ETH Zürich

Master of Science in Robotics, Systems and Control

# Stereo Depth Estimation using Event-Based Cameras & Active Laser Features

*Author:*
Frédéric Debraine

*Supervisors:*
Dr. Yulia Sandamirskaya
Prof. Dr. Margarita Chli
Dr. Julien Martel
Ignacio Alzugaray

*Master's Semester Project*

*in the*

Institute of Neuroinformatics & Vision for Robotics Lab
University of Zürich & ETH Zürich

February 5, 2019

# Contents

# 1 Introduction

## 1.1 Motivation

Depth perception is an essential component in many robotics applications as it helps the robot understand its surroundings and localize itself in its environment. Typical systems used in modern robotic platforms include stereo camera setups, time of flight cameras such as LIDAR or structured light such as Kinect. In the case of stereo camera setups, extracting depth from RGB images can be a challenging task to perform in real time.

**Underlying principles of depth perception.** For humans, it's the binocular vision that helps retrieve the notion of depth by relying on computing disparities, i.e. the difference in image location of an object seen by the left and right eye. In biological systems, this is effortlessly done by the brain but it requires intensive processing in artificial vision systems. Computer vision requires indeed to know which pixel on the left view image corresponds to which pixel on the right view image. This essential step is called matching, also known as stereo correspondence. One way to do it, it to naively perform a 2D search in the whole image. But more efficient methods use the geometric constraints bounding the two camera views to reduce the search to a 1D line (epipolar geometry). This process can however still be cumbersome as it highly depends on the image resolution.

**Motivation of this project** Our goal in this semester project is to prove that we can reduce the stereo matching to a trivial problem by leveraging both event based vision and active laser features. The former will be used to allow for fast and efficient feature extraction while the latter will solve the matching problem.

## 1.2 Related Work

**Stereo event-based reconstruction.** [2] introduces a robotic head supporting two Dynamic Vision Sensor capable of panning and tilting movement. Because disparities can only be computed from a dynamic environment, they demonstrate active perception through mimicking the microsaccades eye movements. [4] performed 3D reconstruction using two DVS with a spiking stereo neural network implemented on a massively parallel neuromorphic processor in which they fed the events. his work tries to mimic the visual cortex function that retrieves depth information. propose a model that solves the stereo-correspondence problem with a spiking neural network that can be directly implemented on a neuromorphic chip that is with massively parallel, compact, low-latency and low-power. [5] introduces a pipeline for semi-dense 3D reconstruction using a stereo event camera. To do so they propose a global energy minimization problem to estimate the inverse depth of an event in the reference view from a number of stereo observations and use a depth fusion strategy to improve the density of the resulting reconstruction.

**Active features.** [1] proposes a frequency detector for event based cameras using blinking LEDs. We will implement it and use it in our pipeline to detect events stemming from a known frequency pattern.

**Previous work.** Our project relies on an idea presented in [3] to combine event-based vision with active laser features to reconstruct a depth map of a static scene. We introduce a new implementation in C++ of the full pipeline, which has the advantage to be simpler and more lightweight.

The following contributions stem from our work:

- **All-in-one Camera & Laser Calibration procedure.** An easy and quick calibration procedure to extract intrinsics and extrinsics parameters for each cameras and laser. We use a chessboard pattern widely used for calibrating conventional cameras.
- **Laser controller.** Visual interface with trackbars to precisely control the laser point along with a tunable sweep mode (boundaries, speed, frequency). We also added a pointing system that control the laser such as it points at a certain pixel in the camera frames.
- **Filter.** Before matching, we want to discard all events that may stem from noise, platform & background movements as well as any events that doesn't come from the laser. For that purpose we implement a robust frequency filter [1] that only outputs events coming from a source blinking at a given frequency.
- **Temporal Matcher.** As we receive new filtered events from both cameras we just assess their time consistency, i.e. we constraint a match up to a fixed threshold in time distance.
- **Triangulation.** We propose two methods to achieve per-pixel depth estimation. The first relies on a standard stereo camera-camera triangulation. The second makes the assumption that the laser can be modeled as a camera and hack the same stereo triangulation pipeline.

