2022 2nd International Conference on Electronic and Electrical Engineering and Intelligent System (ICE3IS), 4-5 November 2022

2022 2nd International Conference on Electronic and Electrical Engineering and Intelligent System (ICE3IS) | 978-1-6654-6541-0/22/$31.00 ©2022 IEEE | DOI: 10.1109/ICE3IS56585.2022.10010142

# Detection and tracking of honeybees using YOLO and StrongSORT

1st Hadj Kouider Benahmed
*Computer Science Department*
*University of Laghouat*
Laghouat, Algeria
h.benahmed@lagh-univ.dz

2nd Mohamed Lahcen Bensaad
*Computer Science Department*
*University of Laghouat*
Laghouat, Algeria
l.bensaad@lagh-univ.dz

3rd Noureddine Chaib
*Computer Science Department*
*University of Laghouat*
Laghouat, Algeria
n.chaib@lagh-univ.dz

***Abstract*—Understanding the behavior of honey bees is essential for maintaining a healthy bee colony. Hive monitoring systems are crucial for this purpose. In the last few years, computer vision and deep learning have become widely used in such systems. This paper uses a deep learning approach for detecting and tracking honey bees. Firstly, for detection, we employed the YOLO model using a data set of 1000 ground truth images. Secondly, for tracking, we used the StrongSORT approach. Results show that the detector performs well in both classes of honey bees (with or without pollen). The models based on this approach provide considerably good average tracking accuracy with low latency. Thus, this procedure is reliable and can be used in the future for real-time monitoring systems.**

***Index Terms*—HoneyBee, Detection, Tracking, YOLO, StrongSORT**

## I. Introduction

Honeybees are incredibly vital all over the world, and they are one of the most important elements of the ecosystem. Honey bees are not only important for the honey they produce but also for their pollination abilities. Additionally, pollination is aided when beekeepers move their beehives from one farm to another [1]. In fact, *Apis mellifera* offer around 14% of the entire pollination services in agriculture [2].

To keep the bee hives healthy, beekeepers must routinely verify them. Some of these activities can be hazardous to bees. Researchers have proposed various methods for monitoring the inside and outside of the hive.

In hive monitoring systems, computer vision is frequently employed [3] such as automatic tracking, and behavior analysis [4] [5], and 3D real-time stereo vision-based monitoring of flying honeybees [6]. With the emergence of deep learning [7], accurate object detection and recognition became possible, encouraging many researchers to use both deep learning and computer vision in their projects, particularly the benefits of extracting information from image data other than honey bees [8] [9] [10], and build honey bees hive monitoring systems for flight activity and background mortality [11] [12].

The Convolutional Neural Network (CNN) has progressed pretty fast in the last decade. In the realm of honeybees, we can find CNN-based approaches, identifying bees according to whether they are carrying pollen or not [13], concentrating on the observation of bees exiting and entering the hive [14], even the usage of CNN provides outstanding detection results, but that is insufficient for the immediate inference, especially when it aims to identify and locate bees that are carrying pollen. Using a Faster-R CNN object detector, the performance is much better than old approaches based on CNN [15].

Other deep learning approaches for object detection have emerged recently, such as Single Shot Multi-Box Detector (SSD) [16] and You Only Look Once (YOLO) [17], and these detection approaches have been applied in real-time bee hive monitoring systems.

The work in [18] focuses on tracking a single bee in a captured scene and plotting its path in a 2D plane using the background subtraction technique and YOLOv2 [18]. Another work in [19] employed a colony flow monitoring system with a one-way multi-frame detector SSD and the Faster R-CNN. Authors in [20] detect Varroa destructor mite using YOLOv5 and SSD. Another study in [21] used YOLOv3 tiny architecture to detect bees and identify pollen sacks with a Kalman filter and the Hungarian algorithm for tracking [21]. In [22], YOLOv4 and DeepSORT algorithms are used in the honeybee in-out monitoring system for tracking.

