Revista de Informática Teórica e Aplicada - RITA - ISSN 2175-2745
Vol. 33, Num. 2 (2026) 81-88

RESEARCH ARTICLE

Multi-Fruit Tracking and 3-D Structure Recovery via CoTracker

Rastreamento Simultâneo de Frutos e Recuperação da Estrutura 3-D utilizando CoTracker

Thiago Teixeira Santos[1]*

Abstract: In agricultural robotics and orchard automation, tasks such as fruit detection, tracking, and spatial localization are essential for applications like yield prediction and harvesting. However, these tasks are challenging due to the similar appearance of fruits, occlusions, and the inherent difficulties of field robotics, including uncontrolled lighting conditions and the variability in orchard environments. This work leverages CoTracker, a transformer-based model for point tracking using cross-track/cross-time attention, to simultaneously perform multiple fruit tracking, 3-D fruit localization, and camera pose estimation. The proposed approach demonstrates promising results in fruit counting, tracking, and scene reconstruction, highlighting its potential in agricultural automation.
Keywords: fruit detection — fruit tracking – agricultural robotics – digital agriculture

Resumo: Na robótica agrícola e automação de pomares, tarefas como detecção de frutos, rastreamento e localização espacial são essenciais para aplicações como predição de produtividade e colheita. Contudo, essas tarefas apresentam desafios devido à aparência similar dos frutos, oclusões e às dificuldades inerentes da robótica de campo, incluindo condições de iluminação não controladas e a variabilidade em ambientes de pomar. Este trabalho utiliza o CoTracker, um modelo baseado em transformer para rastreamento de pontos usando atenção cruzada entre trajetos e tempos, para realizar simultaneamente o rastreamento múltiplo de frutos, localização tridimensional de frutos e estimação da pose da câmera. A abordagem proposta demonstra resultados promissores na contagem de frutos, rastreamento e reconstrução de cena, evidenciando seu potencial na automação agrícola.
Palavras-Chave: detecção de frutos — rastreamento de frutos — robótica agrícola — agricultura digital

[1] *Embrapa Agricultura Digital, Empresa Brasileira de Pesquisa Agropecuária (Embrapa), Brazil*
***Corresponding author**: thiago.santos@embrapa.br
DOI: http://dx.doi.org/10.22456/2175-2745.150960 • **Received:** 15/10/2025 • **Accepted:** 03/11/2025

1. Introduction

Image-based fruit detection in orchards is widely used for yield estimation [1, 2] and automated harvesting [3]. However, agricultural robotics still faces significant challenges in sensing and perception, mainly due to environmental variability [4]. Two critical issues in this context are fruit similarity and occlusion. As a moving agent traverses orchard rows, it captures images from various angles, resulting in a video stream where fruits may be intermittently occluded by leaves, branches, or other fruits, and may reappear multiple times. Developing a fruit counting system capable of avoiding double counting is particularly challenging due to the high similarity between fruits and the lack of distinctive features. Some authors frame this problem as an agronomical case of Multiple-Object Tracking (MOT): the task of assigning unique identities to objects, handling occlusions, and maintaining accurate trajectories despite appearance changes and noisy detections [5, 6, 2].

Some researchers have proposed the use of *3-D localization* as an approach to track or re-identify reappearing fruits [7, 8, 2]. The common idea behind such approaches is to maintain a three-dimensional representation of the orchard, associating 2-D detections and 3-D fruit locations by projective geometry [9]. *Structure from Motion* (SfM) [10] is employed to recover the needed 3-D model: the projective matrices for each camera pose and 3-D points on the surfaces of the objects in the scene (fruits, leaves, trunks, the ground, etc.). An advantage of this approach for fruit counting is the localization of the fruits in the three-dimensional space, useful for other tasks such as pruning and harvesting.

Both MOT and SfM rely on some data association method that allows the linking between observations in the previous

frames to observations in the next frames. Such a procedure produces inter-frame *tracks* of observations, employed in camera pose estimation in SfM and object tracking in MOT. In SfM, the most commonly used association approach is to employ the Scale-Invariant Feature Transform (SIFT) [11] for feature detection and description and the Fast Library for Approximate Nearest Neighbors (FLANN) [12] for feature matching based on the similarity of the descriptors. In MOT, association relies on methods such as Simple Online and Realtime Tracking (SORT) [13] and ByteTrack [14], built over the Kalman Filter and the Hungarian association algorithm. A key aspect of these approaches is that *they assume independence between the tracked observations*. Furthermore, they focus on the association between frame *pairs*, so long-term tracking depends on a successful chain of correct pairwise matchings. In other words, there is a lack of *spatial and temporal integration*.