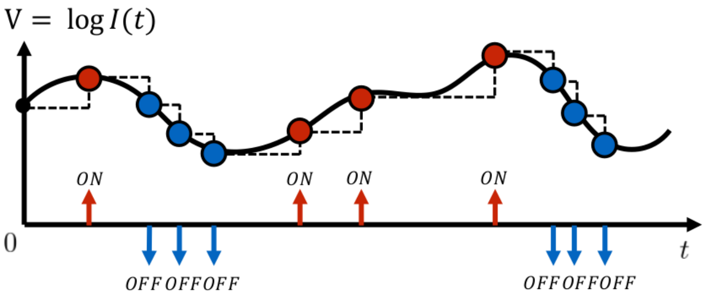
# 2 Method

## 2.1 Hardware setup

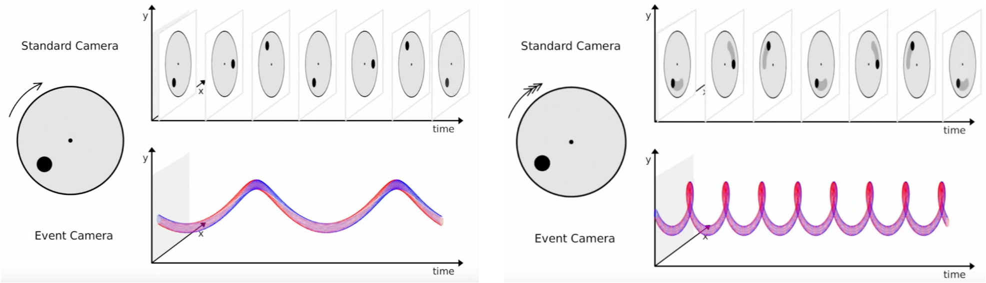
### 2.1.1 DAVIS cameras

**Event-Based Vision.** Event-based cameras such as the DVS and DAVIS have the main characteristic of having each of the pixel updated asynchronously with a binary encoding of the intensity. Such sensors implement a level-crossing sampling rather than a uniform time sampling like in conventional cameras and reacts to logarithmic brightness changes (see 2.1a) and therefore offer a new constraint to solve the correspondence problem: time. This has several advantages:

- **Efficiency.** Pixels only react to a certain threshold in contrast change leading to much less redundant information and a highly sparse data stream (see 2.1b). This allows for the use of more efficient algorithms.
- **Speed.** Traditional cameras have an update rate in the range of the millisecond making it prone to motion blur effects (see 2.1c). Whereas event-based sensors have an much higher temporal resolution, close to the microsecond which prevents this problem to occur.
- **High Bandwidth.** Because of the asynchronicity of each pixel, event-based cameras are also able to detect intensity changes in a wide range of luminosity intensity - also known as High Dynamic Range (HDR) - without saturation while traditional cameras would lead to underexposed and overexposed image areas.



**(a) Principle of level-crossing sampling in DVS.**



**(b) Sparse data stream.**

**(c) High temporal resolution.**

**Figure 2.1: (a)** shows how events are triggered for a single pixel given the change of brightness in a logarithmic scale. **(b)** & **(c)** illustrates the main advantages of a Dynamic Vision Sensor (DVS) over a standard cameras.

**Fixed stereo rig of DAVIS.** In our setup, we use two DAVIS cameras as a stereo pair on a fixed rig (see 2.2). The main difference between the DAVIS and the DVS is that the former also outputs 240 by 180 gray-scale frames aligned with the event pixels allowing for the use of standard stereo calibration pipelines.

**Challenges.** The main challenge in using event-based sensors in our project is to correlate in time and space the events coming from both cameras. This requires keeping the devices synchronized and coming up with a solution for the stereo matching. The synchronization problem is solved by connecting the two DAVIS using a 3.5mm plug patch cable allowing for the event time-stamps to be synchronized. To cope with the matching problem, we introduce active laser features which reduce the traditional stereo correspondence search into a trivial problem.

### 2.1.2 Active laser features

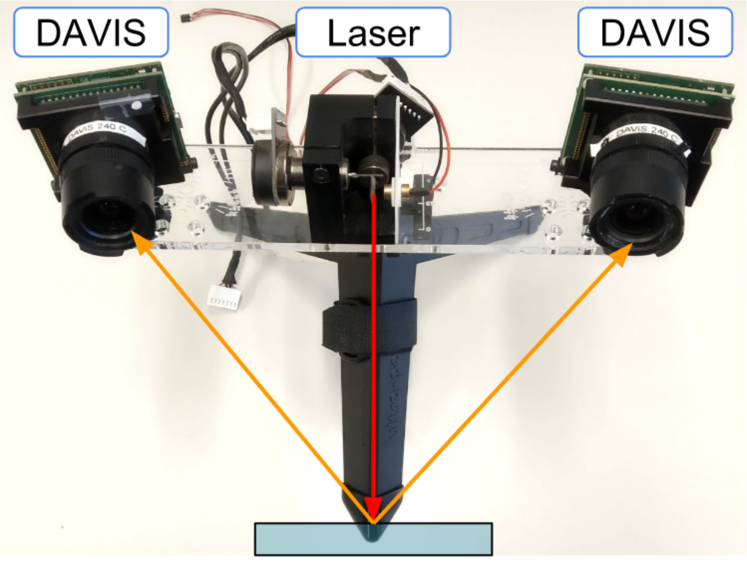


**Figure 2.2: Hardware setup.** Stereo rig composed of a laser and two DAVIS. The laser beam is reflected by a two-axis mirror-driven galvanometer and blinks at a known frequency which trigger events in the DVS at the same time.

The main idea behind active laser features is to create a known pattern of laser stimuli that is easily and instantaneously detectable in the DAVIS event streams.