Unfortunately, the study in [22] has not provided an analysis of the designed system in terms of tracking. To that effect, our current work deals with this issue. In this paper, we use YOLOv5 and StrongSORT to detect and track honey bees and provide the detection and inference time analysis. Data is collected using honeybees videos available online. The frames of these videos clearly show honeybees and pollen sacs. We use these frames to create our data set. After that, we use our YOLOv5 model for detection, and we transfer the detected images to the input of StrongSORT, which combines motion and appearance information to track detected objects.

The paper begins with an introduction, followed by an explanation of the methodology we utilized for detection and tracking, how these approaches function, how we labeled our data, and where we used it. Finally, we discuss the obtained results of the used approach.

## II. Materials and Methods

In our work, we use both detection and tracking. After the image is detected using YOLOv5, StrongSORT is used to track it.

### A. Data-set

For our experiment, we listed many public honeybee data sets. However, none of them provided the suitable data set we were looking for. All of the images we found were captured in bird's view, and we wanted them in front of the beehive. Furthermore, most of the data sets are for classification rather than detection.

Two videos were captured at the entrance doors of the hives [23] [24] . The frame rate for both videos was 25 fps, and after frame extraction, we obtained approximately 1000 clear images for both classes (bees with or without pollen). After labeling using CVAT tool [28], as shown in Fig. 1, we collected 2583 pollen-carrying bees and 7092 bees as shown in Table I. To evaluate precision of tracking models, we labeled 300 frames from the same video that was utilized earlier with the CVAT tool.

TABLE I
NUMBER OF LABELS PER CLASSES.

| Bee | | | Bee carrying pollen | | |
|---|---|---|---|---|---|
| **Train** | **Val** | **Test** | **Train** | **Val** | **Test** |
| 7092 | 1672 | 139 | 2583 | 809 | 76 |

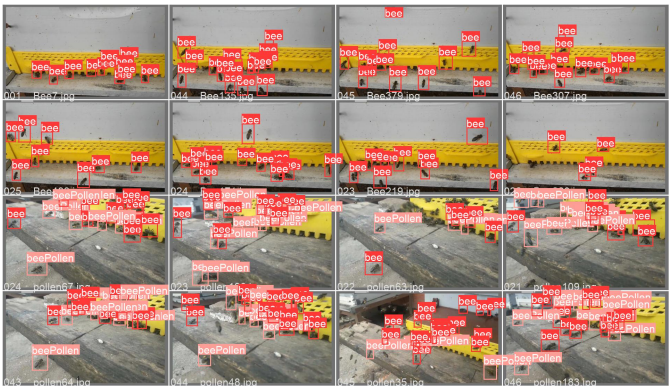

Fig. 1. Example of labeled images.

### B. Detection and tracking Networks

For the detection part, we use YOLOv5, an end-to-end detection model that we choose because it is the fastest among all past versions (YOLO, YOLO9000, YOLOv3, YOLOv4). In the YOLO architecture, the input is an image, and the output is an array of properties with bounding box center coordinates (x, y), bounding box (width, height), and confidence of the recognized object. The model selected is an Ultralytics implementation [25], accessible as an open-source project. YOLOv5 is divided into three parts: the backbone, the neck, and the head, as shown in Fig. 2. For the backbone, it uses CSPDarknet53 [31] to extract the features, and for the neck, it uses PANet [32] with a pyramid network feature for fusion features. YOLOv5 uses the same methods as YOLOv4 and YOLOv3 in the head section to achieve multi-size prediction and generate a predicted array.

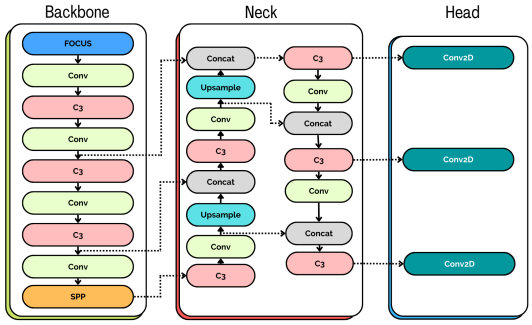


Fig. 2. Architecture of YOLOv5.