Recent methods such as CoTracker [15] exploit the dependency between tracked points to improve tracking performance. CoTracker employs a transformer-based architecture, using self-attention operators to consider each track as a whole for a short video segment (window), transferring information between tracks. As stated in [15], the joint tracking of points is especially useful in cases of occlusion, where the information from the visible points, exploited by the attention mechanism, helps in tracking the occluded ones.

This study demonstrates the applicability of CoTracker for simultaneous data association in MOT and SfM for fruit tracking, three-dimensional localization, and camera pose estimation. The proposed method assumes that fruits are detected by an accurate yet imperfect object detector, such as a YOLO-based neural network [16] or a similar bounding-box-based detection system. In addition, fruits are assumed to serve as landmarks for vision-based odometry [17]. The method initializes CoTracker using object detections, generating temporally and spatially consistent tracks. These tracks are then utilized in visual odometry to estimate the three-dimensional positions of the fruits, as well as the camera poses for novel viewpoints. The poses and fruit positions are continuously optimized using factor graphs [18]. The final output consists of a set of fruit positions in three-dimensional space, camera poses for each frame in the video sequence, and visibility indicators specifying whether a fruit is visible or occluded in each frame.

2. Methods

2.1 Dataset

The data used to evaluate the system comprise a subset of MOrangeT [19], a dataset for tracking multiple objects in sweet oranges presented in previous work [2]. The dataset consists of image sequences of orange trees with annotations following the MOT16 text format [20]. In this format, each visible orange is assigned a unique numerical identifier, and its position is delineated by a bounding box in every frame in which it appears.

The original set consists of 12 sequences, each sequence associated with a single tree. However, assuming that in this proof of concept the fruits are also employed as navigation landmarks, a subset of 7 sequences was selected, eliminating sequences where there are not enough fruits for successful visual odometry, an issue that will be discussed in Section 4. A summary of the dataset is presented in Table 1, where the final column represents the count of visible fruits rather than the total fruit population per tree. For a full description of the MOrangeT dataset, including details on annotation and field locations, refer to [2].

Table 1. The MOrangeT subset employed on multiple fruit tracking tests.

Seq.	Gimbal	Variety	Frames	Oranges
V01	No	*Pera*	408	110
V04	No	*Valencia*	384	105
V05	No	*Natal*	293	148
V06	Yes	*Valencia*	447	122
V08	Yes	*Valencia*	544	192
V11	Yes	*Hamlin*	268	87
V12	Yes	*Hamlin*	307	115

2.2 Fruit detection

The proposed system takes as input a sequence of images corresponding to video frames recorded in the field, along with a set of fruit detections for each image, represented in the well-established format of bounding boxes. In this work, these bounding boxes are generated by a YOLOv5 neural network trained on OranDet, a publicly available dataset for sweet orange detection [21], which consists of 3,065 images containing a total of 9,769 annotated fruits. The detections undergo non-maximum suppression (NMS) with an Intersection over Union (IoU) threshold of 0.2. YOLO-based detectors also provide a *score* index indicating confidence for each detection. In this study, bounding boxes with a confidence score below 0.7 were discarded. The rationale behind this decision is to minimize false positives (spurious detections), even at the cost of increasing false negatives (missed detections). This trade-off is motivated by the assumption that the tracking process can more robustly handle missing detections than it can compensate for spurious ones: missed fruits can be observed from another viewpoint.

2.3 Point tracking with CoTracker

CoTracker is a *point tracker*, designed to track a set of N points in a short window of T images I_t, $t = 1,\ldots,T$. Together with the T input images, the tracker receives a query prompt, an $N \times 3$ tensor where each row (t, x^i, y^i) indicates a point (x^i, y^i) observed in the image I_t that should be tracked in the entire window. The output is a $T \times N \times 2$ tensor, $\mathscr{P}$, informing the positions (x_t^i, y_t^i) for each one of the N input points for each one of the T images. CoTracker also returns a *visibility flag*

