gabriel A. Santiago-Plaza[1], Luke Meyers[2], Fanfan Noel[1], Eduardo J. Figueroa-Santiago[1], Rémi Mégret[1(✉)], Carlos Corrada Bravo[1], José L. Agosto-Rivera[1], and Tugrul Giray[1]

[1] University of Puerto Rico, Rio Piedras, San Juan, PR 00925, USA
{josue.rodriguez10,remi.megret}@upr.edu
[2] Seattle University, Seattle, WA 98122, USA

Abstract. Honey bees are vital to global agriculture, playing a key role in pollinating crops and supporting our food supply. Understanding their foraging behaviors has traditionally involved labor-intensive field experiments, which often come with the risk of human error, especially when tracking and identifying individual bees within large groups. This paper introduces a system that uses artificial intelligence (AI) models, along with an NVIDIA Jetson Xavier for edge computing, to enhance real-time detection, tracking, and data processing of honey bee flower visits in the field. Our system combines AI detection and a specially designed video processing pipeline, offering a practical solution with a user-friendly interface for field biologists. This approach not only makes field experiments more efficient and feasible, but also enables precise, real-time video data processing, crucial for making on-the-spot decisions during experiments. This article delves into the methodology, performance, and potential future work of the system, showcasing how this combination of technology and biology opens new possibilities for conducting accurate and high-throughput field experiments, ultimately improving our understanding and management of honey bee populations.

Keywords: edge computing · embedded AI · artificial intelligence (AI) · computer vision · real-time · honey bees · pollination · foraging behavior · experimental biology

1 Introduction

Honey bees play an essential role in agriculture, being responsible for the pollination of crops that amount to 12 billion dollars in the United States each year, and are crucial for global food security [1]. Given the importance of their role, understanding their foraging behavior is essential for optimizing the health and well-being of their colonies [2]. This, in turn, can lead to improved agricultural productivity.

One of the traditional methods used by biologists to understand foraging behaviors and task specialization of individuals in bee colonies involves performing artificial flower patch experiments [3]. These experiments require the identification and tracking of bees at the individual level, which is typically achieved

M. Kadoch et al. (Eds.): ISICN 2024, LNNS 1094, pp. 365–375, 2024.
https://doi.org/10.1007/978-3-031-67447-1_27

through marking techniques involving gluing unique identifiable tags to the bees, or painting the bees. However, tracking a large number of bees simultaneously, which is essential to achieve statistically relevant sample sizes, is a challenging endeavor and has the potential for human error.

Advancements in artificial intelligence (AI) and computer vision present an opportunity to streamline and improve the accuracy of this detection and tracking process. However, AI image/video detection models require robust hardware for fast inference, often resulting in the need for cumbersome equipment and substantial energy requirements, which are not well suited for experiments in the field [4].

Addressing this challenge, we introduce an integrated system designed for edge computation for studying honey bee foraging behaviors, specifically employing the NVIDIA Jetson Xavier. This system not only makes field experiments more feasible but also allows for in-field calibration and real-time video data processing, which is indispensable for easy deployment in the field and for on-the-spot experimental decisions such as capturing an individual with a peculiar behavior or phenotype for further analysis.

The significance of this technology lies in its ability to perform accurate and high-throughput field experiments, ultimately improving our understanding and management of honey bee populations. This article discusses the challenges with traditional approaches, highlights the opportunities presented by AI and computer vision, and introduces an integrated system designed for edge computation, offering a practical solution for studying honey bee behaviors in real-world conditions.

2 Related Work

The synergy between computational methods and biological field experiments has spurred a variety of research efforts with diverse approaches to understanding and monitoring insect pollinators. In the context of automated video monitoring, several studies provide insights and foundational strategies relevant to our research.

Pegoraro et al. published a review article of strategies involving automated video monitoring systems for insect pollinators in field conditions [5]. Their review includes systems that leverage motion detection to trigger video recording of pollinator visits, relying on post-experiment analysis. Although certain systems under their review employ machine learning algorithms, they discuss no implementations of real-time detection and analysis capabilities.

