Bio-Inspired Optic Flow Sensors for Artificial Compound Eyes

by

Seok Jun Park

A Dissertation submitted in partial fulfillment
Of the requirements for the degree of
Doctor of Philosophy
(Electrical Engineering)
in The University of Michigan
2014

To my father Chul Jae Park and my mother Kyung Suk Park
To my father-in-law Seung Kyu Lee and my mother-in-law Myoung Soon Oh
To my lovely wife Kyoung Ah, my son Kyoung Hwi and daughter Yidam
You are the meaning of my life.

ACKNOWLEDGEMENTS

First and foremost, I would like to thank and dedicate all my success to God, my parents and parents-in-law. Without their love and care, I would not be able to accomplish this dissertation.

I want to express my sincere gratitude and appreciation to my advisor Prof. Euisik Yoon for his warm heart, continuous support, encouragement and guidance throughout my studies. He taught me the most valuable lesson in my life, how to trust myself. This will lead and inspire me to a better place for my remaining life. I could not have imagined having a better advisor and mentor for my Ph.D study. I would like to extend my sincere gratitude to my other committee members, Prof. James W. Cutler, Prof. Michael P. Flynn, Kamal Sarabandi, and Kensall D. Wise for devoting their time to review this thesis and advising me with valuable suggestions.

I am also grateful to Dr. Yong-In Han for his support and guidance during my industrial experience as a digital design engineer. I was able to have a solid background and skill sets in digital hardware implementation for customized signal processing algorithms at his group.

This list will not be complete without acknowledging my wife Kyoung Ah for her support during the past years in Michigan. If at all I am successful in life, it is not sheer luck or my brilliance; it is all my lovely wife Kyoung Ah's endless support and sacrifice. My children Kyoung Hwi and Yidam always bring me a great joy and happiness.

I would like to thank my colleagues and friends at Prof. Yoon's group and EECS for constructive and insightful discussions. I thank Jihyun Cho, Sung-Yun Park, Dr. Sun-Il Chang, Dr. Khaled AlAshmouny, Dr. Jaehyuk Choi, Dr. Xia Lou, Patrick Ingram, Yu-Chih Chen, Fan Wu, Kyunghwan Na, Adam Mendrela, Yu-Ju Lin, Kyuseok Lee, Komal Kampasi, Yu-Heng Cheng, Zhixong Zhang, Dr. John Seymour, Dr. Arata Nakajima, Dr. Bo Yang, Dr. Songjun Lee, Dr. Il-Joo Cho, and Dr. Maesoon Im, Yejoong Kim, Dongmin Yoon, Dr. Dae Young Lee, Dr. Sangwook Han, Dongsuk Jeon, Jaehun Chung, Yong Lim, …

I would like to acknowledge the financial support that I received from the U.S. Army Research Laboratory (ARL) under contract W911NF and prepared through collaborative participation in the microelectronics Center of Micro Autonomous Systems and Technology (MAST) Alliance (CTA). I would like to thank Dr. Joseph Conroy in ARL, Dr. Patrick Shoemaker in Tanner Research, and Prof. Sean Humbert in University of Maryland for their tremendous support and valuable discussion during the participation of the project. I am also thankful to Samsung Electronics for awarding me the Samsung Fellowship that generously funded the five years of my Doctoral research during 2008-2012 academic year.

TABLE OF CONTENTS

DEDICATION ii

ACKNOWLEDGEMENTS iii

LIST OF FIGURES x

LIST OF TABLES xv

ABSTRACT xvi

Chapter 1 Introduction 1

1.1 Motivation 1

1.1.1 Artificial compound eyes 1

1.1.2 Micro-air-vehicles (MAVs) and microsystems sensor capability gap 4

1.1.3 Bio-inspired wide field integration (WFI) navigation methods 7

1.1.4 Current state of the art optic flow sensing solutions for MAVs 8

1.2 Thesis outline 11

1.2.1 Chapter 2: Visual information processing in an insect’s eyes 12

1.2.2 Chapter 3: Optic flow estimation algorithms in MAVs 12

1.2.3 Chapter 4: Time-stamp-based optic flow estimation algorithm 12

1.2.4 Chapter 5: Bio-inspired 1D optic flow sensor for physical verification of the time-stamp-based optic flow algorithm 12

1.2.5 Chapter 6: Bio-inspired time-stamp-based 2D optic flow sensor for artificial compound eyes 13

1.2.6 Chapter 7: Conclusions, contributions, and future work 13

Chapter 2 Visual information processing inside a flying insect’s eyes 15

2.1 Anatomy of an insect’s compound eye 17

2.2 Elementary motion detector model 19

2.3 Wide-field optic flow analysis: lobular complex 21

2.4 Bio-inspired autonomous navigation 24

2.5 Summary 25

Chapter 3 Optic flow estimation algorithm in MAVs 27

3.1 Introduction 27

3.2 Gradient-based algorithms 28

3.2.1 1D gradient-based algorithm 28

3.2.2 Lucas and Kanade (L-K) algorithm 29

3.2.3 Pyramidal Lucas and Kanade (L-K) algorithm 30

3.3 Bio-inspired elementary motion detector (EMD) algorithms 31

3.3.1 Photoreceptor model 33

3.3.2 High pass filter, delays, and correlators 34

3.4 Summary 41

Chapter 4 Time-stamp-based optic flow estimation algorithm 43

4.1 Introduction 43

4.2 General concept of the proposed time-stamp-based motion sensing scheme 44

4.3 Time-stamp-based 1-D optic flow estimation algorithm 46

4.3.1 Time-of-travel measurement and velocity conversion 47

4.3.2 Facilitate-trigger-and-inhibit (FTI) algorithm.. 48

4.4 Time-stamp-based 2-D optic flow estimation algorithm 50

4.5 Performance comparison with respect to MAVs applications 53

4.5.1 Optic flow performance evaluation for rotational velocity extraction............ 53

4.5.2 Optic flow algorithm performance evaluation for MAV navigation 60

4.6 Summary 63

Chapter 5 Bio-inspired 1D optic flow sensor for physical verification of Time-stamp-based optic flow algorithm 64

5.1 Introduction 64

5.2 Chip architecture 65

5.2.1 A/D partitioning 66

5.2.2 Two modes of operation: optic flow generation, and normal image capture . 67

5.3 Pixel architecture 68

5.4 Column circuits 69

5.4.1 The circular time-stamp counter and sign conversion circuits 70

5.4.2 Column-parallel time-stamp sign conversion example from chip measurement… 74

5.5 Time-stamp-based optic flow estimation core 75

5.6 Chip characteristics 81

5.6.1 Measurement setup 82

5.6.2 Measurement results 83

5.6.3 Characteristics summary 87

5.7 Summary 88

Chapter 6 Bio-inspired time-stamp-based 2D optic flow sensor for artificial compound eyes 90

6.1 3D semi-hemispherical module platform for artificial compound eyes 91

6.2 System requirements of 2D optic flow sensors on the 3D module platform 93

6.3 Implemented bio-inspired 2D time-stamp-based optic flow algorithm 95

6.4 Sensor architecture 101

6.5 Circuit implementation and proposed sensor 104

6.5.1 Pixel architecture 104

6.5.2 Column parallel feature detection circuits 105

6.5.3 Digital time-stamp-based optic flow estimation core 109

6.5.4 Digital peripheral circuits: SPI controller, compressor, down-sampler, and WFI………… 112

6.5.5 Optic flow data compression core 113

6.5.6 Down-sampler and WFI 116

6.6 Experimental results 117

6.6.1 2D optic flow performance 119

6.6.2 Sample images and optic flow from real objects 121

6.6.3 Down-sampler and WFI 124

6.6.4 Performance summary and comparison 126

6.7 Conclusions 128

Chapter 7 Conclusions, contributions, and future work 130

7.1 Suggestions for future work 132

BIBLIOGRAPY 134

LIST OF FIGURES

Figure 1-1: 3D wide FoV artificial compound eyes: (a) 180°(H)×60°(V), 42×15 spatial resolution curved artificial compound eye (CurvACE) [1], (b) 160° FoV, 16×16 spatial resolution bio-inspired digital camera [2]................................ 2

Figure 1-2: 3-D semi-hemispherical module platform for the artificial compound eyes ... 4

Figure 1-3: Payload versus wingspan plot for unmanned aerial vehicles (UAVs) and MAVs [1] 6

Figure 1-4: Demonstrated WFI platform: (a) ground robot [25], (b) quadrotor [26] 8

Figure 2-1: The compound eyes of flying insects adapted by http://lis.epfl.ch/curvace .. 15

Figure 2-2: A schematic representation of the fly's visual and central nervous system (cross section through the fly's brain). Photoreceptor signals are transmitted to the lamina, which accentuates temporal changes. A retinotopic arrangement is maintainted through medulla. The lobula plate is made up of wide-field, motion-sensitive tangential neurons that send information to the controlateral optic lobe as well as to the thoracic ganglia, which control the wings [7-8]. 18

Figure 2-3: The Hassenstein-Rechardt or correlational elementary motion detector (EMD) 21

Figure 2-4: Self-motion and optic flow. Left: self-motion parameter of the fly can be described in terms of their components along the three cardinal body axes.

Right: self-motion results in optic flow fields representing how the visual world moves relative to the fly’s eyes. [52]. 23

Figure 2-5: A hypothetical filter neuron matched to a particular optic flow field induced by rotating self motion adapted by [53]. 23

Figure 2-6: Perturbations of azimuthal optic flow and their correlations to the relative state of the vehicle. Amplitude, phase, and asymmetry of the azimuthal optic flow pattern... 25

Figure 3-1: The Hassenstein-Rechardt or correlational elementary motion detector (EMD) 32

Figure 3-2: Adaptive photoreceptor [57] 33

Figure 3-3: Adaptive photoreceptor response to the ambient light [1] 34

Figure 3-4: EMD output using LPFs as delay cells 35

Figure 3-5 Facilitate-and-inhibit (FI) optic flow estimation algorithm 36

Figure 3-6: Photoreceptor linear array (left) and electronic board (right) with 12x12mm FPGA [68] 38

Figure 3-7: On/off pathways and feature arrival/departure relationship 40

Figure 4-1: General concept of the time-stamp-based motion sensing scheme................ 46

Figure 4-2: Time-stamp-based 1-D time-of-travel measurement 48

Figure 4-3: Traditional time-of-travel measurement, facilitate-and-inhibit (FI) 49

Figure 4-4: Facilitate-trigger-and-inhibition (FTI) time-of-travel measurement............. 50

Figure 4-5: 2-D optic flow estimation example.. 52

Figure 4-6: 3x3 mask examples at different time and locations 53

Figure 4-7: A 360 ° wide-field high-resolution imagery example for rotational velocity evaluation 54
Figure 4-8: Flow chart for evaluation of rotational velocity extraction............................ 55
Figure 4-9: Sensor configuration to analyze the detectable velocity range 57
Figure 4-10: Generated optic flows from rotational velocity evaluation.......................... 59
Figure 4-11: Detected velocity plot for different input rotational velocities 59
Figure 4-12. A flybot flight simulation: (a) Virtual flight arena, (b) ten fight paths of an MAV [85] 61
Figure 4-13. Optic flow sensor configuration and a captured image for the performance evaluation. 61
Figure 4-14. Navigation performance evaluation environment setup 62
Figure 5-1: First prototype time-stamp-based optic flow sensor architecture 66
Figure 5-2: Pixel architecture and equivalent circuit 69
Figure 5-3: Column circuits and implemented 1-D time-stamp-based optic flow core ... 70
Figure 5-4: The implemented circular time-stamp counter and value reordering 73
Figure 5-5: Column-parallel no-stamp code decoding logic 73
Figure 5-6: Column-parallel time-stamp value reordering logic for the 8-b circular counter 73
Figure 5-7: Column-parallel recorded time-stamp clearing logic..................................... 74
Figure 5-8: Column-parallel digital time-stamp circuits implemented using 0.9V supply 74
Figure 5-9: Column-level time-stamp sign conversion 75
Figure 5-10: Chip-level time-stamp-based 1-D optic flow processing core 79