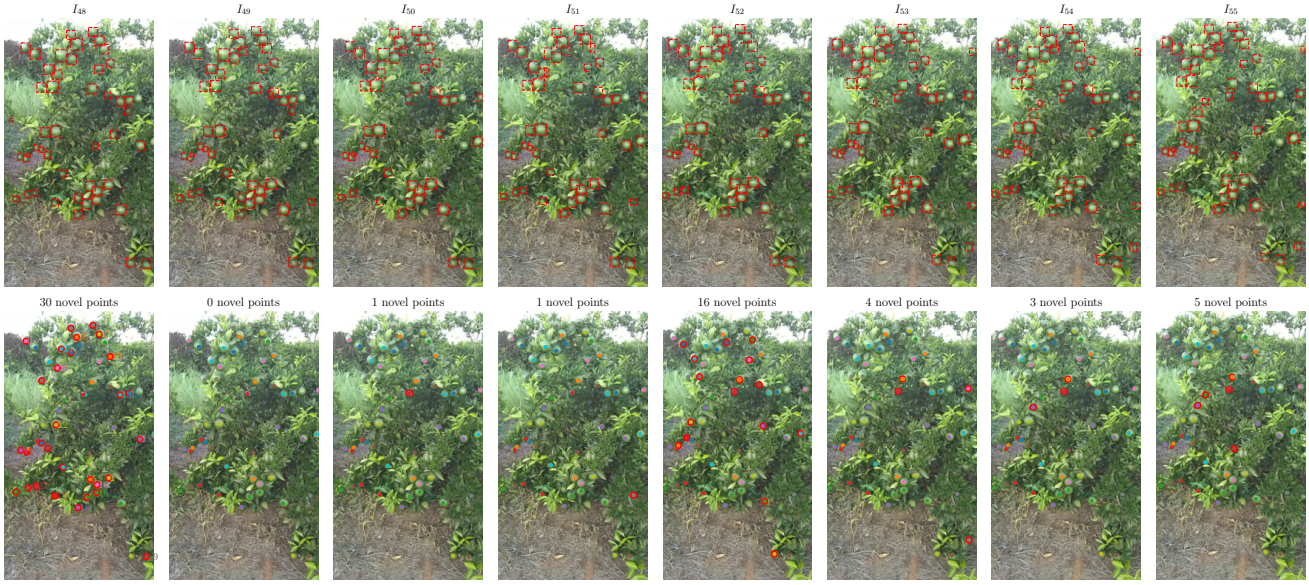

Figure 1. The tracking loop for window $I_{48},\ldots,I_{55}$. Top row: bounding box detections. Bottom row: tracked points. The landmarks are projected on $t = 48$, initiating CoTracker (big red dots). Novel, uncovered bounding boxes are found in the other frames, marked as big red dots. Small colored dots represent CoTracker point tracking results.

$v_t^i \in \{0,1\}$ indicating whether the i-th point is occluded or visible in I_t.

Basically, CoTracker integrates convolutional neural networks (CNNs) and transformers. A CNN is used as a feature extractor and the produced feature map is used to generate feature tokens $Q_t^i \in \mathbb{R}^d$ for the vicinity of each point i in each image I_t. The tensor $\mathscr{P}$ is used as another source of input tokens, and then transformers are employed for cross-track/cross-time attention. Such an attention system could be computationally intractable due to its original complexity $O(N^2T^2)$. Using *factorized attention* [22], CoTracker reaches an $O(N^2+T^2)$ complexity, being efficient even for large sets of N points if the time windows are short, around $T \approx 8$. For further details, readers should refer to [15].

2.4 Landmarks, landmark tracks and bounding boxes

The detections for each image I_t are sets where the j-th bounding box is represented by $\mathtt{b}_t^j = (x_1, y_1, x_2, y_2)$ where (x_1, y_1) and (x_2, y_2) correspond to the top left and bottom right corners of the box, respectively. The center of such a box is naturally defined as $\bar{\mathbf{x}}_t^j = (\bar{x}_t^j, \bar{y}_t^j)$ where $\bar{x}_t^j = (x_1 + x_2)/2$ and $\bar{y}_t^j = (y_1 + y_2)/2$.

In this work, a *landmark* $l_i = \langle \mathbf{X}^i, \mathtt{t}^i \rangle$ is composed of a vector $\mathbf{X}^i = (X^i, Y^i, Z^i, 1)^\intercal$, corresponding to a three-dimensional point in homogeneous coordinates, and a *landmark track* $\mathtt{t}^i$. The landmark track is a set $\mathtt{t}^i = \{\mathtt{b}_k^j\}_{k \in K}, K \subseteq \{0,\ldots,M\}$, the set of observed bounding boxes associated with the landmark l_i. The landmark represents a fruit with center $\mathbf{X}^i$ in the 3-D space and observed in bounding boxes $\mathtt{b}_k^j$ in a subset of the M-image sequence $\{I_m\}_{m=1}^M$. Consider $T \ll M$, which means that the entire image sequence can be larger than the short CoTracker windows.

