embeddedworld2018
Exhibition&Conference
... it's a smarter world

Embedded Velocity Measurement using a Sensor based on a FlyEye

Hans Dermot Doran
ZHAW / Institute of Embedded Systems
Winterthur, Switzerland
donn@zhaw.ch

***Abstract*—This paper describes a new algorithm for velocity detection using the principle of a fly-eye. Called the Distance of Travel algorithm, it is optimized for low cost commercially available cameras.**

Keywords—Velocity Sensor, Optical Sensors, Robot Vision.

I. INTRODUCTION

A. Motivation

The motivation for this body of work was to find a robust and computationally cheap way of enabling a robot to navigate down a corridor. The well-understood fly eye was, given its biological precedent, considered a suitable sensor for this task however we were unable to find any implementations on low-cost COTS (Commercial off the Shelf) cameras so the novelty of our paper is therefore to analyse and implement a fly-eye algorithm on this class of platform.

In order to achieve the task the well-known fly-eye algorithms were analysed and it was discovered that the linearity of response at the typical sampling times of low-cost cameras leaves something to be desired so we developed, simulated and implemented a new algorithm, called the Distance of Travel Algorithm. It can be shown that the algorithm functions with better linearity of response at the lower sampling rates expected by the COTS camera. This informs the structure of the paper.

We briefly discuss related work, methodology and algorithms followed by simulation results of the well-known algorithms. We then discuss the new algorithm, the Distance of Travel (DoT) algorithm and simulations. Section IV discusses the practical implementation of the DoT algorithm and the final section draws conclusions and proposes future work.

B. Related Work

Scientists have been examining the perception and navigation of insects in general and the fly in particular, for some 60 years and this has led to much work in replicating capabilities in technology. The substantial body of literature includes some informative summaries, f.i. Franceschini (2014), Sirnivasan (1999,) and Orchard (2014.)

Biological research generated several replicable models, for instance Hassenstein and Reichardt (Hassenstein, 1961) who described their Elementary Motion Detector (EMD - Figure 1.) Other models include the Barlow and Levick model (1965) or the Watson and Ahumada model (1985.) Many of these models have been replicated in engineering, often using analogue electronics for instance Tanner (Tanner, 1986)

Little work has done to apply known principles to CMOS detectors. Arreguit (1996) describes a pointing device and Basch (2010, 2011) Hassenstein-Reichardt based collision detectors.

C. Methodology

Current published (research) solutions use EMD's of size 8 to 254 pixels. Our contention is that data from standard cameras will provide data to perform several tasks in parallel, most probably with the use of Field Programmable Gate Arrays (FPGA) devices. A low computational cost task like the fly-eye algorithm can intuitively be bound to a CMOS camera.

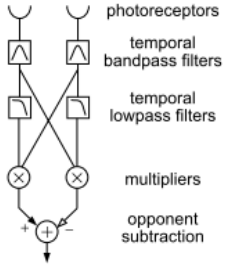

Figure 1: Elementary Motion Detector (EMD) according to Harrison [4]

The methodology used was to examine commonly used algorithms in simulation, and if suitable in a real-world environment to choose an algorithm to implement, prove the use-case in software and measure

the real-time properties of the solution to decide whether hardware offloading is necessary or beneficial.

D. Algorithms

The EMD generates a metric from the time taken for an image feature to pass from one receptor to a second. The EMD is unidirectional. Intensity based and token based are the two general categories of artificial implementations of the EMD. As we placed value on low computational complexity as well as resource expense, using these criteria, we chose two algorithms for an initial implementation, specifically Harrison and Koch (1999) who based their work on intensity based correlation using the Hassenstein-Reichardt principle and that of Roubieu et. al. (2013) who used token based methods.

II. Simulation of Known Algorithms

The two algorithms were simulated in Matlab. Given the typical size of a EMD scaled up to the kind of images we are using the area of the camera was sub-divided into photoreceptors of and array of 5*5 pixels (Figure 2.)