Kane et al. [6] introduced DeepLabCut-Live!, which was developed for the estimation of low-latency real-time pose of animals. The system achieves approximately 71fps on a video of 917×698 resolution with a MobileNetV2-0.35 model, using powerful hardware (Titan RTX). Notably, when transposed to an edge computing environment (NVIDIA Jetson Xavier), the performance dropped to 27fps. Additionally, DeepLabCut-Live! is not designed for multi-animal real-time tracking.

Droissart et al. [7] developed PICT (plant-insect interactions camera trap), a camera trap system based on a Raspberry Pi Zero, designed to record animal activity when motion is detected. The system is inexpensive and weatherproof, adequate for recording video in the field, but videos must be analyzed via post-processing.

Delisle et al. [8] published a systematic review on 'next-generation camera trapping'. The efforts they describe are geared towards expanding knowledge on wildlife ecology and conservation via acquisition of data at large spatio-temporal scales. The authors emphasize the potential of machine learning models to "break the bottleneck" stemming from analyzing the vast quantities of images and videos acquired by camera trapping systems.

Bjerge et al. [9] built a system prioritizing real-time insect monitoring, using computer vision and deep learning. Deployed on an NVIDIA Jetson Nano, the system demonstrated on-site real-time insect species classification (utilizing the YOLOv3 architecture) at 0.33 frames per second on 1080p video. While insect species classification was performed on site, filtering and tracking of insects was performed remotely.

Rodriguez et al. [10] used video recordings from the entrance of honey bee colonies to monitor foraging activity. Deep Learning models running in High Performance Computing facility were used to analyze the data at scale in order to estimate entrance, exit, and pollen intake.

Ratnayake et al. [11] analyzed pollination behavior in wildflower clusters by combining background substraction and deep-learning, with the potential for future low-power implementation on a low-power device.

Building upon previous research, the current study aims to leverage real-time detection, enhance practical field usability, and provide an intuitive interface. This approach capitalizes on the strengths of previous systems while addressing their limitations.

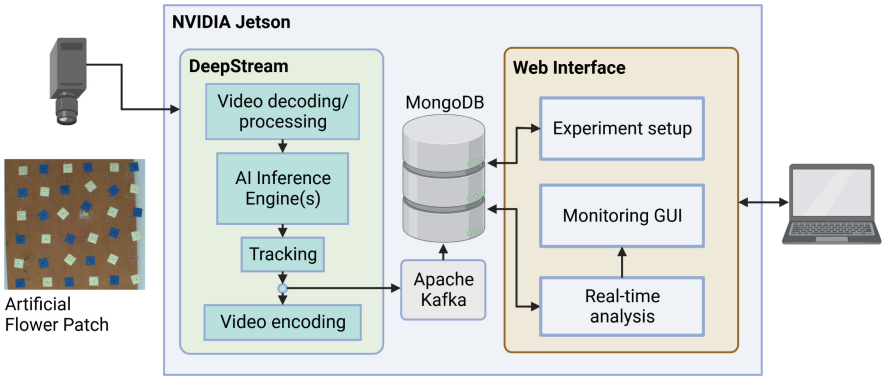

Fig. 1. Overview of the system architecture.

3 Methodology

3.1 System Components

Our system incorporates several hardware and software components (Fig. 1). A Basler USB3 camera (a2A2840-48ucPRO) is used to capture high-quality video streams of honey bees in the artificial flower patch. This video feed is then processed by the NVIDIA Jetson Xavier. A web application interface can be accessed directly on the Jetson or on an auxiliary computer. Figure 2 shows the system deployed at the experimental agricultural station in Gurabo, Puerto Rico.