The pose of the camera for the image I_m is encoded in a 3×4 projective matrix $\mathtt{P}_m$. A landmark $l_i = \langle \mathbf{X}^i, \mathtt{t}^i \rangle$ is projected[1] onto the 2-D point $\mathbf{x}_m^i = \mathtt{P}_m \mathbf{X}^i$ in I_m, and such projection must be contained in a bounding box $\mathtt{b}_m^j \in \mathtt{t}^i$ if the landmark is *visible* in image I_m. Landmarks and camera poses will be associated for long-term optimization in the form of *projective factors*, as will be presented in Section 2.7.

The presented formulation gives a simple but effective framework for linking point tracking from CoTracker to MOT and SfM based on bounding boxes. A landmark l_i can be projected onto point $\mathbf{x}_t^i = (x_t^i, y_t^i, 1)^\intercal$, and the row (t, x_t^i, y_t^i) can be inserted into the input prompt tensor for CoTracker. Meanwhile, if the returned visibility flag v_t^i indicates that the tracked point is visible in I_t, then the 2-D point on the CoTracker output tensor $\mathscr{P}[t,i] = (x_t^i, y_t^i)$ should be within the limits of a bounding box.

2.5 Initialization

On initialization, there are no landmarks or camera poses available. The first window of $T = 8$ consecutive frames

[1] The vector $\mathbf{x}_m^i = (x_m^i, y_m^i, 1)^\intercal$ represents a 2-D point in homogeneous coordinates. Homogeneous coordinates allow projective transformations to be expressed as a simple matrix multiplication, as $\mathtt{P}_m \mathbf{X}^i$, providing not just an elegant mathematical representation, but also making computations more efficient [9].

$\{I_t\}_{t=1}^{8}$ is loaded for CoTracker and the input prompt tensor is filled with the centroids $\bar{\mathbf{x}}^j$ for each of the $\mathtt{b}_1^j$ bounding boxes detected by YOLOv5 in the first image. The output tensor $\mathscr{P}_1$ from this first CoTracker call is used to create tracks $\mathtt{t}^j$, adding bounding boxes $\mathtt{b}_t^j$ to the track if they contain the corresponding point $\mathscr{P}_1[t,j] = (x_t^j, y_t^j)$ —i.e., if (x_t^j, y_t^j) is inside $\mathtt{b}_t^j$. Such boxes $\mathtt{b}_t^j$ are considered *covered* and will not add new lines to the input prompt in the following iterations.

The same window will be used in the following iterations, now addressing the *uncovered* detected boxes. For the next $t = 2,\ldots,8$, the centroids of the uncovered boxes add new rows (t, x_t^j, y_t^j) to the prompt tensor, and new calls to the CoTracker routine are made for each t. At the end of each call, point tracks with 4 or more observations (visible tracked points) are used to create bounding box tracks $\mathtt{t}^j$, selecting boxes $\mathtt{b}_t^j$ from the detector that contain the point $\mathscr{P}_t[t,j] = (x_t^j, y_t^j)$ from CoTracker's output for the t-th iteration. A threshold of $T/2 = 4$ was applied to ensure that points were visible in at least 50% of the observation window.

2.5.1 Visual odometry initialization

A set of tracks can be used to initialize the visual odometry module. The initialization used here is the most common one used in 3-D vision, computed from a set of 2-D point correspondences for a pair of frames I_{t_a} and I_{t_b}. The essential matrix $\mathtt{E}$ is estimated using the [23] 5-point algorithm in a robust RANSAC procedure with outliers. The pose at t_a is trivial: a camera at the center of the 3-D space with no rotation or translation. The pose at t_b is recovered from $\mathtt{E}$, the camera rotation and translation relative to t_1. The OpenCV computer vision library presents efficient implementations for this procedure, and the interested reader can find in-depth descriptions in [23, 9], and [17]. After the camera projective matrices $\mathtt{P}_{t_a}$ and $\mathtt{P}_{t_b}$ are determined, by integrating the pose data and the internal camera matrix, which contains focal distance information, a triangulation procedure is used to estimate the 3-D points associated with the 2-D point correspondences. Subsequently, a first set of landmarks l_i is instantiated from the three-dimensional points and their source tracks.

2.6 Integrating windows: the main loop

After initialization, the system can rely on the landmarks to (i) determine the camera pose for each frame I_t and (ii) initialize the first CoTracker prompt, continuing to follow the previously observed landmarks. The pose for t is estimated from the landmarks l_i, which include the 3-D positions and their observed locations in I_t, determined by the tracking procedure. This estimation of pose from 3-D to 2-D correspondences is known as *perspective from n points* (PnP or *resection*) [17]. In this work, the estimation of all projective matrices is done using the landmark set and resection, except for the initial pose pair.