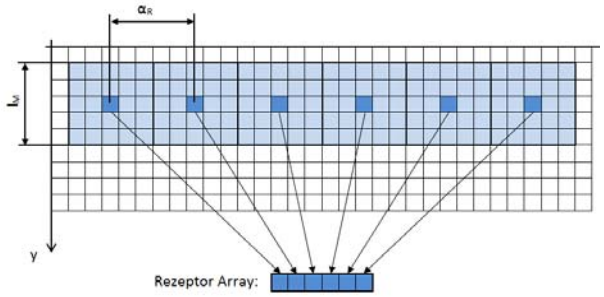

Figure 2: Assembling receptors from camera pixels

We used several forms of test patterns including a square-wave pattern (black and white bars) as well as randomly chosen pictures of wood, brick and concrete (Figure 3.) The algorithms were tested by passing the patterns in front of the receptors at different velocities.

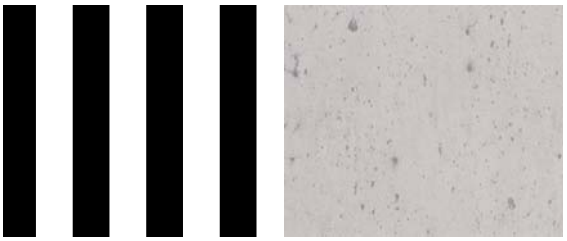

Figure 3: Black and white (left) and concrete (right) patterns for simulation of tokens

A. Hassenstein – Reichardt Detection

The Hassenstein-Reichardt detector was configured as follows:

Table 1: Fly-Eye Parameters for the Hassenstein-Reichardt detector

Parameter	Value
Distance between receptors (α_r)	4°
Mask Size	43 Pixels
Number of receptors	12 Receptors

We established the response at different angular velocities for a camera sampling time of 20 ms (Figure 4.) The simulation at a sampling rate 5 ms (Figure 5.) Inspecting the response we can see that correlation with the expected result is considerably better at sampling rates that are infeasible for a standard COTS camera so the Hassenstein-Richardt does not appear to be suited for implementation for our use-case.

B. Time Of Travel Algorithm

This procedure was repeated for the time of travel algorithm (ToT, Roubieu (2013.)) configured according to the set of values in Table 2.

Table 2: Fly-Eye Parameters for the Time of Travel detector

Parameter	Value
Distance between receptors (α_r)	7°
Mask Size	77 Pixels
Number of receptors	6 Receptors

The results, that largely correspond with experiences from the Hassenstein-Richardt detector in that the correlation between expected and actual velocity are weak, are shown in Figure 6 and are the reason it was considered necessary to develop a new algorithm.

III. Distance of Travel Algorithm and Simulation Results

A. Distance of Travel Algorithm

The new algorithm, the Distance of Travel (DoT) algorithm is similar to the time of travel algorithm in that it uses token tracking. The concept of the EMD is broken with by detecting tokens across an entire line rather than just the next receptor. The principle of operation is shown in Figure 7.

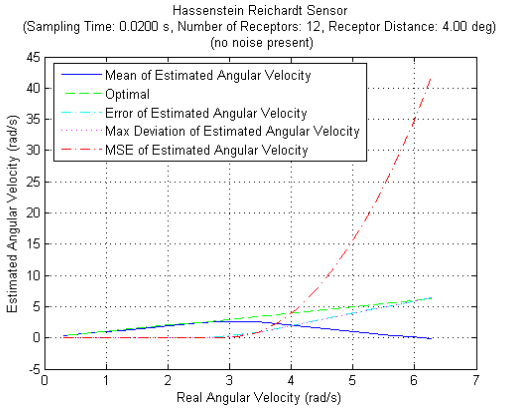

Figure 4: Performance of the Hassenstein-Richardt algorithm simulation at sampling rates of 50 Hz.

