Towards 3D Visualization of Insect Trajectories Using Stereo Event Camera Data and RGB-Based SfM Point Clouds

Regina Pohle-Fröhlich[1,2][a], Tobias Bolten[1][b], Colin Gebler[1,2][c] and Felix Lögler[1][d]

[1]*Institute for Pattern Recognition, Niederrhein University of Applied Sciences, Krefeld, Germany*
[2]*Doctoral School NRW, Department Computer and Data Science, Bochum, Germany*
{*regina.pohle, tobias.bolten, colin.gebler, felix.loegler*}*@hs-niederrhein.de*

Keywords: Event Camera, Insect Monitoring, Visualization, Structure-from-Motion

Abstract: To investigate the factors influencing insect behaviour, a monitoring system is needed that can automatically record insect activity and environmental conditions over extended periods. For this purpose, a stereo setup with two event cameras to capture the flight paths of insects is used. In this paper, the visualization of these flight trajectories within the context of their environment is focused on. The processing steps required to automatically detect reference markers in both the event data and the corresponding RGB video frames captured by a smartphone camera are described. A coloured point cloud of the environment is reconstructed from the video data using Structure-from-Motion and is aligned with the recorded insect flight paths. This enables novel insights into insect–flower interactions by integrating environmental context into event data visualisation, allowing analyses such as tracking the sequence of an individual insect's flower visits or quantifying the number of visitors to a single flower within a given time frame.

1 INTRODUCTION

In recent years, insect biomass and biodiversity have declined markedly on a global scale. This decline is driven by multiple factors, including climate change, intensified agriculture, the spread of invasive species through globalisation and travel, as well as rising atmospheric CO_2 levels (Saleh et al., 2024). Insects play crucial ecological roles as prey for other animals, as natural enemies in biological pest control, and as pollinators of around 90% of flowering plants. Therefore, it is essential to identify and understand the stressors that have triggered this development (Landmann et al., 2023). Equally important is the establishment of robust methodologies for assessing the effectiveness of protective measures. It is evident that a considerable number of relationships between insects and their interaction with the environment remain unexplored. To address this gap, a range of monitoring techniques is used (Van Klink et al., 2024). Manual observations, such as counting floral visitors, continue to be widely used, but they are time-consuming, poorly reproducible, and limited to small spatial and temporal windows. Automated optical and acoustic methods offer promising alternatives. We examined the limitations of manual observation using eye-tracking glasses in comparison with automatic optical monitoring, and these findings will be discussed in more detail in Section 4.1.

Conventional RGB cameras employed to investigate insect–environment interactions can, in practice, only observe a limited field of view. As with manual assessments, the reliable detection of small, fast-moving insects is constrained by motion blur and the visual complexity of natural backgrounds. Acoustic monitoring techniques also have their limitations, as they do not inherently permit spatial mapping and are highly susceptible to interference from background noise, particularly in environments with traffic. For this reason, the use of event cameras for insect monitoring has been investigated in recent years (Gebauer et al., 2024; Pohle-Fröhlich et al., 2024; Pohle-Fröhlich et al., 2025; Arning et al., 2025; Pohle-Fröhlich and Bolten, 2023). Such cameras operate on a fundamentally different sensing principle compared to conventional frame-based imaging systems (Gallego et al., 2022).

Rather than capturing images at fixed frame rates, they consist of independently operating pixels that

[a] https://orcid.org/0000-0002-4655-6851
[b] https://orcid.org/0000-0001-5504-8472
[c] https://orcid.org/0009-0006-4654-032X
[d] https://orcid.org/0009-0003-7204-4686

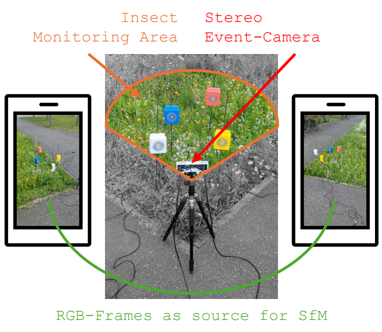


Figure 1: RGB- and Event-Camera setup used for calibration and measurement. The green curve represents the range of motion of the smartphone camera.

asynchronously report changes in local brightness. When the change at a pixel exceeds a defined threshold, an event is generated and encoded as a tuple $\{(x,y),t,p\}$, where (x,y) is the pixel location, t is the timestamp (with microsecond precision), and p indicates the polarity of the brightness change. This results in a data stream that is sparse and driven by the dynamics of the scene, with no redundant information from static backgrounds. The extremely high temporal resolution make event cameras particularly well-suited for observing fast and irregular motion, such as the flight of small insects.

However, a drawback of event cameras lies in the difficulty of interpretation by domain experts, since static elements of the scene, such as the position of flowers in the absence of wind, are not included in the recorded data. This paper aims to address this gap in the literature by proposing a processing approach that integrates event and RGB data for 3D visualization of insect trajectories. The most significant contributions are the extraction of markers in both the point cloud recorded with an event camera and the point cloud generated with Structure-from-Motion (SfM) for environment representation. Additionally, the method used to calculate depth for the segmented insect flight paths is presented. Finally, we demonstrate the new evaluation possibilities resulting from our representation method.

2 RELATED WORK

First, we introduce the fundamental measurement setup to provide context for the related work discussed in this section.

A stereo event-based camera setup is used to capture insect flight trajectories. The system consists of two SENSING SE1-S4-USB event cameras, which use the SONY IMX646 event-based image sensor, each with a resolution of 1280×720 pixels, mounted

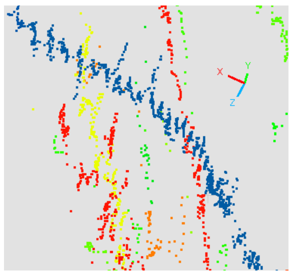


Figure 2: Part of a labeled 3D flight curve over a meadow. The depth was calculated using block matching for a 60 ms projection window. The blue trajectory clearly shows uniform calculated depth values per pixel throughout the entire time window.

on a tripod with their field of view (FoV) directed toward the target measurement area. This area spans several square meters and remains fixed throughout each recording session. In addition, RGB images of the scene are captured from various viewpoints using a video from a standard smartphone camera. These images are subsequently used to reconstruct a 3D model of the measurement environment via a Structure-from-Motion pipeline.