- **Active Feature Extraction.** Traditional methods for stereo depth estimation usually rely on keypoints detection at the early steps, which is difficult in the case of textureless/uniform surfaces since it needs corners or blobs to be present in the scene. Instead of passively extracting features from the scene, our method relies on actively creating features by using a blinking laser inside the field of view of the cameras. This always lead to keypoints as long as the surface is reflective enough to trigger events in the cameras. We use a laser beam reflected by a two-axis mirror driven galvanometer enabling us to control the laser dot and make it sweep the scene (see 2.2).

- **Temporal matching.** Using the laser to trigger localized events in both cameras, the matching of keypoints becomes trivial. Traditional pipeline requires computationally expensive matching methods to match keypoints from each camera frame. Furthermore, the accuracy and robustness of such matching technique heavily depends on the descriptor used to describe each keypoint. In our case, we only need to consider the arrival time - called event timestamp - of each keypoint event as descriptor. To ensure time consistency and increase robustness as the laser is moving, we match two filtered events if their temporal distance is below a certain threshold before sending them to the triangulation block in our pipeline.

## 2.2 Software pipeline

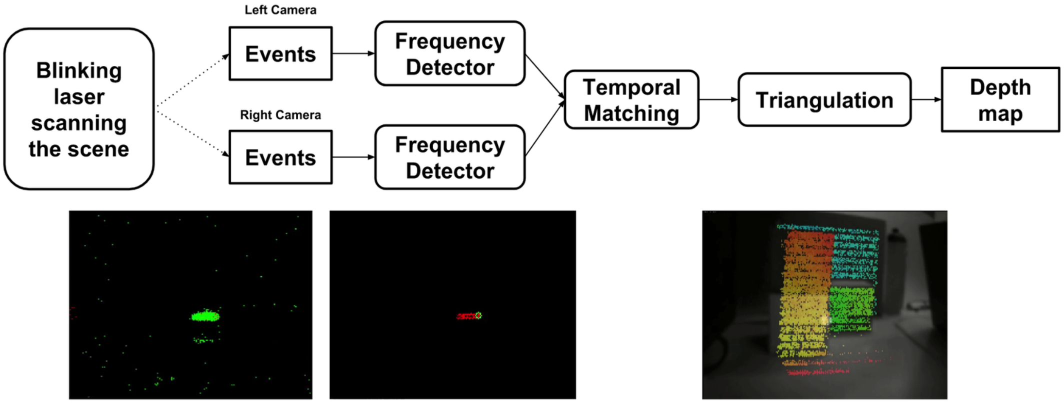


**Figure 2.3: High level view.** Our pipeline consists in a series of functional blocks that process the events coming from both DAVIS as input and return a depth map as output. A visualization of 3 different steps in the pipeline is representing (from left to right): the incoming raw events, the output of the frequency detector and the depth map.

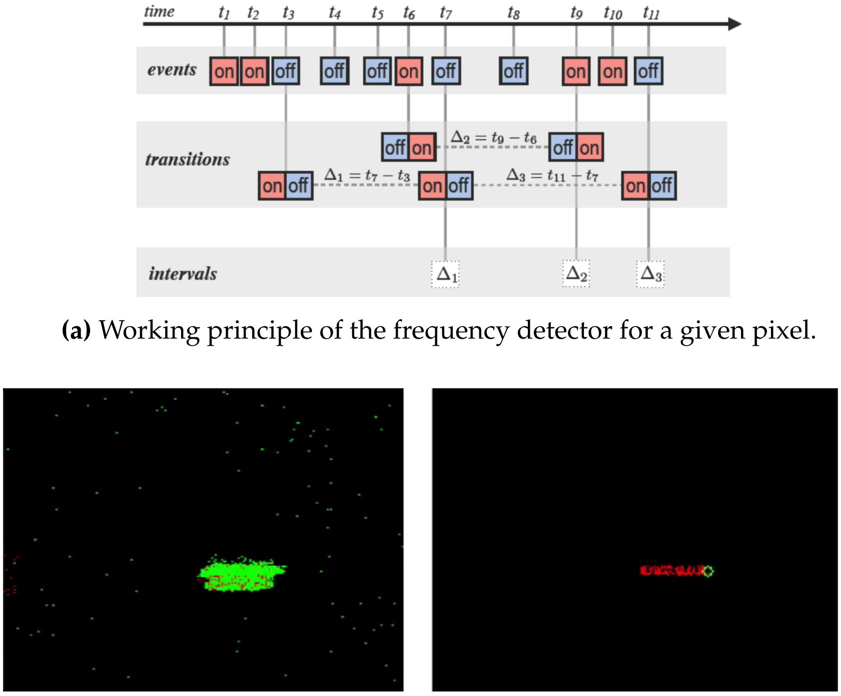
**Laser Control.** The laser dot can be pointed and moved to different spots in the scene by controlling two mirrors - on which the laser beam is reflected - around their rotation axis. We keep track of the laser state using a command space similar to the (x-y) pixel space of a camera. We can also control the laser to sweep through the scene while blinking.