Figure 5-11: Implemented digital facilitate-trigger-and-inhibition (FTI) circuits........... 80

Figure 5-12: Valid stamps detection logic 80

Figure 5-13: A velocity conversion look-up table (LUT).. 80

Figure 5-14: Chip micrograph 81

Figure 5-15: Block diagram of measurement environment for the first prototype chip... 83

Figure 5-16. 1-D optic flow test plots with the on-chip digital core................................ 85

Figure 5-17. 2-D optic flow test result constructed from 1-b feature information generated by the fabricated chip 87

Figure 6-1: Artificial compound eyes on the 3-D semi-hemispherical module platform. 90

Figure 6-2: The conceptual diagram of 3D semi-hemispherical module platform........... 92

Figure 6-3: The uni-axial sensing module of the 3D semi-hemispherical module platform 92

Figure 6-4: System architecture of the artificial compound eye module.......................... 95

Figure 6-5: Concept diagram of 2D time-stamp information 96

Figure 6-6: A bio-inspired moving object arrival/departure detection 98

Figure 6-7: Implemented time-stamp-based 2D optic flow estimation algorithm.......... 101

Figure 6-8: Prototype time-stamp-based 2D optic flow sensor architecture 102

Figure 6-9: Pixel architecture: schematic and layout... 105

Figure 6-10: Feature detection circuits: schematic and timing diagram......................... 108

Figure 6-11: Digital time-stamp-based optic flow estimation core and peripheral circuits 109

Figure 6-12: The implemented circular time-stamp counter and value reordering 111

Figure 6-13: Implemented 2D optic flow data compression algorithm.......................... 115

Figure 6-14: Measurement setup 118

Figure 6-15: Chip micrograph 118

Figure 6-16: Characterized 2D Optic flow performance 121

Figure 6-17: Sample images and captured optic flow from real moving objects 123

Figure 6-18: Down-sampler performance.. 124

Figure 6-19: WFI core performance 126

LIST OF TABLES

Table 1-1: Payload and power consumption of available sensors for a MAV's flight navigation 7

Table 1-2: Estimated total power consumption in the pure digital approaches 9

Table 3-1: The relationship between on/off edges and the departure/arrival of objects... 40

Table 4-1: Correlation between the optic flow generated from each optic flow algorithm and the correct optic flow using the test images 63

Table 5-1: Hardware resource comparison of a discrete temporal high-pass filter implementation 67

Table 5-2: Measurement plan for the first prototype chip 83

Table 5-3: Chip characteristics 88

Table 5-4: Performance comparison with previous works 88

Table 6-1: Serial bus protocols used in MAV systems 93

Table 6-2: Equations for 2D time-stamp-based optic flow estimation 112

Table 6-3: Chip characteristics 127

Table 6-4: Performance comparison of optic flow sensors and systems 128

ABSTRACT

Bio-Inspired Optic Flow Sensors for Artificial Compound Eyes

by

Seok Jun Park

Compound eyes in flying insects have been studied to reveal the mysterious cues of vision-based flying mechanisms inside the smallest flying creatures in nature. Especially, researchers in the robotic area have made efforts to transfer the findings into their less than palm-sized unmanned air vehicles, micro-air-vehicles (MAVs). The miniaturized artificial compound eye is one of the key components in this system to provide visual information for navigation. Multi-directional sensing and motion estimation capabilities can give wide field-of-view (FoV) optic flows up to 360 ^{o}solid angle. By deciphering the wide FoV optic flows, relevant information on the self-status of flight is parsed and utilized for flight command generation. In the last few years, several artificial compound eyes have been demonstrated by forming a hemispherical lens configuration to realize an independent optical path to each photoreceptor. However, they require complicated fabrication processes, limiting their applications due to poor yield and high cost.

In this work, we take a simple and practical approach. We realize the wide-field optic flow sensing in a pseudo-hemispherical configuration realized by mounting a number of 2D array optic flow sensors on a flexible PCB module. The flexible PCBs can be bent to form a compound eye shape by origami packaging. From this scheme, the multiple 2D optic flow sensors can provide a modular, expandable configuration to meet low power constraints.

The 2D optic flow sensors satisfy the low power constraint by employing a novel bio-inspired algorithm. We have modified the conventional elementary motion detector (EMD), which is known to be a basic operational unit in the insect's visual pathways. We have implemented a bio-inspired time-stamp-based algorithm in mixed-mode circuits for robust operation. By optimal partitioning of analog to digital signal domains, we can realize the algorithm mostly in digital domain in a column-parallel circuits. Only the feature extraction algorithm is incorporated inside a pixel in analog circuits.   In addition, the sensors integrate digital peripheral circuits to provide modular expandability. The on-chip data compressor can reduce the data rate by a factor of 8, so that it can connect a total of 25 optic flow sensors in a 4-wired Serial Peripheral Interface (SPI) bus. The packaged compound eye can transmit full-resolution optic flow data through the single 3MB/sec SPI bus. The fabricated 2D optic flow prototype sensor has achieved the power consumption of 243.3pJ/pixel and the maximum detectable optic flow of 1.96rad/sec at 120fps and 60 ºFoV.

CHAPTER 1
INTRODUCTION

1.1 Motivation

1.1.1 Artificial compound eyes

For a recent decade, research efforts have been made on building miniaturized artificial compound eyes. The bio-inspired devices mimic a flying insect's visual organs and signal pathways to provide valuable information from wide field-of-view (FoV) surroundings [1]–[3]. The wide FoV multi-directional sensing is the most interesting capability of the organs that covers nearly 360 ° from both compound eyes. In addition, the collected visual information is processed to generate optic flow of the encompassing environment. Then, the neuron nerves in the insect's eyes parse the wide FoV optic flow to extract key cues for its safe and agile flying navigation [4]. Many researchers in robotics are focusing on understanding the flying mechanism and visual information sensing. With the information, they are trying to implement the mechanism on their robot platforms, especially on micro air vehicles (MAVs) which are the smallest class of unmanned aerial vehicles [5]–[15]. This effort is natural in that researchers examine the mechanism inside the smallest flying creature in the world to discover solutions that will work under the situations and harsh design constraints that cannot be achieved by conventional UAV soultions.Two recently demonstrated 3D-shaped wide FoV artificial compound eyes are shown in Figure 1-1. The curved artificial compound eye (CurvACE) in Figure 1-1(a) supports 180 ° FoV in horizontal and 60 °FoV in vertical directions. The module is assembled on a flexible PCB; 42 linear array including 15 photodetectors are located on top of the PCB with the customized micro lenses. Then, the linear array is carefully cut along with all the vertical lines to give flexibility in horizontal direction. Finally, the PCB is bent to form a cylinder shape configuration to achieve 180 °FoV in horizontal direction. Because in this configuration the PCB is not able to bend in the vertical direction, the vertical FoV is limited. However, by patterning the apertures in different spaces from the photodetectors, the device guides different angles of incident light path in the vertical direction. Thus, 60 °FoV is achieved. Both acceptance and inter-photodetectors angles are also carefully designed in the CurvACE sensor. The preliminary work for investigating the relationship can be found in [3], [16]–[18].

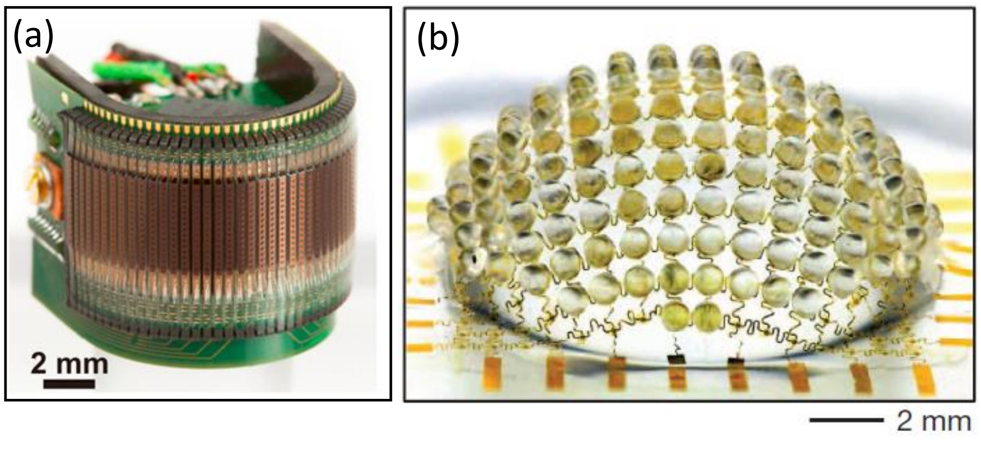


Figure 1-1: 3D wide FoV artificial compound eyes: (a) 180 °(H)×60 °(V), 42×15 spatial resolution curved artificial compound eye (CurvACE) [1], (b) 160 °FoV, 16×16 spatial resolution bio-inspired digital camera [2]

A different approach to form 3D hemispherical artificial compound eyes is shown in Figure 1-1(b). The device combines elastometric compound optical elements with deformable arrays of thin silicon photodetectors into integrated sheets that can be elastically transformed from the planar geometries in which they are fabricated to hemispherical shapes for integration into apposition cameras Figure 1-1. The stretchable micro lens and photodetector arrays form the shape on hemispherical supporting substrate of black silicone. Thus, this approach achieves 160° FoV for all surroundings. The photodetectors underneath the hemisphere collect light from 16×16 array and deliver the measured photo currents through the contact matrix at the bottom stretchable layer. The sensor does not integrate optic flow or motion sensing capability yet. The CurvACE embedded a customized optic flow estimation algorithm in the MCU on the discrete thick PCB, which is also used for supporting its cylindrical shape. The CurvACE system consumes 0.9W maximum power, which could be reduced by integrating the optic flow and interface circuits in the same silicon.

The aforementioned systems require complicated fabrication and assembly processes to form a hemispherical lens configuration and secure an independent optical path to each photodetector. Instead of relying on the customized technology, we take a simple and more practical approach to realize wide-field optic flow sensing in a pseudo-hemispherical configuration by mounting a number of 2D array optic flow sensors on a flexible PCB module as shown in Figure 1-2. Multiple pairs of a lens and 2D optic flow sensors, which are implemented using the conventional image sensors assembly process, are located on top of each segment of a flexible PCB. Then, the origami packaging technique folds the segments to form desired inter-sensor angles to build a semi-hemispherical shape configuration. Thus, the artificial compound eyes can be assembled only applying existing technologies. The main challenge of the approach is customizing 2D optic flow sensors being capable of modular expandability so that identical sensors can capture and transmit optic flow on the multiple sensors in a module platform. Each sensor is required to support modular expandability being able to easily attach and detach from the system. In addition, 2D optic flow estimation must consume extremely low power to engage multiple sensors on MAV applications.

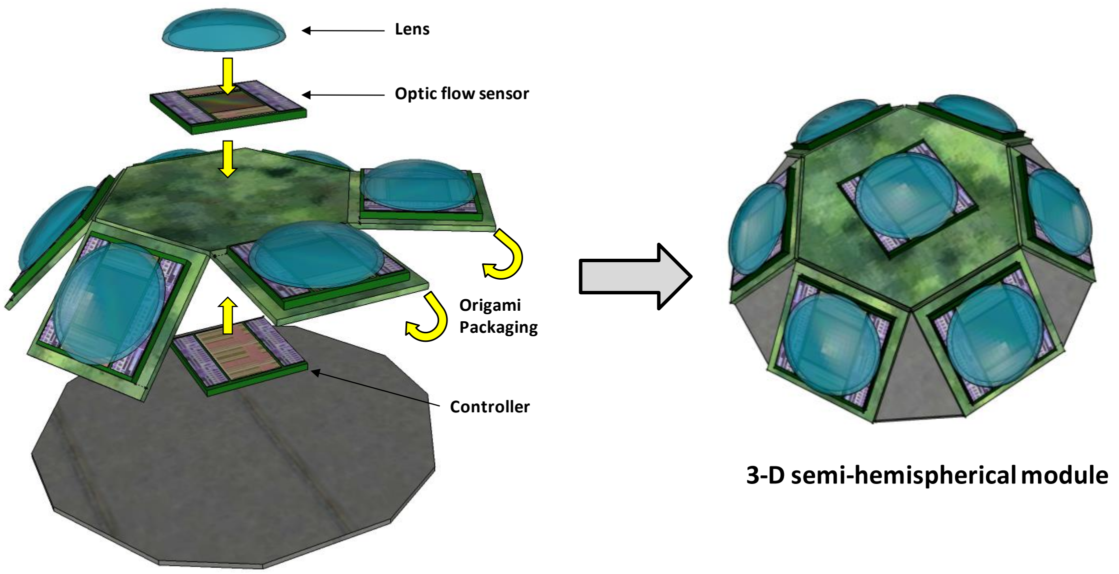


Figure 1-2: 3-D semi-hemispherical module platform for the artificial compound eyes

1.1.2 Micro-air-vehicles (MAVs) and microsystems sensor capability gap

This small-size MAV, whose size ranges up to a few centimeters, is designed to perform various missions in dangerous places that threaten human life. For example, Micro Autonomous Systems and Technology (MAST) alliance [19] defines mission scenarios for MAVs: cave searches, demolished building searches, and perimeter defense.