A schematic overview of this recording setup is provided in Figure 1. This setup was chosen because a combination of only one event camera and a RGB camera, as used in (Gebauer et al., 2024), does not allow depth estimation for small insects such as wild bees, since they are not present in the RGB data.

2.1 Event-based Vision Domain: Depth Estimation

Stereo event-based camera setups, such as the one used in this work, follow a classical hardware configuration that enables depth estimation through spatial disparity. Such systems have been adopted in a broad range of application scenarios, including autonomous driving and robotics.

A recent survey (Ghosh and Gallego, 2025) provides a structured overview of stereo depth estimation techniques based on event data, including image-reconstruction, time-correlation, and learning-based approaches. Due to the extremely high flight speeds of insects, encoding events in time-surface[1] image data over short intervals (e.g., 60 ms) for block matching (Konolige, 1998) often results in fragmented flight trajectories, as illustrated in Figure 2. This limitation arises because correlation-based standard methods, such as block matching, compute only a single depth value per projected pixel. While using smaller,

[1] https://docs.prophesee.ai/stable/tutorials/ml/data_processing/event_preprocessing_python_legacy.html#time-surface

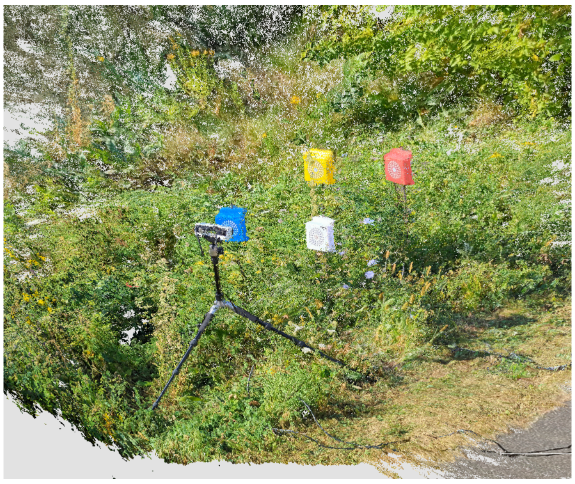


Figure 3: Point cloud of the scene generated with Agisoft Metashape Pro.

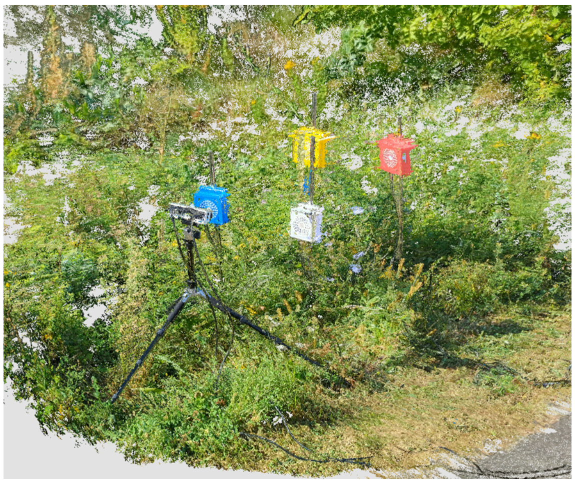


Figure 4: Point cloud of the scene generated with COLMAP.

overlapping time windows could mitigate this issue, it would also greatly increase computational complexity.

Similarly, it is also unreliable to depend solely on timestamp correlation (Rogister et al., 2012). When strong plant movements generate a large number of simultaneous events, the limited capacity of the sensor's output bus introduces timestamp delays, leading to relative misalignment even between otherwise synchronized cameras.

Finally, the third approach commonly discussed in the literature, using learning-based methods, e.g. (Cho and Yoon, 2022; Chen et al., 2024), is also impractical due to the lack of suitable training data. Together, these challenges highlight the limitations of conventional and learning-based approaches for accurately capturing high-speed insect flight. Therefore, in Section 3.1.2, we present an approach that is directly designed to determining depth in insect flight paths.

2.2 Frame-based RGB Vision Domain: Structure-from-Motion

Structure-from-Motion is a well-established technique for reconstructing 3D scene geometry from multiple overlapping 2D images (Özyeşil et al., 2017). In this work, videos of the measurement area are recorded, and RGB images from various viewpoints are extracted from them, enabling the generation of a detailed 3D point cloud.

For the reconstruction process, various open-source software packages (e.g. OpenDroneMap[2], VisualSfM[3], COLMAP[4], and OpenMVG[5]) as well as commercial software (e.g. Agisoft Metashape Pro[6]) can be used. Some of these packages did not produce usable results in our tests because the reconstructed point cloud broke up into many non-matching models due to slight wind movement of the plants. However, Agisoft Metashape Pro and COLMAP produced high-quality results (see Figure 3 and Figure 4). We used Agisoft Metashape Pro for our example implementation because it showed slightly fewer disturbances in the output (see the incorrect blue structures between the white and yellow box in the point cloud generated with COLMAP).

2.3 Visualization & Registration

The integration of monitoring data into a 3D model enables effective visualization and opens up new possibilities for meaningful interpretation and analysis. Such approaches are frequently applied in urban development, for instance by embedding building models within SfM data (Iheaturu et al., 2022). Another promising field of application is the visualization of designated air corridors for drone operations (Kamal et al., 2020). Finally, 3D representations are increasingly used for the visual exploration and analysis of climate networks (Nocke et al., 2015).