When a new window of T frames is processed, $I_{t_1},\ldots I_{t_8}$, the set of landmarks l_i is projected onto t_1 using $\mathbf{x}_{t_1}^i = \mathtt{P}_{t_1}\mathbf{X}^i$, and these points are used to initialize the first CoTracker prompt. For the remaining seven frames, new query rows are added to the prompt, adding novel uncovered detections to the tracking, in the same way described in Section 2.5. The tracks generated for these new detections lead to the creation of additional landmarks, which are subsequently incorporated into the existing landmark set.

2.7 Factor graphs and global optimization

At its core, SfM involves a joint optimization problem: the goal is to estimate the poses $\mathtt{P}_m$ and landmark positions $\mathbf{X}^i$ that minimize the reprojection error, defined as the distance between the projected point $\mathbf{x}_m^i = \mathtt{P}_m\mathbf{X}^i$ and the observed point $\bar{\mathbf{x}}_m^i$, which, in our case, corresponds to the centroid of the observed bounding box.

Factor graphs [18] provide an elegant and efficient representation of the reprojection problem, allowing a structured approach to optimization. These bipartite graphs consist of variable nodes, representing unknowns such as camera poses and landmark positions, and factor nodes, encoding probabilistic constraints or error terms, such as reprojection factors. In this context, binary reprojection factors connect camera pose and landmark nodes, enforcing constraints based on observed image measurements (Figure 2). By leveraging the sparse structure of the graph, optimization can be performed efficiently using iterative methods such as the Levenberg–Marquardt (LM) algorithm, focusing computation only on directly connected dependencies. This factorized approach significantly improves scalability and computational performance compared to direct bundle adjustment formulations.

Whenever a landmark or pose is added to the model, corresponding graph nodes are created to represent them, and reprojection factors (edges) are incorporated into the graph, storing the observed points for subsequent reprojection error minimization. This minimization process constitutes a nonlinear least-squares problem, which is solved using the LM algorithm [9]. Once a new set of landmarks is introduced, LM optimization is performed. It is crucial to note that the LM algorithm requires well-initialized values; if the initial estimates for poses and landmark positions deviate significantly from the optimal values, the algorithm may yield suboptimal results. In practice, resection values for poses and triangulation values for landmarks serve as effective initializations for the LM algorithm.

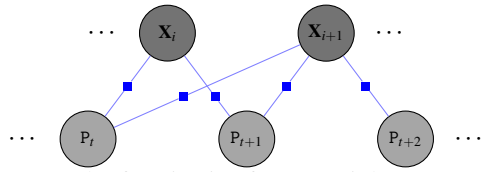

Figure 2. Graph of projective factors. Light gray: projective matrices for each camera pose. Dark gray: landmark 3-D positions.

2.8 Post-processing

To remove false landmarks caused by erroneous fruit detections, fruits that were visible for fewer than $V_{\min}$ frames

throughout the entire sequence (according to the visibility flag) are removed, where $V_{\min} = 8$. This threshold was chosen to match the CoTracker window size $T = 8$, as reliable landmark initialization requires tracking across at least one complete window to establish consistent spatial correspondences. Landmarks visible in fewer frames are likely spurious detections or highly occluded fruits that provide insufficient constraints for accurate 3-D localization.

2.9 Implementation details

The system was implemented in Python 3.11. The implementations used for essential matrix estimation, pose recovery, point triangulation, and resection (PnP) are from OpenCV 4.9. Factor graph representation and LM optimization are performed using GTSAM 4.3. The original CoTracker 2 code[2] by [15] was used for point tracking. Camera intrinsic parameters and lens distortion coefficients were estimated using COLMAP's automatic calibration [10], which simultaneously recovers camera parameters and scene structure during SfM reconstruction. The calibrated parameters were used for all subsequent projective operations in the proposed pipeline.

3. Results

Figure 3 shows frames I_{174}, I_{179}, I_{184}, and I_{189} from video sequence V04 in MOrangeT. A five-frame step was used here to display a large temporal window and pose variation. The scene highlights the challenging conditions typical of this type of application: green fruits amidst green foliage, varying lighting conditions (with the upper fruits under direct sunlight and the lower ones in shadows), and occlusions. Notice how fruits l_{103} (bottom) and l_{109} (top) transition from occluded to visible states.

It is important to note that the points $\mathbf{x}_t^i$, which represent the reprojections of the landmarks in the frames, are not located in the center of the bounding boxes. This is because the 3-D positions of the landmarks, $\mathbf{X}^i$, are optimized using the LM algorithm to minimize the reprojection error. Additionally, since oranges are not perfectly spherical, the center of the bounding box serves as a rough approximation of the center of the fruit, accounting for pose variations.