For our data set's training, we use an Nvidia (Tesla T4) GPU. In YOLOv5 architecture, there are five models that are distinct one from another (YOLOv5n, YOLOv5s, YOLOv5m, YOLOv5l, YOLOv5x). YOLOv5x has 567 layers and 86.47 million parameters. YOLOv5m has 369 layers and 21.19 million parameters. YOLOv5n has 270 layers and 7.23 million parameters. We choose these three models for our experiment because of the disparities in neural network size.

For the tracking part, we use StrongSORT [26]. It is a known Multi-Object Tracking method, and it is an improvement to DeepSORT [33] in both branches. For the appearance branch, Resnet50 [34] replaces CNN, and for the feature bank, the feature updating strategy [27] replaces it. We used StrongSORT on two different networks rather than Resnet50. One is MobileNetV2 [29]. The other one is OSnet [30]. We chose those two last models because they are lightweight when compared to ResNet-based approaches, and they outperform every other model in the person re-identification benchmarks. Then, we select MobileNetV2 x1.0 with 3.4M input parameters and MobileNetV2 x1.4 with 6.9M input parameters, as well as OSnet x1.0 with 2.2M input parameters, OSnet x0.5 with 1.3M input parameters, and OSnet x0.25 with 1.3M input parameters. Each model was trained using a large-scale public benchmark data-set for person re-identification, such as Market-1501 [36], DukeMTMC-reID [37], and Msmt17 [38].

### C. Metrics for detection and tracking

To assess the success of a deep learning model, we must employ measures that provide insight into the model's performance after training. Before we explain the metrics used in this work, we need to explain some terminology to understand how metrics get calculated. The first one is the Intersection over Union (IoU). It determines whether two bounding boxes are too close to each other. The value of IoU varied between 0 and 1, as shown in Fig. 3. The IoU must be greater than the threshold value in order to receive a correct prediction. Based on the IoU with the predicted bounding boxes and the threshold and ground truth, we calculate the following four metrics. The first one is True positive: The results indicate that the predicted box is in the desired location. The second one is False positive: the predicted box is in the wrong area of the ground truth. The third one is a false negative: the model

did not predict where the ground truth. The last one is a true negative: the area is correct, the model did not predict it, and there is no ground truth.

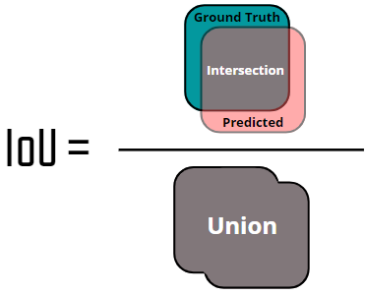


Fig. 3. Intersection over Union.

We used the calculated value of precision, as shown in equation (1), for our evaluation in order to assess how accurate our prediction is, i.e., what is the percentage of our correct estimation? We also utilized recall to see how effectively we found all of the true positives as shown in equation (2).

$$precision = \frac{TruePositive}{TruePositive + FalsePositive} \quad (1)$$

$$recall = \frac{TruePositive}{TruePositive + FalseNegative} \quad (2)$$

Also, we used mean average precision (mAP) as mAP@.5 and mAP@.5:.95. The mAP@.5 would be the average of all average *precisions* for all classes when the IoU threshold is greater than 0.5, and for the mAP@.5:.95, we calculate the average of all average precisions for all classes for each IoU threshold from 0.5 to 0.95 with a 0.05 step.

For the tracking metric, we use average tracking accuracy (ATA) from Local metrics for multi-object tracking [35]. It establishes a correspondence between whole tracks to maximize the total number of frames in which corresponding tracks overlap and then evaluates the fraction of correct tracks.