3.2 Video Processing Pipeline

The central part of the system is the video processing pipeline. This pipeline is built atop the NVIDIA DeepStream SDK, which is highly optimized to take advantage of NVIDIA hardware, performing tasks that include video decoding and encoding, AI inference, and real-time multi-object tracking. Each frame is processed through the pipeline, where bee detection takes place through a YOLOv5n model as the primary GPU inference engine, optimized for inference with NVIDIA TensorRT. The system captures essential metadata for each bee detection, including frame number, timestamp, tracking ID, and bounding box coordinates that encapsulate the detected bee. The tracking ID organizes the detections into tracks, starting when an insect enters the field of view, until it is no longer visible. This metadata is streamed via Apache Kafka to be stored in a MongoDB database for storage and retrieval. Additionally, for every detected and tracked bee, a JPEG image is generated and stored to disk, to be used for

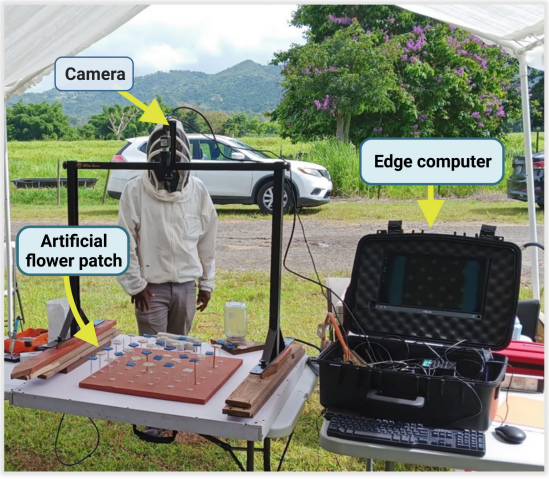

Fig. 2. System deployed on the field.

individual reidentification. This visual snapshot helps researchers by providing a visual reference of each visiting bee, which can be rendered in the experiment web interface.

3.3 Web Application

To ensure that the system is not only robust but also user-friendly, it is managed through an interactive web application. This application, constructed using the Flask micro web framework, is depicted in Fig. 3. The choice of this framework was guided by its lightweight nature, which makes it particularly suitable for real-time edge computing. The web application serves multiple purposes: experiment configuration, flower patch calibration, and real-time monitoring of bee flower visits.

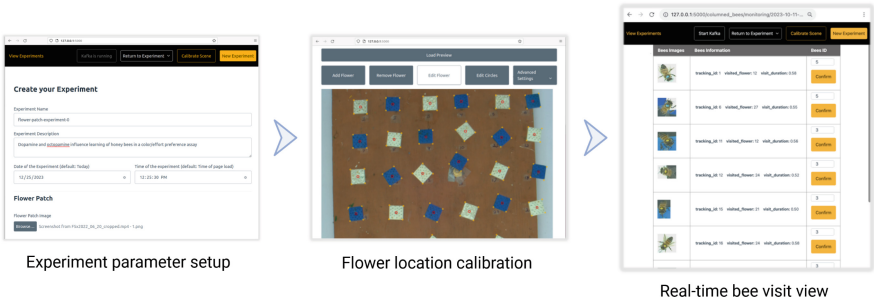

Fig. 3. Web application user interface - main screens.

An additional advantage lies in the accessibility of the application. While it can be directly accessed on the Jetson, its design also supports remote access from auxiliary devices such as laptops. This feature is especially beneficial during field experiments, allowing multiple researchers to simultaneously monitor the experiment and aiding in data logging and dynamic decision making.

3.4 Bee Visit Detection

Developing an automated approach to discern visits from bees is crucial to make the bee monitoring process feasible for a large number of individuals. A bee visit is quantitatively defined as a tracked bounding box that persists beyond a user-determined temporal threshold within the center of an artificial flower, where the experimenters deposit artificial nectar. The visit detection algorithm involves several stages:

– **Experiment Configuration**: A graphical user interface was developed to allow researchers to set up the experiment parameters. The configuration includes the experiment name, the number of flowers within the artificial

patch, the spatial coordinates of each flower (and its central point), and the time threshold before confirming a bee visit. The interface provides automatic detection of the square flowers using OpenCV, as well as interactive editing from an image snapshot in a graphical canvas built using Fabric.js.

– **DeepStream Pipeline Integration**: After configuration, the interface lets the user signal the initiation of the DeepStream pipeline, activating real-time experiment monitoring.
– **Real-Time Polling and Analysis**: While the experiment is active, bee detections are processed to continuously monitor the state of all visible bees and detect visit events, which are then stored back into the MongoDB database. The web interface retrieves these visit events to display them in the interface.