*Tunable parameters: Blinking laser frequency, vertical & horizontal step size and sleeping time between each command update.*

**Frequency Detector.** The goal of the frequency detector is to associate a frequency to each of the incoming DAVIS polarity events and only output the ones that match the laser frequency (see 2.4). Our implementation is based on [1] and consists in three steps for each pixel:

- **First layer - Polarity event.** Keeps track of the last event that arrived at the given pixel. Each time there is a new incoming event, its polarity is compared with the previous one. If the polarity changed, a transition event is triggered in the second layer. We then update the last event.
- **Second layer - Transition event.** We trigger a transition event and for each two types of transitions (on-off or off-on), we keep track of the last transition event
- **Third layer - Transition interval.** Finally we can compute the frequency associated to each transition event by calculating the intervals between consecutive transitions timestamps and this discard the original event or not.

*Tunable parameters: Target frequency and frequency window tolerance.*


time
$t_1$ $t_2$ $t_3$ $t_4$ $t_5$ $t_6$ $t_7$ $t_8$ $t_9$ $t_{10}$ $t_{11}$
events
on on off off off on off off on on off
transitions
off on
$\Delta_2 = t_9 - t_6$
off on
on off
$\Delta_1 = t_7 - t_3$
on off
$\Delta_3 = t_{11} - t_7$
on off
intervals
$\Delta_1$
$\Delta_2$
$\Delta_3$

**(a)** Working principle of the frequency detector for a given pixel.

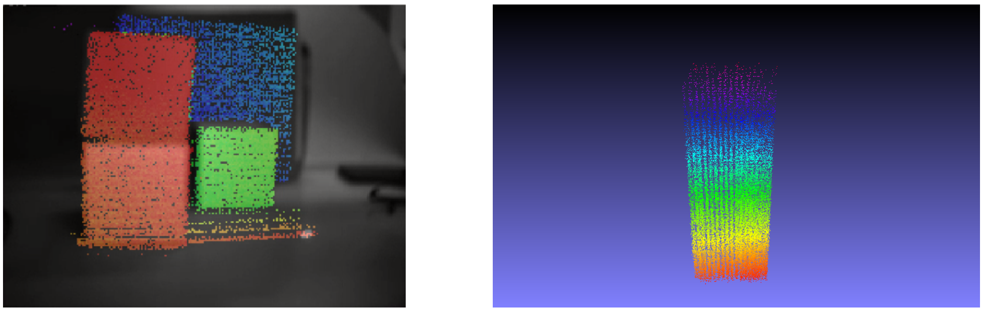
**(b)** (Left) - Input raw events. (Right) - Output events of the frequency detector.

**Figure 2.4: Frequency detector block.** (a) illustrates the working principle of the frequency detector implemented from [1]. (b) shows a typical input of incoming raw events from a DAVIS and the associated output where most of the noisy events have been discarded.

**Events Temporal Matching.** The matching block receives the events from the frequency detector associated to each stereo view in two buffers and outputs event pairs that satisfy a given temporal constraint. Such temporal constraint is necessary as both buffers are filled at two different paces. This is mostly due to variations in the internal characteristics of the sensors as well as their different viewpoints leading to more or less events being triggered - noisy or not. Two events are paired up and sent to the triangulation block if their associated timestamps are closer in time than a given threshold.

*Tunable parameters: Buffer size (maximum event age) and temporal matching threshold.*

**Triangulation.** Finally, for each events pair, we can compute the associated 3D point using the calibration parameters of our setup. The 2D depth maps are obtained by populating a 240 by 180 matrix using the left event coordinates and the depth value of the 3D point. We can also visualize the full 3D point cloud after a bit of post-processing using Python and MeshLab.



**(a)** 2D depth map output visualized in real-time

**(b)** 3D point cloud output visualized offline with Meshlab

**Figure 2.5: Triangulation block outputs.** (a) shows a 2D depth map sample for a scene composed of 3 planar objects at different distance. (b) represents a 3D point cloud of a tilted plane after recording the triangulated points.

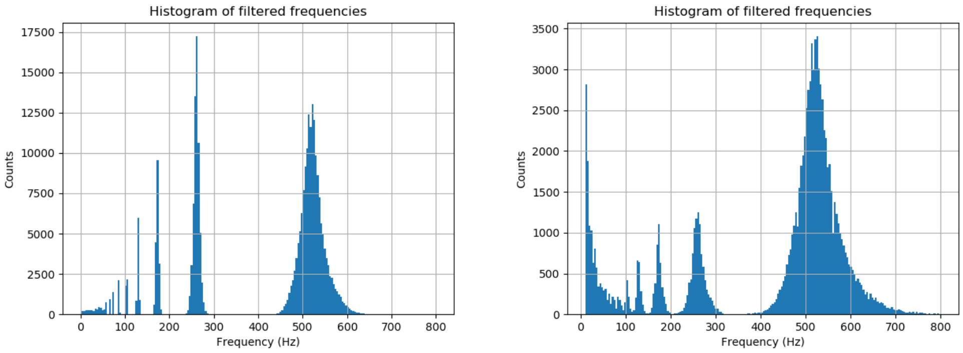
**Laser & Camera Calibration.** Before running the pipeline, we need to calibrate the setup to estimate the intrinsic and extrinsic parameters of each camera and the laser. This procedure needs to be done carefully as it will greatly influence the quality of the triangulation block output. We use a conventional stereo calibration pipeline from the OpenCV library using a chessboard pattern of 8 by 5 squares with a length size of 2.86cm. The calibration of the DAVIS cameras relies on detecting the chessboard corners using the frame information under different view points. This leads to intrinsics parameters such as the calibration matrix and the distorsion parameters as well as the extrinsics parameters, i.e. the rigid transformation between the two cameras. A similar procedure is used at to calibrate the laser as we make the assumption that we can fit a camera model. Instead of knowing where the event is triggered in the second camera, we know what command (x,y) we used to trigger the event in the first camera. Using this image space / command space analogy we can retrieve the extrinsics and the intrinsics of the laser using a standard calibration pipeline. We detect the blinking laser dot using the events stream outputted by the frequency detector and update the laser command to make the dot converge towards each detected chessboard corner.