While performing cave searches and demolished building search missions, soldiers are exposed to unpredictable dangers; in perimeter defense, soldiers are always facing to an abrupt enemy attack. Therefore, the mission success of MAVs is thankfully saving tremendous amounts of human life. The demolished building mission can also save people from the dangers of fire or toxic gas leakage.

The design constraints of MAVs in terms of payload and wingspan, the distance from one to the other wingtips, are shown in Figure 1-3 [20]; the constraints for previously implemented unmanned aerial vehicles (UAVs) are also shown. Because both constraints for MAVs are less than two orders of magnitude than those of previously implemented UAVs, most mechanical components and electronics are no longer reused by just scaling its physical dimensions; those aggressive constraints have changed everything to build the smallest class flying system. In addition to the constraints, an extremely limited power source due to the MAV’s capability of loading only a very small-size battery is another big hurdle to overcome. As a result, researchers have borrowed mechanisms and vision-based navigation schemes from inside flying insects to meet those constraints of MAVs: aerodynamics modeled from inside the wings of a fly [21]–[24], a control theory based on an insect’s vision-based flight navigation [25]–[29], and a motion sensing scheme of from insect’s eyes [30]–[32]. These mechanisms are being proposed as several solutions for the aforementioned constraints; a vision-based navigation system is a good example to achieve the goals.

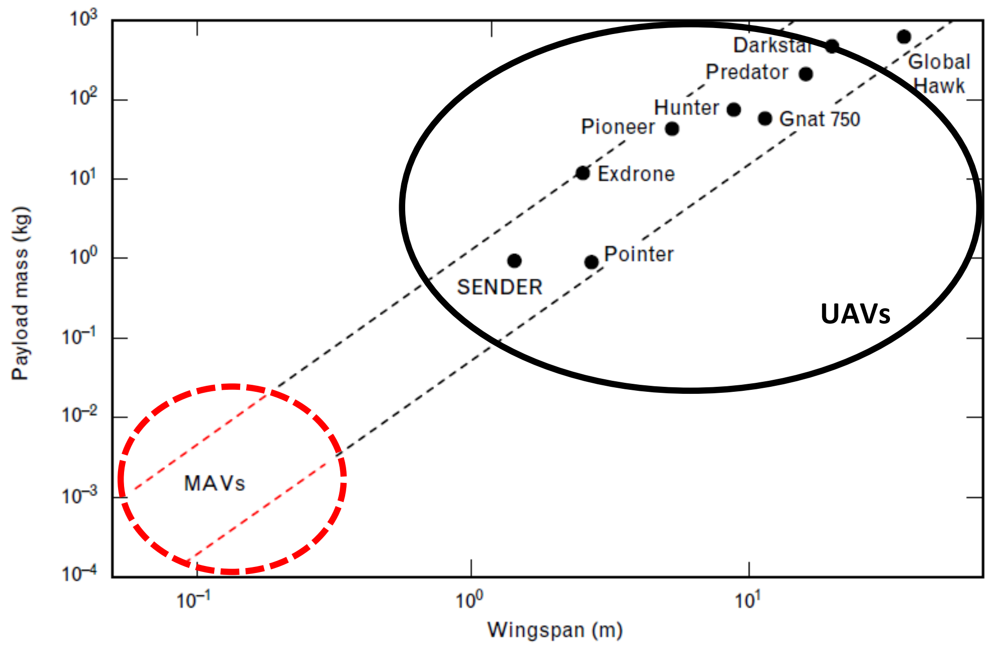


Figure 1-3: Payload versus wingspan plot for unmanned aerial vehicles (UAVs) and MAVs [1]

Sensors that are utilized for UAVs or able to be used for MAVs for fight navigation are listed in Table 1-1. Target constraints for payload and power consumption are around 1g and 100mW ranges for electronic devices on a MAV system. A global positioning system (GPS), the most popular solution for UAVs, is no longer available for MAVs not only because the places of a MAV's missions are too harsh for a GPS to correctly operate (indoor or between buildings), but also because the payload and power consumption of the GPS are much higher than the constraints as shown in Table 1-1. An inertial measurement unit (IMU), a laser rangefinder, and an ultrasonic device do not satisfy the constraints in an order of magnitude. An analog neuromorphic optic flow sensor, which is implemented by mimicking the motion sensing scheme in an insect's compound eyes, is the only sensor to meet both payload and power constraints.

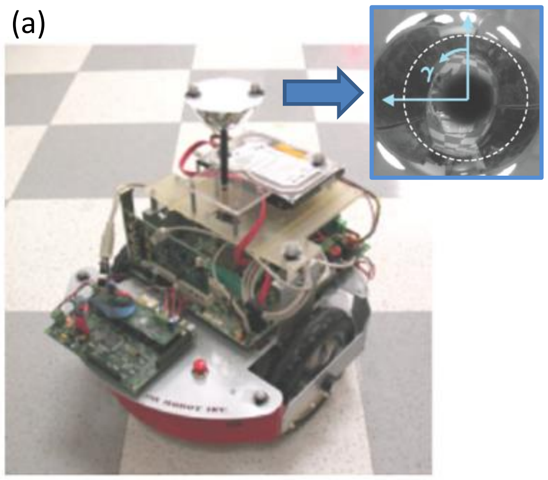
Table 1-1: Payload and power consumption of available sensors for a MAV's flight navigation

Sensor	Weight (g)	Power (mW)
GPS	16-28	770-910
IMU	3	250
Laser Rangefinder	26	40
Ultrasonics	10	1000
Analog VLSI Optic Flow Sensor	0.25	0.01
Target	1	100

1.1.3 Bio-inspired wide field integration (WFI) navigation methods

Research findings on a flying insect's visual pathway reveal that neurons in the compound eyes interpret the information about self-flight status from wide FoV optic flow [4]. For example, the velocity of piloting, a distance from possible obstacles, and even self-motion can be parsed by the neurons. This mechanism has inspired to develop the wide field integration navigation (WFI) methods [25]–[27], [33]. The methods apply matched filters on the measured wide-field optic flow to extract cues containing self-status of flight. Then, the motor commands feedback and move the vehicles to maintain the target status. The demonstrated systems in Figure 1-4 verify 3 degree-of-freedom (DoF) controls are possible in a corridor by the methods [25], [26], [33]. As shown in Figure 1-4(a), the system mounts an image sensor or camera and a parabolic mirror on top of the sensor to monitor wide FoV scenes. Then, optic flow is extracted on concentric circles, which correspond to the wide-field flow from the surrounding equator.

(a)



(b)

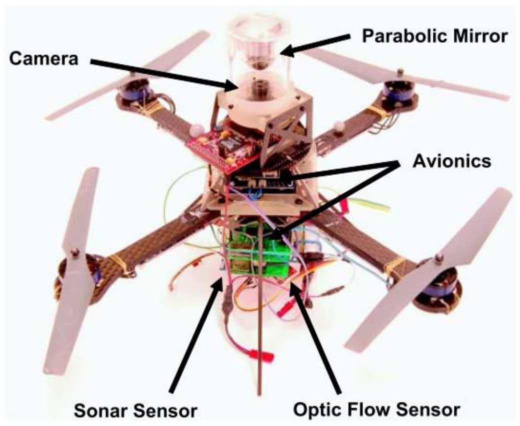


Figure 1-4: Demonstrated WFI platform: (a) ground robot [25], (b) quadrotor [26]

In addition, a theory expanding the WFI for 6 DoF control is published [27]. For 6 DoF control, the theory requires optic flow from the 3D surrounding sphere, not 2D concentric circles in the Figure 1-4. Therefore, the artificial compound eyes are key components in the system.

1.1.4 Current state of the art optic flow sensing solutions for MAVs

In addition to the WFI methods, a variety of control schemes are successfully demonstrated on MAV systems by integrating optic flow sensing systems or devices for collision avoidance [34] and altitude control [35]. Two state-of-the-art approaches for optic flow sensing implementation are described: the pure digital system approach and pure analog VLSI approach.

1.1.4.1 Pure digital optic flow sensing approach

The pure digital optic flow sensing approach implements computation intensive signal processing algorithms such as Lucas and Kanade (L-K) [36], which are the state-of-the-art highly accurate optic flow estimation algorithms, on hardware. This hardware consists of an image sensor to extract video scenes and a microcontroller (MCU) or a field programmable gate array (FPGA) to compute the high computational cost algorithms [37]. Though this approach provides very accurate optic flows, it is difficult to meet a MAV's payload and power consumption constraints: discrete components increase payload, image sensors, MCUs, and FPGAs consume too much power to compute the algorithms. For example, the SXGA image sensor consumes 120mW [38]; the Blackfin processor, which is a commercially available low power MCU adopted for the implementation in Conroy [26], consumes around 64mW [39]. FPGAs consume 50mW [40]. Therefore, even though pure digital approach provides accurate optic flows, this hardware implementation has a huge gap for MAV applications.

Table 1-2: Estimated total power consumption in the pure digital approaches

	CIS+MCU [1]	**CIS+FPGA [2]**
CMOS Image Sensor	120*	-
Blackfin Processor	64**	-
FPGA	-	50**
Total	184	>50 including CIS

* From deatsheet, ** estimated Unit [mW]

1.1.4.2 Pure analog VLSI optic flow sensors

Pure analog VLSI optic flow sensors, which are neuromorphic circuits mimicking neuro-biological architectures in an insect's visual pathways, are categorized by two purposes. The first type is implementing the full insect's visual signal processing chain as similar as possible in order to verify a hypothesis made to explain physiological experiment results. As a byproduct, the sensors also can be used for the MAV applications [30]. The other type of pure analog optic flow sensors selectively chooses the aspects inside a flying insect's eyes to effectively implement on the sensor [32]. Thus, the sensors are compactly designed by modifying the original models.

Both pure analog neuromorphic approaches provide a monolithic solution and consume only ranges from nW to μW; therefore, it is considered a promising solution to overcome MAVs payload and power constraints. However, the pure analog neuromorphic circuit is sensitive to temperature and process variations; thus, output from the circuits is susceptible to noise. In addition, neuromorphic signal processing is implemented in pixel-level; a pixel consists of quite a few transistors. Thus, a small pixel design is difficult. Also, this in-pixel implementation causes pixel-to-pixel output variations due to circuit mismatches.

1.1.4.3 Analog/digital (A/D) mixed-mode optic flow sensor

The focus of this thesis is to address the drawback of the aforementioned pure digital and analog approaches by introducing the A/D mixed-mode approach. The approach is realized by the time-stamp-based optic flow algorithm, which is developed for the optimal A/D partitioning. The algorithm borrows motion estimation concepts from a flying insect’s neuromorphic algorithm; however, it differently maps the concepts into circuits so that the approach selectively takes advantages of both pure digital and analog circuits. The time-stamp-based algorithm implements the moving feature extraction using a temporal high-pass filter in the analog domain as the compound eye’s Lamina lobe does. Then, the algorithm performs time-of-travel measurement and velocity conversion such as the neuromorphic EMD; however, the time-of-travel value is calculated by simple digital arithmetic operators for robust optic flows. As a result, the proposed approach achieves low power consumption in mainly two ways: reducing digital calculation power inherited by the compact bio-inspired algorithm and minimizing static power consumption by only activating analog amplifiers for the readout period, which is possible to operate the sensor in discrete-time domain.