If the data to be visualized, such as the extracted insect monitoring results and the 3D point clouds generated with SfM in our case, originate from fundamentally different paradigms and data sources, an

[2]https://www.opendronemap.org/

[3]http://ccwu.me/vsfm/

[4]https://colmap.github.io/

[5]https://github.com/openMVG/openMVG

[6]https://www.agisoft.com/

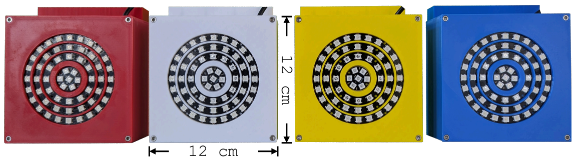


Figure 5: Active LED-Marker used for calibration.

alignment step is required, commonly referred to as registration.

While fully automated registration methods exist, such as the recent learning-based approach in (Lin et al., 2023), our work opts for the use of physical markers placed within the measurement scene. This approach offers straightforward integration into the monitoring process during an initial calibration phase and ensures high-precision alignment.

A comprehensive overview of marker systems for event-based vision is provided in (Tofighi et al., 2025). For instance, active LED markers have been successfully employed for low-latency and real-time tracking (Censi et al., 2013; Ebmer et al., 2024).

Instead of combining blinking patterns with 2D marker patterns as proposed in (Kitade et al., 2024), we utilize another hybrid approach: blink patterns detected by the event camera paired with color and shape cues identified in the RGB domain, which facilitates detection in both the event data and the SfM-generated point cloud.

3 DATA PROCESSING PIPELINE FOR VISUALIZATION

The physical markers used in the measurement setup consist of 3D-printed, monochromatic enclosures measuring $12 \times 12 \times 5$ cm. Each marker integrates four flush-mounted LED rings on its front face. These rings are equipped with 24, 16, 12, and 7 WS2812 LEDs, arranged concentrically from the outer to the inner ring. Figure 5 illustrates the marker design used in this study.

The following sections describe the processing steps required to generate the desired visualizations. A comprehensive representation of the complete workflow is provided in Figure 6.

For reproducibility, a reference implementation covering the configuration, detection, and extraction of the LED-marker reference points is available online[7].

[7]https://gitlab.com/hsnr-insect/dvs_blink_marker

Table 1: Timing parameters of active LED-Markers (values are given in milliseconds).

Marker Color	Cycle Time ON / OFF	Cycle Time ON-to-ON	Ring Delay
RED	268.5	537	55
BLUE	333.5	667	70
YELLOW	441.5	883	85
WHITE	551.5	1103	100

3.1 Event-based Vision Domain

3.1.1 Active LED Marker

Configuration: As the scene remains static during calibration, except for the markers themselves, high-frequency blinking patterns at kilohertz-level are unnecessary. The marker system was therefore designed to operate at lower update rates using a standard Arduino-based microcontroller and comparatively slow WS2812 LEDs, which operate with limited update rates due to their serial control protocol.

The implemented blinking patterns follow a bullseye-inspired configuration, with each ring operating independently. The blinking sequence initiates at the outermost ring and progresses inward toward the center, creating a dynamic animation that emphasizes the concentric structure of the marker. To improve temporal separability in the recorded event data, adjacent LED rings were programmed to avoid simultaneous transitions between ON and OFF states. Each marker was assigned a unique blinking cycle, characterized by (a) the total cycle duration (from one ON transition to the next ON transition of the same ring), and (b) the delay between the activations of adjacent rings within the same marker.

The timing parameters used for each marker are summarized in Table 1.

Detection: The recorded event stream includes events not originating from the active marker system, even when the event cameras are mounted on a static tripod. In addition to potential scene disturbances such as minor plant or insect motion during calibration, the raw output of event-based sensors typically contains background activity caused by sensor noise rather than actual luminance changes in the scene (see Figure 7a). To ensure robust marker detection, the first processing step involves a spatio-temporal filtering of the event stream.

The event data is first filtered by polarity. Only events with positive polarity ($p = 1$) are retained, as these correspond to positive contrast changes from darker to brighter image regions and are associated with the ON transitions of the LED blinking pattern.

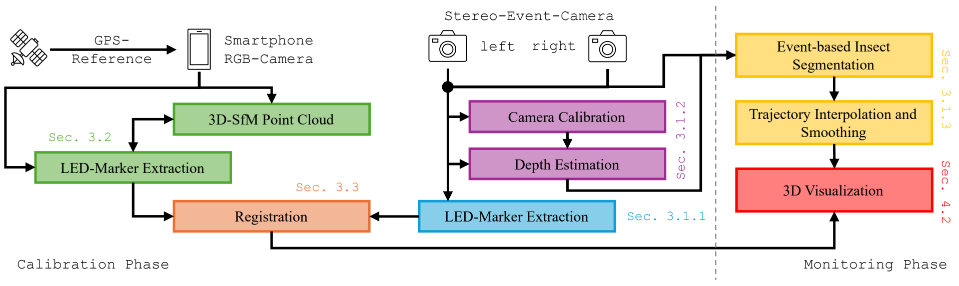
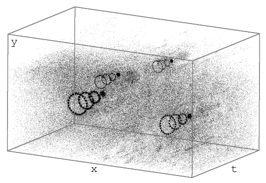
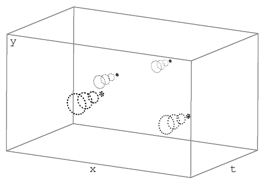


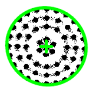
Figure 6: Processing workflow.



(a) Raw ON-Events



(b) Filtered ON-Events



(c) *xy*-projection of a single marker with fitted ellipse

Figure 7: Triggered Events (shown are 800 ms).

