Applied Computer Systems
ISSN 2255-8691 (online)
ISSN 2255-8683 (print)
December 2021, vol. 26, no. 2, pp. 173–177
https://doi.org/10.2478/acss-2021-0021
https://content.sciendo.com

The Process of Data Validation and Formatting for an Event-Based Vision Dataset in Agricultural Environments

Maris Galauskis[1*], Arturs Ardavs[2]
[1,2] *Riga Technical University, Riga, Latvia*

***Abstract* – In this paper, we describe our team's data processing practice for an event-based camera dataset. In addition to the event-based camera data, the Agri-EBV dataset contains data from LIDAR, RGB, depth cameras, temperature, moisture, and atmospheric pressure sensors. We describe data transfer from a platform, automatic and manual validation of data quality, conversions to multiple formats, and structuring of the final data. Accurate time offset estimation between sensors achieved in the dataset uses IMU data generated by purposeful movements of the sensor platform. Therefore, we also outline partitioning of the data and time alignment calculation during post-processing.**

***Keywords* – Dataset creation, event-based vision, neuromorphic vision dataset.**

I. Introduction

When it comes to visual sensing in robotics, conventional frame-based cameras have been an industry standard for decades [1]. However, as with many significant inventions, it is often beneficial to seek inspiration from nature. In recent years, a new camera technology inspired by the working principles of the retina has emerged, promising significant improvements in key computer vision weaknesses [2]. A dynamic vision sensor (DVS), also known as an event-based camera, generates single-pixel light intensity change events instead of full frames. Each event contains information about x and y coordinates of the pixel, a timestamp, and the light intensity change polarity. The asynchronous nature of the DVS allows it to achieve microsecond-level latency as an event is sent out almost instantly. Furthermore, event cameras have a high dynamic range, low power consumption, take up less bandwidth, and require less processing power than standard cameras. DVSs generate next to no events at a standstill and are less affected by motion blur than frame-based cameras [3].

Since event cameras are a novel technology, most modern algorithms have not yet been adapted for the asynchronous nature of the DVS sensors [4]. Event-based cameras are expensive or difficult to obtain. Therefore, several datasets have been created for researchers to develop and test event-based algorithms. Datasets are useful since they provide a reproducible environment, so the input data do not change between algorithm updates and iterations. Also, researchers do not necessarily have to invest in expensive equipment to test out their ideas, allowing them to iterate quickly and effectively and reduce the potential cost of research in event-based vision.

There are multiple event-based and conventional computer vision datasets available online. Among the more popular ones, along with the general description of the datasets in their accompanying papers, researchers mainly concentrate on the sensor calibration process [1], [3]–[6], time synchronisation [5], [7], and ground truth generation [4], [8]. In contrast, the data preparation, processing, and validation processes are not so extensively discussed in research papers; the few available articles on the topic include [7] and [8].

Our team has created a dataset for researchers interested in agricultural environment event-based camera data or event-based data in general [9]. The dataset includes DVS, LIDAR, RGB, and depth data recorded during remote-controlled wheeled platform movements in various agricultural scenes.

Creating such a dataset involves several tasks: designing a time synchronisation method between sensors, designing and building a recording equipment bundle, scheduling various locations for recording, and post-processing the obtained data to ensure their quality before making them available publicly.

In this paper, we describe our team's approach to dealing with the last step of dataset creation, i.e., data processing. The need for human input for actions like visual validation of the data and, specifically in our case, the time offset calculation between the sensors hinders a fully automated workflow. Therefore, we created a semi-automated solution involving both human input and automated software.