1.2 Thesis outline

This thesis presents bio-inspired optic flow sensors that are customized for the semi-hemispherical artificial compound eyes platform. Visual signal pathways in a flying insect’s compound eyes and a variety of bio-inspired optic flow estimation algorithms are discussed as background research. Bio-inspired time-stamp-based optic flow algorithm and its circuit implementation are following to address the A/D mixed-mode approach. Then, two prototype sensors are introduced. The 1D optic flow sensor is fabricated and characterized to verify the developed bio-inspire time-stamp-based algorithm; the 2D optic flow sensor demonstrates the optic flow core performance as well as the feasibility to be mounted on the artificial compound eyes platform by integrating additional peripheral circuits.

1.2.1 Chapter 2: Visual information processing in a flying insect’s eyes

This chapter covers the basic anatomy of an insect’s compound eyes to elucidate its mechanism of vision-based flying control. I describe an elementary motion detector model that is known to estimate optic flows in the compound eyes. Also, the parsing cues from wide FoV optic flow in tangential cells in the visual organ is introduced. Then, I seek to understand how these visual cues are utilized to control a MAV’s autonomous navigation.

1.2.2 Chapter 3: Optic flow estimation algorithms in MAVs

This chapter covers two categories of optic flow algorithms that are adopted on unmanned vehicles for vision-based autonomous navigation. Computationally intensive high-accurate gradient-based algorithms and bio-inspired low computation power EMD-based algorithms are introduced.

1.2.3 Chapter 4: Time-stamp-based optic flow estimation algorithm

This chapter covers the proposed time-stamp-based optic flow algorithm from a conceptual level to detailed 1D and 2D optic flow examples. At the end of the chapter, our two algorithm evaluation methods are introduced and used to assess the algorithm in terms of a MAV’s autonomous navigation requirements.

1.2.4 Chapter 5: Bio-inspired 1D optic flow sensor and physical verification of the

time-stamp-based optic flow algorithm

This chapter covers from the design to measurement of the 1D time-stamp-based optic flow sensor. Firstly, the proposed sensor architecture to efficiently implement the time-stamp-based algorithm in analog/digital (A/D) mixed-mode circuits is introduced. Secondly, each block design of the proposed architecture is explained in the signal processing order: a discrete-time temporal high-pass filter; feature detection circuits; time-stamp latch; and digital time-of-travel and velocity calculation circuits. Finally, the measurement results of the fabricated sensor are discussed.

1.2.5 Chapter 6: Bio-inspired time-stamp-based 2D optic flow sensor for artificial compound eyes

This chapter introduces the bio-inspired 2D optic flow sensor which is not only composed of time-stamp-based 2D optic flow core, but also integrates all the required periphery circuits for the proposed artificial compound eyes platforms. The embedded optic flow data compression core supports data reduction on the 16-b/pixel raw optic flow; as a result, in average only 1.92-b/pixel is required to transmit. Therefore, up to 25 sensors can send the full resolution optic flow data on the 3MB/sec SPI bus without any loss. The digital WFI core is also integrated to provide the parsed cues from measured 2D optic flow for a MAV's autonomous navigation. The future direction of the research is also discussed at the end of the chapter.

1.2.6 Chapter 7: Conclusions, contributions, and future work

This chapter summarizes final conclusions and highlights contributions of this work, and suggests possible and meaningful further works in this research area.

CHAPTER 2
VISUAL INFORMATION PROCESSING INSIDE A FLYING INSECT'S EYES

In a flying insect's body, eyes occupy most of its head and provide important visual information for flying control [41]. As shown in Figure 2-1, the huge eyes are composed of many tiny repeating units (ommatidia), which include a lens and a photoreceptor to independently sense light. Thus, the eyes are called compound eyes. Each hemispherical compound eye can see a wide field of view (FOV), approximately 180 º. Therefore, two compound eyes allow a flying insect to see nearly 360 ºand thereby continuously extract visual information from its full surroundings [42].

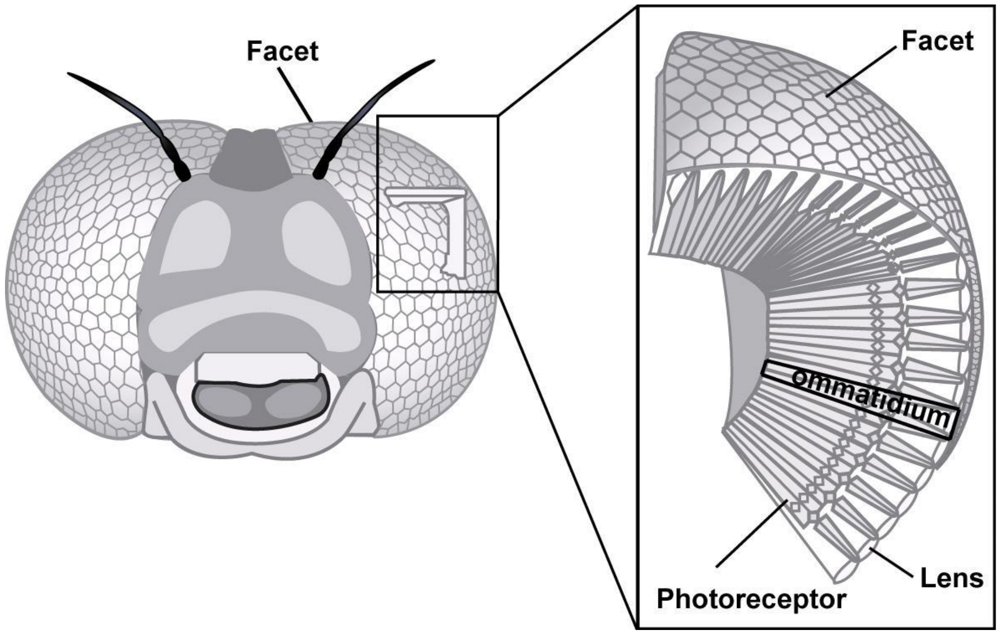


Figure 2-1: The compound eyes of flying insects adapted by http://lis.epfl.ch/curvace

In addition to sensing the aforementioned wide FOV, the insect's eye also has aspect namely it is immobile and consists of fixed-focus optics [43]. With these two features, an insect's eye cannot extract distance information from either stereo vision or focus control, as human eyes do. Instead, the compound eye infers distance information from estimating motions (or optic flows) based on the phenomenon that a closer object causes faster motion than a farther object.

Because compound eyes must efficiently detect motion information, they have high temporal resolution, whose maximum detectable frequency ranges from 200-300 Hz whereas human eye is only sensitive to up to 20 Hz [44]. However, their spatial resolution is poorer than that of human eyes. Spatial resolution is partially attributed to physical limitation that is the small size of an insect eye. Specifically, the total number of ommitidia in a compound eye ranges from only 700 in the fruitfly to merely 6,000 in the blowfly, a member of Diptera, which is a large order of insects that comprises the true flies. In comparison, this number of detection units is significantly smaller than human retina (10^8 for rods and 10^6 for cones) and even in commercially available artificial image sensors ($3x10^5$ for VGA resolution and $2x10^6$ for HDTV resolution) [45]. These differences imply an insect's vision system has evolved in a quite different manner than human vision has and is worth reviewing to understand how visual information is processed to find clues for their flight control.

This chapter covers the basic anatomy of an insect's compound eyes to elucidate its mechanism for vision and flying control. To facilitate understanding the eye, I describe an elementary motion detector model that estimates optic flows. This model explains how neighboring ommatidia to coordinate local motion information. Also, because an insect's eye integrates all the local motions into a combined wide-field motion view, I present a wide-field optic flow analysis model. In total, I seek to understand how an insect uses all clues from optic flow field to control its body in flight.

2.1 Anatomy of an insect’s compound eyes

The anatomy of an insect’s compound eyes that is relating on an flying insect’s navigation is concisely summarized by Zeffery [46]. The anatomy for visual information processing is mainly quoted from the reference. Three main optic lobes for visual information processing in a flying insect’s two compound eyes are shown in Figure 2-1. Those optic lobes are composed of three types of neuropils or ganglia, which are a dense network of interwoven nerve fibers and their branches and synapses: the lamina, the medulla, and the lobula complex (consisting of lobula and lobula plate). Those neuropils correspond to three important visual information processing chains [46]:

- The lamina lies right beneath the photoreceptor layer of the eye and receives direct input from photoreceptors. The neurons in this ganglion act as temporal high-pass filter (HPF) by amplifying temporal contrast reversals or changes. They also provide gain control functionality; thus, ensuring a quick adaptation to variations in background light intensity. Axons from the lamina invert the image from front to back while projecting to the medulla.
- Cells in the medulla are extremely small and difficult to record from. However, behavioural experiments suggest that local optic flow detection occurs at this level. The retinotopic organization is still present in this second ganglion and there are about 50 neurons per ommatidium. The medulla then sends information to the lobula complex.

- The third optic ganglion, the lobula complex, is the locus of massive spatial convergence. Information from several thousand photoreceptors, preprocessed by the two previous ganglia, converges onto a mere 60 cells in the lobula plate. These so-called tangential cells (or Lobular Plate Tangential Cells, LPTC) have broad dendritic trees that receive synaptic inputs from large regions of the medulla, resulting in large visual receptive fields. The lobula complex projects to higher brain centers and to descending neurons that carry information to motor centers in the thoracic ganglia.

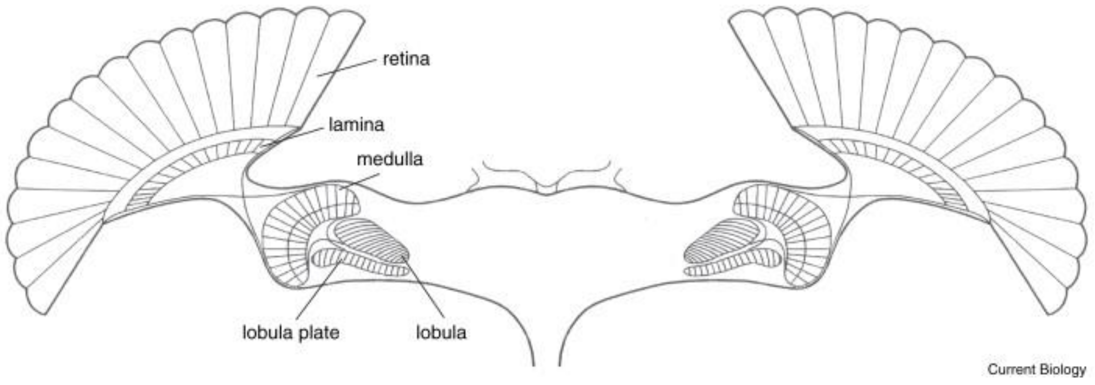


Figure 2-2: A schematic representation of the fly's visual and central nervous system (cross section through the fly's brain). Photoreceptor signals are transmitted to the lamina, which accentuates temporal changes. A retinotopic arrangement is maintainted through medulla. The lobula plate is made up of wide-field, motion-sensitive tangential neurons that send information to the controlateral optic lobe as well as to the thoracic ganglia, which control the wings [7-8].

This vision processing in three optic lobes are modeled by engineers. The lamina is modeled as a temporal HPF. If a moving feature, which has different illuminance from background, is arrived in each ommatidium, the output of the lamina is high due to a temporal contrast change. Therefore, this HPF output represents a moving feature's arrival. The medulla is modeled as a motion detector between spatially neighboring cells.