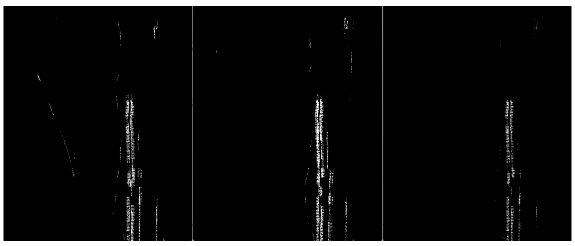


Figure 8: Example of a projection image of events within 1024 milliseconds from the left camera (left), the right camera (center), and the result with all projected events triggered by both cameras.

For each retained event, a local spatio-temporal density check is applied. Events are preserved only if at least six other supporting events occur within their 8-connected spatial neighborhood and within a temporal window of 0.5 ms. This filtering step preserves coherent clusters related to LED activations while suppressing isolated noise and non-marker activity (compare to Figure 7b).

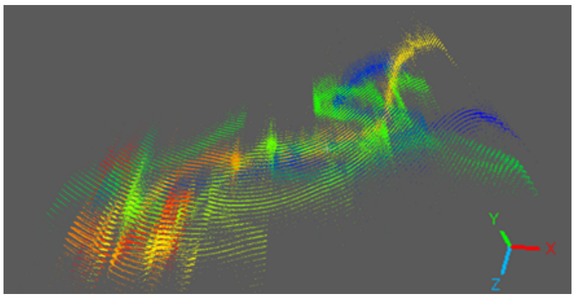
For the remaining event stream, a timestamp buffer is constructed and updated during the processing of each filtered event. This buffer records only the timestamp of the most recent event at each spatial location. By evaluating the temporal differences between consecutive events in this timestamp buffer and comparing them to the configured `ON`-to-`ON` cycle durations of the LED markers, a unique marker assignment is established. The temporal coding is sufficiently robust that explicit consideration of the configured inter-ring delays has not been necessary. For events successfully assigned to a marker, ellipses are fitted to the corresponding event sets in the two-dimensional *xy*-projection (see Figure 7c). This allows reliable estimation of the marker center positions with respect to the individual FoV of each sensor.

3.1.2 Camera Calibration & Depth Estimation

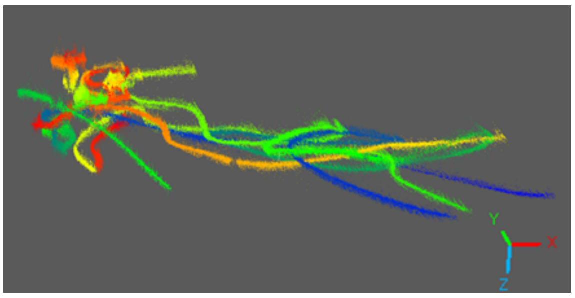
Calibration: Accurate stereo calibration of the cameras is essential for extracting depth information from two stereo images. During this process, the intrinsic and extrinsic parameters of both cameras are estimated. Our applied calibration procedure utilizes a checkerboard pattern that is moved in a controlled manner in front of the cameras. From the recorded event streams, a grayscale video is reconstructed using a neural network (Muglikar et al., 2021). Based on this reconstructed video, the actual calibration is performed using MATLAB and OpenCV calibration functions, enabling the precise determination of the intrinsic and extrinsic parameters of the stereo setup.

Depth-Estimation and Log-Scale: The derived camera matrices are used to rectify the *x* and *y* coordinates in the segmented *xyt* point clouds from the left and right cameras and to align the resulting event streams along the *y*-axis. This geometric alignment significantly simplifies the subsequent correspondence search between individual events. To begin processing, the events from the two point clouds are projected sequentially into two separate *yt* images, using a time window of 1024 ms for *t*. These projection images are then analyzed for overlapping points, which are considered candidates for corresponding points (see Figure Figure 8).

If a unique pair of candidates is found, it is directly used for the depth computation. If multiple potential correspondences exist, an *xt*-representation of a 19×19 neighborhood around each candidate point is generated. The similarity between points is then



(a) Standard transformation (following Equation (1))



(b) Logarithmic transformation (following Equation (3))

Figure 9: 3D point cloud after segmentation and integrated depth calculation (colors corresponds to different points in time).

evaluated based on the number of overlapping pixels in these regions. When several possible combinations occur, for instance, two potential points in the left camera and four in the right camera (resulting in a maximum of two possible pairings), the combination with the highest similarity is selected. If the calculated similarity exceeds a predefined threshold (in our application, a value corresponding to five overlaps), the associated point pairs are accepted for depth computation. Figure 9a shows the result for a sample point cloud in 3D.

When using standard stereo geometry, the calculated depth value z depends on the disparity d, the focal length of the camera f and the distance between the cameras b as follows

$$z = \frac{f \cdot b}{d} \tag{1}$$

In our case, b is 12.5 cm. Depending on the distance between the insect and the camera, a disparity difference of one pixel has a different effect on the calculated depth value. This change can be calculated using

$$\Delta z = \frac{f \cdot b}{d \cdot (d+1)} \tag{2}$$

In our setup, a disparity of 300 pixels corresponds to a distance of approximately $43\,cm$ from the camera. In this case, a change in disparity of one pixel causes a change in the depth value of $1.42\,mm$. With a disparity of 64 pixels, corresponding to a depth value of approximately $2\,m$, a change of one pixel affects the depth value by approximately $3.1\,cm$. At a camera distance of 4 m, the depth values change by approximately $12\,cm$. This makes it difficult to identify coherent trajectories in the data. To achieve a roughly equal change in depth, we calculate a modified z-value using the following equation:

$$z_{mod} = f \cdot b \cdot log(d) \tag{3}$$

This results in an approximately constant value for

$$\begin{aligned} \Delta z &= f \cdot b \cdot (\log(d+1) - \log(d)) \\ &= f \cdot b \cdot \log\left(\frac{d+1}{d}\right) \end{aligned} \tag{4}$$

To achieve consistent results, the x and y coordinates must also be changed as follows

$$x_{mod} = \frac{(x - c_x) \cdot z_{mod}}{f} \tag{5}$$

and

$$y_{mod} = \frac{(y - c_y) \cdot z_{mod}}{f}, \tag{6}$$

with c_x and c_y as coordinates of the image center point. As can be seen in Figure 9b, this transformation results in denser trajectories, particularly for curves that are further away from the camera.