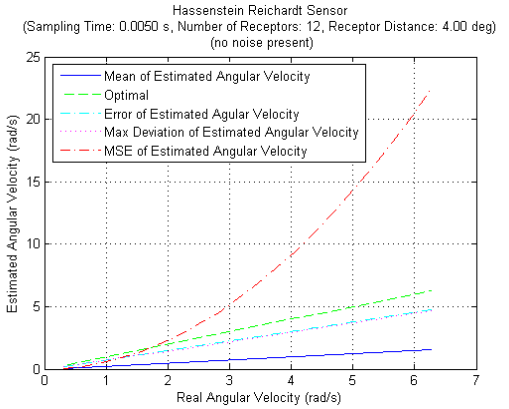

Figure 5: Performance of the Hassenstein-Richardt algorithm simulation at sampling rates of 200 Hz.

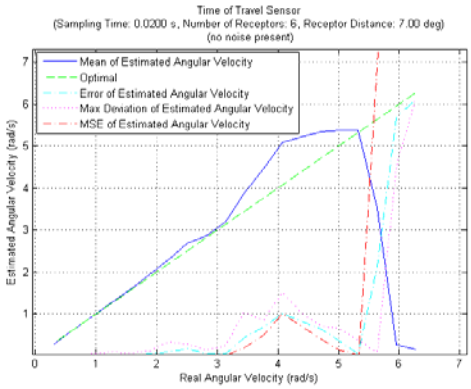

Figure 6: Performance of the Time of Travel algorithm at sampling rates of 50 Hz.

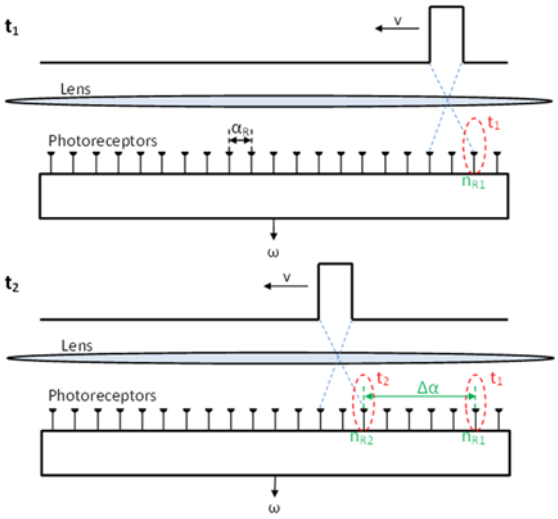

Figure 7: Principle of operation: Distance of Travel Algorithm

At t_1 a token is detected at the second receptor and again at t_2, at the seventh receptor. The angular velocity can be calculated using (1.)

$$\omega = \Delta\alpha/\Delta t = (\eta_{R2} - \eta_{R1}) \bullet \alpha_R / (t_2 - t_1) \qquad (1)$$

This measure may be improved by more detections of the token across the array. If the token reaches the end of the array the search algorithm returns to the beginning of the array. If a second token is detected then the algorithm will produce a stream of velocity values. there is a new token the algorithm will be able to produce a stream of measurement values (Figure 8.)

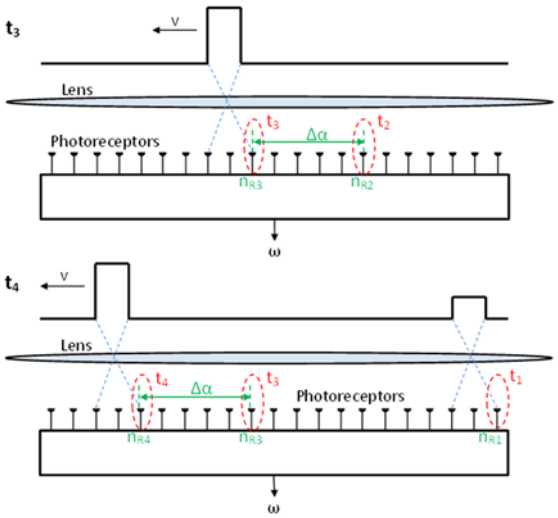

Figure 8: Principle of operation: Distance of Travel Algorithm - case of token leaving field of vision

B. *Simulation Results*

The same benchmark was applied to the Distance of Travel implementation according to the parameters in Figure 9 and Table 3 with results shown in Figure 10.

