Low-latency localization by Active LED Markers tracking using a Dynamic Vision Sensor

Andrea Censi, Jonas Strubel, Christian Braend, Tobi Delbruck, Davide Scaramuzza

Abstract—**At the current state of the art, the agility of an autonomous flying robot is limited by the speed of its sensing pipeline, as the relatively high latency and low sampling frequency limit the aggressiveness of the control strategies that can be implemented. The use of Dynamic Vision Sensors (DVS), which encode changes in the visual field using an address-event representation, not unlike neuronal spikes in biological systems, have a latency that can be measured in the microseconds, thus offering the theoretical possibility of creating a sensing pipeline whose latency is negligible compared to the dynamics of the platform. However, to use this sensor we must rethink the way we interpret visual data. In this paper we present an approach to low-latency pose tracking using a DVS camera and Active Led Markers (ALMs), which are LEDs blinking at very high frequency (>1 KHz). The DVS camera time resolution is able to distinguish different frequencies, thus avoiding the need for data association. We compare the DVS approach to traditional tracking using a CMOS camera, and we show that the DVS performance is not affected by fast motion, unlike the CMOS camera, which suffers from motion blur.**

I. Introduction

- *Current agile robots use alternative tracking systems.*
  - **DS: one-two paragraphs dissing Raff D'Andrea**

Currently, the agility of a mobile robot is limited by the speed of the sensing pipeline. More precisely, "speed" can be quantified in *observations frequency* and *latency* (Fig. I.1). In current state-of-the art autonomous navigation applications (**DS: add citations)** cameras give observations with frequency of *???* and the total latency, from acquiring the images to processing them, is *???* ms (**DS: give reasonable numbers)**. To obtain more agile systems, we need to use faster sensors and low-latency processing.

In this paper, we consider the lowest-latency sensor available, called Dynamic Vision Sensor (DVS) and how it can be incorporated in a robotic system for the application of pose tracking. The main difference of a DVS with respect to a normal CMOS camera is that the data is transmitted as a series of *events* (Fig. II.1). Intuitively, the events generated can be interpreted as the sign of the derivative of the luminance, but this is just an idealization (Section II describes the principles behind the device). These events are fired not unlike spikes in a biological visual system, as they respond to *change* in the perceived luminance in fact, "silicon retina" is a nickname for the DVS. But the DVS circuits are much faster than a slow neuron: events are generated with a latency of *???* µs. Therefore, potentially we could obtain sensing pipelines with a negligible latency compared to dynamics of the platform. Moreover, compared to normal high speed cameras, the data output and thus the processing is reduced, as only change is advertised by the camera.

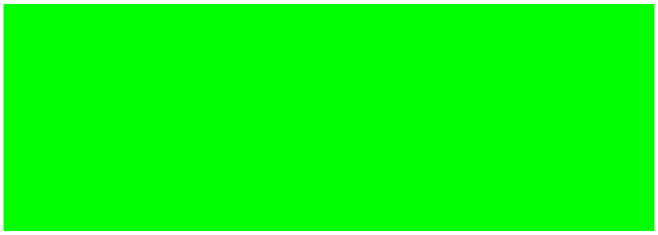


Figure I.1. Discretization and latency

We are a few years to the goal, however. The DVS camera, though currently available commercially, has a few limitations, such as the limited resolution (128 × 128 pixels), and it is too heavy to be attached on current agile drones. These problems will be solved shortly; here we turn our attention on how we could use the data from a DVS for autonomous navigation of flying drones.

The application that we show here is localization based on tracking of Active LED Markers (ALMs). These are blinking LEDs at high frequency ($> 1\,\mathrm{kHz}$). The DVS is fast enough to be able to estimate the blinking frequency. Therefore, we can detect not only the position, but, assuming that each ALM is given a slightly different blinking frequency, also the identity of the markers, thus simplifying data association. We envision that this system could be used for inter-robot localization for high-speed acrobatic maneuvers, or that, in applications such as rescue robotics these markers could be left in the environment to facilitate cooperative mapping.

It is clear that the way we do computer vision must be completely rethought. It is possible to integrate the events of a DVS camera to simulate a regular CMOS frame, on which to do standard image processing, however, that is not what you would want to do, because accumulating. Ideally, to have the lowest latency for the sensing pipeline, one would want each single event to be be reflected in a small but instantaneous change in the commands given to the actuators. Therefore, we really want to consider approaches that possibly use the information contained in each single event.