The first motion detector model was introduced by Hassenstein and Reichardt [49] and was named as the elementary motion detector (EMD). This EMD output is the magnitude of 1-D motion between a pair of neighboring cells. The details of this model are described in the section 2.2. Finally, the lobular complex is modeled as parallel matched filters, whose output is an inner product of 2-D matrices: one matrix is optic flows from surroundings, the other matrix is coefficients designed for self-motion estimation. The details of this matched filter is discussed in the section 2.3.

2.2 Elementary motion detector model

The specialized processing of visual motion begins in the medulla, which maintains a retinotopic architecture with columns of cells each associated with an overlying ommatidium. An early and still influential model of early motion detection, the correlational EMD, was formulated by Hassenstein and Reichardt [49]. This model is based on a correlation of the signal associated with one ommatidium with the delayed signal from a neighboring ommatidium, as depicted schematically in Figure 2-3. The combination of the spatial connectivity, the delay operation, and the nonlinear interaction of the correlator endow the EMD with directional motion sensitivity. The final opponent stage depicted in Figure 2-3, which takes the difference of two mirror-image correlator outputs, enhances the directional properties and rejects the effects of temporal contrast not due to motion. The EMD produces a motion-related output without computing derivatives (a process that would amplify noise).

The correlational EMD model is well-supported by physiological and behavioral evidence, although its neural basis has been hard to pinpoint due to the technical difficulty of electrophysiological recording from the very small medulla cells that are believed to be involved. Evidence suggests that EMDs are formed between at least nearest neighbors and next-nearest neighbors on the hexagonal lattice, and are thus aligned with various directions in visual space [50]. Physiological data in support of the EMD model have been taken primarily from more central neurons in the lobula complex, in particular the tangential cells themselves.

The output of a motion detector model of this kind in response to a moving visual scene is typically unsteady, with transients generated in response to the passage of edges or contrast gradients. In the mean, however, the output is a function of velocity of a moving stimulus, although this dependence is not monotonic, and there is also strong dependence on spatial structure and contrast of the stimulus. In spite of its ambiguities as a motion sensor, theory suggests that this type of detector is inherently well suited to tasks that might be limited by noise [51].

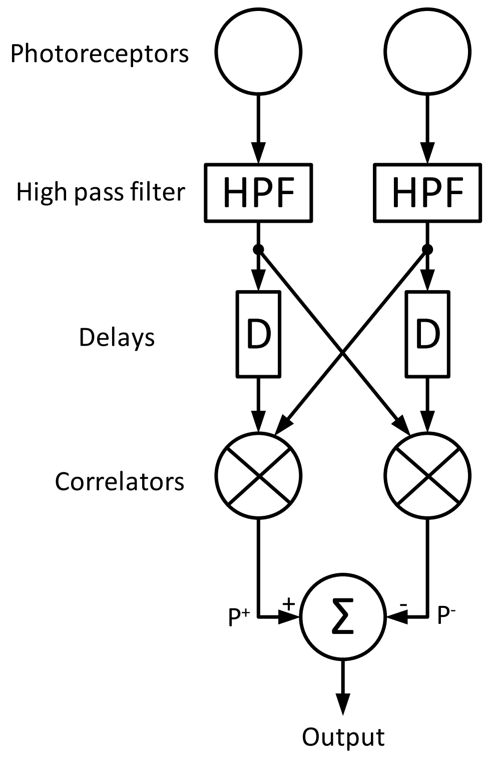


Figure 2-3: The Hassenstein-Rechardt or correlational elementary motion detector (EMD)

2.3 Wide-field optic flow analysis: lobular complex

Visual motion stimuli occur when an insect moves in a stationary environment, and their underlying reason is the continual displacement of retinal images during self motion. The resulting optic flow fields depend in a characteristic way on the trajectory followed by the insect and the 3-D structure of the visual surroundings. Therefore, these motion patterns contain information indicating to the insect its own motion and the distances from potential obstacles. However, this information cannot be directly retrieved at the local level and optic flow from various regions of the visual field must be combined to infer behaviourally significant information as shown in Figure 2-4 [52].

Analysis of the global motion field (or at least several different regions) is generally required in order for the local measurement to be exploited at a behavioural level. Some sort of spatial integration is known to take place after the medulla (where local motion detection occurs retinotopically), mainly in the lobular plate where tangential neurons receive input from large receptive fields. Thus, the lobula plate represents a major center for optic flow field analysis. Some of the 60 neurons of the lobula plate are known to coherent large-field motion, for example the VS, HS, and Hx-cells, whereas other neurons, the Figure detection cells (FD-cells), are sensitive to the relative motion between small objects and the background [53].

The response fields of VS neurons resemble rotational optic flow fields that would be induced by the fly during rotations around horizontal axes [53]. The response field of Hx cells have the global structure of a translational optic flow field. The response fields of HS cells are somewhat more difficult to interpret since it is believed that they do not discriminate between rotational and translational components. In summary, it appears that tangential cells in the lobula act as neuronal matched filters tuned to particular types of visual wide-field motion as shown in Figure 2-5 [53].

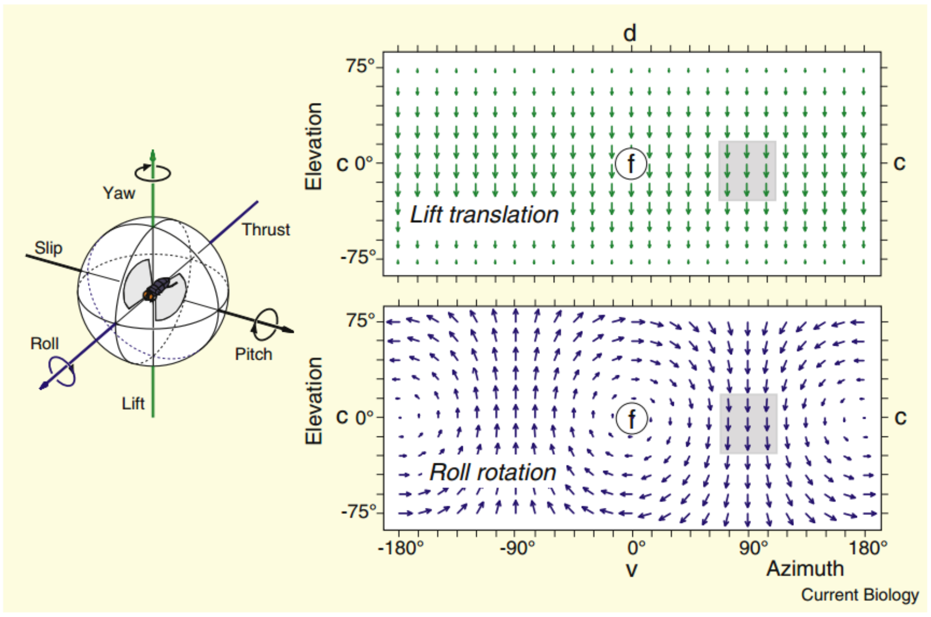
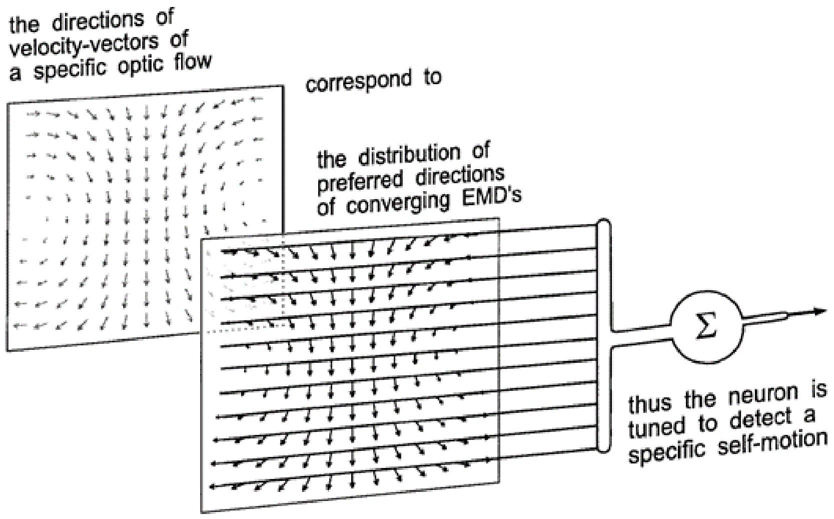


Figure 2-4: Self-motion and optic flow. Left: self-motion parameter of the fly can be described in terms of their components along the three cardinal body axes. Right: self-motion results in optic flow fields representing how the visual world moves relative to the fly's eyes. [52].



the directions of velocity-vectors of a specific optic flow
correspond to
the distribution of preferred directions of converging EMD's
Σ
thus the neuron is tuned to detect a specific self-motion

Figure 2-5: A hypothetical filter neuron matched to a particular optic flow field induced by rotating self motion adapted by [53].

2.4 Bio-inspired autonomous navigation

Bio-inspired autonomous navigation, which extracts control parameters by integrating wide-field optic flow, is implemented and demonstrated [14-15]. This navigation method extracts control parameters from several matched filters, which are tuned to be sensitive for optic flow patterns induced by a robot's specific movement.

As an example, a fly is moving forward along a corridor, which is located between two patterned side walls [26]. Geometric parameters in the given example are: γ is the body-fixed viewing angle, a is the half of the corridor width, δy is lateral displacement, $\delta\psi$ is a phase shift, and δu is a forward speed increase. The three parameters, δy, $\delta\psi$, and δu, are controlled in flight in order for the fly to be able to maintain a stable state.

The stable flight state is described in Figure 2-6 A, which a fly is moving forward with a desired speed in the center of the corridor by looking straight (no orientation shift); thus, a generated optic flow pattern, which is plotted as a function of the angle γ, is approximately a sine wave. A focus of expansion is in the front of view, and a focus of contraction is in the rear. These two focuses generate the smallest motion, and the sides generate the largest motion because of the closest distance.

Then, the three parameters are extracted by monitoring perturbations of a generated optic flow pattern. If the fly laterally shifts to right, a generated optic flow pattern moves downward by reflecting lateral displacement δy as shown in Figure 2-6 B. This is because the right side wall becomes closer than the left side wall. If the fly shifts its orientation, the focus of expansion and the focus of contraction are rotated; thus, a generated optic flow pattern shifts along with the γ axis (Figure 2-6 C). Finally, the fly is moving faster or slower than desired speed, the generated optic flow pattern is scaled as shown in Figure

2-6 D. Therefore, a lateral shift (δy), orientation shift (δψ), and a forward speed increase (δu) are all extracted from a generated optic flow pattern. By a close-loop flight control, the fly can maintain the stable fight state (Figure 2-6 A).

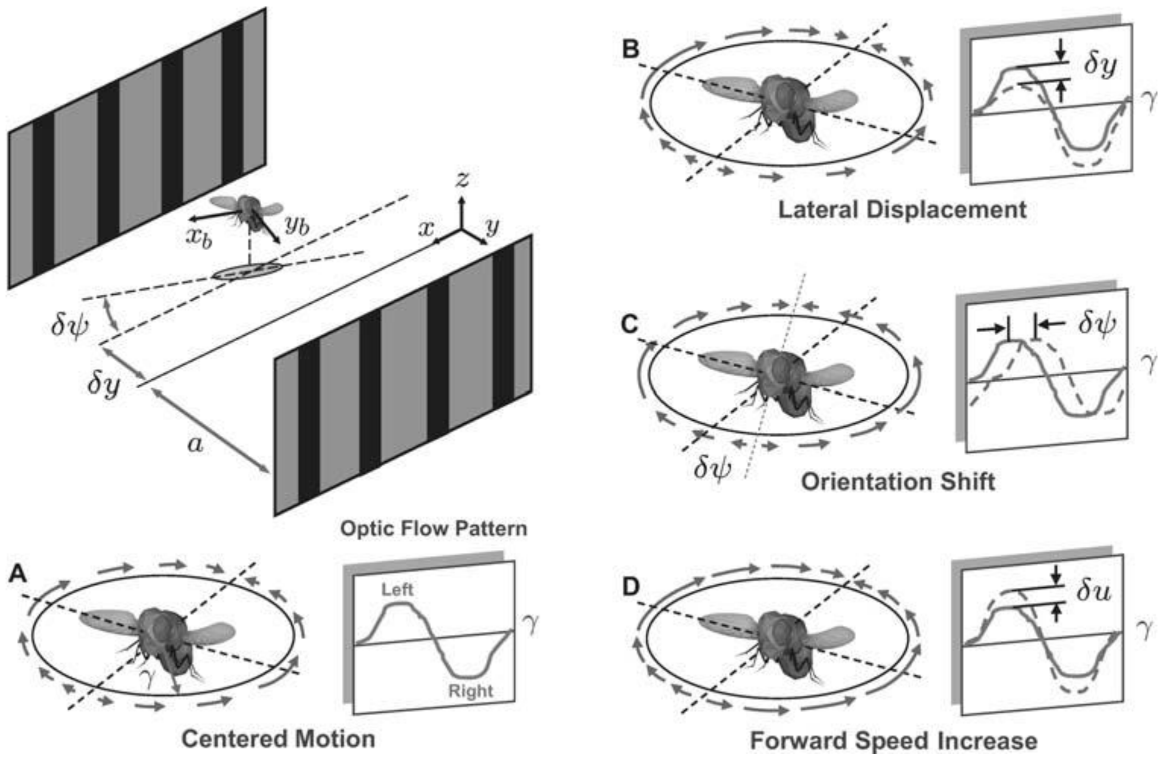


Figure 2-6: Perturbations of azimuthal optic flow and their correlations to the relative state of the vehicle. Amplitude, phase, and asymmetry of the azimuthal optic flow pattern encode relative proximity and speed with respect to obstacles in the environment [16].