## III. RESULTS AND DISCUSSION

Two different setups were used to train and test our approach. In order to train our detection model, we used an Nvidia (Tesla T4) GPU card with 16 GB GDDR6, 32 GB RAM, and 8 Virtual Cores on the CPU. The resolution of images used in training was 640 x 640 px. We also employed picture augmentation with the options of a blur, random gamma, horizontal and vertical flip, and noise to boost data training. We chose 16 for the validation of batch size, and we also trained our models for 300 epochs. We measured precision and recall, as well as mAP@.5 and mAP@.5:.95, at the end of each epoch.

Based on the results, we make the following observation in Figure 4, the YOLOv5x and YOLOv5m models both reached 0.990 after 100 epochs, but the YOLOv5n could not.

Despite the fact that the models achieve a precision score of 0.990, the margin of error on how close the predicted

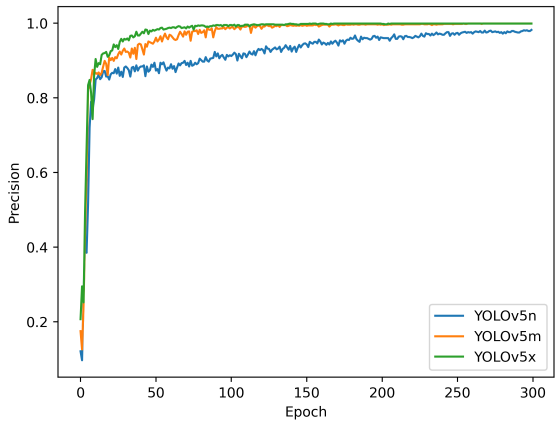


Fig. 4. Precision.

bounding box is to the ground truth for YOLOv5n is around 30%, while the margin of error for the two models YOLOv5m and YOLOv5x is less than 20%, as shown in Fig. 5

Table II represents all performance metrics obtained by the YOLOv5 model's validation test on our data set. The YOLOv5x model consistently outperforms all other models in terms of performance, although, as we can see, all models have precision and recall higher than 0.9.

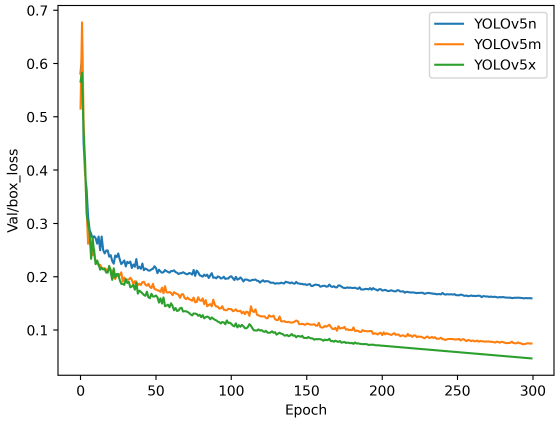


Fig. 5. Box loss.

In order to test our detection models with tracking models, we used a different configuration with an Nvidia GTX 970 GPU, 16 GB of RAM, and a six-core I7 5820K processor. We simply utilized the same video that we used before in detection for tracking. According to Table III, which indicates the average time it takes to detect and track a frame, we can see that YOLOv5x inference and tracking takes longer than 120 ms, even though that OSnet is smaller in size than MobileNetV2. Furthermore, MobileNetV2 x1.0 msmt17 is the fastest with both YOLOv5n and YOLOv5m, while MobileNetV2 x1.4 msmt17 is the slowest with YOLOv5n.