# 3 Experimental evaluation

This chapter aims at evaluating the main components of our system. We will first focus on how our frequency detector behaves when discarding events and then evaluate the depth reconstruction w.r.t several parameters such as the laser speed, the scene distance and the temporal matching constraint.

## 3.1 Frequency detector

We analyzed the output of the frequency detector for a laser blinking at about 520Hz in two states: static and sweeping. This frequency was chosen as it triggered a higher number of events in both DAVIS and was far away from the noisy frequency range - 0 to about 100Hz - due to intrinsic electronic noise and background movements.



**(a)** Static mode

**(b)** Sweeping mode

**Figure 3.1: Frequency analysis.** Histograms of detected frequencies in a static mode (only blinking) and sweeping mode (blinking & sweeping the scene) for a laser blinking at 520Hz.

**Static mode.** There is a significant spike at the corresponding frequency of 520Hz as well as smaller spikes at the harmonic frequencies. We make the conjecture that these events correspond to the pixels triggered at the boundary of the laser blob that is prone to small fluctuations. Between two consecutive triggers, the elapsed time should be a multiple of the laser period which is what is observed.

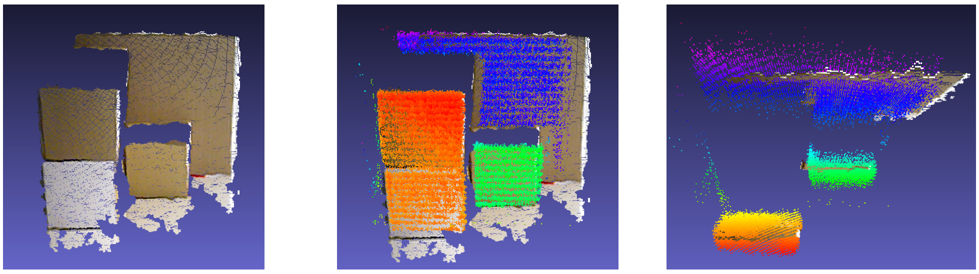
**Sweeping mode.** The vast majority of the events are located around a spike at the expected frequency (yet more spread and smaller then in the static mode). We can still observe the harmonics like in the static case. The rest of the events - in the 0-100 frequency range - are probably triggered by the laser blob movement and noisy events.

In our pipeline the output of the frequency detector are the events corresponding the one close to the spike of the laser frequency.

## 3.2 Depth reconstruction

### 3.2.1 Ground truth and evaluation method

To assess both qualitatively and quantitatively the reconstruction quality of our system, we recorded the depth ground truth of a same scene under different conditions using an ASUS Xtion Pro. By aligning the ground truth point cloud with our triangulated point cloud, we can assess qualitatively our results (see 3.2).



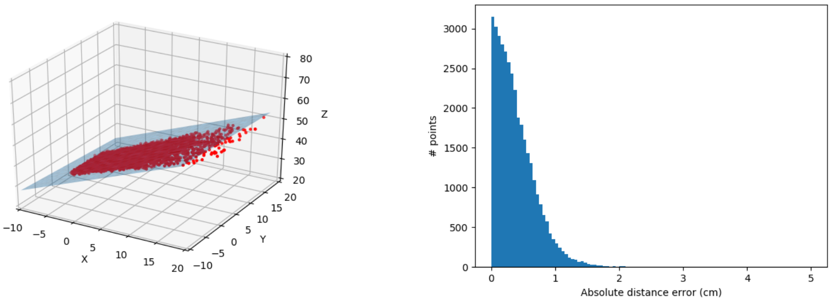
**(a)** Ground truth point cloud.

**(b)** Front view of superposed point clouds.

**(c)** Top view of superposed point clouds.

**Figure 3.2: Point clouds comparison.** (a) shows the ground truth point cloud collected by a ASUS Xtion. (b) & (c) displays the superposition of the ground truth with the system's triangulated point cloud.

