MENG INDIVIDUAL PROJECT

IMPERIAL COLLEGE LONDON

DEPARTMENT OF ELECTRICAL AND ELECTRONIC ENGINEERING

SLAM Using A Neuromorphic Camera to Estimate the Trajectory of a Randomly Moving Robot and to Localise Objects of Interest in the Surrounding Environment.

Author:
Tejas Dandawate

Supervisor:
Prof. Pier Luigi Dragotti

Second Marker:
Dr. Adam Smith

January 17, 2022

Abstract

Neuromorphic sensing is a novel way of encoding analogue signals, inspired by the biological processing of information in our brains. Neuromorphic sensing is based on time encoding, where rather than recording the amplitude of the input signal at predefined times, one records the time instants where the amplitude surpasses a certain trigger mark. This leads to a very efficient way of acquiring and processing signals.

Simultaneous localisation and mapping (SLAM) is a popular application of robotics and computer vision, in which a robot moves autonomously along an unknown trajectory. A useful application of this would be in cases such as finding the location of natural fires, in which it is important to know both the mapping of the surrounding environment as well as the locations of both the robot and the objects of interest.

The main aim of the project is to utilise a neuromorphic camera to carry out SLAM (moving in an unknown trajectory), mapping out the environment around the robot and finding the location it is in. This could possibly leverage spiking neural networks to further emulate biological responses to the stimuli the camera would receive. Then, if time permits, the next step would be to carry out some object recognition to localise objects or points of interest in the local environment.

Acknowledgements

Thanks mum!

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1   Objectives

1.2   Challenges

1.3   Contributions

Chapter 2

Background

This chapter outlines background information required for understanding the basis for the project. The theory and literature serves to outline the main concepts used for neuromorphic SLAM, as well as to reveal gaps in existing research that require solidifying.

2.1 Event Cameras

Event based cameras can be described as 'bio-inspired sensors that differ from conventional frame cameras: Instead of capturing images at a fixed rate, they asynchronously measure per-pixel brightness changes, and output a stream of events that encode the time, location and sign of the brightness changes' [1].

2.1.1 Benefits

Event-based cameras are purported to provide a number of benefits including, but not limited to;

- **Very high temporal resolution**

  The reason for this is that whereas frame-based cameras have a certain frame-rate, event-based cameras do not have this limitation, meaning the "blind time" between frames is eliminated. The reason for this is that the function of a frame-based camera is dependent on the global shutter to capture the light at a particular instant, whereas event-based cameras can be thought of as having individual shutters for each pixel that are shut whenever an even occurs.

- **High dynamic range**

  The reason for this is again the fact that each pixel has its own individual shutter, but as well as this they all use a logarithmic scale, meaning they function well from very bright to very dim environments as well as fast shifts between the two.

- **Low power consumption**

- **High pixel bandwidth**

  Each pixel can capture events at the rate of kHz. This has the effect of reducing blur since there is a very high temporal resolution to begin with. This makes the system very responsive and therefore ideal for real-time systems.

- **Efficient Encoding**

  Since events are asynchronous and spatially sparse (i.e there are mainly 0 values in the matrix), the encoding is very efficient, as opposed to frame-based cameras that produce data that is very spatially dense.

The above benefits are very persuasive reasons to adopt neuromorphic cameras in many different applications. It is conceivable that if algorithms can make use of these benefits (since most classical algorithms play to the strengths of the data generated by frame-based cameras), real-time systems could be completely revolutionised.

2.1.2 Function

Event-based cameras differ from frame based cameras fundamentally, in that they do not rely on a global shutter closing at regular intervals to record information of a scene. Instead each pixel closes whenever it detects an 'event' occurring. The way to detects such events are dictated by the 'event generation model'[1].