A. Description of the Dataset

The Agri-EBV-autumn dataset [10] contains data sequences acquired in various agricultural environments. Each of the data sequences includes 1–4 min of the actual recording of the environment with OS-1 LIDAR, DVS240 event-based camera, as well as Realsense (RS) RGB and depth cameras. A sample frame with data from each of the sensors is shown in Fig. 1. We also provide metadata with recording visualisations and supplementary data subsets that offer additional data for sensor calibration and time alignment. The dataset is intended

* Corresponding author's e-mail: Maris.Galauskis@rtu.lv

primarily as a source of DVS event data with supplementary visual data from the RGB camera and depth information from the LIDAR and the depth camera. Additionally, basic information about the environment is recorded, such as temperature, humidity and air pressure. IMU data of the LIDAR, Realsense, and DVS are used for time alignment estimation between the sensors. All the recorded sequences use trajectories with loop-closure to allow for evaluation of SLAM algorithms. The initial version of the dataset consists of 21 data sequences recorded in autumn. The complete dataset will include sequences for all four seasons. We also provide all the recorded data in rosbag format for the convenience of ROS users.

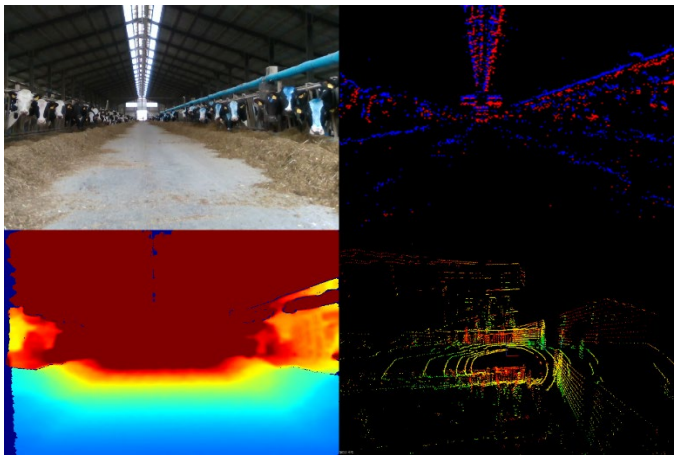

Fig. 1. Sample of RGB, DVS, Depth, and LIDAR data.

II. Data Processing

A. Overview

The general outline of the data processing is illustrated in Fig. 2. We tried to automate and batch-process most of the operations since there are 21 recordings in the first Autumn section of the dataset alone, and we are going to add Winter, Spring and Summer sections as well.

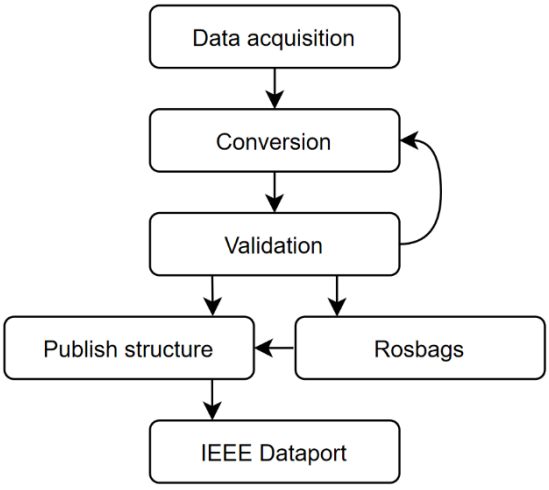

Fig. 2. The main data processing steps.

B. Data Acquisition from Sensor Bundle

The data sequences were recorded using a custom robot platform with a sensor bundle mounted on top of it, as shown in Fig. 3. External hard drives are used to facilitate the copying

Fig. 3. Platform with sensor bundle mounted during recording.

process. Therefore, the sensor bundle was only used for data acquisition, and all the necessary data processing happened afterwards on another computer. To automate the process and thus minimise potential human errors, a python script is used for copying the data from the sensor bundle to a local computer. The software searches for log files generated by the recorder program to identify recordings.

C. Conversion from Binary to Readable Data

All the data from the sensors are recorded into separate raw binary files to eliminate conversion operations for the onboard computer. Therefore, the first step in data processing is to convert the raw binary data to commonly used individual data formats for each recorded sensor (see Table I). This process is summarised in the following graph (Fig. 4).