We have found two main approaches to handling event data. So far the DVS events are processed using features (such as lines or points) that are tracked through time (**CB: add citations/expand)**. This approach works well when the camera is static, because the output is very

A. Censi is with the Computing & Mathematical Sciences department, Division of Engineering and Applied Sciences, California Institute of Technology, Pasadena, CA. E-mail: andrea@cds.caltech.edu. J. Strubel and D. Scaramuzza are with Department of Informatics, University of Zurich. E-mail: *???* and davide.scaramuzza@ieee.org. C. Braend and T. Delbruck are with the Institute of Neuroinformatics, University of Zurich and ETH Zurich. E-mail: *???* and *???*.

sparse. We found out that mounting a DVS camera on a flying robot creates a new set of challenges. Because of the apparent motion of the environment, the input is not sparse anymore. Moreover, while in controlled conditions the DVS camera parameters can be tuned to obtain the best performance, a robot must be able to work in a wider range of environmental conditions and be robust to interferences.

To achieve this robustness we have developed an approach that can sacrifice a bit of latency to be more robust to noise and unmodeled phenomena. We accumulate the events perceived in thin slices of times corresponding to the blinking frequency (1ms slice for 1 kHz data). This allows to do detection of the ALMs position in image space. On top of this, we use a particle filter for tracking the position in image space of each detection, and a resolution stage to obtain coherent hypotheses on the joint position of the markers. Finally we reconstruct the pose using a standard approach to rigid reconstruction.

We evaluate our method in tracking the pose of a drone during an agile maneuver (a flip). We compare our methods with a traditional CMOS-based approach using PTAM. We verify that our method, with a latency of 1 ms, is able to reacquire tracking instantaneously regardless of the fast motion, while the CMOS data is corrupted by motion blur. We evaluate the reconstruction accuracy using an OptiTrack system (however, at 250 Hz resonse, this is way slower). We obtain values that are compatible with the low spatial resolution (128×128) of the DVS, which proves to be the current limitation.

Software, datasets, and animations illustrating the method are available at the website ???

II. The DVS camera

CB/TB: could you write a short description (~1.5 columns) of the DVS camera that could be understandable to a computer scientist? Also feel free to write more, I will edit it down to space constraints. I wrote down a few points that I'd like to make regarding future improvements that are relevant to robotics.

[1][2][3][4]

- Future improvements:
  - miniaturization
  - higher resolution
  - buffer able to handle more events
  - inclusion of a normal CMOS camera
  - easier to tune

(a) DVS camera (b) Response to circular stimulus (c) Drone rotor

Figure II.1. DVS camera and its output

Figure III.1.

Figure III.2.

III. Hardware setup and event data

- This section describes the basic hardware setup and how the data looks like.

A. Active LED Markers (ALMs)

- We have a set of blinking LEDs
- Each LEDs blinks at a different frequency.

B. Events

C. Events data in practice

- Fig. III.1 shows how the data looks like.
  - In this case, the LEDs are fixed in the environment and a *fixed* camera is looking at them.
  - Fig. III.1*a* shows the histogram of events from one pixel
  - Fig. III.1*b* shows the sequence of events from one particular pixel.
- Note also the halo in Fig. III.1*a* cannot be explained by the refractive properties of the optics of the camera and is probably due to properties.
- The idea
- Experimentally the interval is actually very repeatable

D. Alternate events and motion

- We go from events to "alternate events"
- This needs a buffer
- This series now has the polarity

- Fig. III.3*a* shows the histogram
- The frequency peaks are clearly visible in this histogram
- What about motion?
  - We see that, following motion, in Fig. III.3*b* the peaks are clearly visible.
  - There is also a
  - Of course all of this depends on the statistics of the image.

Figure III.3.

IV. Tracking algorithm

A. From raw events to sequence events

B. Particle filters

C. Estimation

D. 3D Reconstruction

V. Experiments

The goal of our experimental evaluation is to consider the advantages of a DVS-based tracking solution compared with a tracking solution based on a traditional CMOS camera. We compare the DVS+LED-based tracking with vision-based tracking using the PTAM algorithm, using the output of an OptiTrack system as the ground truth. The data show that the DVS+LED-based solution is able to deal with faster motions due to the minimal latency, however, the reconstructed quadrotor pose is not as accurate, as it could be expected from the lower resolution.