3.1.3 Segmentation & Insect Trajectories

Segmentation: Although event cameras exclusively capture moving objects, a segmentation step is still required to differentiate between events generated by moving plants and those induced by insects. To this end, numerous algorithms have been proposed for the semantic segmentation of event data. In our pipeline, we use a U-Net with tiling to segment insect events, following the approach described by Gebler (Gebler et al., 2026).

Flight Trajectory Processing: In the subsequent step, the results of semantic segmentation, together with the computed logarithmically scaled depth data, are utilized for instance segmentation, in which the events are decomposed into individual trajectories. In our pipeline, we apply an adapted version of the algorithm originally proposed by Arning (Arning et al., 2025), which has been extended to four dimensions (x, y, z, t) for our application.

For visualization, individual trajectories are converted into continuous 3D curves using the unscaled 3D coordinates. First, a moving median filtering is applied to produce a consistent curve structure from the widely scattered point clouds. Then, a seven-point moving average is used to smooth the curve points along the time axis.

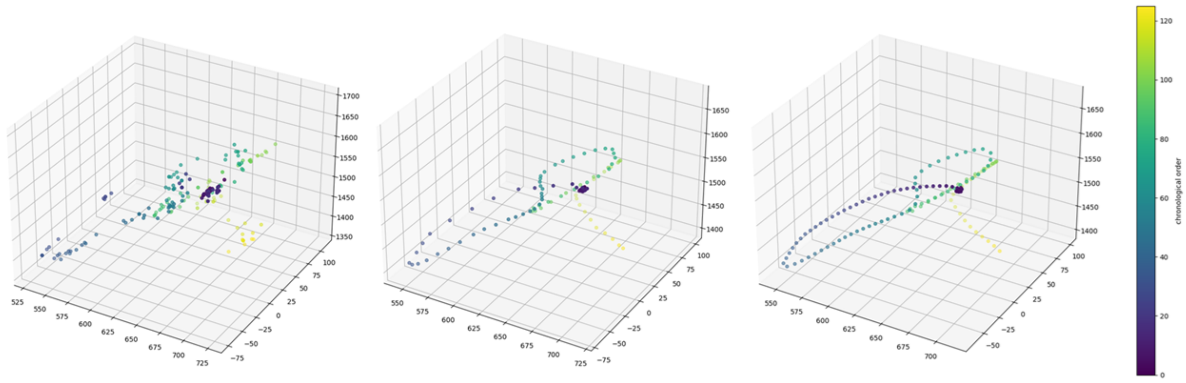


Figure 10: Steps from the point cloud (left) to a single smoothed trajectory (center) to a uniformly sampled 3D curve (right).

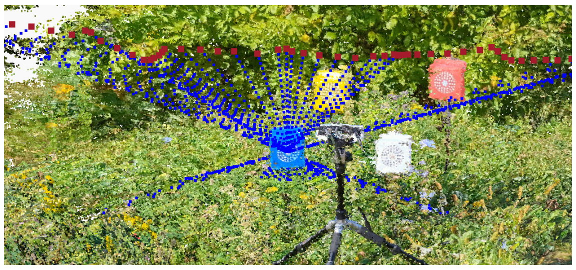


Figure 11: Examples of identified rays passing through the center of the `BLUE` marker (local camera positions are shown in dark red).

Subsequently, the resulting curve points are further smoothed using a second-order Savitzky–Golay filter, reducing local noise while preserving the overall movement dynamics. The cumulative curve length is then calculated to enable uniform spatial resampling of the trajectory. Finally, new points are interpolated along this length axis, resulting in a trajectory with evenly spaced points. This uniform spatial distribution is necessary for the later calculation of curve characteristics, such as curvature. The results of the described steps can be seen for an example curve in Figure 10.

3.2 Frame-based RGB Vision Domain: Marker Extraction & Processing

The marker extraction process is performed on the 2D source images that are captured from the recorded videos and used to generate the SfM model. The results are subsequently mapped into the 3D coordinate system of the generated point cloud.

For each input image, a color-based segmentation is carried out in the HSV color space to isolate the regions corresponding to the four individual markers. This approach uses pre-defined color thresholds, exploiting the monochromatic design of the LED marker enclosures. The color segmentation results are then post-processed using morphological operations. An opening step is applied to remove small unwanted speckles, such as background elements (e.g., flowers) that share similar hues to the markers, followed by a closing step to fill holes within the color masks in order to make the detected marker regions solid and continuous.

For further analysis, each extracted marker region is cropped into a separate image patch in order to determine the marker center. This is achieved by applying the symmetry detection algorithm outlined in (Dalitz et al., 2019). Considering that the markers incorporate concentric LED rings, the center of each marker corresponds to the origin of a rotational symmetry. Therefore, the image point ranked with the highest rotational symmetry score above a predefined threshold is considered to be the marker's center and stored for subsequent processing.

As already mentioned, the SfM processing is carried out using the commercially available software Agisoft Metashape Pro, which, by providing GPS metadata embedded in the input images, is able to generate a correctly scaled 3D model. In addition to the model, estimates of the intrinsic camera calibration parameters for the smartphone camera used to capture the images, as well as the camera transformation matrices (i.e., rotation and translation) for each input image, can be exported.

Finally, the combination of the camera parameters with the extracted marker center points from multiple 2D images enables the triangulation of the marker centers within the 3D world coordinate system of the generated model. This process is illustrated in Figure 11.