2.5 Summary

In this chapter, the vision processing in an insect's eye is described. Three optic lobes, which are lamina, medulla, and lobula, perform temporal high-pass filtering, motion detection, and wide-field optic flow integration in each lobe. The integrated optic flows by a matched filter provide self-motion, depth and speed information for a flying insect's flight. A 3 degree-of-freedom (3 DoF) autonomous navigation control scheme, which utilizes the extracted motion by WFI, is also described.

CHAPTER 3
OPTIC FLOW ESTIMATION ALGORITHMS IN MAVS

3.1 Introduction

Optic flow or optical flow is the pattern of apparent motion of objects, surfaces, and edges in a visual scene caused by the relative motion between an observer (an eye or a camera) and the scene [54]. This pattern is 3-D in the natural world, and animals have evolved to detect the surrounding optic flow field. The pattern includes key information for animal navigation such as the distance between the observer and its surroundings and the velocity of self-motion or egomotion, which is 3-D motion of an observer. When the observer is a natural object, such as an insect, an eye plays a role to detect the flow field. The eye has special cells to decode the information to control the animal's body while it is flying. Thus, researchers who are building very small size air vehicles have been trying to utilize optic flow for their systems to mimic the smallest flying creature in nature.

The cells of the eye are inspiration or a model for researchers trying to mimic the visual ability of insects and animals. Prior to the initiate for modeling optic flow on the natural eye, researchers used an algorithm with a fixed time period. Displacement measurement in a fixed time period can estimate both the direction and magnitude of motion. The displacement measuring method is adequate to develop algorithms using computers because currently used camera systems provide image sequences with a fixed time period (called frame rate). Therefore, a variety of algorithms based on displacement measurement have been reported with high accuracy working on high computing power processors. With inspiration from an insect's eye, researchers have developed an algorithm for modeling flow that has a free time period and a fixed distance unit. The elementary motion detector (EMD) is the model, which estimates motion information from two fixed neighboring photorecepters. With a free time period, the traveling time of an object is measured as it passes between neighboring cells. Such time-of-travel-measurement algorithms are simple to calculate; however, custom-designed imaging systems must be utilized, which are not required for displacement systems. Moreover, for very small air vehicles, power constraint is also as important as accuracy, thus finding an optimal solution, balancing the calculation power and accuracy is a key point of the algorithm selection for the system.

This chapter covers the variations of these two aforementioned optic flow algorithms. Especially, the algorithms that are implemented on autonomous navigation system are described.

3.2 Gradient-based algorithms

3.2.1 1D gradient-based algorithm

The gradient method is based on the assumption that image intensity is conserved between successive frames in an image sequence [55]. For 1D optic flow, this constraint simplifies to:

$$\frac{\partial I}{\partial x}dx + \frac{\partial I}{\partial t}dt = 0$$

$$\therefore \frac{dx}{dt} = \frac{\partial I/\partial t}{\partial I/\partial x} \qquad (1)$$

Where, I is pixel intensity, t is time and x is pixel position.

Comparing successive frames, and using finite differencing, provides the temporal gradient, while comparing adjacent pixels in a given frame provides the spacial gradient. In implementation, however, this method often suffers the drawback of having erroneous spikes in optic flow in regions with low spatial gradient. For this reason, we typically obtain a spatially dense grid of estimates and then desample this grid (by block averaging of adjacent estimates), ignoring large-shift outliers in the process. Since it assumes the image intensity varies linearly, the algorithm generally becomes ineffective for shifts larger than a few pixels per frame, depending on the environment textures.

3.2.2 Lucas and Kanade (L-K) algorithm

The original L-K algorithm effectively extends the gradient method to two dimensions and includes several improvements [36], namely:

- Final estimates are formed by averaging the raw estimates over several neighboring pixels (in a 2D 'integration' window).
- The gradient-based estimate (based on a first order Taylor expansion) is iterated in a Newton-Raphson fashion to converge on a local optimum.
- The estimator equation has been re-derived based on minimization of L2 norm image-intensity matching error in the equation (2), since the equation (1) does not generalize to n-dimensions. The associated cost function is

$$\varepsilon = \sum_{x=p_x-\omega_x}^{x=p_x+\omega_x} \sum_{y=p_y-\omega_y}^{y=p_y+\omega_y} \delta I_k^2(x,y) \qquad (2)$$

Where, $\delta I_k^2(x,y)$ is defined in the equation (3).

3.2.3 Pyramidal Lucas and Kanade (L-K) algorithm

Accuracy requires a small window size since larger windows may contain pixels with differing velocities. Computation limitations also dictate a smaller window. Robustness, however, requires larger windows, i.e. if the optimum lies within the window then convergence is much more likely. To solve this trade-off, Bouguet proposed a Pyramid iteration scheme [56]. A low resolution version of the image is used to obtain an estimate for the optic flow, and then this estimate is fed to a higher resolution version as the new initial guess. The process repeats until the full resolution image returns the final estimate. This enables shifts to be detected which are greater than the window size and partially overcomes the problem of getting trapped in local optima. The raw L-K algorithm is capable of detecting shifts up to $\frac{1}{2}$ a wavelength of the highest spatial frequency. Thus, the pyramid resolution iteration extends the maximum detectable image shift by first focusing on the lower spatial frequencies in the image.

$$G = \sum_{x=p_x-\omega_x}^{x=p_x+\omega_x} \sum_{y=p_y-\omega_y}^{y=p_y+\omega_y} \begin{bmatrix} I_x^2(x,y) & I_x\,(x,y) I_y\,(x,y) \\ I_x\,(x,y) I_y\,(x,y) & I_y^2(x,y) \end{bmatrix}$$

$$\delta I_k\,(x,y) = I(x,y) - J(x + g_x + v_x^{k-1}, y + g_y + v_y^{k-1})$$

$$b_k = \sum_{x=p_x-\omega_x}^{x=p_x+\omega_x} \sum_{y=p_y-\omega_y}^{y=p_y+\omega_y} \begin{bmatrix} \partial I_x\,(x,y) I_y\,(x,y) \\ \partial I_x\,(x,y) I_y\,(x,y) \end{bmatrix}$$

$$v^k = v^{k-1} + G^{-1} b_k \qquad (3)$$

where (I_x, I_y) are the spatial intensity gradients, (p_x, p_y) is the point in the image which we wish to track, (w_x, w_y) define the integration window size, (g_x, g_y) is the shift estimate from the previous pyramid (resolution) level, v^k is the adjustment to the shift estimate at iteration step k (iteration ceases after 20 steps, default in our simulation, or when $\|G^{-}\|b_k\|<0.3$ (default in our simulation)), I refers to the intensity of the previous image and J the intensity of the current image. At each pyramidal level, G and g are updated then $\delta Ik(x, y)$, v^k and b_k are successively computed until the iteration ceases.

In general, more pyramid-resolution levels allow one to detect greater shifts more robustly. However, if the raw image resolution is already low, and the pixel shifts are typically small, it can be detrimental to use pyramid iteration, as the estimates from lower-resolution levels may be erroneous due to poor image contrast at very low spatial frequencies.

3.3 Bio-inspired elementary motion detector (EMD) algorithms

The first elementary motion detector (EMD) model in a flying insect’s vision system was formulated by Hassenstein and Reichardt [49]. As shown in Figure 3-1(a), the EMD consists of two photoreceptors, temporal high pass filters (HPFs), and delay units modeled by low pass filters, and correlators. The two outputs of the correlators are summed up with different polarity. This configuration estimates motion of a moving object or feature, which is passing through two neighboring photoreceptor cells.

An example in Figure 3-1shows that a feature is moving from the left photoreceptor (at time t=0, shown in a solid line) to the right photoreceptor (at $t=t_1$, shown in a dotted line). As a result, a temporal change is detected by a HPF and is delayed with a time constant τ (Figure 3-1 (b)). Each delayed HPF output is correlated or multiplied with the

HPF output of the neighboring cell; thus, each correlator generates output when a feature is moving to a preferred direction. In the given example, the feature moves from left to right at $t=t_1$; only the left side correlator generates output at time $t=t_1$ as shown in Figure 3-1(c). Therefore, the final output is positive, which means the direction of motion is left to right, and its magnitude corresponds to the speed of motion. Due to the correlation process, the maximum detectable velocity is limited until two inputs of the correlator are exactly matched or overlapped. The condition of the exact overlapping is when t1 is τ; thus, the maximum measurable speed is $1/\tau$ [pixel/sec].

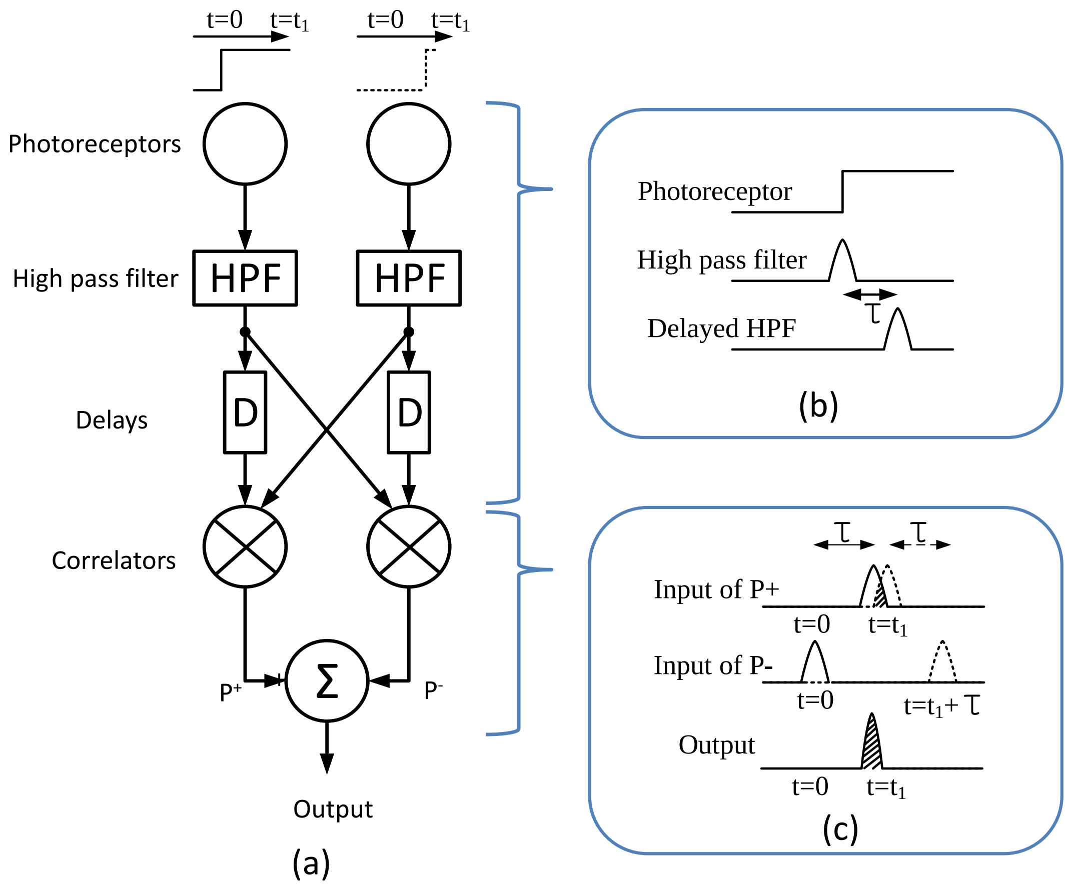


Figure 3-1: The Hassenstein-Rechardt or correlational elementary motion detector (EMD)

3.3.1 Photoreceptor model

The first implemented photoreceptor model by Delbrück and Mead is shown in Figure 3-2 [57]. Two valuable features in an insect eye are implemented in the model: logarithmic response and adaptation to ambient light. The logarithmic response allows the device to collect information from both indoor and outdoor environments. The adaptation to ambient light example is shown in Figure 3-3 [1]. The transient response shown in green lines is centered at the ambient light condition illustrated in red lines. This model is implemented in most EMD-based motion detectors.