To enable the determination of bee visits, a track state object is maintained to build a temporal track for each bee, comprising five (5) distinct states (Fig. 4):

- **Pending**: Triggered when bee detection coordinates are on a flower, but not its center.
- **In center**: Activated once a bee's bounding box overlaps a flower's center, initiating a timer to track its duration within this locus.
- **Visiting**: Transitions from 'in center' when the stay of a bee exceeds the predefined time threshold, logging the duration of the visit.
- **Out of flower**: Assigned when a bee ceases to be detected within the flower, leading to deletion of its track state after a specified duration in this state.
- **Terminated**: The bee object is removed from the state controller if it is 'out of flower' for longer than a predefined threshold.

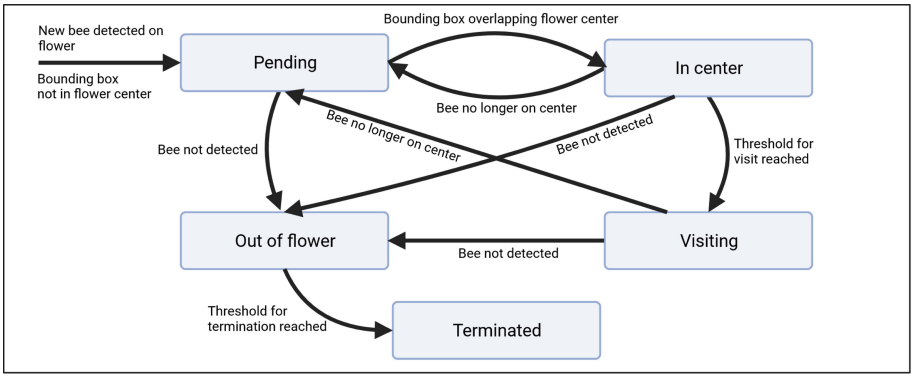

Fig. 4. Bee visit detector state diagram.

Figure 5 illustrates an example of a honeybee being tracked during an artificial flower visit.

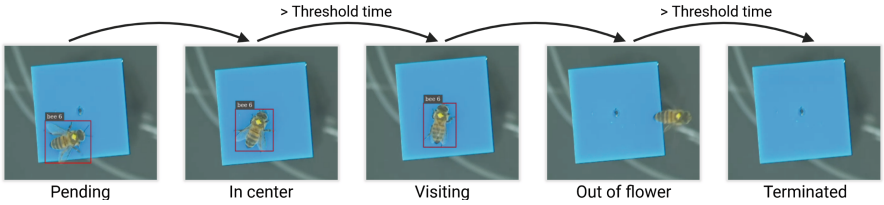

Fig. 5. Example of a bee visit with state transitions.

Regarding multiple visits by the same individual or mere pass-through occurrences, transitions from states 'in center' and 'visiting' back to 'pending' are allowed, allowing for cyclic traversal through the state diagram. To avoid overflowing the state controller of the system, when the bee is not detected, we classify it as 'out of flower' and remove the state object of the bee after a preconfigured amount of time.

3.5 Flower Patch Re-calibration

During an experiment in the field, it is common for the artificial flower patch to be physically moved due to wind or human interference. To account for this, we implemented a feature that allows the user to recalibrate the flower coordinates during the experiment's execution. A dedicated button in the user interface sends a command to the DeepStream pipeline to encode a frame of the current experiment setup as a JPEG image, which is saved as the most recent experiment image. Utilizing OpenCV's Python module, Scale-Invariant Feature Transform (SIFT) keypoints are put in correspondance to compute an affine matrix transform, mapping differences between previous and current flower patch images. Subsequently, this matrix is applied to the original flower coordinates, ensuring the integrity and consistency of the experiment data amidst environmental perturbations. This approach ensures that the intrinsic dynamic nature of field experiments is taken into account, maintaining the precision and reliability of the monitoring system under changing conditions.