3.3 Registration: Merging Modalities

After obtaining the point correspondences from the marker segmentation in both the event-based point cloud and the SfM point cloud, the subsequent step involves computing the transformation matrix for registration. This transformation encapsulates translation, rotation, and scaling, and is optimized to minimize the mean squared error between corresponding points. The computation is performed using MATLAB's `procrustes` function, which implements a generalized Procrustes analysis to determine the best-fit transformation.

4 EXPERIMENT RESULTS

To assess the limitations of human observation, we first conduct an eye-tracking experiment comparing human perception with event-based recordings. We subsequently present visualizations derived from insect monitoring data to illustrate the capabilities of our approach.

4.1 Comparison: Eye-Tracking

As an initial step, we compared insect activity recorded by the event camera with what was perceived by humans. For this, we used Tobii eye-tracking glasses (Onkhar et al., 2024), which provide gaze data along with a synchronized RGB video.

To allow visualization of gaze data despite moderate head movements, the gaze data needs to be mapped into a common coordinate system. As we aimed to approximate the event camera's perspective, a SIFT based alignment of video frames to a reference frame from the eye-tracking video proved sufficient given the recording's limited head motion. A more robust method that uses additional fiducial markers is described in (Niehorster et al., 2025), but this added complexity was unnecessary for our setup.

To enable alignment of the events of the event camera with the reference image, they were integrated over a short temporal window without filtering, ensuring that background structures remained visible for matching. Temporal information was encoded using a blue-green-yellow-red color scale. Figure 12 shows that human observers can actively track only a small number of trajectories. One detected example is a butterfly in the top right of the image, but most insect activity went unnoticed. Moreover, fast motion make individual insects hard to follow by eye, and participants were repeatedly distracted by moving plants in the foreground, drawing their gaze to this area.

It should be noted that the eye-tracking camera only worked reliably in the shade. Under brighter conditions, insect activity would likely be higher, which would make human detection even less effective, motivating the need for automated tracking.

4.2 Visualization Results

Quality: The registration accuracy of the four markers can be used to estimate the accuracy of our method. For our calculations, the centers of the markers were between 0.71 m and 1.130 m from the camera. Due to the inaccuracy in depth determination for more distant objects, as described in the Section 3.1.2, there was a mean Euclidean deviation of 5 cm (1.27 cm, 3.57 cm, 7.54 cm, 7.85 cm) between the markers in the SfM and event point clouds after registration. The deviations in distance between the marker boxes and the camera, as measured manually before recording, are of the same order of magnitude. However, this deviation is similar in magnitude to the movement of flowers on long-stemmed plants in the wind.

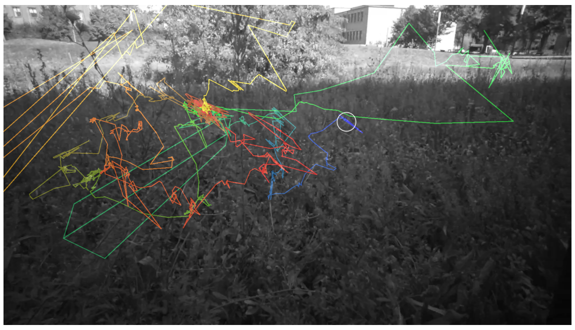
Time required: Currently, the registration process takes only a few minutes longer due to the additional capture of the marker data. Because the amount of data is small, its evaluation also takes only a few minutes. This time is negligible compared to generating the SfM point cloud and evaluating the event data. The time required to generate the SfM point cloud depends on the tool used and is strongly influenced by the number of photos. Likewise, the time needed to analyze the event data scales significantly with wind movement and insect activity, as both increase the number of events that must be processed.

Trajectory Approximation: Figure 13a illustrates the insect's flight over a meadow for approximately 10 minutes, while Figure 13b shows the same scene with flight paths compressed into curves.

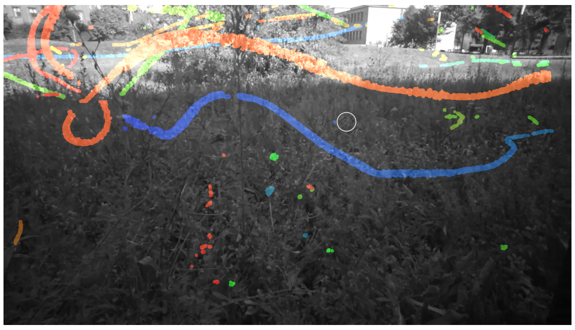
Individual Flight Paths: Our method supports diverse analyses. For instance, it allows us to examine the flight paths of individual insects to gain insights into the chronological sequence of their flower visits (Figure 14 and Figure 15).

Additionally, we can isolate all flight paths associated with a specific flower for further investigation, e.g., to quantify visitor numbers over time. For this purpose, the flower positions are manually annotated. Flower visits are then estimated based on the spatial proximity of the compressed flight paths to the flowers in the SfM point cloud.

The trajectory points around manually located flowers are determined using a *KD-tree* (Bentley,

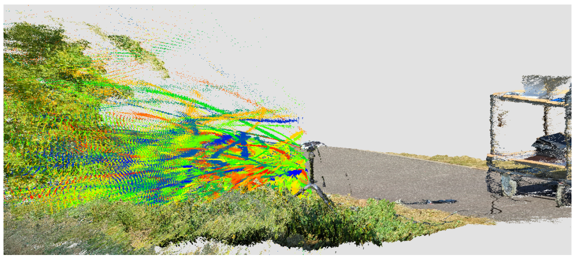


(a) Eye-tracking gaze data projected into a reference image

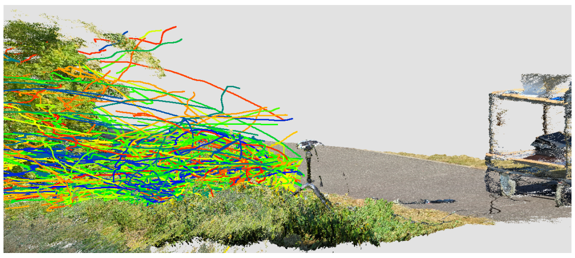


(b) Semantically segmented event camera data

Figure 12: Comparison of human eye tracking and event camera data. Using a BGYR color scale to encode time.