Before drawing any conclusions however, we need to extract meaningful statistics from this 3D point cloud comparison. A first method would be to use the ICP (Iterative Closest Point) procedure to align both point clouds and compute a point-to-point mean distance between each point cloud. Another method would be to manually align both point clouds, fit planes on each planar section of the ground truth scene and then compute a point-to-plane mean distance between for each point cluster of our result point cloud corresponding to a planar section of the ground truth. We will go for the second method as it straightforward and still gives meaningful statistics. The plane fitting method rely on a RANSAC procedure to increase the robustness towards noise (see 3.3). We then considered the absolute error distance between the fitted plane and each 3D point of our setup.



**(a)** 3D point cloud and its fitted plane

**(b)** Histogram of absolute distance error

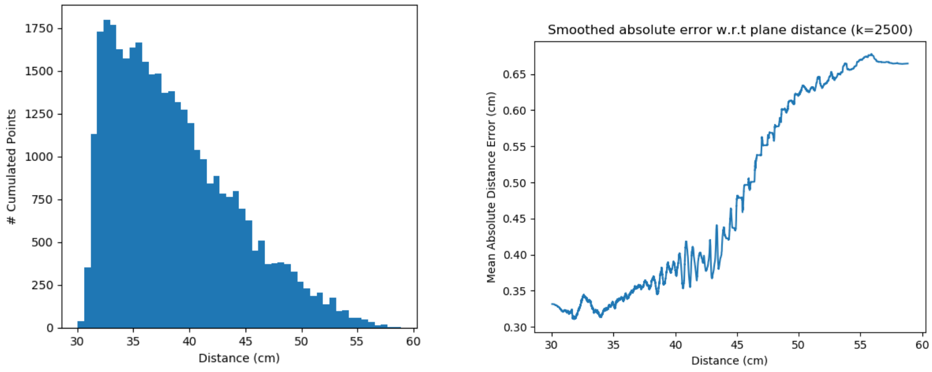
**Figure 3.3: RANSAC plane fitting & error metrics.** (a) illustrates the plane fitting procedure using RANSAC over a 3D point cloud triangulated by our system. (b) represents the histogram of absolute distance error computed between each 3D point and the fitted plane.

### 3.2.2 Influence of the scene distance

The following experiment uses a tilted plane - ranging from 30cm to 60cm w.r.t the setup - to test the influence of the distance on the reconstruction quality. After recording the reconstructed 3D point cloud and the ground truth, we can use the evaluation method introduced in 3.2.1.

**Fixed parameters:**

- Temporal matching constraint: 0.1ms
- Laser speed range: 5 scanned lines per second



**(a)** Histogram of distance for the reconstructed 3D point cloud

**(b)** Evolution of the error w.r.t the distance

**Figure 3.4: Influence of the distance using a tilted plane.** (a) displays the distribution of the reconstructed 3D points w.r.t their distance to the camera. (b) represents the absolute distance error (smoothed with a box filter) between each 3D point and the RANSAC fitted plane w.r.t the distance of the scene to the camera.

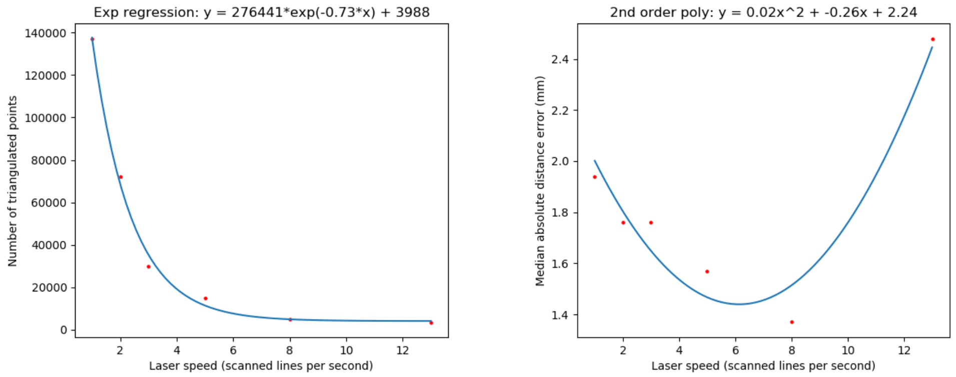
From 3.4a we can conclude that the sparsity of the reconstructed point cloud is highly correlated to the distance: the further the scene, the sparser the reconstruction. In 3.4b we can also notice a degradation in the reconstruction accuracy as the distance to the setup increases, especially above 40cm. The saturation observed above 50cm is likely due to the small number of triangulated points and might not be representative of the true reconstruction quality, as we expect the error to increase exponentially.

### 3.2.3 Influence of the laser speed

To estimate the influence of the laser speed on the reconstruction quality, we recorded the output of the triangulation block - i.e. the triangulated 3D point cloud - using a planar scene. In our experiments, we define the laser speed by the number of lines scanned in the scene per second and make it vary in a range from 1 to 13 scanned lines per second .

**Fixed parameters:**

- Distance to the scene: 30cm
- Temporal matching constraint: 0.1ms



**(a)** Evolution of the point cloud sparsity w.r.t the laser speed

**(b)** Evolution of the median error w.r.t the laser speed

**Figure 3.5: Influence of the speed for a planar scene.** (a) displays the evolution of the sparsity of the reconstructed 3D point cloud w.r.t the laser speed. (b) represents the evolution of the median absolute distance error between the reconstructed 3D point cloud and the ground truth fitted plane with the laser speed.