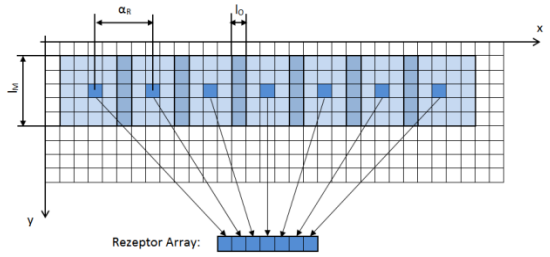

Figure 9 Assembling receptors from camera pixels for the Distance of Travel Algorithm

Table 3: Fly-Eye Parameters for the Distance of Travel detector

Parameter	Value
Distance between receptors (α_r)	0.36 °
Receptor size	5 * 5
Receptor Overlap	1 * 5
Number of Receptors	100

IV. Distance of Travel Algorithm Implementation and Tests

A. *Implementation*

With the principle of the algorithm confirmed by these simulations the algorithm was implemented on the low-cost, but discontinued, COTS leanXcam from SCS. This camera, is based on a 1/3" CMOS colour sensor and features a 500MHz Blackfin under uClinux, (SCS 2016.)

The implementation runs in an endless loop with both the run time of the software and the image capturing/transfer time determining the timing characteristics. We also measure the runtime of the algorithm to ensure an adequate frame-rate is achieved.

The application can be setup up using a web interface. The camera chip is polled, using supplied libraries, for a new image. The velocity value output is via the camera's Ethernet console output.

The implementation also performs an intensity check before deciding whether to search for a token or not.

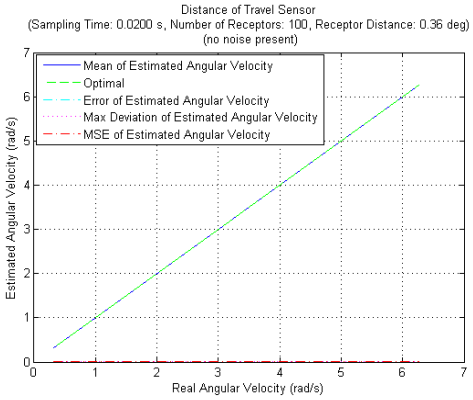

Figure 10: Performance of the Distance of Travel Algorithm simulation at sampling rates of 50 Hz.

B. *Measurements*

In order to get a direct correlation between the simulation and the implementation the same test patterns were used. A test-jig based around a conveyer belt was built (Figure 11.) The images previously generated were printed and stuck onto the conveyer belt (black and white image on the conveyer belt in Figure 11) and the speed of the conveyer belt could be adjusted.

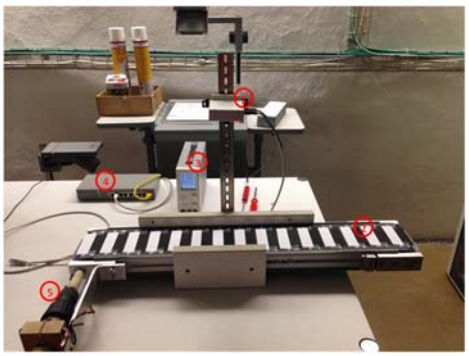

Figure 11: Picture of test-jig

The schematic in Figure 12 shows the jig parameters used in the tests and in most of the tests the parameters noted in Table 4 were used.

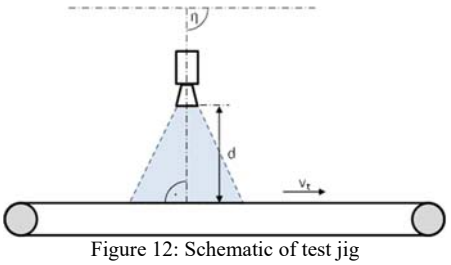

Figure 12: Schematic of test jig

Table 4: Parameter Settings of Test-Jig

Parameter	Value
Distance camera to the belt (d)	0.2 m
Angle of Camera to belt (η)	90°
Velocity of belt (vt)	0.2 – 0.8 m/s
Illumination	room
Illumination time	0.05 s
Test pattern	black/white stripe

The first results showed a strong correlation between the measured velocity and that of the conveyer belt. Oscillations were also noticed and further investigation showed that the motor was periodically sticking. The measurement was repeated and current and voltage were graphed against the velocity (Figure 13). The estimated angular velocity tracks the "stickiness" of the motor/belt, as measured by motor voltage/current, very well.