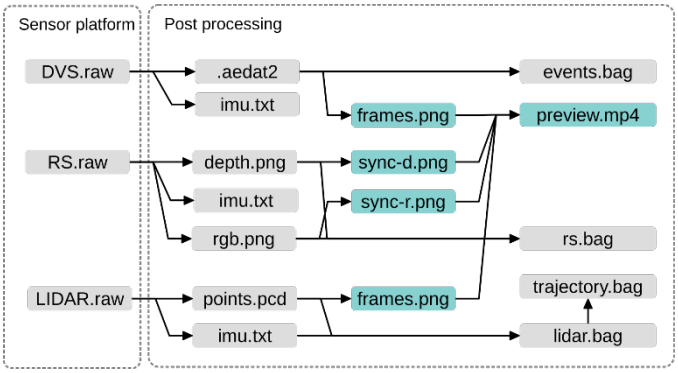

Fig. 4. File formats used during dataset processing.

A dedicated C++ converter program for each sensor type reads a single raw data file generated by the sensor and exports

the data to a respective standard file format. This process is automated by a python script that systematically launches those converters and captures their outputs to determine whether a conversion was successful. A log file is generated and updated after each step to keep track of the progress in case of the main python script failure.

Our typical recording contains four distinct parts, as shown in Fig. 5. These parts are separated during post-processing. There are two periods for time alignment – b and d, a sequence for LIDAR-RGB extrinsic calibration – a, and the main recording that contains data recorded during the platform motion in each recording – c.

For each C++ converter, it is possible to select a start and an end time to export only a particular part of the binary file. This mechanism is used to partition the recording into corresponding sections.

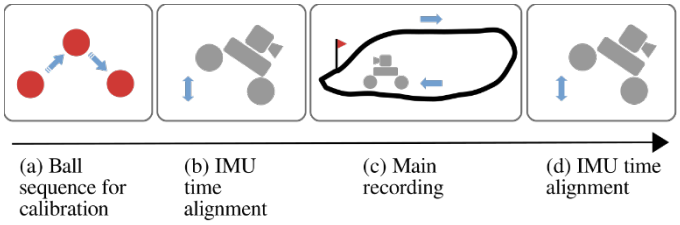

Fig. 5. Time dimension structure of each recording.

We first estimate approximate time offsets between the sensors by comparing absolute time values of the DVS, Realsense and LIDAR. It is necessary because not all the sensors use absolute time – for example, the DVS starts its time count after the reset command at startup. Then, we manually examine the DVS IMU gyroscope data graphs and note the start and end times of both platform tilting/lowering procedures. After that, the time alignment script uses estimates of time offsets to match the corresponding periods from the LIDAR and RS IMU gyroscope data. Accurate time offset values are obtained by finding a time offset with maximum cross-correlation value within a predetermined search window. LIDAR and RS gyroscope data are re-sampled to match the DVS gyroscope sample rate, which is 1 kHz. The time drift between these sensors is calculated by comparing the time between two controlled platform tilting events, the first one before the main recording and the second one shortly after.

To produce separate data for RGB – LIDAR extrinsic calibration, we manually browse RGB frames to visually find the start and end frame numbers of this part of the recording. The time alignment script correlates these frame numbers to their time values according to the LIDAR time conversion coefficients found by the time offset calculation.

The converter script is used again on the raw data when all recording periods are marked into a JSON file. This time, the converter uses the previously acquired time interval values to convert only the distinct four parts.

D. Preview Video Generation

For users to preview the data sequences quickly, we provide a video containing DVS event visualisation frames, RGB frames and depth images side by side. Since Realsense frame durations are not constant, there are occasional occurrences of longer frame times followed by a sequence of shorter frame periods. First, the Realsense RGB and depth images corresponding to the desired recording fragment start and end times are copied. Then, the frames of DVS events are generated by drawing all the positive and negative events of the corresponding RGB frame duration into a png image.