TABLE II
BEST TEST-SCORE ACHIEVED BY MODELS.

| YOLOv5n | | | | |
|---|---|---|---|---|
| **Class** | **precision** | | **mAP@.5** | **mAP@.5:.95:** |
| **Bee** | 0.977 | 0.978 | 0.989 | 0.716 |
| **BeePollen** | 0.986 | 0.975 | 0.987 | 0.802 |
| **All** | 0.982 | 0.977 | 0.988 | 0.759 |
| **YOLOv5m** | | | | |
| **Class** | **precision** | | **mAP@.5** | **mAP@.5:.95:** |
| **Bee** | 0.979 | **0.989** | 0.993 | 0.932 |
| **BeePollen** | 0.985 | 0.979 | 0.990 | 0.938 |
| **All** | 0.983 | 0.985 | 0.992 | 0.935 |
| **YOLOv5x** | | | | |
| **Class** | **precision** | | **mAP@.5** | **mAP@.5:.95:** |
| **Bee** | **0.995** | **0.989** | 0.994 | 0.944 |
| **BeePollen** | **0.990** | **0.982** | 0.993 | 0.956 |
| **All** | 0.992 | 0.980 | 0.994 | 0.950 |

Additionally, MobileNetV2 x1.0 market1501 was the fastest with YOLOv5x, while OSnet x0.5 msmt17 was the slowest.

TABLE III
AVERAGE DETECTION AND TRACKING TIME PER FRAME IN MS.

| | **Detection Model** | | |
|---|---|---|---|
| **Tracking Model** | **YOLOv5n** | **YOLOv5m** | **YOLOv5x** |
| **MobileNetV2 x1.0** *market1501* | 84.9 ms | 76.3 ms | **123.7 ms** |
| **MobileNetV2 x1.0** *msmt17* | **66.4 ms** | **75.6 ms** | 129.5 ms |
| **MobileNetV2 x1.4** *market1501* | 89.6 ms | 84.9 ms | 136.7 ms |
| **MobileNetV2 x1.4** *dukemtmcreid* | 83.2 ms | 100.4 ms | 141.8 ms |
| **MobileNetV2 x1.4** *msmt17* | 103.6 ms | 89.1 ms | 139.9 ms |
| **OSnet x1.0** *market1501* | 95.5 ms | 94.9 ms | 146.1 ms |
| **OSnet x1.0** *dukemtmcreid* | 86.2 ms | 95.6 ms | 146.5 ms |
| **OSnet x1.0** *msmt17* | 86.6 ms | 93.6 ms | 149.3 ms |
| **OSnet x0.75** *market1501* | 81.1 ms | 88.3 ms | 142.3 ms |
| **OSnet x0.75** *dukemtmcreid* | 78.1 ms | 88.9 ms | 142.5 ms |
| **OSnet x0.75** *msmt17* | 85.4 ms | 89.1 ms | 142.0 ms |
| **OSnet x0.5** *market1501* | 89.9 ms | 91.1 ms | 144.3 ms |
| **OSnet x0.5** *dukemtmcreid* | 86.7 ms | 86.2 ms | 147.2 ms |
| **OSnet x0.5** *msmt17* | 74.6 ms | 89.2 ms | 152.5 ms |
| **OSnet x0.25** *market1501* | 76.2 ms | 86.1 ms | 151.5 ms |
| **OSnet x0.25** *dukemtmcreid* | 102.1 ms | 91 ms | 148.8 ms |
| **OSnet x0.25** *msmt17* | 89.2 ms | 88.6 ms | 137.4 ms |

To evaluate tracking accuracy, we used ATA as a metric. Out of all the models we used for tracking, as shown in Fig. 8, 6 and 7, we chose the top 5 tracking models for each detection model. All tracking models start with an accuracy of 0.8, which decreases to 0.5 after 50 frames, then to 0.2 after 100 frames. The drop means that the part of the video we utilized began with a few bees, but after a few seconds, a large number of bees gathered on the left side of the hive's entrance, so the detection model is not trained to distinguish between a bee and a group of bees. In this experiment, OSnet x0.75 DukeMTMC-reID was the best tracking model with YOLOv5n, followed by MobileNetV2 x1.0 market1501 with YOLOv5m, and OSnet x0.75 market1501 with YOLOv5x. Samples of tracking and detection frames are in appendix A