Each pixel responds to changes in its log photo-current ($L = log(I)$, where I is the perceived brightness), giving the system a very high dynamic range. A recorded event 'k' has the format $e_k = (\mathbf{x}_k, t_l, p_k)$. THis is known as the Address Event Representation (AER). The first value is the spacial location of the event ($\mathbf{x}_k = (x_k, y_k)^\top$), the second value t_k is the temporal location, and the final value $p_k \in 1, -1$ indicates the polarity of the event (i.e in which direction the brightness gradient was changing). The brightness increment between two events at the same pixel is given by the equation $\Delta L(\mathbf{x}_k, t_k) = L(\mathbf{x}_k, t_k) - L(\mathbf{x}_k, t_k - \Delta t_k)$. In a perfect (noise free) environment an event is fired whenever the brightness increment reaches a temporal contrast threshold given by the equation $\Delta L(\mathbf{x}_k, t_k) = p_k C$ ($C > 0$). It should be noted that the value of C could be variable and therefore different for $p_k = \pm 1$.

Additionally, we can approximate the temporal derivative of a pixels brightness by utilising the following Taylor expansion:

$$\Delta L(\mathbf{x}_k, t_k) \approx \frac{\delta L}{\delta t}(\mathbf{x}_k, t_k)\Delta t_k$$

$$\frac{\delta L}{\delta t}(\mathbf{x}_k, t_k) \approx \frac{\Delta L(\mathbf{x}_k, t_k)}{\Delta t_k} = \frac{p_k C}{\Delta t_k}$$

The above approximation, however, is only true under the assumption that Δt_k is exceedingly small. Since unlike frame-based cameras we do not measure absolute brightness, this is an indirect way of measuring and keeping track of the brightness within the frame.

Figure 2.1 shows the basic functionality of an event based camera. **(a)** is the simplified circuit diagram of the DAVIS pixel, which in **(b)** is used to convert light into events (shown in real life in images **(c)** and **(d)**). **(e)** shows how this setup would view a white square rotating on a black disk. It is a stream of events going from the past in green to the present in red. These events can then be seen overlaid on a natural scene in **(f)**.

2.1.3 Optic-flow Methods

More classic computer vision techniques using optic-flow constraints can now be utilised to characterise the events detected by pixels. In frame-based systems, optic flow methods create a flow-field that describe the displacement vector (signifying direction and magnitude of movement) for each pixel in the frame. A core constraint in this derivation is that the intensity of a local time-varying image region is constant under motion (for at least a short amount of time)[4].

NEED TO READ AND UNDERSTAND WHY THE FOLLOWING IS TRUE [4] [1]