4 Performance

The performance assessment, based on the deployment of the proposed pipeline, was directed towards measuring its performance under defined parameters. With an input resolution and size of 1184×1184 pixels for the YOLOv5n bee detection model, optimized with TensorRT, crucial metrics, specifically framerate and percent CPU load of the DeepStream process, were benchmarked in various test scenarios for both the FP32 and INT8 network modes (Table 1).

Hardware video encoding was performed with the H.264 codec at a bit rate of 8 Mbps. Enabling video encoding did increase CPU load, but did not decrease the framerate, demonstrating the Jetson Xavier's video encoding capabilities.

Table 1. System Performance Benchmarks. CPU load for the DeepStream process accounts for multiple cores.

Network Mode	Video Encoding	Output Rendering	Framerate (fps)	DeepStream CPU load (1 min avg)
FP32	Enabled	Enabled	60 fps	170%
FP32	Disabled	Enabled	55 fps	139%
FP32	Enabled	Disabled	67 fps	106%
FP32	Disabled	Disabled	64 fps	79%
INT8	Enabled	Enabled	60 fps	128%
INT8	Disabled	Enabled	60 fps	112%
INT8	Enabled	Disabled	110 fps	141%
INT8	Disabled	Disabled	110 fps	105%

Turning off display output rendering allowed the framerate to increase to the camera's maximum output of 110 fps.

When comparing the quantized INT8 model with the FP32 model, the INT8 model performed faster in almost all cases. An exception was found in the CPU load of the DeepStream process when video encoding and output rendering were both disabled, possibly because the framerate was capped at the display's maximum of 60 fps.

Table 2. System Visit Detection Performance.

Video clip ID	Precision	Recall	F1 Score
1	0.75	1.0	0.86
2	0.60	0.90	0.72
3	0.63	0.94	0.74
Overall	0.65	0.94	0.77

To assess the performance of the automatic visit detection system, humans annotated three 1-minute clips from different videos of honey bees visiting artificial flowers, for a total of 37 visits with duration ≥ 1 s. The video clips were used as input to the system with the faster INT8 quantized model and the system's visit detection performance was calculated (Table 2). Although recall was high (0.94 overall), precision was lower (0.65 overall) due to the system overdetecting visits. The main reason for the additional visits reported by the system is due to the way a bee visit is calculated: the overlap of a bee's bounding box and the flower's center. This caused bees walking close to the center of the flowers to sometimes be detected as visiting by the system. A possible solution to this issue is integrating a pose detection model to identify the bee's head, enabling better spatial detection of bee drinking events during flower visits, and reducing false positives.

This level of performance illustrates the system's capacity to deliver joint real-time analytics and concurrent video recording, which is vital for

post-experimental reviews and data validation, without sacrificing temporal resolution or computational efficiency.

5 Discussion

The integration of artificial intelligence (AI) into biological research has opened up new opportunities which were previously challenging due to technical and computational limitations. This study outlines a system that combines Deep Learning AI techniques, edge computing, and a user-friendly interface to monitor bee behavior in a practical and effective manner. Understanding the foraging patterns of bees is crucial, not only for conservation efforts, but also for the impact it has on our agricultural systems. Using AI to track and analyze their behavior in real time allows for a detailed examination of data and provides a deeper understanding of bee behavior in natural environments.

The adaptability of our system to monitor other insects or even larger animals, with modifications to the detection and tracking algorithms, poses an interesting possibility. Embedding more complex behavioral analysis algorithms could provide deeper insights into behavioral patterns and interactions between different species, expanding the ecological understanding that can be achieved with the system.

6 Conclusions and Future Work

The system described in this article integrates an AI detection architecture with a hardware-optimized video processing pipeline and an intuitive user interface, crafting a cohesive hardware/software solution that is applicable in the field by experimental biologists. Through the lens of the proposed system, we illustrate the transformative potential of edge AI in field biological experimentation, with the potential to enable the execution of biological assays with higher throughput while increasing the accuracy, precision, and reproducibility of measurements. Integration of AI into edge systems enables the emergence of innovative tools, which markedly enhance experimental field work.

We believe that by using technology to better understand and analyze bee foraging behavior, not only do we provide a versatile tool for field biologists but also lay solid groundwork for future research.