As expected, 3.5a clearly shows an increase in the point cloud sparsity as we make the laser sweep faster. However 3.5b reveals an interesting phenomenon, that is, the reconstruction accuracy is worse a low speed and reaches an optima around 7 scanned lines per second before deteriorating. While the right part of the plot makes sense, the left part - slow laser speed - seems a bit counter intuitive. We don't have any conjecture about this phenomenon.

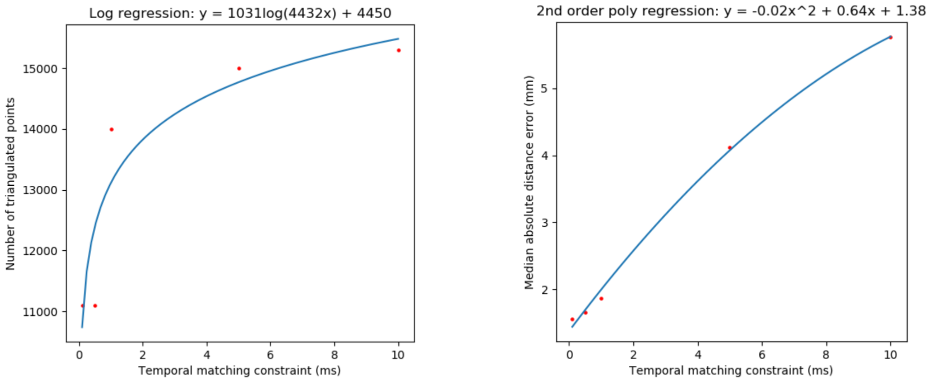
### 3.2.4 Influence of the temporal matching constraint

When matching the extracted features of both event streams, the temporal constraint also plays an important role in the reconstruction accuracy. To assess it, we vary the temporal matching constraint in a range from 0.1ms to 10ms with the same scene as the last experiment. If the timestamp distance of two events (one from each stereo pair) is below a given threshold, we match them, otherwise we discard the oldest event.

**Fixed parameters:**

- Distance to the scene: 30cm
- Laser speed range: 5 scanned per second

As we relax the temporal matching constraint, we observe an denser 3D point cloud but also a clear loss in reconstruction accuracy (see 3.6).



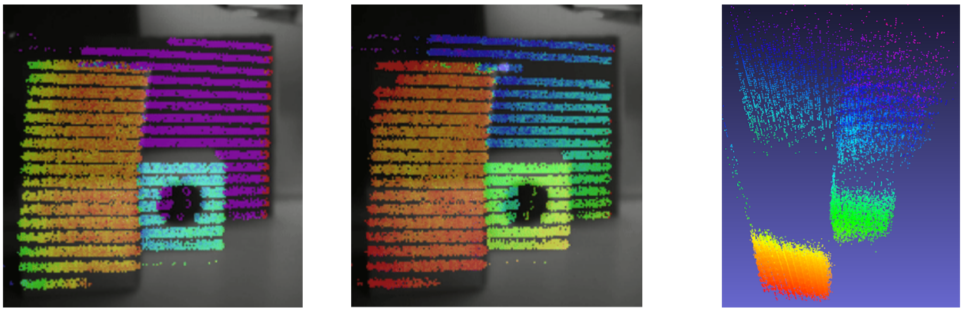
**(a)** Evolution of the point cloud sparsity w.r.t the matching constraint

**(b)** Evolution of the median error w.r.t the matching constraint

**Figure 3.6: Influence of the temporal matching constraint for a planar scene.** (a) displays the evolution of the sparsity of the reconstructed 3D point cloud w.r.t matching constraint. (b) represents the evolution of the median absolute distance error between the reconstructed 3D point cloud and the ground truth fitted plane w.r.t the matching constraint.

### 3.2.5 Laser-DAVIS stereo pair

In the following experiment, we make the assumption that a camera model can be applied to the laser (see 2.2). As we know the laser command, we can use the same stereo triangulation pipeline as before. However we don't apply any temporal matching constraint as the commands aren't given any precise timestamps. Despite a loss in reconstruction accuracy and some offset between the two pairs, we believe this method may become useful when incorporated into a data fusion pipeline.



**(a)** (Left Camera - Laser) stereo pair depthmap

**(b)** (Right Camera - Laser) stereo pair depthmap.

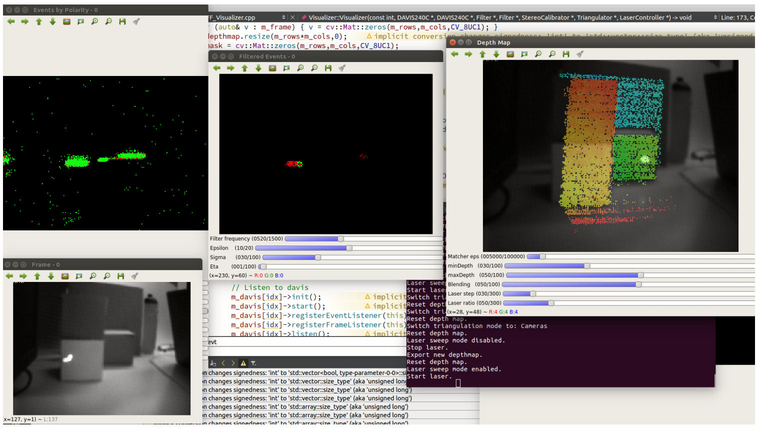
**(c)** (Left Camera - Laser) stereo pair point cloud.