A. Setup

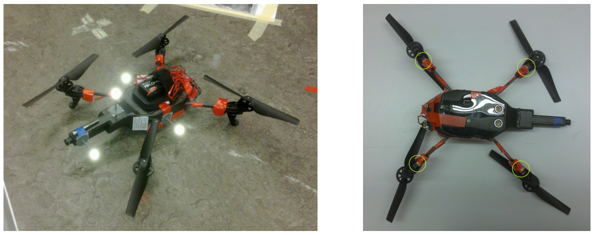
1) Robot platform: We used the commercially available ARDrone 2.0, a remarkable platform for its low price of *???*. We attached four custom-built ALMs (V.1). Each LED was fixed facing downwards, one under each of the four rotors, so that the four where lying on a plane forming a square of 20cm side length. The USB connector available on the drone provided power to the microcontroller and ALMs.

The drone has also a front-facing 720 × 980 CMOS camera that is used in these experiments, while the ground-facing camera is not used.

2) DVS : The DVS128 camera was used for the tests. This model is currently commercially available from INI labs. It has a resolution of 128 × 128 pixels. The lens attached was a *???*, with a FOV of approximately 90 deg, giving a resolution of 0.7 pixels/degree. For tracking the quadcopter, the DVS was installed on the floor facing upwards. Note that the relative motion between DVS and quadcopter would be the same if the positions were switched (ALMs on the floor or another vehicle and DVS on the platform).

3) OptiTrack: To measure the pose estimation accuracy we used a OptiTrack tracking system from NaturalPoint[1], which is a marker-based optical motion tracking system using active infrared light and reflective marker balls. Four markers have been applied to the drone (Fig. V.1a).

Our lab setup comprised 10 cameras in an 6 m × 8 m area; the cost of this system is approximately 20,000 CHF. The sampling frequency used was 250 Hz. The accuracy is stated as ∼ 1 mm by the manufacturer, but this seems an optimistic estimate based on our experience with the system.



(a) Infrared markers

(b) ALMs configuration

Figure V.1. The ARDrone 2.0 equipped with reflecting markers for the OptiTrack (shown in *a*) and the LEDs tracked by the DVS (shown in *b*).

a) Motion: The prototypical aggressive maneuver that we use is a "flip" of the quadcopter, i.e. a 360° roll (Fig. V.2a). This happens in approximately *???* seconds. During the flip the frontal camera images are severely blurred (Fig. V.2b).

b) Interference OptiTrack / DVS: We encountered an unexpected incompatibility between OptiTrack and DVS camera. The OptiTrack uses high-power infrared spotlights. In the OptiTracks standard configuration, the spotlights are pulsed at a high frequency. This is of course invisible to normal sensors and to the human eye, but it was a spectacular interference for the DVS. Like most cameras, the DVS is very sensitive to the infrared spectrum and is much faster than the OptiTrack strobing frequency. Strong generated a buffer overflow on the DVS as the electronics could not handle the large number of events to be processed contemporaneously. Eventually we understood how to deactivate the strobing for all the cameras prior to recording. Still there was a slight residual interference by the infrared illumination from the OptiTrack, but it should have relatively little impact to the results of our experiments.

B. Methods

We compare three ways to track the pose of the quadcopter: 1) The output of our method; 2) The OptiTrack output; 3) The output of a traditional feature-based tracker using the data from the conventional CMOS camera mounted front-facing on the drone. The image data was streamed to a computer via network interface, were the parallel tracking and mapping algorithm (PTAM)[**PTAM**] was employed for pose estimation.

1) Data recording, synchronization, and alignment: Using this setup we did several recordings, in which we recorded the OptiTrack tracking data, using its native format, the image data using a ROS interface, as well as the raw event data from the DVS in the native format. All these data, plus other data used for preliminary experiments, are available at the website *???*.

To synchronize the data from different sources we used a motion induced cue. We moved manually the drone up and down, generating an approximated sinusoid curve in the position data. Matching this trend in the data allowed easy synchronization of the various recordings.