Moving forward, there are several paths for development and improvement:

- **UI Improvements**: Refining the user interface to enhance the user experience, making it even more intuitive and easy to use. One such addition is a gallery view in which experimenters can examine the most recent bee detections on a single condensed page.
- **Identity annotation**: Allowing researchers to annotate individual identities during the experiment within the web application will allow for easier analytics and decision-making during the assay.

- **Additional Inference Capabilities**: Adding AI inference engines for automatic classification and re-identification of individual bees during an experiment will increase the system's capabilities, providing more detailed data and allowing for a more in-depth analysis of the results.
- **Model Optimization**: Improving the bee detection model to increase accuracy and inference speed. Techniques such as model distillation might help boost its performance and efficiency without losing accuracy. Moreover, implementing a pose detection model can aid in accuracy by providing information on the exact position of the head, potentially decreasing false positives in the visits detection.

By continuing to merge additional technological advances with biological research, the system described in this paper provides a platform for future exploration, with the potential to be extended to new applications in exploring and understanding behaviors and dynamics in behavioral biology.

Acknowledgments. This research was supported in part by the intramural research program of the U.S. Department of Agriculture, National Institute of Food and Agriculture under Grant No. 2021-67014-34999. The findings and conclusions in this paper have not been formally disseminated by the U. S. Department of Agriculture and should not be construed to represent any agency determination or policy. This material is in part based upon work supported by the National Science Foundation under Grant No. 2318597. G. S-P acknowledges support from the PR-LSAMP Bridge to the Doctorate Program, NSF award 2306079. L. M. acknowledges support by IQ-BIO REU, NSF award 1852259. Figures were created with BioRender.com.

References

1. Khalifa, S.A.M., et al.: Overview of bee pollination and its economic value for crop production. Insects **12**(8), 1–23 (2021)
2. Fewell, J.H., Page, R.E.: Colony-level selection effects on individual and colony foraging task performance in honeybees, APIs mellifera L. Behav. Ecol. Sociobiol. **48**(3), 173–181 (2000)
3. Giray, T., et al.: Effect of octopamine manipulation on honeybee decision making: reward and cost differences associated with foraging. Anim. Behav. **100**, 144–150 (2015)
4. Sze, V., Chen, Y.H., Emer, J., Suleiman, A., Zhang, Z.: Hardware for machine learning: challenges and opportunities. In: Proceedings of Custom Integrated Circuits Conference, vol. 2017-April (2017)
5. Pegoraro, L., Hidalgo, O., Leitch, I.J., Pellicer, J., Barlow, S.E.: Automated video monitoring of insect pollinators in the field. Emerg. Top. Life Sci. **4**(1), 87–97 (2020)
6. Kane, G.A., Lopes, G., Saunders, J.L., Mathis, A., Mathis, M.W.: Real-time, low-latency closed-loop feedback using markerless posture tracking. Elife **9**, 1–29 (2020)
7. Droissart, V., Azandi, L., Onguene, E.R., Savignac, M., Smith, T.B., Deblauwe, V.: PICT: a low-cost, modular, open-source camera trap system to study plant-insect interactions. Methods Ecol. Evol. **12**(8), 1389–1396 (2021)

8. Delisle, Z.J., Flaherty, E.A., Nobbe, M.R., Wzientek, C.M., Swihart, R.K.: Next-generation camera trapping: systematic review of historic trends suggests keys to expanded research applications in ecology and conservation. Front. Ecol. Evol. **9**(Feb), 1–18 (2021)
9. Bjerge, K., Mann, H.M.R., Høye, T.T.: Real-time insect tracking and monitoring with computer vision and deep learning. Remote Sens. Ecol. Conserv. **8**(3), 315–327 (2022)
10. Rodriguez, I.F., et al.: Automated video monitoring of unmarked and marked honey bees at the hive entrance. Front. Comput. Sci. **3**, 769338 (2022)
11. Ratnayake, M.N., Dyer, A.G., Dorin, A.: Tracking individual honeybees among wildflower clusters with computer vision-facilitated pollinator monitoring. PLoS ONE **16**(2), e0239504 (2021)