Table 2. Results: HOTA, predicted number of fruits, ground-truth for number of fruits, false positives, false negatives, and percentage error in fruit counting.

Seq.	HOTA	Pred	Truth	FP	FN	Error
V01	30.183	117	110	7	0	6.36%
V04	66.485	100	105	0	5	-4.76%
V05	50.228	143	148	0	5	-3.38%
V06	40.583	175	122	53	0	43.44%
V08	42.477	228	192	36	0	18.75%
V11	45.431	98	87	11	0	12.64%
V12	45.810	117	115	2	0	1.74%
All	47.457	978	879	109	10	11.26%

Table 2 presents the results for fruit counting and tracking, evaluated using HOTA tracking metrics [24]. These metrics assume flawless tracking as score 100. These metrics consolidate various errors into a single score, including identity switches, misalignment of the bounding box, missed detections, and missed tracks. Achieving accurate tracking is inherently more challenging than accurate counting, as even a short sub-track may suffice for fruit counting but would be heavily penalized under tracking metrics. The combined counting error in the seven sequences was 11%, largely influenced by a significant error in sequence V06 (43%). This sequence exhibits numerous false positives caused by errors on re-identification of reappearing fruits, possibly caused by poor pose estimation.

The total processing time for 3-D tracking and localization ranged from approximately 3 minutes (V05) to 15 minutes (V08). This represents a substantial improvement over methods based on COLMAP [7, 2]: a standard run could require half an hour solely for SfM processing on the shortest sequences, using identical hardware (Intel Xeon CPU at 3.40GHz with RTX 4000 GPU). The primary advantage stems from the reduced number of landmarks, as COLMAP employs thousands of SIFT-based feature points, whereas the proposed method restricts landmarks exclusively to fruit features. However, when COLMAP is limited to a few hundred landmarks —a number comparable to the fruit count used in this study— its SfM processing time drops significantly.

Figure 4 displays the estimated 3-D model, including the reconstructed camera poses. The estimated motion across all seven sequences aligns with the intended path followed by the camera operators, who were instructed to first contour the lower part of the tree, then return to record the middle section of the canopy, and finally cover the top of the tree, directing the camera toward the tallest part of the plant.

4. Discussion

4.1 Extending the Landmark Framework

In this study, fruits served as landmarks for visual odometry, but this framework can be naturally extended to incorporate *additional landmark types*. For instance, sampling a grid of arbitrary points across multiple frame locations would provide supplementary tracks, enhancing the visual odometry system's overall robustness. This extension represents a form of *semantic visual odometry*, where tracks carry distinct semantic meanings (e.g., fruits, trunks, soil) and can be leveraged differently within the SfM framework depending on their characteristics and reliability.

4.2 Robustness to Environmental Conditions

While the proposed method was not explicitly tested under adverse weather conditions, its potential robustness can be analyzed. Rain and dust primarily affect image appearance, and these effects can be effectively emulated through image

[2] Co-Tracker ⟨https://co-tracker.github.io⟩

Figure 3. Final landmarks for sequence V04. The points show the projections of the 3-D landmark on each frame. The rectangular boxes show the detected bounding boxes from YOLOv5 associated with that landmark. Dashed boxes indicate occluded (non-visible) landmarks.