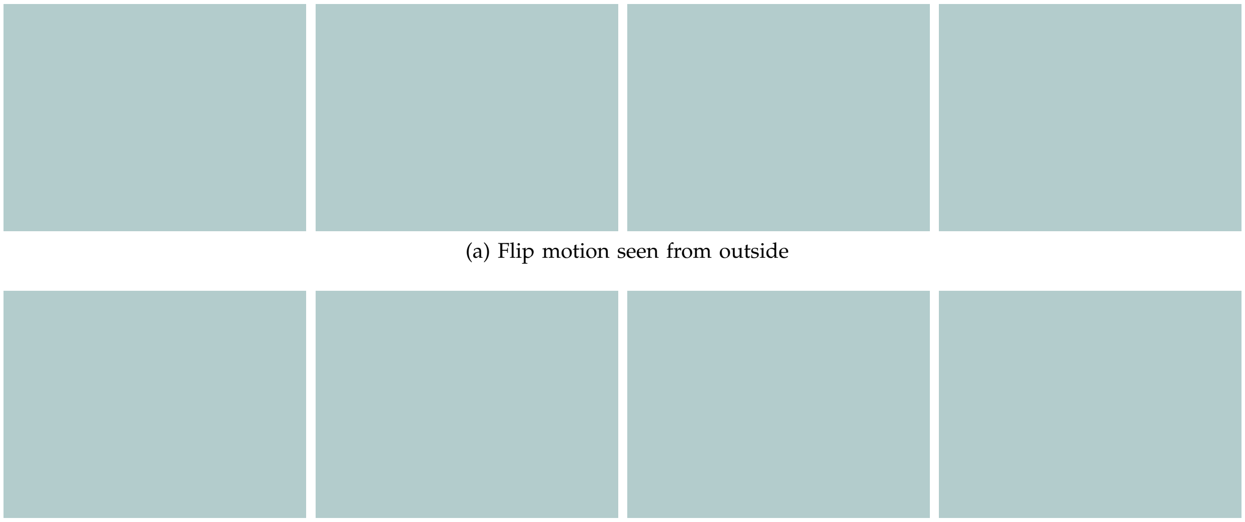
After adjusting for the delay, the data sets were brought to the same number of samples with a common time stamps. As our algorithm's output has a lower sampling rate than the OptiTrack (1KhZ vs *???*), the OptiTrack data was resampled by linear interpolation.

After time synchronization we put all data in the same frame of reference. Given two sequences of points $\boldsymbol{x}_k, \boldsymbol{y}_k \in \mathbb{R}^3$, the rototranslation $\langle \boldsymbol{R}, \boldsymbol{t} \rangle \in \mathsf{SE}(3)$ that matches them can be found by solving the optimization problem

$$\min_{\langle \boldsymbol{R}, \boldsymbol{t} \rangle \in \mathsf{SE}(3)} \sum_k \| \boldsymbol{x}_k - (\boldsymbol{R}\boldsymbol{y}_k + \boldsymbol{t}) \|^2, \qquad \text{(V.1)}$$

which can be easily solved using *???*.

[1] http://www.naturalpoint.com/optitrack/



(a) Flip motion seen from outside

(b) Flip motion seen from the frontal camera

Figure V.2. Flip motion

C. *Results*

We recorded data from 18 flips, of which only 6 were succesful. During the recordings we met a number of unforeseen difficulties due to our modifications to the drone. Having attached the LEDs and microcontroller to the drone we found that it had become unstable during flight and hard to control due to the additional weight, so while it could hover normally, it did not have enough thrust to stabilize itself after a flip.

1) Tracking downtimes: During a flip, both the DVS and PTAM lose tracking: PTAM loses tracking while the image is blurred; the DVS loses track when the ALMs are not visible from the ground. The comparison of these “blackout times” gives a direct measurement of the latency of the two systems.

The length of a flip was measured by considering the roll data from the OptiTrack, taking the interval between the last measurement before the flip and the first measurement after the flip when the helicopter was in a level orientation to the floor.

To measure the onset and offset of the blackout for the DVS, we considered the last sample before losing track (i.e. where the interval position samples were considerably higher than the mean sampling rate) and the first sample of reacquiring track (regaining a steady sample rate). The equivalent operation was performed on the PTAM data.

Figure V.3 shows a statistical comparison of the blackout time intervals for the two approaches compared to the duration of flips. The red bar indicates the median while the blue box marks the first and third quartile. The whiskers extend to a range of 1.5 times the range between the first and third quartiles. All data points outside this range are considered outliers and are marked with a red plus.

