Lab session project: Event Camera Data Processing

Thijs Alens, Bernd Dockx, and Xander Verberckt

I. INTRODUCTION

During the lab sessions of 'Computationele Beeldvorming en Visualisatie', we were tasked to evaluate various techniques for processing event camera data, specifically targeting image reconstruction and object detection. Event camera data is saved in a text file where the each line contains:

- a timestamp
- an x-coordinate
- a y-coordinate
- the polarity of the pixel

II. METHODS

In this section we will discuss the methods we used to reconstruct the images using classical methods as well as AI-based approaches.

The image reconstruction process follows a relatively simple pipeline. It uses a starting image, captured with a "normal" camera. During the capturing of the video, the event camera, who can capture data much faster then a regular camera, captures data between the actual frames. Using this extra data, the original image is incrementally updated using the high-temporal-resolution data captured by the event camera. Each timestamp, all the events get accumulated, which means all the pixels with a change in polarity get updated. This process repeats itself for the whole video, which results in a video with much more frames, and therefore a much higher frame rate.

For this lab, we only used 8-bit grayscale images

A. Image Reconstruction from events using non-AI models

a) Accumulate events: The first classical approach is very simple. It just accumulates the polarities for each pixel and adds it to the original image. The implementation of this, as shown below, is also not complex. The variable "consider_polarity", determines whether the polarity of each event is factored into the reconstruction process.

```
for i in range(len(t)):
  if consider_polarity:
    if p[i] == 1:
      orig_image[y[i]-1, x[i]-1] += 1
    else:
      orig_image[y[i]-1, x[i]-1] -= 1
  else:
    orig_image[y[i]-1, x[i]-1] += 1
return orig_image
```

b) Time surface: The second approach is a bit more complex. It uses the idea of memory decay, which comes down to dynamically updating the pixels according to a decay-function (in our case an exponential function). Every event that occurs on a set pixel location updates the pixel, after which its effect decays, according to the function. This way the most recent events are more resound then the older ones. The code for this can be found below.

```
# Initialize the time surface
time_surface = np.zeros(image_size,
                        dtype=np.int8)
current_time = max(t) if t else 0

last_event_time = np.full(image_size, -np.inf)

# Update the last event timestamp
for each pixel
for i in range(len(t)):
    last_event_time[y[i], x[i]] = t[i]

# Compute time difference and apply decay
delta_t = current_time - last_event_time
# Ignore pixels with no events
delta_t[delta_t < 0] = np.inf
time_surface = decay_function(delta_t)

return time_surface
```

B. Image Reconstruction from Events Using AI Models

The implementation of this method utilized publicly available code accompanying the study by Zhang et al. [1], adapted to operate on our custom event data. To get everything to work, we had to add an flow.np file to tell the model in which ways the camera was moving and/or rotating. Their approach formulates event-based image reconstruction as a linear inverse problem, integrating classical and learning-based regularization techniques. By leveraging optical flow information, the method reconstructs intensity images from event data without relying on recurrent neural networks, thus enhancing explainability and reducing computational complexity.

To reconstruct an image from event-based data, two key inputs are required:

- Event data captured by the sensor
- A flow file that describes the motion of the camera

In contrast to traditional (non-AI) reconstruction methods, this approach does not rely on an initial image. Instead, it directly utilizes the event data and the camera motion

information to synthesize the final image. A crucial dependency of this method is the accuracy of the flow file: if the estimated motion deviates significantly from the actual camera trajectory, the reconstructed image will suffer from noise and artifacts.

We explored multiple AI models for this task; however, most implementations were incomplete or failed to run due to missing files or denoising components, despite claims of open-source availability in the respective research papers.

C. Object Detection Using Event Data

As discussed in the section on AI-based image reconstruction, much of the research in this area claims to be open source, but in practice many models and required files are inaccessible or incomplete. Due to these limitations, we opted not to apply object detection directly on raw event data. Instead, we developed a pipeline that first reconstructs intensity images from event streams, and then applies conventional object detection techniques to those images. We experimented with several YOLO models on the reconstructed outputs and ultimately selected YOLOv11x based on its performance. For inference, we employed the open-source Ultralytics implementation [2].

III. RESULTS

This section presents the outcomes of the different event data processing methods evaluated in the lab sessions, including both classical and AI-based image reconstruction, as well as object detection performance.