The DoT algorithm shows an excellent correlation between expected and measured velocity. The deviation observable at speeds greater 0.75 m/s is the result of an array error discovered later.

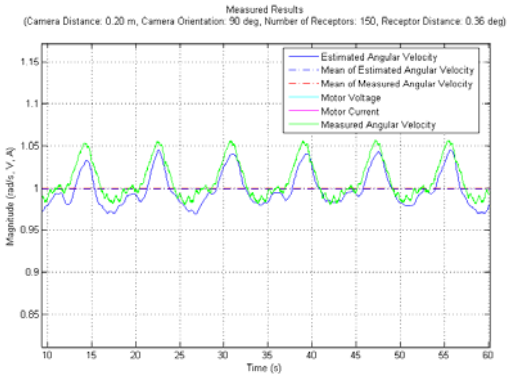

Figure 13: Correlation of Distance of Travel algorithm with motor current

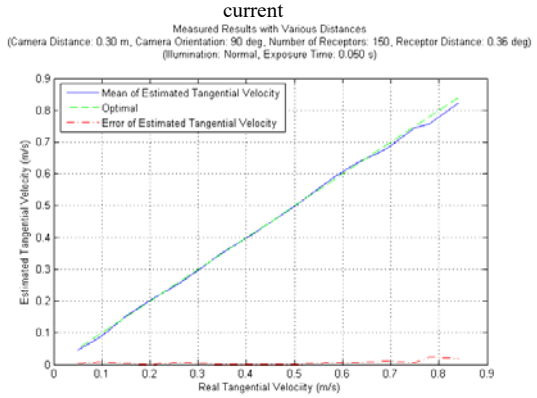

Figure 14: Performance of the Distance of Travel algorithm implementation at sampling rates of 50 Hz. and using the stripes pattern

1) Wall pattern,

The DoT algorithm was also applied to the wood, brick and concrete patterns with results listed in Table 5.

Table 5: Measurement error of Distance of Travel implementation using various test patterns

Test Pattern	Average measured angular velocity (rad/s)	Average estimated angular velocity (rad/s)	Error %
Stripes	0.67	0.66	1.5
Brick	0.67	0.73	9.0
Concrete	0.67	0.72	7.5
Wood	0.67	0.74	10.5

2) Real-Time Constraints

The real-time performance of the algorithm is of interest so the time for the algorithm to determine velocity on a per captured frame basis was measured over a time period of 60 seconds. An average CPU time of 171 µs was determined (Figure 15.) The operating system interfering with the execution of the tasks is responsible for the peaks seen > 200 µs. In contrast the de-Bayering algorithm took an average of 514 µs for execution on this platform.

V. Conclusion, Discussion and Further Work

A. *Conclusion*

We have shown by simulation that current implementations of the fly-eye algorithm for auto-velocity detection are unsuitable for implementation on low-cost commercially available cameras.

We have proposed a new algorithm, called the Distance of Travel algorithm, which is suitable for implementation on low-cost commercial cameras operating at low sampling rates. We implemented this algorithm and showed through tests that the promise shown through simulation is reflected in real-world measurements. The real-time characteristics of the implementation are also attractive.

B. *Discussion*

The Distance of Travel algorithm appears to be quite useful in structured environments. In less structured environments inspection shows that the nature of the pattern – in essence a derivative of Shannon's sampling theory – determines the achievable accuracy.