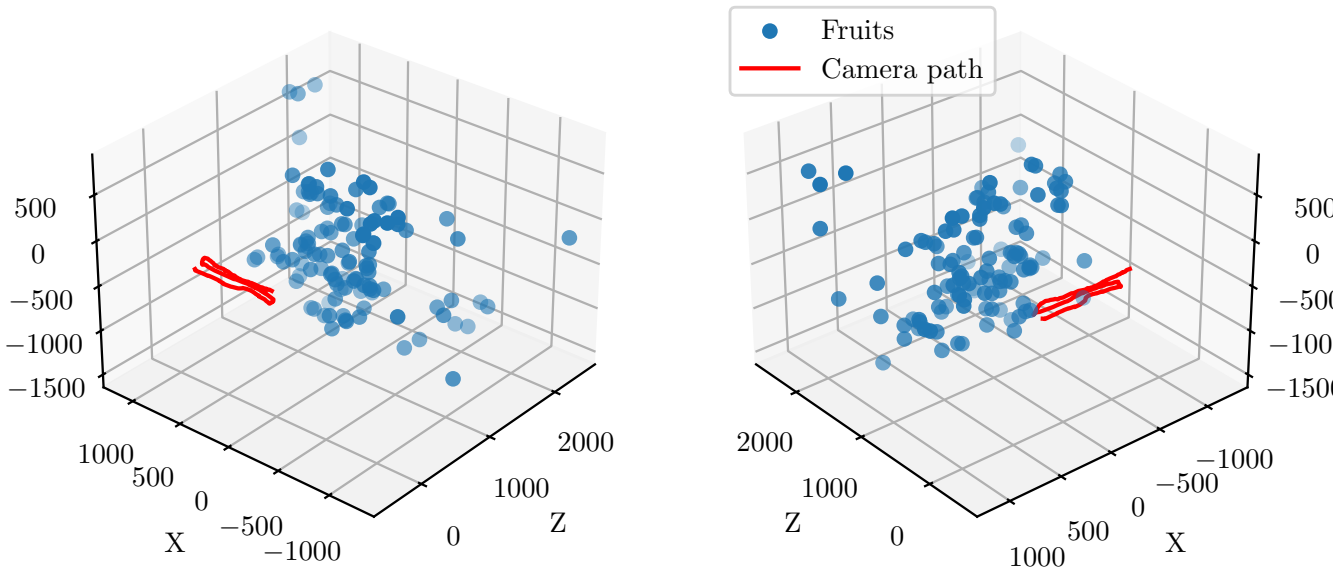

Figure 4. Three-dimensional reconstruction showing camera poses and 3-D landmarks for sequence V12, produced by the proposed system.

augmentation techniques such as filters and added noise during training, making fruit detection more resilient to such conditions.

Wind presents a more fundamental challenge. Since the SfM framework inherently assumes static landmarks, wind-induced movement of fruits and foliage could significantly compromise tracking accuracy. Here, semantic visual odometry offers a practical solution: by restricting landmarks to rigid structures such as trunks, poles, and fences for pose estimation, while treating fruits as dynamic objects solely for detection and counting, the system could maintain robustness even under windy conditions. Validating performance under these challenging field conditions remains an important direction for future work.

4.3 Performance Analysis and Future Directions

The methodology proposed here exploits CoTracker's point tracking capabilities in an object tracking context. However, this approach faces inherent limitations: individual points poorly represent the full appearance and geometry of three-

dimensional objects like fruits. This representational gap is reflected in the results. Compared to our previous work [2], which achieved a HOTA of 56.039 and a median counting error of 6.95%, the current method does not demonstrate superior performance. For this dataset at least, *cross-tracking offered no clear advantage over the tracking independence assumption*.

Nevertheless, the potential of cross-track and cross-temporal attention should not be dismissed entirely. In species where fruits typically appear in clusters (e.g., apples), such spatiotemporal reasoning could prove more beneficial. Moreover, CoTracker's CNN-based feature detector backbone could be adapted for object detection, enabling a more integrated approach. A promising architectural direction would involve a multi-head neural network capable of jointly handling detection and tracking within a unified multitask learning framework, potentially overcoming the limitations of point-based representation while retaining the benefits of attention mechanisms. These directions warrant further investigation in future research.

5. Conclusion

This work presents a successful proof-of-concept integrating MOT and SfM for sweet orange tracking and 3-D localization. The methodology is also applicable to tasks that require 3-D position estimation, such as robotic navigation. Critically, results indicate that modern attention-based trackers —leveraging spatiotemporal attention for multipoint tracking— show significant promise for agricultural field robotics. Future studies should validate these findings across diverse datasets, including different fruit and orchard training systems, to determine whether cross-track/cross-time attention confers substantial advantages over alternative trackers.

Acknowledgements

This study was financed, in part, by the São Paulo Research Foundation (FAPESP), Brazil, process number 2022/09319-9.

Author Contributions

Single author.

References

[1] HÄNI, N.; ROY, P.; ISLER, V. A comparative study of fruit detection and counting methods for yield mapping in apple orchards. *Journal of Field Robotics*, v. 37, n. 2, p. 263–282, 2020.

[2] SANTOS, T. T. et al. Multiple orange detection and tracking with 3-d fruit relocalization and neural-net based yield regression in commercial sweet orange orchards. *Computers and Electronics in Agriculture*, Elsevier BV, v. 224, p. 109199, set. 2024. ISSN 0168-1699. Disponível em: ⟨http://dx.doi.org/10.1016/j.compag.2024.109199⟩.

[3] LI, T. et al. A multi-arm robot system for efficient apple harvesting: Perception, task plan and control. *Computers and Electronics in Agriculture*, Elsevier BV, v. 211, p. 107979, ago. 2023. ISSN 0168-1699.

[4] SHAMSHIRI, R. R. et al. Agricultural robotics to revolutionize farming: Requirements and challenges. In: *Mobile Robots for Digital Farming*. [S.l.]: CRC Press, 2025. p. 107–155.