TABLE I
DATA FORMATS USED FOR EACH SENSOR

Sensor type	Converted file	Data stored
Realsense	png	RGB and Depth frames
LIDAR	pcd	Point cloud data
DVS	aedat	Pixel events data
Sensorboard	csv	Temperature, humidity, air
IMU	csv	3-axis acc, gyro

Afterwards, all of the three sensor images are combined into a single frame. Finally, frames are encoded into a video using the FFmpeg[2] tool.

E. Reference Trajectory Generation

With each recording section of the Agri-EBV dataset, we provide a reference trajectory of the sensor platform estimated by the Cartographer SLAM tool [11]. The ROS version of the Cartographer is used. Therefore, we first create a rosbag file that contains both LIDAR point cloud and IMU data. Then, a script systematically launches the Cartographer for each of the recordings. The output trajectory points of the Cartographer are then captured by a custom ROS node, as there is no provision for writing these data directly to a file. The trajectory is provided in the regular dataset version as a list of points and orientations in a csv file. However, the trajectory is a separate rosbag file containing a single topic of the tf2 type in the ROS version. Afterwards, a Gnuplot[3] diagram tool is used to draw the graphs of XY coordinates of each trajectory for visual validation.

F. Validation

We validate data in multiple steps. An overview of data validation is shown in Table II. The first data validation is done after raw data files are uploaded to a computer. We check recording software log files and convert only those recordings showing up as completed in the log. This action is helpful because there may be recording attempts that have failed due to equipment or environmental issues. The main validation step is done after the raw data are converted to the standard formats.

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

[3] http://www.gnuplot.info/

TABLE II
LIST OF ITEMS VALIDATED DURING DATASET PROCESSING

Validation description	Data	Processing step	Criticality
Completeness of the recording	recording log file	upload	critical
Time alignment calculation	IMU data of DVS LIDAR and RS	conversion	critical
Quality of DVS and RGB calibration	calibration output file	upload	
Quality of data for LIDAR – RS calibration	RGB frames	conversion	
Trajectory loop closure	RGB frames	conversion	critical
The ratio of lost IMU samples	DVS and RS IMU files	conversion	
Balance of polarity of DVS events	DVS convertor log file	conversion	
Quality of DVS and RGB data	preview video	partitioning	
Quality of LIDAR data	trajectories of SLAM tool	trajectory generation	

The IMU-based time alignment approach implies that the sensor platform is tilted and lowered before and after the main recording. Therefore, these moments have to be located visually in the IMU gyroscope data of the DVS sensor. The data graph at these moments has a distinct shape, so it is relatively easy to do by hand. The recording is processed further only if the platform tilting parts can be found in the IMU data so that time alignment can be calculated.

We provide data for postponed LIDAR to RGB extrinsic calibration that contains a sphere held in discrete positions, using the methodology from [12]. Therefore, we also check if the whole sphere is visible in the captured RGB frames. RGB frames are also used to visually find the start and the end of the main recording part.

Individual data converters provide information that is used to assess data quality. In particular, we consider the ratio of the number of positive and negative DVS events and the number of lost IMU samples. For example, some recordings may have missing data samples due to connection bandwidth limitations. Converter software compares the actual number of IMU samples with an estimate based on a known IMU sample rate and recording length. We provide this ratio of lost IMU samples in the metadata file of the dataset because it may also indicate possible missing data from other sensors.

After the preview videos are generated, we visually validate the content of RGB frames and frames generated by accumulating DVS events. In RGB images, we check if the desired scene and scenario are presented as intended and whether the images are sharp and clear. In DVS images, we check for excessive noise and sharp edges, as blurred edges may indicate issues with focal length adjustment of the DVS sensor.

The quality of LIDAR data is validated directly by visualisation of point cloud data and assessing trajectories obtained with the SLAM tool. We check if trajectories are closed and have a distinctive shape in the XY plane.