**Figure 3.7: Laser-DAVIS stereo pair reconstruction.** (a) & (b) display the superposition of the triangulated depth map and the frame for each (Laser-DAVIS) stereo pair. (c) shows the triangulated point cloud corresponding to (a).

### 3.2.6 Other considerations

Several other parameters and choice of setting could have been considered for this evaluation such as the laser frequency, the ambient lightning and the choice between a DVS or a DAVIS.

# 4 Conclusion


Events by Polarity - 0
F_Visualizer.cpp
Visualizer::Visualizer(const int, DAVIS240C *, DAVIS240C *, Filter *, Filter *, StereoCalibrator *, Triangulator *, LaserController *) -> void
Filtered Events - 0
Depth Map
Frame - 0

```
 (auto& v : m_frame) { v = cv::Mat::zeros(m_rows,m_cols,CV_8UC1); }
lepthmap.resize(m_rows*m_cols,0);      ⚠ implicit conversi
ask = cv::Mat::zeros(m_rows,m_cols,CV_8UC1);
```

Filter frequency (0520/1500)
Epsilon (10/20)
Sigma (030/100)
Eta (001/100)
(x=230, y=60) ~ R:0 G:0 B:0

Matcher eps (005000/100000)
minDepth (030/100)
maxDepth (050/100)
Blending (050/100)
Laser step (030/300)
Laser ratio (050/300)
(x=28, y=48) ~ R:4 G:4 B:4

```
    // Listen to davis
    m_davis[idx]->init();                    ⚠ implicit
    m_davis[idx]->start();                   ⚠ implicit
    m_davis[idx]->registerEventListener(this)
    m_davis[idx]->registerFrameListener(this)
    m_davis[idx]->listen();                  ⚠ implicit
```

```
n changes signedness: 'int' to 'std::vector<bool, type-parameter-0-0>::si
n changes signedness: 'int' to 'std::vector::size_type' (aka 'unsigned long
n changes signedness: 'int' to 'std::vector::size_type' (aka 'unsigned long')
n changes signedness: 'int' to 'std::vector::size_type' (aka 'unsigned long')
n changes signedness: 'int' to 'std::array::size_type' (aka 'unsigned long')
n changes signedness: 'int' to 'std::array::size_type' (aka 'unsigned long')
n changes signedness: 'int' to 'std::array::size_type' (aka 'unsigned long')
```

```
Laser swee
Start lase
Switch tri
Reset dept
Switch tri
Reset depth map.
Switch triangulation mode to: Cameras
Reset depth map.
Laser sweep mode disabled.
Stop laser.
Export new depthmap.
Reset depth map.
Laser sweep mode enabled.
Start laser.
```

Figure 4.1: GUI.

We introduced in this semester project a new way of estimating depth in stereo using event-based sensors and laser active features. The main contributions consists in establishing a full pipeline in C++ from the calibration of the camera/laser, the frequency detection, event matching and triangulation as well as OpenCV user interface to visualize each step of the pipeline (see 4.1).

Several tracks for improvements of this project include:

- Range increase: Higher laser output power to collect more events in a wider range.
- Laser synchronization: Adding timestamps to the laser commands would allow proper measurements from the (laser-DAVIS) stereo pair triangulation.
- Depth map fusion: Optimizing a single depth map using the triangulated points from different stereo pairs and inferring depth at missing pixel values.
- Make it fly: Allow the platform to move and reconstruct the scene in 3D while having it on a drone. This could be beneficial for close range drone inspection.

I want to thank V4RL and INI for offering this project opportunity.

# Bibliography

[1] Andrea Censi et al. "Low-latency localization by Active LED Markers tracking using a Dynamic Vision Sensor". In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2013.

[2] Jacques Kaiser et al. "Microsaccades for Neuromorphic Stereo Vision". In: *International Conference on Artificial Neural Networks (ICANN)*. 2018.

[3] Julien N. P. Martel et al. "An Active Approach to Solving the Stereo Matching Problem using Event-Based Sensors". In: *IEEE International Symposium on Circuits and Systems (ISCAS)*. 2018.

[4] Marc Osswald et al. "A Spiking Neural Network Model of 3D Perception for Event-based Neuromorphic Stereo Vision Systems". In: *Scientific Reports*. 2017.

[5] Yi Zhou et al. "Semi-Dense 3D Reconstruction with a Stereo Event Camera". In: *European Conference on Computer Vision (ECCV)*. 2018.