[5] JONG, S. de et al. APPLE MOTS: Detection, Segmentation and Tracking of Homogeneous Objects Using MOTS. *IEEE Robotics and Automation Letters*, Institute of Electrical and Electronics Engineers (IEEE), v. 7, n. 4, p. 11418–11425, out. 2022. ISSN 2377-3774.

[6] VILLACRÉS, J. et al. Apple orchard production estimation using deep learning strategies: A comparison of tracking-by-detection algorithms. *Computers and Electronics in Agriculture*, Elsevier BV, v. 204, p. 107513, jan. 2023. ISSN 0168-1699.

[7] LIU, X. et al. Monocular camera based fruit counting and mapping with semantic data association. *IEEE Robotics and Automation Letters*, v. 4, n. 3, p. 2296–2303, 2019.

[8] MATOS, G. P. et al. Tracking and counting apples in orchards under intermittent occlusions and low frame rates. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*. [S.l.: s.n.], 2024. p. 5413–5421.

[9] HARTLEY, R.; ZISSERMAN, A. *Multiple View Geometry in Computer Vision*. [S.l.]: Cambridge University Press, 2004. ISBN 9780511811685.

[10] SCHÖNBERGER, J. L.; FRAHM, J.-M. Structure-from-motion revisited. In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. [S.l.: s.n.], 2016. p. 4104–4113.

[11] LOWE, D. G. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, Springer Science and Business Media LLC, v. 60, n. 2, p. 91–110, nov. 2004. ISSN 0920-5691.

[12] MUJA, M.; LOWE, D. G. Scalable nearest neighbor algorithms for high dimensional data. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, v. 36, n. 11, p. 2227–2240, 2014.

[13] WOJKE, N.; BEWLEY, A.; PAULUS, D. Simple online and realtime tracking with a deep association metric. In: *2017 IEEE International Conference on Image Processing (ICIP)*. [S.l.]: IEEE, 2017.

[14] ZHANG, Y. et al. Bytetrack: Multi-object tracking by associating every detection box. In: ____. *Computer Vision – ECCV 2022*. [S.l.]: Springer Nature Switzerland, 2022. p. 1–21. ISBN 9783031200472.

[15] KARAEV, N. et al. CoTracker: It Is Better to Track Together. In: LEONARDIS, A. et al. (Ed.). *Computer Vision – ECCV 2024*. Cham: Springer Nature Switzerland, 2025. p. 18–35. ISBN 978-3-031-73033-7.

[16] TERVEN, J.; CÓRDOVA-ESPARZA, D.-M.; ROMERO-GONZÁLEZ, J.-A. A Comprehensive Review of YOLO Architectures in Computer Vision: From YOLOv1 to YOLOv8 and YOLO-NAS. *Machine Learning and Knowledge Extraction*, v. 5, n. 4, p. 1680–1716, 2023. ISSN 2504-4990.

[17] SCARAMUZZA, D.; FRAUNDORFER, F. Visual odometry [tutorial]. *IEEE Robotics and Automation Magazine*, v. 18, n. 4, p. 80–92, 2011.

[18] DELLAERT, F. Factor graphs: Exploiting structure in robotics. *Annual Review of Control, Robotics, and Autonomous Systems*, Annual Reviews, v. 4, n. 1, p. 141–166, maio 2021. ISSN 2573-5144.

[19] SANTOS, T. T. et al. *eContaFruto MOrangeT*. Redape, 2024. Disponível em: ⟨https://www.redape.dados.embrapa.br/citation?persistentId=doi:10.48432/OI7BFG⟩.

[20] MILAN, A. et al. *MOT16: A Benchmark for Multi-Object Tracking*. 2016. Disponível em: ⟨https://arxiv.org/abs/1603.00831⟩.

[21] SANTOS, T. T. et al. *eContaFruto OranDet: Base de dados para detecção de laranjas por imagem.* Redape, 2024. Disponível em: ⟨https://www.redape.dados.embrapa.br/citation?persistentId=doi:10.48432/IG6VIQ⟩.

[22] BERTASIUS, G.; WANG, H.; TORRESANI, L. *Is Space-Time Attention All You Need for Video Understanding?* [S.l.]: arXiv, 2021.

[23] NISTER, D. An efficient solution to the five-point relative pose problem. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Institute of Electrical and Electronics Engineers (IEEE), v. 26, n. 6, p. 756–770, jun. 2004. ISSN 0162-8828.

[24] LUITEN, J. et al. HOTA: A higher order metric for evaluating multi-object tracking. *International Journal of Computer Vision*, v. 129, p. 548–578, 2021.