Table I shows the mean standard deviation of the different approaches. Our algorithm lost track during the average time of 0.35 seconds. PTAM lost track for a mean of 0.8 seconds, which is more than twice the time of the DVS and takes longer than the average duration of a flip.

Table I
MEAN AND STANDARD DEVIATION OF TRACKING DOWNTIME INTERVALS AND THE FLIP DURATION.

	mean [s]	std.dev. [s]
DVS	0.35	0.10
PTAM	0.80	0.33
flip motion	0.56	0.15

One can clearly see that the time where tracking is lost is much shorter with our approach in respect to PTAM. As Figure V.4 further illustrates, the downtime for the DVS stays inside the interval of the flip duration. The results emphasize that the DVS is faster in recovering lost tracks than the PTAM approach due to the shorter latency. As verified with our recordings, the downtimes of the DVS correspond to losing sight of the LED markers because of their emission angle.

We reckon that with a suitable configuration of either more markers or dynamic vision sensors, tracking could be maintained during the whole flip. PTAM shows to lose track for a longer duration than the flip takes. In contrary to the DVS the the camera of the drone loses sight of its tracked features due to blurring in the camera image and thus takes a longer time to recover.

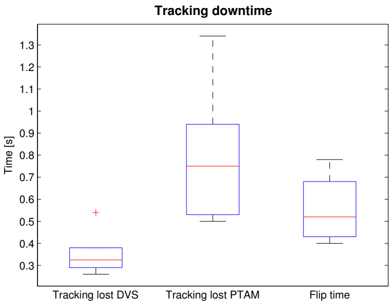


Figure V.3. Statistical plot of measure time interval. The boxplots show the time interval in which tracking is lost for the our algorithm and PTAM compared to the duration of a flip.

Table II
ESTIMATION ERROR OF DVS AND PTAM COMPARED TO OPTITRACK

(A) TRANSLATION

	mean [cm]	std.dev. [cm]
DVS	8.9	12.6
PTAM	19.0	12.4

(B) ROLL

	mean [°]	std.dev. [°]
DVS	19	27
PTAM	7	22

(C) PITCH

	mean [°]	std.dev. [°]
DVS	17	18
PTAM	5	11

(D) YAW

	mean [°]	std.dev. [°]
DVS	6	15
PTAM	3	10

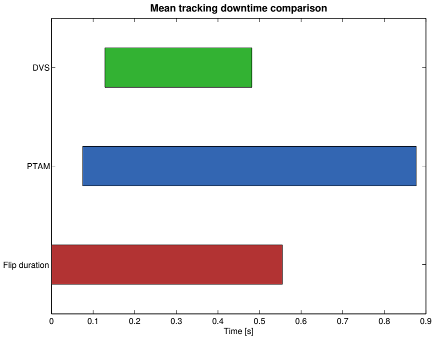
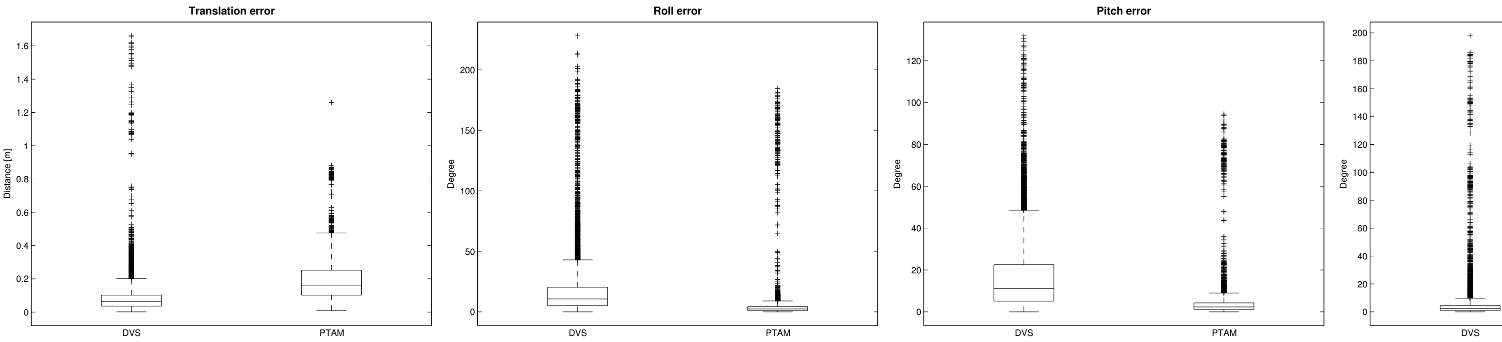


Figure V.4. Comparison of the mean tracking downtime intervals. The mean time intervals of both algorithms are compared against the mean flip time on a time line.