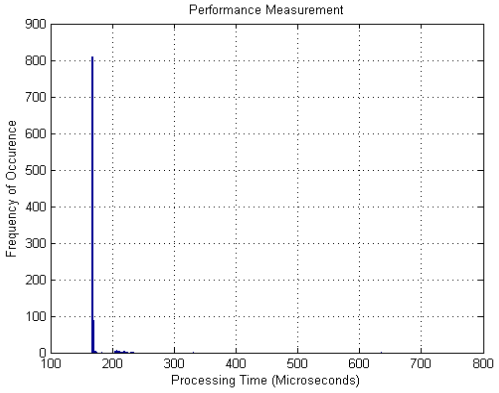

Figure 15: Processing time distribution Distance of Travel implementation

C. *Further Work*

The first focus for further research is developing 2 and 3-D versions of the algorithm as well as mounting the camera(s) on a mobile robot and allowing it to drive semi-autonomously down a corridor at speeds of up to 1 m/s.

The suitability for offloading the algorithm into an FPGA is also to be examined as we believe that the combination camera, CPU and FPGA – as opposed to the use of GPUs - to be the most cost efficient for mobile robotics.

ACKNOWLEDGMENTS

Thanks are due to Erich Ruff of InES for his kind support in building the measurement and test systems.

REFERENCES

X. Arreguit, F. A. Van Schaik, F. V. Bauduin, M. Bidiville, and E. Raeber, 1996 ‘‘A CMOS motion detector system for pointing devices,’’ IEEE J. Solid-State Circuits, vol. 31, no. 12, pp. 1916–1921, Dec. 1996.

H. B. Barlow and W. R. Levick, 1965 ‘‘The mechanism of directionally selective units in rabbit’s retina,’’ J. Physiol., vol. 178, no. 3, p. 477, 1965.

M.E. Basch, D.G. Cristea, V. Tiponuţ, and T. Slavici. 2010 “Elaborated motion detector based on Hassenstein-Reichardt correlator model”. In Proceedings of the 14th WSEAS international conference on Systems:Vol. I. (WSEAS), Stevens Point, Wisconsin, USA, 192-195. 2010.

M.H. Basch, D.G. Cristea, R.I. Lõrincz, and V. Tiponut. 2011 “A bio-inspired obstacle avoidance system concept for visually impaired people”. Proceedings of the 15th WSEAS international conference on Systems, World Scientific and Engineering Academy and Society (WSEAS), Stevens Point, Wisconsin, USA, 288-297. 2011

D. Floreano et al., 2009 Flying Insects and Robots. Berlin, Heidelberg: Springer-Verlag, pp. 101 -114, 2009.

N. Franceschini, 2014 "Small Brains, Smart Machines: From Fly Vision to Robot Vision and Back Again," in Proceedings of the IEEE, vol. 102, no. 5, pp. 751-781, May 2014.

R. R. Harrison and C. Koch, 1999 ”A Robust Analog VLSI Motion Sensor Based on the Visual System of the Fly,” Autonomous Robots 7, pp. 211 - 224, 1999.

G. Orchard and R. Etienne-Cummings, 2014 "Bioinspired Visual Motion Estimation," in Proceedings of the IEEE, vol. 102, no. 10, pp. 1520-1536, Oct. 2014.

W. Reichardt, 1961 ‘‘Autocorrelation, a principle for the evaluation of sensory information by the central nervous system,’’ Sensory Communication. Cambridge, MA, USA: MIT Press, 1961, pp. 303-317.

F. L. Roubieu et al., 2013 ”Two-Directional 1-g Visual Motion Sensor Inspired by the Fly’s Eye,” IEEE Sensors Journal, Vol. 13, Issue 3, pp. 1025 - 1035, 2013.

M. V. Sirnivasan et al., 1999 ”Motion detection in insect orientation and navigation,” Vision Research 39, pp. 2749–2766, 1999.

SCS, 2016 https://www.scs.ch/fileadmin/images/leanXcam/SCS_leanXcam_Datasheet_D.pdf, Last accessed 01.03.2016

J. E. Tanner, 1986 ‘‘Integrated optical motion detection,’’ Ph.D. dissertation, Eng. Appl. Sci., California Inst. Technol., Pasadena, CA, USA, 1986.

A. B. Watson and A. J. Ahumada, Jr., 1985 ‘‘Model of human visual-motion sensing,’’ J. Opt. Soc. Amer. A, vol. 2, no. 2, pp. 322–341, 1985