G. Conversion to Rosbag Format

The team also decided to publish the data in the rosbag file format as it provides a unified representation of all the different data types obtained from our sensor platform. All the data from various sensors written in the rosbag files can be simultaneously played back in the ROS environment and processed with built-in tools like Rviz or C++ and Python code API. We use two-step conversion: first, we convert each source file or a set of files to its ROS representation. This way, we obtain rosbag files that contain data from a single sensor, e.g., a different file for DVS events and DVS IMU, etc. Then, in the second step, we can combine the desired data of any of the sensors into a single merged file. We chose to publish four rosbag files of each recording. If needed, rosbag files can be combined further using rosbag API or ready-to-use ROS codebase from open-source packages.

H. Structure for Publishing

To create the final data structure for publishing (see Fig. 6), we use a python script that copies and renames the necessary data, compresses the data and generates a unique metadata file for each data sequence. The metadata contains information about the sensors used, data structure description, data about the time synchronisation and known issues. We publish three archives for each of the recordings: the main recording data, data for postponed calibration, and the main recording data in the rosbag format.

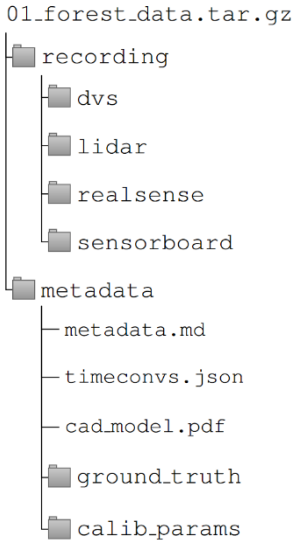

Fig. 6. Publish ready file structure tree of the main data archive.

For internal purposes, we used recording names consisting of a date followed by a time, for example, "2020-09-25_11-21-18". It is a simple approach to generate a unique identifier for each of the recordings and a quick way to evaluate the time and place of the data recording session. For the end-user, this naming scheme would not make any sense. Therefore, when generating the final data structure, all the occurrences of the internal names are renamed to a recording number followed by the name of the recorded environment, e.g., "01_forest".

For hosting the dataset, IEEE DataPort was selected. It is a widely recognised unified open-access platform with a great variety of datasets. It offers good collaboration opportunities and allows updating datasets and getting feedback from other users.

III. Conclusion

In this paper, we have described procedures involved in processing data for the Agri-EBV dataset. Due to the requirements to achieve accurate time alignment of the sensors used, a necessity for postponed calibration data and conversion to the rosbag format, the whole procedure demands multiple stages of data conversion, formatting and validation.

The procedure we described could include more automation to reduce the human time required during dataset processing. We believe that finding loop closure and calibration balls in RGB frames could be done by automated computer vision software. Still, the development of such a system was not feasible for our number of recordings. Also, the IMU-based time alignment method could be improved to automatically find the time offset and drift values without the need for dedicated platform movement procedures.

Acknowledgements

The authors would like to acknowledge the rest of the Agri-EBV dataset team for their contribution to the data processing: Dr.sc.ing, researcher Andrejs Zujevs and PhD student, researcher Mihails Pudzs for their significant contribution to developing the data processing software and methodology. The authors express their gratitude to Dr.sc.ing Vitalijs Osadcuks, PhD student Aldis Pecka and PhD student Janis Galins from Latvia University of Life Sciences and Technologies for providing the robotic platform, performing most of the data recordings, and giving valuable feedback about the data recoding process.

The research has been supported by the Latvian Council of Science (lzp-2018/1-0482).