$$\Delta L \approx -\nabla L \cdot v \Delta t_k$$

2.2 Neural Heterogeneity

Above are very persuasive reasons for utilising neuromorphic systems, but there still many challenges posed when attempting to do so. For example, each pixel only responds to brightness change, but the problem is that such a change could be a result of not only scene changes, but also the position of the camera within the scene. For this reason most neuromorphic systems have currently been limited to stationary cameras. As well as this the system is especially prone to stochastic noise, due to inherent shot noise in photons and from transistor circuit noise [1]. In order to tackle such issues, it is useful to look at existing examples of spiking neural systems, such as biological brain with neurons working based on a spiking function. It is known that the brain is heterogeneous on every scale, in the past this was though to be simply a by-product of noisy processes, but more recently it can be shown that by evolving our largely homogeneous spiking neural networks (SNNs), we can create more stable and robust systems [5].
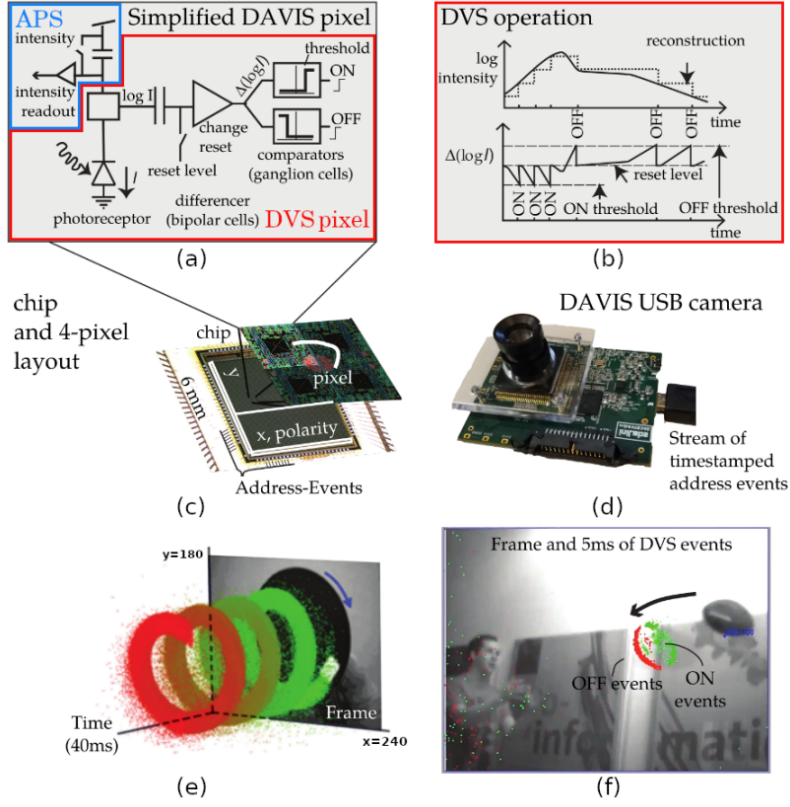
Figure 2.1: Summary of DAVIS event based camera[1]

2.3 Existing Algorithms for Event Analysis

For SLAM and pose estimation the problem again is that classical systems heavily rely on the structure of conventional cameras, and so there needs to be a radical paradigm shift in order to take events as inputs instead. The reason for this is that their function depends on iterative changes to the location probabilities using inputs, for which spiking inputs are ideal.

2.3.1 Probabilistic Filters

Bayesian Inference

Unlike most other previous systems, probabilistic filters such as Bayesian filters inherently work for the new scenario with event-based cameras. Bayes's theorem can be derived from simple probabilistic rules[6]. We know $P(X|Y) = \frac{P(X,Y)}{P(Y)}$ and similarly $P(Y|X) = \frac{P(Y,X)}{P(X)}$. Therefore we can re-arrange both to give $P(X|Y)P(Y) = P(Y|X)P(X)$, since $P(X,Y) = P(Y,X)$. Then from there formula for Bayesian inference can be trivially obtained:

$$P(XZ) = P(Z|X)P(X) = P(X|Z)P(Z)$$

$$P(X|Z) = \frac{P(Z|X)P(X)}{P(Z)}$$

In the above equations X is known as the prior (which is the assumed location of the camera) and Z is known as the posterior (which is the measurement taken by the sensor or camera). $P(Z|X)$ is known as the likelihood function, which indicates how likely it is to have received the particular reading given the assumed position.

Monte-Carlo Localisation

Now that we have the concept of Bayesian inference we can adapt it to create an efficient localisation algorithm. It includes initialising a number of particles that act as predictors of where in the map the camera is. We can now give the probability of a posterior camera position given a sensor reading.

The probability distribution $P(X|X)$ is a continuous function, and so updating each of the posteriors for every value of X is a computationally difficult problem. We can instead break it up into smaller bins to alleviate this issue. When generating these particles we can represent the probability distribution, albeit at a lower granularity. The benefit of this is that even though we cannot see the full distribution the peaks (i.e. the locations the camera is most likely to be in) are very well defined.

Now we can use the above simplification to carry out the following steps:

1. Randomly assign particle distribution across map

2. Apply Bayes' law to measurement to update particle distribution

   Bayes' rule can be simplified to be:

   $$w_{i_{x+1}} = P(z|x_{i_x}) \times x_{i_x}$$

   This can be done since the ignored multiplier on the right hand side will be normalised in the next step.