A. Image Reconstruction from Events Using Non-AI Models

The reconstruction method based on accumulated polarities produces images in which key subjects are clearly distinguishable, and the results closely resemble the ground truth. However, both the reconstructed and ground truth images exhibit a considerable amount of noise. While this classical approach effectively preserves image structure, it lacks mechanisms for suppressing noise.

In contrast, the AI-based reconstruction model we used [1] incorporates noise suppression techniques as part of its pipeline. As a direction for future work, it may be valuable to explore the integration of similar noise reduction strategies into the non-AI reconstruction method to improve visual quality.

B. Image Reconstruction from events using AI-models

The method returns images in which the subjects are clearly visible and identifiable. Although the image is clearly different from the ground truth image. The image contains much less noise when compared to the ground truth image, but it also contains much less sharpness. Objects with the same color seem to get merged together, such as a white book laying on a white desk.

The AI-based reconstruction method also produces visually coherent images, albeit with distinct characteristics when compared to the ground truth. Specifically, the output exhibits substantially reduced noise levels; however, this is accompanied by a loss of detail and sharpness. Homogeneous regions, such as a white book placed on a white desk, tend to merge. This suggests a trade-off between denoising and the preservation of fine spatial features.

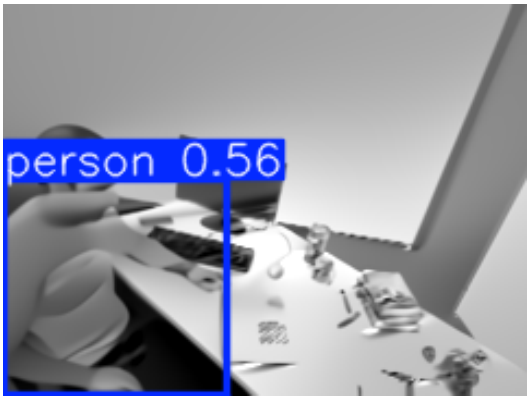

Fig. 1. YOLO detection on the reconstruced image by AI

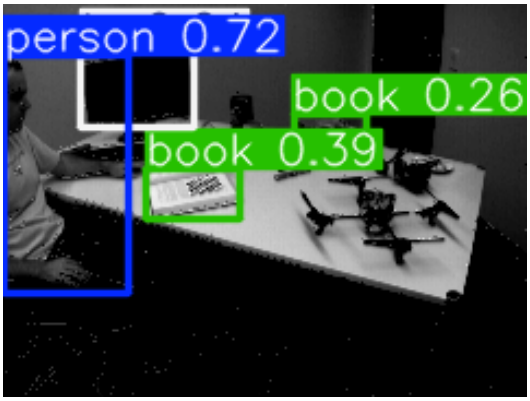

Fig. 2. YOLO detection on the reconstruced image using accumulated events

C. object detection using event data

Object detection was performed on reconstructed images using the YOLOv11x model. While detection results were generally correct, the associated confidence scores were lower than those typically achieved with standard frame-based images under well-lit conditions. Notably, the image reconstructed via accumulated events yielded higher detection confidences than the AI-reconstructed counterpart. This discrepancy may stem from the blurring and homogenization effects introduced by the AI model. Nevertheless, event cameras retain advantages in low-light scenarios, where traditional cameras often under perform.

IV. CONCLUSION

The lab sessions provided valuable insights into the functionality and advantages of event-based cameras, as well as current methodologies for processing their data. The exploration of both classical and AI-driven techniques highlighted the trade-offs between simplicity and performance. Notably, the AI-based reconstruction approach, combined with YOLO-based object detection, demonstrated promising

results despite challenges such as motion estimation accuracy. These findings clarified the potential of event cameras in dynamic and low-light environments, suggesting further opportunities for research in real-time applications.

REFERENCES

[1] Z. Zhang, A. Yezzi, and G. Gallego, "Formulating event-based image reconstruction as a linear inverse problem with deep regularization using optical flow," *IEEE Trans. Pattern Anal. Mach. Intell.*, 2022.

[2] G. Jocher and J. Qiu, "Ultralytics yolo11," 2024.

APPENDIX

All our code can be found in the following git repository: `https://github.com/ThijsAlens/computationele_beeldvorming.git`