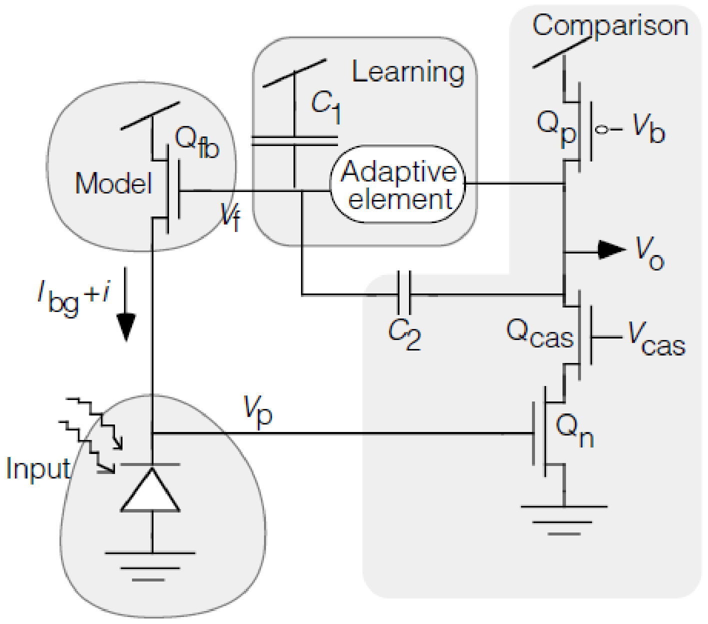


Figure 3-2: Adaptive photoreceptor [57]

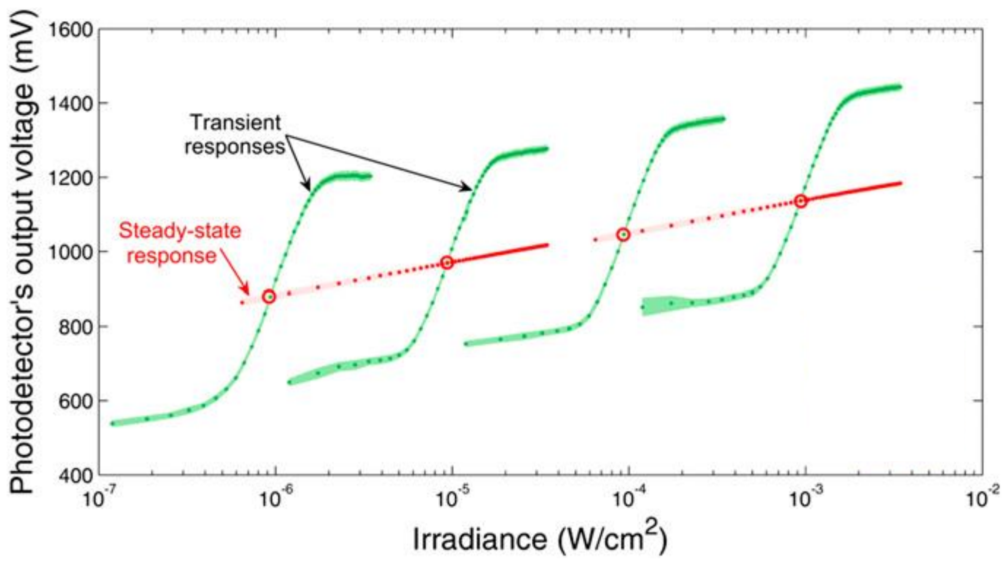


Figure 3-3: Adaptive photoreceptor response to the ambient light [1]

3.3.2 High pass filter, delays, and correlators

3.3.2.1 Pure analog EMD implementation

The high pass filter (HPF), delays, and correlators in Figure 3-1 measure velocity for a preferred direction of two neighboring photoreceptor cells. The HPF finds temporal change by rejecting DC ambient light. Thus, the HPF responds at the arrival or departure of a moving object on the location. The delays are modeled using a low pass filter (LPF). An ideal all-pass LPF delivers perfect delaying functionality. However, other types of LPFs can be also used for motion measurement. Correlators are the component that has a variety of circuit implementation. An analog multiplier is the first option for finding correlation between two analog signals [58]–[60].

3.3.2.2 Pulse-based EMD implementation

In order to see the variations of the implementation, the EMD including LPFs as delay cells are shown in Figure 3-4. The difference from using ideal delay cells in Figure 3-1 can be found in Figure 3-4(b). The delayed HPF output is expressed as the impulse response of LPF in delay cells. Here, an interesting observation is made. The HPF output can be considered as an impulse signal which indicates the time of the moving feature's arrival (or also called token). In addition, due to impulse-like HPF output, the correlator output can be understood as a sampled delayed response of the LPF, which is captured at time t_1 in the example. This concept is implemented by utilizing pulse-based analog circuits [32], [61]. The implementation replaces the analog multipliers to sampling circuits and pulse generators.

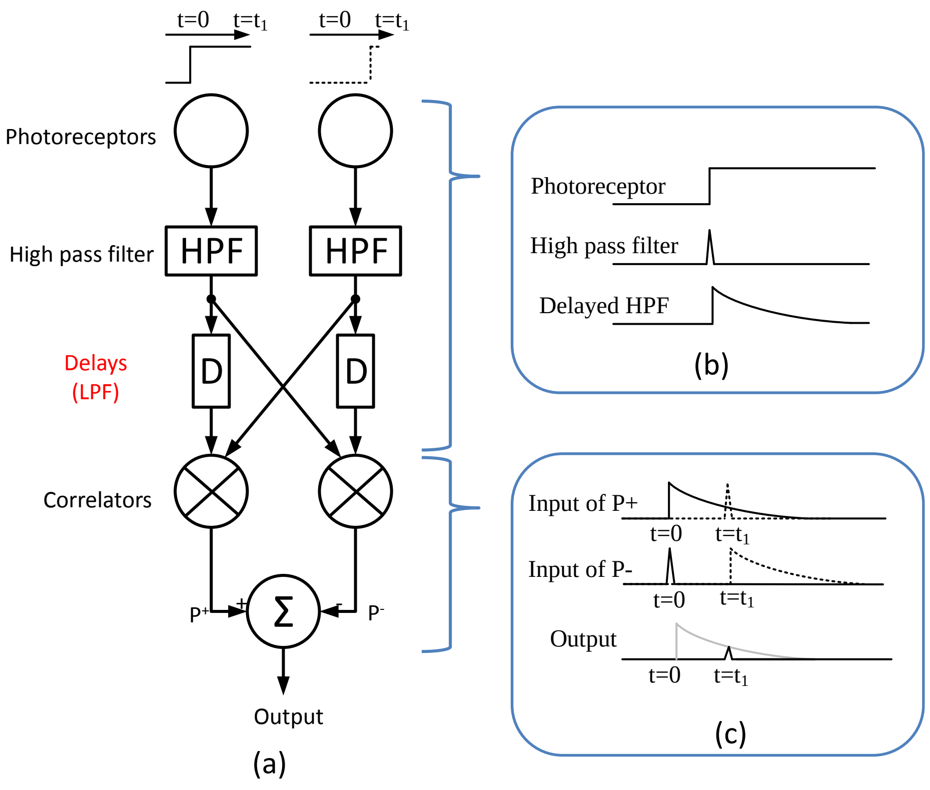


Figure 3-4: EMD output using LPFs as delay cells

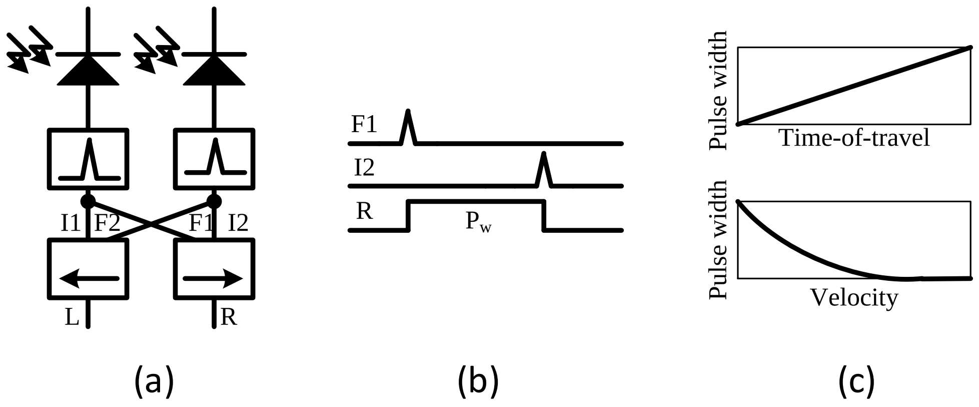


Figure 3-5 Facilitate-and-inhibit (FI) optic flow estimation algorithm

Another pulse-based implementation is shown in Figure 3-5. This algorithm is named “facilitate-and-inhibition” because the time-of-travel is measured by facilitating and inhibiting paired pixels as shown in Figure 3-5 (a) [32], [62], [63]. The timing diagram in Figure 3-5 (b) describes time-of-travel measurement for an object that moves from left to right direction. A moving object is represented by a contrast edge, which can be implemented using a continuous-time (CT) high-pass filter (HPF). In the example, a contrast edge moves from a left pixel to right pixel in a certain time interval; thus a facilitating signal (F1) is generated at the left pixel and an inhibiting signal (I2) is followed from the right pixel. Using the signals (F1 and I2), a pulse is accordingly generated at node R, and its pulse width (P_w) indicates the time-of-travel of the moving object. Finally, using an inversely proportional relationship between the measured time-of-travel and velocity shown in Figure 3-5 (C), motion to the right direction is estimated. Motion to the left direction is estimated by the same procedure with F2 and I1 signals at output node L. Note: time-of-travel is directly measured as the pulse width of the output signal. This inspires many variations of implementation, however the circuits basically perform time-of-travel measurement. Facilitate-and-trigger (FT), facilitate-trigger-and-inhibit (FTI) [32], [61], and facilitate-and-compare (FC) [64]. The review on the aforementioned pulse-based algorithms are found in [65].

3.3.2.3 Digital EMD implementation

The research efforts were made on implementing the EMD algorithm in digital circuits in FPGAs [66]–[68]. The motivation is straightforward. On FPGAs, porting digital algorithms is simple and fast. Nowadays, the algorithm developed in Matlab can be directly ported on FPGAs [69]. Therefore, customized algorithms can be quickly integrated on a robot platform. The modification and verification process is also simple and fast. In addition, the digital EMD directly interfaces to the host MCU or CPU by providing digital output. The FPGA-EMD implementation can be considered as the A/D boundary is moved to the front of the sensor.