3. Normalise particle weights

   Now the particles will have weights that no longer sum to 1, and so we need to normalise them again to follow the usual rules of probability:

   $$w_{i_{x+1}} = \frac{w_i}{\sum_{i=1}^{N} w_i}$$

4. Re-sample particle distribution

   We now need to create a new set of particles that all have the same weight ($\frac{1}{N}$, but whose spacial distribution now reflects the probability density.
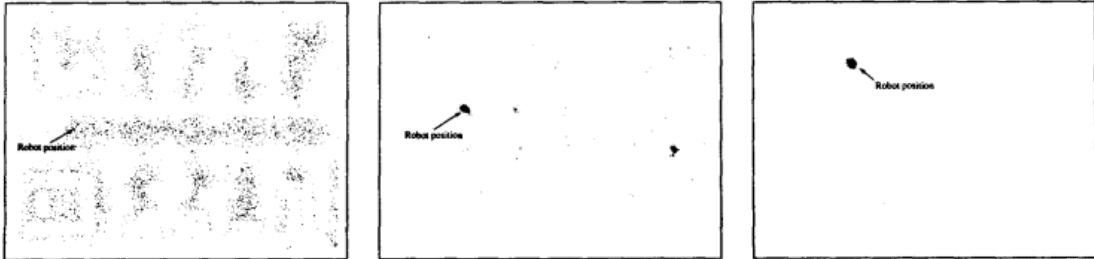

Figure 2.2: Example of Monte-Carlo localisation[2]

Figure 2.2 shows a typical example of the algorithm. The leftmost panel shows the random initialisation (or previous particle distribution), which then becomes the centrally shown distribution after one iteration. Since only one measurement is taken and the room is symmetrical it is possible that it could be in one of two location (hence the two dense clusters). After one more reading in the next iteration the algorithm is quickly able to narrow down the location of the robot.

This algorithm, however, is classical and has therefore been mostly superseded by deep neural networks in most modern day applications. As well as this it is only applicable to localising the robot, whereas we want to be able to simultaneously map it. SLAM can be achieved using similar methods by identifying and moving around located points of interest and estimating their positions as well as the robots. These tasks have been efficiently solved by neural networks for classical frame based data, but there is still much ongoing research on how to do the same with spiking data. NEED TO FIND CLASSICAL EXAMPLE OF SLAM ALGORITHM TO REFERENCE AS WELL AS IN DEPTH EXPLANATION OF LOOP CLOSURE ETC.

2.4 Computational Neuroscience

The foundations of Spiking Neural Networks (SNNs) are in computational neuroscience. The mechanisms of neurons in the brain are the inspiration behind creating artificial neural networks with neurons that spike in the same way. Neurons in an a typical Artificial Neural Network (ANN) have a weight, bias and activation function. this means the output of the neuron can be specified as:

$$y = \theta(\sum_{j=1}^{n} w_j x_j - u_j)$$

WRITE SOME STUFF FROM COURSERA COURSE[7]

2.5 Existing Datasets

There already exists many repositories of recorded neuromorphic data to get familiar with spiking data.

2.5.1 Neuromorphic MNIST

This data-set is a spiking version of the original frame-based MNIST dataset [8]. The dataset is identical to the original MNIST dataset in all ways (including scale, size and sample split) bar one - it was captured using an ATIS sensor mounted on a motorised pan-tilt unit. This sensor moved while viewing the MNIST examples on an external monitor.

For each item in the dataset there is a binary file which has a list of events. Each event is characterised by a 40 bit unsigned integer. The integer gives the following information of a particular event:

- bit 39 - 32: X location (in pixels)

- bit 31 - 24: Y location (in pixels)

- bit 23: Polarity (0 for OFF, 1 for ON)

- bit 22 - 0: Timestamp (in microseconds)

More datasets include"

- DVS128

- Other data-sets such as fashion MNIST could also be converted to spiking times by treating image intensities as input currents to model neurons, so that higher intensity pixels would lead to earlier spikes, and lower intensity to later spikes

- Heridelberg Spiking data-sets (SHD and SSC)

2.6 Image Reconstruction Algorithms

Image reconstruction has been implemented for event data using on the direct optimised versions of Convolutional Neural Networks (CNNs). An example of this is the network named 'U-net'[9] which managed to reconstruct a video using 10M parameters to analyse events from an AER camera protocol. Recent work by Rebecq *et al.* illustrates a novel network architecture that reconstructs a video from a stream of events [3]. These methods are purported to allow the introduction of mainstream computer vision research to event cameras. Figure 2.3 shows an example of how converting spiking data to a video stream allows for use of classical computer vision algorithms.