(a) All segmented trajectory points



(b) Trajectories approximated by curves

Figure 13: Visualization of the insect's flight over a meadow for about 10 minutes. The timing of each flight within this interval is encoded using a linear rainbow color scale, ranging from blue through green to red.

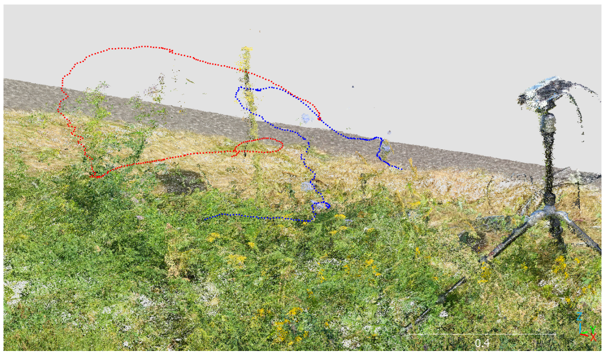


Figure 14: The example shows the paths taken by two insects as they move from flower to flower. The color represents the two different instances.

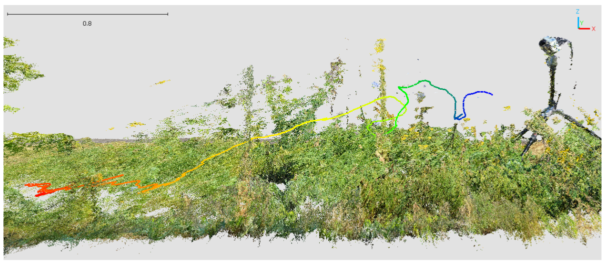
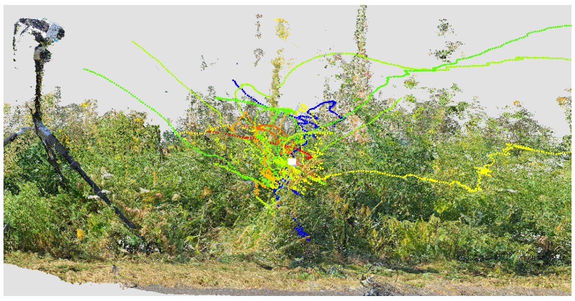


Figure 15: The example shows the path of an insect. The color encodes the timestamp.


(a) Chicory flower (highlighted by white square)

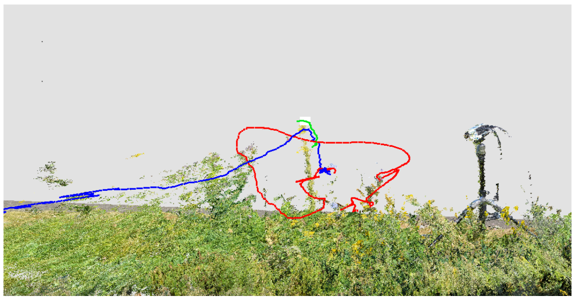

(b) Mullein flower (highlighted by white square)

Figure 16: Paths of all insects that visit a selected flower within 10 minutes. The color indicates the timestamp.

1975) in the (x,y,z) point cloud. All trajectories containing points within a radius of 10 cm around a flower (accounting for wind-induced movement and depth estimation errors) are subsequently considered visiting trajectories for the visualizations. Results for two different flowers (chicory and mullein) are shown in Figure 16a and Figure 16b.

5 CONCLUSION AND FUTURE WORK

In this paper, we present a method that combines insect flight paths recorded with an event camera and corresponding environmental point clouds reconstructed from mobile phone videos using SfM. To align these two data sources, we employ a marker-based registration approach. In the event data, we record distinct flashing patterns that correspond to the colours of reference boxes visible in the video data. After segmenting the marker positions, we compute the optimal transformation matrix to accurately register both data sources.

This integrated representation allows for deeper insights into insect behaviour and insect-plant-interactions. For example, we can determine the sequence of flower visits made by an individual insect or quantify how many insects visit a particular flower within a given time frame. In future work, we plan to extend the pipeline by incorporating insect classification into distinct groups. Furthermore, we will analyse additional characteristics of the flight trajectories, such as flower visit duration, flight altitude, and flight speed.

A further extension of our approach will be the automatic detection and classification of flowers in the video frames, followed by the determination of their positions in the SfM point cloud, analogous to the procedure described in Section 3.2. This will enable the automatic counting of flower visits.

ACKNOWLEDGEMENTS

This work was funded by the Carl-Zeiss-Foundation as part of project BeeVision.

REFERENCES

Arning, J., Dalitz, C., and Pohle-Fröhlich, R. (2025). Separation of Insect Trajectories in Dynamic Vision Sensor Data. In *Proceedings of the 20th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications - Volume 3 VISAPP*, pages 69–77. INSTICC, SciTePress.

Bentley, J. L. (1975). Multidimensional Binary Search Trees Used for Associative Searching. *Communications of the ACM*, 18(9):509–517.

Censi, A., Strubel, J., Brandli, C., Delbruck, T., and Scaramuzza, D. (2013). Low-latency localization by Active LED Markers tracking using a Dynamic Vision Sensor. In *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 891–898.

Chen, W., Zhang, Y., Sun, X., and Wu, F. (2024). Event-Based Stereo Depth Estimation by Temporal-Spatial Context Learning. *IEEE Signal Processing Letters*, 31:1429–1433.

Cho, H. and Yoon, K.-J. (2022). Event-Image Fusion Stereo Using Cross-Modality Feature Propagation. *Proceedings of the AAAI Conference on Artificial Intelligence*, 36(1):454–462.