References

[1] E. Mueggler, H. Rebecq, G. Gallego, T. Delbruck, and D. Scaramuzza, 'The event-camera dataset and simulator: Event-based data for pose estimation, visual odometry, and SLAM," *International Journal of Robotics Research*, vol. 36, no. 2, pp. 142–149, Feb. 2017. https://doi.org/10.1177/0278364917691115

[2] G. Gallego *et al.*, "Event-based vision: A survey," *arXiv*, pp. 1–30, Apr. 2019. https://doi.org/10.1109/tpami.2020.3008413

[3] D. Weikersdorfer, D. B. Adrian, D. Cremers, and J. Conradt, "Event-based 3D SLAM with a depth-augmented dynamic vision sensor," in *IEEE International Conference on Robotics and Automation (ICRA)*, Hong Kong, China, Sep. 2014, pp. 359–364. https://doi.org/10.1109/ICRA.2014.6906882

[4] A. Z. Zhu, D. Thakur, T. Özaslan, B. Pfrommer, V. Kumar, and K. Daniilidis, "The multivehicle stereo event camera dataset: An event camera dataset for 3D perception," *IEEE Robotics and Automation Letters*, vol. 3, no. 3, pp. 2032–2039, Jul. 2018. https://doi.org/10.1109/LRA.2018.2800793

[5] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The KITTI dataset," *International Journal of Robotics Research*, vol. 32, no. 11, pp. 1231–1237, Aug. 2013. https://doi.org/10.1177/0278364913491297

[6] F. Barranco, C. Fermuller, Y. Aloimonos, and T. Delbruck, "A dataset for visual navigation with neuromorphic methods," *Frontiers in Neuroscience*, vol. 10, pp. 1–9, Feb. 2016. https://doi.org/10.3389/fnins.2016.00049

[7] J. Binas, D. Neil, S. C. Liu, and T. Delbruck, "DDD17: End-to-end DAVIS driving dataset," *arXiv Computer Vision and Pattern Recognition* , pp. 1–9, Nov. 2017.

[8] J. Wulff, D. J. Butler, G. B. Stanley, and M. J. Black, "Lessons and insights from creating a synthetic optical flow benchmark," in *Computer Vision – ECCV 2012. Workshops and Demonstrations. ECCV 2012 (Lecture Notes in Computer Science)*, A. Fusiello, V. Murino, R. Cucchiara, Eds. Springer, Berlin, Heidelberg, vol. 7584, 2012, pp. 168–177. https://doi.org/10.1007/978-3-642-33868-7_17

[9] A. Zujevs, M. Pudzs, V. Osadcuks, A. Ardavs, M. Galauskis and J. Grundspenkis, "An Event-based vision dataset for visual navigation tasks in agricultural environments," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, Xi'an, China, Oct. 2021, pp. 13769–13775. https://doi.org/10.1109/ICRA48506.2021.9561741

[10] A. Zujevs, M. Pudzs, V. Osadcuks, A. Ardavs, M. Galauskis, and J. Grundspenkis, "Agri-EBV-autumn dataset," 2021. [Online]. Available on: https://ieee-dataport.org/open-access/agri-ebv-autumn. Accessed on: Aug 20, 2021.

[11] W. Hess, D. Kohler, H. Rapp, and D. Andor, "Real-time loop closure in 2D LIDAR SLAM," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*, Stockholm, Sweden, June 2016, pp. 1271–1278. https://doi.org/10.1109/ICRA.2016.7487258

[12] T. Toth, Z. Pusztai, and L. Hajder, "Automatic LiDAR-camera calibration of extrinsic parameters using a spherical target," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*, Paris, France, 2020, pp. 8580–8586. https://doi.org/10.1109/ICRA40945.2020.9197316

Maris Galauskis received a Bachelor degree in Intelligent Robotic Systems from Riga Technical University in 2020. He is a 2nd year Master student at Riga Technical University. He is currently a Research Assistant in a project related to computer vision.
E-mail: Maris.Galauskis@rtu.lv

Arturs Ardavs received a Master degree in Intelligent Robotic Systems from Riga Technical University Faculty in 2020. He has worked as a Research Assistant in projects related to computer vision and multi-robot systems.
E-mail: Arturs.Ardavs@rtu.lv