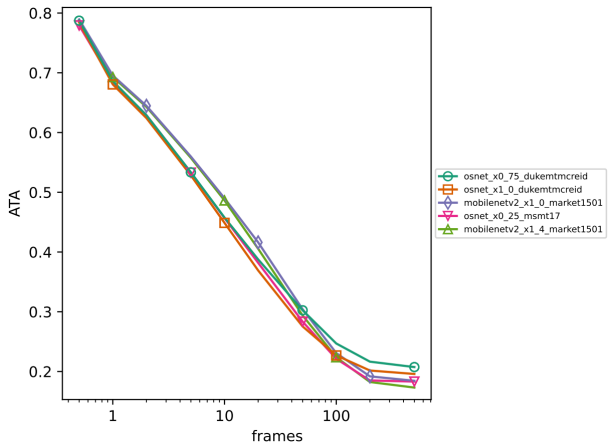


Fig. 6. Average track accuracy using YOLOv5n as the detector for various tracking models.

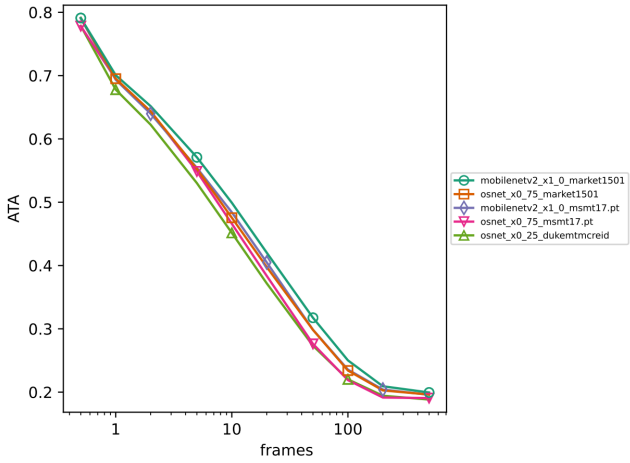


Fig. 7. Average track accuracy using YOLOv5m as the detector for various tracking models.

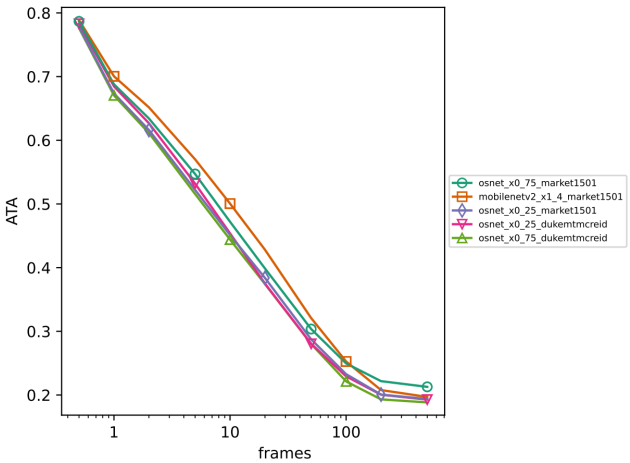


Fig. 8. Average track accuracy using YOLOv5x as the detector for various tracking models.

## IV. Conclusion

In this paper, we utilized another approach for the detection and tracking of honeybees where the angle of the camera is in front of the entrance of the hives. We used YOLOv5 for detection and StrongSORT for tracking. Firstly, we extracted images from two videos and then labeled them with two classes (with and without pollen) of bees. After that, we labeled 300 frames again from the video to test the accuracy of the tracking models. For the detection model, we got a score greater than 0.9, and as we see, all tracking models work great when the bees are not grouped. Additionally, the inference and detection times for a single frame vary from 70 to 150 ms, which motivated us to work on future projects to address this issue of tracking honeybees when they are gathered, and even test new tracking and detection models for real-time monitoring systems on edge.

## References


[1] R. R. Rucker, W. N. Thurman, M. Burgett, 'Honey bee pollination markets and the internalization of reciprocal benefits', American Journal of Agricultural Economics, vol. 94, page. 4, 956–977, 2012.