Dalitz, C., Wilberg, J., and Jeltsch, M. (2019). The Gradient Product Transform: An Image Filter for Symmetry Detection. *Image Processing On Line*, 9:413–431.

Ebmer, G., Loch, A., Vu, M. N., Mecca, R., Haessig, G., Hartl-Nesic, C., Vincze, M., and Kugi, A. (2024). Real-time 6-DoF Pose Estimation by an Event-based Camera using Active LED Markers. In *2024 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pages 8122–8131.

Gallego, G., Delbrück, T., Orchard, G., Bartolozzi, C., Taba, B., Censi, A., Leutenegger, S., Davison, A. J., Conradt, J., Daniilidis, K., and Scaramuzza, D. (2022). Event-Based Vision: A Survey. *IEEE Trans-*

actions on Pattern Analysis and Machine Intelligence, 44(1):154–180.

Gebauer, E., Thiele, S., Ouvrard, P., Sicard, A., and Risse, B. (2024). Towards a Dynamic Vision Sensor-Based Insect Camera Trap. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pages 7157–7166.

Gebler, C., Funk, J., Pohle-Fröhlich, R., and Wagner, A. (2026). DVS-StereoInsect: An Event-based Stereo Dataset for Foreground-Background Insect Segmentation. In Castrillón-Santana, M. et al., editors, *Computer Analysis of Images and Patterns*, pages 209–219, Cham. Springer Nature Switzerland.

Ghosh, S. and Gallego, G. (2025). Event-Based Stereo Depth Estimation: A Survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 47(10):9130–9149.

Iheaturu, C., Okolie, C., Ayodele, E., Egogo-Stanley, A., Musa, S., and Speranza, C. I. (2022). A simplified structure-from-motion photogrammetry approach for urban development analysis. *Remote sensing applications: Society and Environment*, 28:100850.

Kamal, A., Javaid, A. Y., Devabhaktuni, V. K., Kaur, D., Zaientz, J., and Marinier, R. (2020). A Novel Approach to Air Corridor Estimation and Visualization for Autonomous Multi-UAV Flights. In *2020 IEEE International Conference on Electro Information Technology (EIT)*, pages 370–377. IEEE.

Kitade, T., Yamada, W., Ochiai, K., and Imai, M. (2024). Bicode: A Hybrid Blinking Marker System for Event Cameras. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2939–2945.

Konolige, K. (1998). Small Vision Systems: Hardware and Implementation. In Shirai, Y. and Hirose, S., editors, *Robotics Research*, pages 203–212. Springer.

Landmann, T., Schmitt, M., Ekim, B., Villinger, J., Ashiono, F., Habel, J. C., and Tonnang, H. E. (2023). Insect diversity is a good indicator of biodiversity status in Africa. *Communications Earth & Environment*, 4(1):234.

Lin, X., Qiu, C., cai, z., Shen, S., Zang, Y., Liu, W., Bian, X., Müller, M., and Wang, C. (2023). E2PNet: Event to Point Cloud Registration with Spatio-Temporal Representation Learning. In Oh, A. et al., editors, *Advances in Neural Information Processing Systems*, volume 36, pages 18076–18089. Curran Associates, Inc.

Muglikar, M., Gehrig, M., Gehrig, D., and Scaramuzza, D. (2021). How to Calibrate Your Event Camera. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 1403–1409.

Niehorster, D. C., Hessels, R. S., Nyström, M., Benjamins, J. S., and Hooge, I. T. C. (2025). gazeMapper: A tool for automated world-based analysis of gaze data from one or multiple wearable eye trackers. *Behavior Research Methods*, 57(7):188.

Nocke, T., Buschmann, S., Donges, J. F., Marwan, N., Schulz, H.-J., and Tominski, C. (2015). Review: visual analytics of climate networks. *Nonlinear Processes in Geophysics*, 22(5):545–570.

Onkhar, V., Dodou, D., and de Winter, J. C. F. (2024). Evaluating the Tobii Pro Glasses 2 and 3 in static and dynamic conditions. *Behavior Research Methods*, 56(5):4221–4238.

Özyeşil, O., Voroninski, V., Basri, R., and Singer, A. (2017). A Survey of Structure from Motion. *Acta Numerica*, 26:305–364.

Pohle-Fröhlich, R. and Bolten, T. (2023). Concept study for dynamic vision sensor based insect monitoring. In *VISIGRAPP (4: VISAPP)*, pages 411–418.

Pohle-Fröhlich, R., Gebler, C., Böge, M., Bolten, T., Gehlen, L., Glück, M., and Traynor, K. S. (2025). Features for classifying insect trajectories in event camera recordings. *Proceedings of the 20th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications - Volume 3 VISAPP*, pages 355–364.

Pohle-Fröhlich, R., Gebler, C., and Bolten, T. (2024). Stereo-Event-Camera-Technique for Insect Monitoring. In *Proceedings of the 19th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications - Volume 3 (VISAPP)*, pages 375–384. INSTICC, SciTePress.

Rogister, P., Benosman, R., Ieng, S.-H., Lichtsteiner, P., and Delbruck, T. (2012). Asynchronous Event-Based Binocular Stereo Matching. *IEEE Transactions on Neural Networks and Learning Systems*, 23(2):347–353.

Saleh, M., Ashqar, H. I., Alary, R., Bouchareb, E. M., Bouchareb, R., Dizge, N., and Balakrishnan, D. (2024). Biodiversity for ecosystem services and sustainable development goals. In *Biodiversity and Bioeconomy*, pages 81–110. Elsevier.

Tofighi, N. J., Robic, M., Morbidi, F., and Vasseur, P. (2025). A Survey on Event-based Optical Marker Systems.

Van Klink, R., Sheard, J. K., Høye, T. T., Roslin, T., Do Nascimento, L. A., and Bauer, S. (2024). Towards a toolkit for global insect biodiversity monitoring. *Philosophical Transactions of the Royal Society B*, 379(1904):20230101.