A naive approach would be take take each would be to take each event $e_k = (\mathbf{x}_k, t_l, p_k)$ and assuming that the firing was due to the brightness change was due to a brightness change above a threshold $\pm C$ which is a constant that could be set by the user. If this was the case events could be directly integrated to recover the intensity map of images. however, the value C in reality does not remain constant and is heavily dependent on other factors such as event rate, temperature,
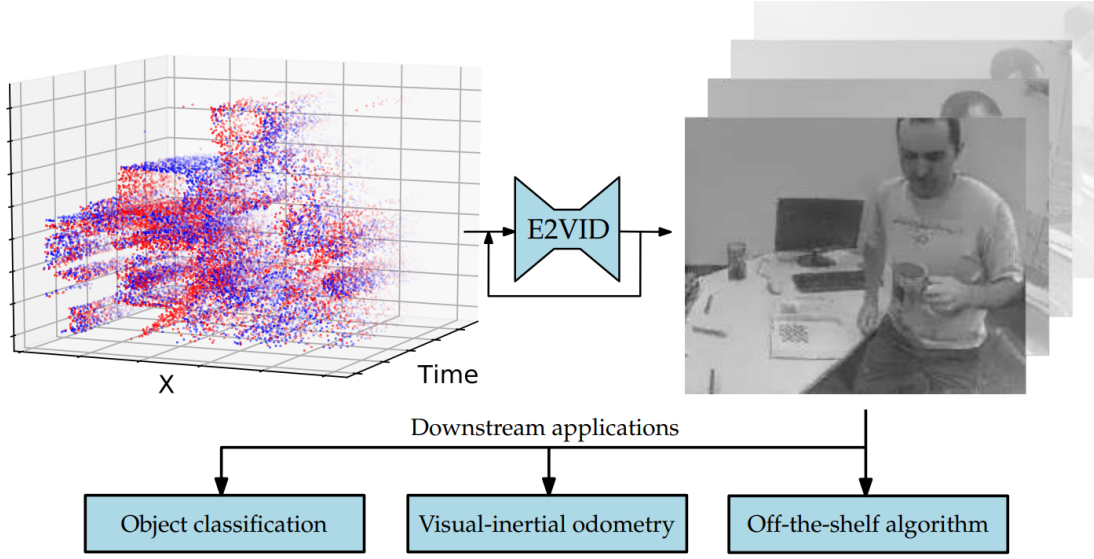
Figure 2.3: Illustration of mapping of spiking data to video stream to apply off-the-shelf algorithms to[3]