[2] L. A. Garibaldi , 'Wild pollinators enhance fruit set of crops regardless of honey bee abundance', science, vol. 339, page. 1608–1611, 2013.

[3] C. Chen, E.-C. Yang, J.-A. Jiang, T.-T. Lin, 'An imaging system for monitoring the in-and-out activity of honey bees', Computers and electronics in agriculture, vol. 89, page. 100–109, 2012.

[4] D. J. Kale, R. Tashakkori and R. M. Parry, "Automated beehive surveillance using computer vision," SoutheastCon 2015, 2015, pp. 1-3.

[5] A. Veeraraghavan, R. Chellappa and M. Srinivasan, "Shape-and-Behavior Encoded Tracking of Bee Dances,"in IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 30, no. 3, pp. 463-476, March 2008.

[6] G. Chiron, P. Gomez-Krämer, M. Ménard, 'Detecting and tracking honeybees in 3D at the beehive entrance using stereo vision', EURASIP Journal on Image and Video Processing, vol. 2013, page. 1–17, 2013.

[7] Y. LeCun, Y. Bengio, G. Hinton, 'Deep learning', nature, vol. 521, page. 436–444, 2015.

[8] Z. Babic, R. Pilipovic, V. Risojevic, G. Mirjanic, 'Pollen bearing honey bee detection in hive entrance video recorded by remote embedded system for pollination monitoring', ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences, vol. 3, page. 51, 2016.

[9] C. Yang, J. Collins, M. Beckerleg, 'A model for pollen measurement using video monitoring of honey bees', Sensing and Imaging, vol. 19, page. 1–29, 2018.

[10] C. Yang J. Collins, 'Deep learning for pollen sac detection and measurement on honeybee monitoring video', 2019 International Conference on Image and Vision Computing New Zealand (IVCNZ), 2019, page. 1–6.

[11] J. Marstaller, F. Tausch and S. Stock, "DeepBees - Building and Scaling Convolutional Neuronal Nets For Fast and Large-Scale Visual Monitoring of Bee Hives," 2019 IEEE/CVF International Conference on Computer Vision Workshop (ICCVW), 2019, pp. 271-278,

[12] F. Tausch, K. Schmidt, M. Diehl, 'Current achievements and future developments of a novel AI based visual monitoring of beehives in ecotoxicology and for the monitoring of landscape structures', bioRxiv, 2020.

[13] I. F. Rodriguez, R. Megret, E. Acuna, J. L. Agosto-Rivera and T. Giray, "Recognition of Pollen-Bearing Bees from Video Using Convolutional Neural Network," 2018 IEEE Winter Conference on Applications of Computer Vision (WACV), 2018, pp. 314-322.

[14] V. Kulyukin S. Mukherjee, 'On video analysis of omnidirectional bee traffic: Counting bee motions with motion detection and image classification', Applied Sciences, vol. 9, 2019.

[15] C. R. Yang, 'The use of video to detect and measure pollen on bees entering a hive', Auckland University of Technology, 2018.

[16] W. Liu , 'Ssd: Single shot multibox detector', European conference on computer vision, 2016, page. 21–37.

[17] J. Redmon, S. Divvala, R. Girshick, A. Farhadi, 'You only look once: Unified, real-time object detection', Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, page. 779–788.

[18] M. N. Ratnayake, A. G. Dyer, A. Dorin, 'Tracking individual honeybees among wildflower clusters with computer vision-facilitated pollinator monitoring', Plos one, vol. 16, page. 2, e0239504, 2021.

[19] J. Zhuang, X. Huang, X. Ye, 'Bee colony flow monitoring system based on SSD algorithm', ICETIS 2022; 7th International Conference on Electronic Technology and Information Science, 2022, page. 1–3.