2) Pose estimation: Table IIIa shows the mean and standard error for the translation in both approaches. The DVS average error is roughly two times lower than PTAM. V.5a shows the statistical distribution of the pose errors. Although the spread of outliers is higher in our approach compared to PTAM, the translation errors of the latter technique show a broader distribution around their median. Overall this proves that the DVS approach has higher accuracy with less spread, if we neglect the extreme tails of the distribution.

Figures V.5b, V.5c and V.5d show the error distribution for roll, pitch and yaw respectively. The DVS performs worse in roll and pitch compared to yaw. This was to be expected, because of the position of the ALMs. As roll and pitch play a minor roll in qaudrotor pose estimation these can be neglected for finding the drone's orientaion. Table IIId demonstrates that the DVS performs slightly worse than PTAM with a mean error of 6 degrees and a deviation of 15 degrees. This is explained by the much lower resolution of the DVS (128 × 128) compared to the PTAM data (???).



(a) Translation error (b) Roll (c) Pitch

Figure V.5. Distributions of the errors of DVS/PTAM in reference to the OptiTrack measurements. The data is synthetized in Table II.

VI. Conclusions

Fast robots need fast sensors. A dynamical vision sensor (DVS) returns changes in the visual field with a latency in the microsecond resolution. This technology is the most promising candidate for enabling highly aggressive autonomous maneuvers for flying robots. The current prototypes suffer a few limitations, such as a relatively low resolution, and a large weight that cannot be put onboard an agile drone. However, these problems will be resolved in a few years. In the mean time, the sensing pipeline must be completely re-designed to take advantage of the low latency.

This paper has presented the first pose tracking application using DVS data. We have shown that the DVS can detect Active LEDs Markers (ALMs) and disambiguate their identity if different blinking frequencies are used, thus thus avoiding the complexity of data association. The algorithm that the we developed uses a Bayesian framework, in which we accumulate evidence of every single event into "evidence maps" that are tuned to a particular frequency. The temporal interval can be tuned and it is a tradeoff between latency and precision. In our experimental conditions it was possibile to have a latency of only 1 ms. After detecion, we used a particle filter and a multi-hypothesis tracker.

We have evaluated the use of this technology for tracking the motion of a quadrotor during an aggresive maneuver. Experiments show that the DVS is able to reacquire stable tracking with negligible delay as soon as the LEDs are visible again, without suffering from motion blur, which limits the traditional CMOS-based conventional feature tracking solution. However, the precision in reconstructing the pose is limited because of the low 128×128 resolution. Future work involving the hardware include improving the ALMs by increasing their power and their angular emittance field, as we have found these to be the main limitations.

In conclusion, DVS-based ALM tracking promises to be a feasible technology that can be used for fast tracking in robotics.

Acknowledgements: We gratefully acknoweledge the contribution of Christian Saner for *???*, of Yves *???* as our pilot, and Matia Pizzoli for the CMOS-based tracking experiments.

References

[1] M. Boerlin, T. Delbruck, and K. Eng. "Getting to know your neighbors: unsupervised learning of topography from real-world, event-based input". In: *Neural computation* 21.1 (2009). DOI: 10.1162/neco.2009.06-07-554.

[2] P. Lichtsteiner, C. Posch, and T. Delbruck. "A 128 × 128 120 dB 15 μs Latency Asynchronous Temporal Contrast Vision Sensor". In: *IEEE Journal of Solid-State Circuits* 43.2 (2008). DOI: 10.1109/JSSC.2007.914337.

[3] R Etienne-Cummings. "Intelligent robot vision sensors in VLSI". In: *Autonomous Robots* 7.3 (1999). DOI: 10.1023/A:1008968319725.

[4] M Oster, Y Wang, R Douglas, and S. C. Liu. "Quantification of a spike-based winner-take-all VLSI network". In: *IEEE Transactions on Circuits and Systems I: Regular Papers* 55.10 (2008). DOI: 10.1109/TCSI.2008.923430.