and sign of brightness change. The implementation outlined instead makes use of a Recurrent Neural Network (RNN), that takes as input sets of events within a spatio-temporal window. For example, a stream of events will be broken down into sequences given by $\epsilon_i \; \forall i \in [0, N-1]$. Since each sequence is of fixed length N the framerate of the output video from the RNN is proportional to the event rate. Figure 2.4 shows demonstrates the functionality of such a network. Each event window ϵ_k is converted to a 3D event tensor and passed into the network along with the last K constructed images to generate the latest iteration of the image. It is clear from this that each new image is constructed by fusing the previous K images with the new stream of events.
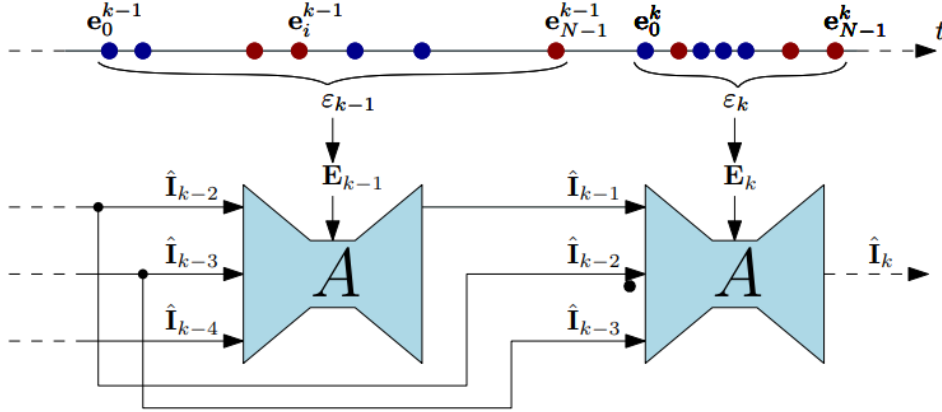

Figure 2.4: Overview of RNN used to generate video from sets of events[3]

Chapter 3

Requirements Capture

- Main:

  - Create and evaluate different NN models or SNN models on neuromorphic data from external datasets

  - Create practical set-up to obtain data from own neuromorphic camera

  - The main aim of the project is to utilise a neuromorphic camera to carry out SLAM (moving in an unknown trajectory), mapping out the environment around the robot and finding the location it is in.

- Fallback:

  - Only using existing datasets rather than practical set-up

  - Rather than focusing on SLAM the emphasis may be on object detection/recognition

- Extensions:

  - Carry out some object recognition to localise objects or points of interest in the local environment.

  - Creating further visualisations of the area map and locations of the robot and objects of interest.

Chapter 4

Analysis and Design

Chapter 5

Implementation

Chapter 6

Testing

Chapter 7

Results

Chapter 8

Evaluation

Chapter 9

Conclusion and Further Work

Appendix A

First Appendix

Bibliography

[1] G. Gallego, T. Delbruck, G. Orchard, C. Bartolozzi, B. Taba, A. Censi, S. Leutenegger, A. Davison, J. Conradt, K. Daniilidis *et al.*, "Event-based vision: A survey," *arXiv preprint arXiv:1904.08405*, 2019.

[2] F. Dellaert, D. Fox, W. Burgard, and S. Thrun, "Monte carlo localization for mobile robots," in *Proceedings 1999 IEEE International Conference on Robotics and Automation (Cat. No. 99CH36288C)*, vol. 2.   IEEE, 1999, pp. 1322–1328.

[3] H. Rebecq, R. Ranftl, V. Koltun, and D. Scaramuzza, "Events-to-video: Bringing modern computer vision to event cameras," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 3857–3866.

[4] G. Gallego, C. Forster, E. Mueggler, and D. Scaramuzza, "Event-based camera pose tracking using a generative event model," *arXiv preprint arXiv:1510.01972*, 2015.

[5] N. Perez-Nieves, V. C. Leung, P. L. Dragotti, and D. F. Goodman, "Neural heterogeneity promotes robust learning," *bioRxiv*, pp. 2020–12, 2021.

[6] T. C. Wallstrom, "The marginalization paradox and the formal bayes' law," in *AIP Conference Proceedings*, vol. 954, no. 1.   American Institute of Physics, 2007, pp. 93–100.

[7] R. P. Rao and D. H. Ballard, "Predictive coding in the visual cortex: a functional interpretation of some extra-classical receptive-field effects," *Nature neuroscience*, vol. 2, no. 1, pp. 79–87, 1999.

[8] G. Orchard, A. Jayawant, G. K. Cohen, and N. Thakor, "Converting static image datasets to spiking neuromorphic datasets using saccades," *Frontiers in neuroscience*, vol. 9, p. 437, 2015.

[9] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *International Conference on Medical image computing and computer-assisted intervention*.   Springer, 2015, pp. 234–241.