[20] S. Bilik. 'Visual diagnosis of the Varroa destructor parasitic mite in honeybees using object detector techniques', Sensors, vol. 21, 8, 2021.

[21] T. N. Ngo, D. J. A. Rustia, E.-C. Yang, T.-T. Lin, 'Automated monitoring and analyses of honey bee pollen foraging behavior using a deep learning-based imaging system', Computers and Electronics in Agriculture, vol. 187, 2021.

[22] J.-S. Ryu, J.-W. Jung, C.-H. Jeong, B.-J. Choi, M.-L. Lee, H. W. Kwon, 'Honeybee In-Out Monitoring System by Object Recognition and Tracking from Real-Time Webcams', Journal of Apiculture, vol. 36, page. 273–280, 2021.

[23] bee video. (Mar. 05, 2022). Accessed: Jul. 16, 2022. [Online Video]. Available: https://www.youtube.com/watch?v=CRfql5vrfos.

[24] bee video. (Mar. 06, 2022). Accessed: Jul. 17, 2022. [Online Video]. Available: https://www.youtube.com/watch?v=F18lFr14hY4.

[25] G. Jocher , ultralytics/YOLOv5: v3.1 - Bug Fixes and Performance Improvements. Zenodo, 2020.

[26] Y. Du, Y. Song, B. Yang, Y. Zhao, 'Strongsort: Make deepsort great again', arXiv preprint arXiv:2202. 13514, 2022.

[27] Z. Wang, L. Zheng, Y. Liu, Y. Li, S. Wang, 'Towards real-time multi-object tracking', European Conference on Computer Vision, 2020, page. 107–122.

[28] B. Sekachev , opencv/cvat: v1.1.0. Zenodo, 2020.

[29] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, L.-C. Chen, 'MobileNetV2: Inverted residuals and linear bottlenecks', Proceedings of the IEEE conference on computer vision and pattern recognition, 2018, 4510–4520.

[30] K. Zhou, Y. Yang, A. Cavallaro, T. Xiang, 'Omni-scale feature learning for person re-identification', Proceedings of the IEEE/CVF International Conference on Computer Vision, 2019, 3702–3712.

[31] C.-Y. Wang, H.-Y. M. Liao, Y.-H. Wu, P.-Y. Chen, J.-W. Hsieh, I.-H. Yeh, 'CSPNet: A new backbone that can enhance learning capability of CNN', Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops, 2020, page 390–391.

[32] S. Liu, L. Qi, H. Qin, J. Shi, J. Jia, 'Path aggregation network for instance segmentation', Proceedings of the IEEE conference on computer vision and pattern recognition, 2018, page. 8759–8768.

[33] N. Wojke, A. Bewley, D. Paulus, 'Simple online and realtime tracking with a deep association metric', 2017 IEEE international conference on image processing (ICIP), 2017, 3645–3649.

[34] K. He, X. Zhang, S. Ren and J. Sun, "Deep Residual Learning for Image Recognition," 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 770-778.

[35] J. Valmadre, A. Bewley, J. Huang, C. Sun, C. Sminchisescu, C. Schmid, 'Local metrics for multi-object tracking', arXiv preprint arXiv:2104. 02631, 2021.

[36] L. Zheng, L. Shen, L. Tian, S. Wang, J. Wang, Q. Tian, 'Scalable Person Re-identification: A Benchmark', Computer Vision, IEEE International Conference on, 2015.

[37] Z. Zheng, L. Zheng, Y. Yang, 'Unlabeled Samples Generated by GAN Improve the Person Re-identification Baseline in vitro', arXiv preprint arXiv:1701. 07717, 2017.

[38] L. Wei, S. Zhang, W. Gao, Q. Tian, 'Person transfer gan to bridge domain gap for person re-identification', Proceedings of the IEEE conference on computer vision and pattern recognition, 2018, page. 79–88.

# APPENDIX A
# EXAMPLES OF DETECTION AND TRACKING

Fig. 9. Example on prediction.


Fig. 10. Example on tracking.