Compared to the analog EMDs, which are implemented in parallel at every pixel, the digital EMDs share computation cores. Pixel and intermediate data are stored in digital memory. Logarithmic transform for modeling photoreceptor response and temporal high pass filtering are performed in high speed, 100-160MHz [66]–[68]. Temporal HPF cores are carefully designed in recursive fashion using stored intermediate data in RAMs. High precision (36-b) is required for intermediate data to avoid information loss in processing.

The delay cells are implemented in discrete-time manner. The frame rate determines minimum delay unit. The correlators can be implemented using digital multipliers [66]. Also, time-of-travel can be measured at each pixel by implementing a counter at every pixel [68]. The counter is triggered by the temporal edge information generated by current location and stopped by the edge of neighboring cells. The operation is similar with timing using a stopwatch. Thus, the recorded time in the counter is time-of-travel. To measure 1D velocity information, two counters are required per pixel to measure right and left directions.

The digital EMDs proved the neuromorphic algorithm can be integrated in the digital domain. Compact size implementation is also demonstrated as shown in Figure 3-6. However, hardware cost to store frame data and to process temporal high pass filtering is expensive. High speed operation due to recursive computation increases power consumption. Thus, to be applied on MAVs additional efforts to be monolithic integration and low power consumption are required.

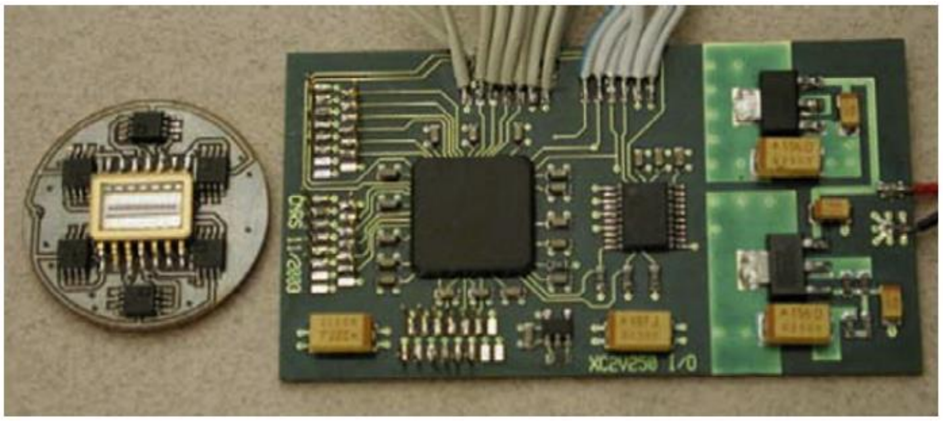


Figure 3-6: Photoreceptor linear array (left) and electronic board (right) with 12x12mm FPGA [68]

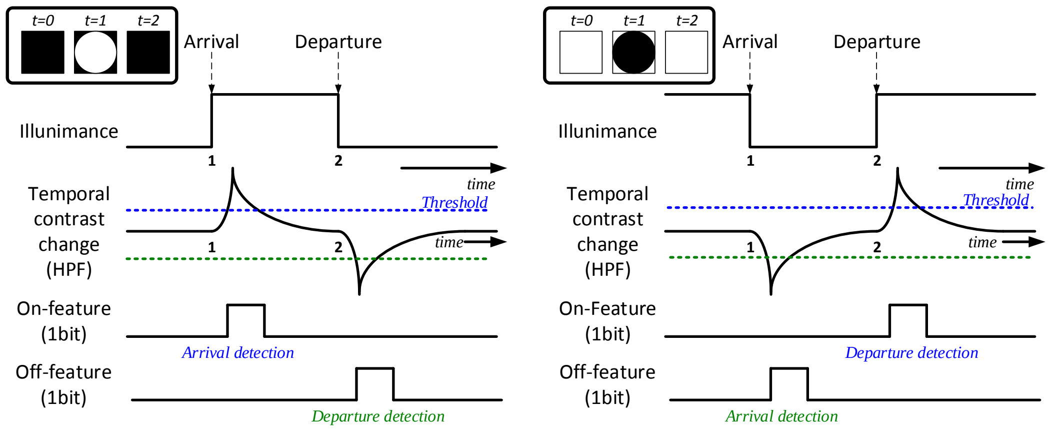
3.3.2.4 Contrast adaptation

The mechanism inside a flying insect's visual pathway is still mysterious only basic motion detection and processing behaviors are uncovered. The Reichardt EMD model strongly depends on contrast strength; under the condition of the same speed applied, the high contrast signal induces faster velocity output, and the low contrast signal induces slower velocity output. These results from correlators are simply multiplication of two analog contrast signals. However, physiology tests of flying insects show the contrast dependency is not severe in actual pathways. Thus, additional hypothesis is made on contrast adaptation in the pathways [30], [70], [71]. The saturation block is introduced after HPFs in EMDs to limit high contrast signals to affect the correlator output [59], [70]. Another assumption introduces adaptive gain control on HPF output depending on global velocity [30], [71], [72]. Though research efforts on validating the aforementioned hypothesis are under progressing, the circuit implementation can engage the contrast adaptation or normalization function for the sensor's robust operation.

3.3.2.5 On and off pathways

On and off pathways in the vertebrate retina are recently reported to also exist in the insect's vision [73]. The on and off pathways process the on-edge and off-edge of contrast change separately [74], [75]. The example in Figure 3-7 describes the on and off edge processing with respect to a moving object's departure and arrival. The contrast change is measured by the HPF in EMDs. The rising edge (on-edge) and falling edge (off-edge) of the contrast changes encode the arrival and departure of a moving object. When a bright object is passing through a dark background, the on-edge represents the arrival as shown in Figure 3-7 (a). At the same condition, off-edges occur at the object's departure. The opposite relationship is made when a dark object is passing on a bright background as shown in Figure 3-7 (b). This relationship between on/off edges and the departure/arrival of moving objects are summarized in Table 3-1.

Two notes on this property are important. The first note is if on-edge occurs, then off-edge follows; if off-edge occurs, then on-edge follows. This is because arrival is always followed by departure. The other note is motion information or time-of-travel between neighboring cells must be measured by using the same edges. This condition confirms the measurement is the time between the arrivals (or departures); thus, a moving object's staying time on the location can be extracted for velocity measurement.



(a) A bright object is passing a dark background (b) A dark object is passing a bight background

Figure 3-7: On/off pathways and feature arrival/departure relationship

Table 3-1: The relationship between on/off edges and the departure/arrival of objects

	Bright objects	Dark objects
On-edge	Arrival	Departure
Off-edge	Departure	Arrival

The property recommends on-edges and off-edges be processed in parallel datapaths in hardware implementation. However, it requires twice the resources for computation and memory. For example, the stopwatch like digital EMDs require 4 counters in each pixel. Therefore, optimal solution in the accurate optic flow and hardware resource needs to be considered before implementation.

3.4 Summary

Optic flow algorithms that are adopted on autonomous navigation systems are described. In digital systems, the state-of-the-art gradient-based L-K and algorithm is programmed on MCUs or CPUs. Thus, substantial hardware is required. Bio-inspired optic flow algorithms based on EMDs address low computation complexity and massive implementation on analog-VLSI chips. The EMD algorithm models the visual pathways in a flying insect to measure motion information. Adaptive photoreceptor to ambient light enlarges dynamic range and amplifies contrast. HPFs in the pathway find a temporal contrast change, which occur at a moving object's arrival or departure. Delay cells are modeled LPFs, and correlators are implemented using analog multipliers. The delayed correlation function is also interpreted to time-of-travel measurement of neighboring cells. The time-of-travel measurement enables simple implementation of delayed correlation. Pulse-based time-of-travel measurement is popular implementation. Contrast adaptation is also considered in a few implementations by limiting the measured contrast before correlators. On and off pathways in the insect's vision contain an important hint for time-of-travel-based algorithms. The measurement must be made between the same edges; thus, parallel datapaths are required. Finally, digital EMDs are also implemented on FPGAs; however, substantial hardware resources for temporal filtering and data storage are hurdles to be overcome if to be applied on MAVs.

In the next chapter, the time-stamp-based algorithm which is modified from time-of-travel EMDs is introduced. The algorithm maintains bio-inspired features such as in-pixel temporal contrast measurement. The algorithm also introduces the time-stamp imaging, which is effectively used for intermediate information for simple and robust digital time-of-travel computation. Thus, the algorithm addresses the drawback of pure analog and pure digital EMDs.

CHAPTER 4
TIME-STAMP-BASED OPTIC FLOW ESTIMATION ALGORITHM

4.1 Introduction

In the previous chapter, I discussed calculation intensive displacement-based and bio-inspired optic flow estimation algorithms. These algorithms have been implemented on hardware to provide real-time, front-end information for bio-inspired micro air vehicles (MAVs). The most popular one, the Lucas and Kanade (L-K) algorithm, was implemented using a programmed microcontroller and successfully demonstrated autonomous navigation of unmanned vehicles [26]. This same algorithm was also implemented solely using hardware, which consisted of field-programmable gate arrays (FPGAs) for algorithm processing and external dynamic random access memories (DRAMs) for frame memory storage [37]. Although both solutions have proved to be feasible for MAVs, they are too bulky to satisfy a MAV's low power consumption and payload constraints because the complicated algorithm consumes huge amounts of power and requires too much physical space. In addition, a huge frame memory requirement makes it impossible to be integrated on a single chip with an image sensor.

Bio-inspired neuromorphic optic flow algorithms were also implemented on hardware by either pure analog circuits [30], [32], [76] or FPGAs [6-7] . Due to the simple nature of the algorithm, pure analog processing implementation proved to be a monolithic solution, which integrated both photoreceptors and processing circuits in a single chip.

However, pure analog signal processing is too easily susceptible to temperature and process variations to provide a stable optic flow. Also, the analog processing must be implemented in pixel-level circuits; as a result, it is difficult to either scale the pixel size or apply low power design techniques in [78]–[81]. The FPGA solution overcame the weakness of pure analog signal processing in terms of providing stable optic flow, but unexpected huge hardware resources were required to perform temporal filtering in the digital domain, making this approach unsuitable for MAVs.

In order to address the aforementioned problems, I propose a time-stamp-based optic flow algorithm, which is modified from the conventional neuromorphic algorithm to give an optimum partitioning of hardware blocks in both analog and digital domains as well as adequate allocation of pixel-level, column-parallel, and chip-level processing. Temporal filtering, which requires huge hardware resources if implemented in the digital domain, can be implemented in a pixel-level analog processing unit to maintain minimal hardware resources. The rest of the blocks can be implemented using digital circuits in column-parallel or chip-level processing units in order to provide stable optic flows.

This chapter covers the proposed time-stamp-based optic flow algorithm from a conceptual level to detailed 1D and 2D optic flow examples. Lastly, two algorithm evaluation methods are introduced and used to assess the algorithm with respect to the MAV's autonomous navigation requirements.

4.2 General concept of the proposed time-stamp-based motion sensing scheme

The basic idea of the proposed time-stamp based motion sensing algorithm is illustrated in Figure 4-1. As shown in Figure 4-1 (a), an object is moving from t=10 to t=20 following the trajectory depicted as an arrow. Each pixel records the time when the feature arrives in the pixel; thus the recorded value in each pixel is the time stamped at each arrival. Figure 4-1 (b) shows the recorded values in a 2D array that is named a time-stamp image. Most zero values in the image mean that there was no object appearance at the corresponding location for the time period, yet non-zero values contain useful information for motion estimation.

Since the non-zero value of each pixel encodes information of when a moving object arrives at the spatial location, a new image sequence that locates a moving object at specific time can be decoded as shown in Figure 4-1 (c). In the decoded images, a black square means the location that an object arrives at that current time, and a gray square means the location that an object arrived at that previous time. Then, the motion information at a specific time can be extracted by finding a position change (Δx, Δy) over a given time period (Δt). The arrow in Figure 4-1 (d) shows the extracted motion at the specific time t=14 by calculating a position change from the previous time (t=13) over time period $\Delta t=1$.

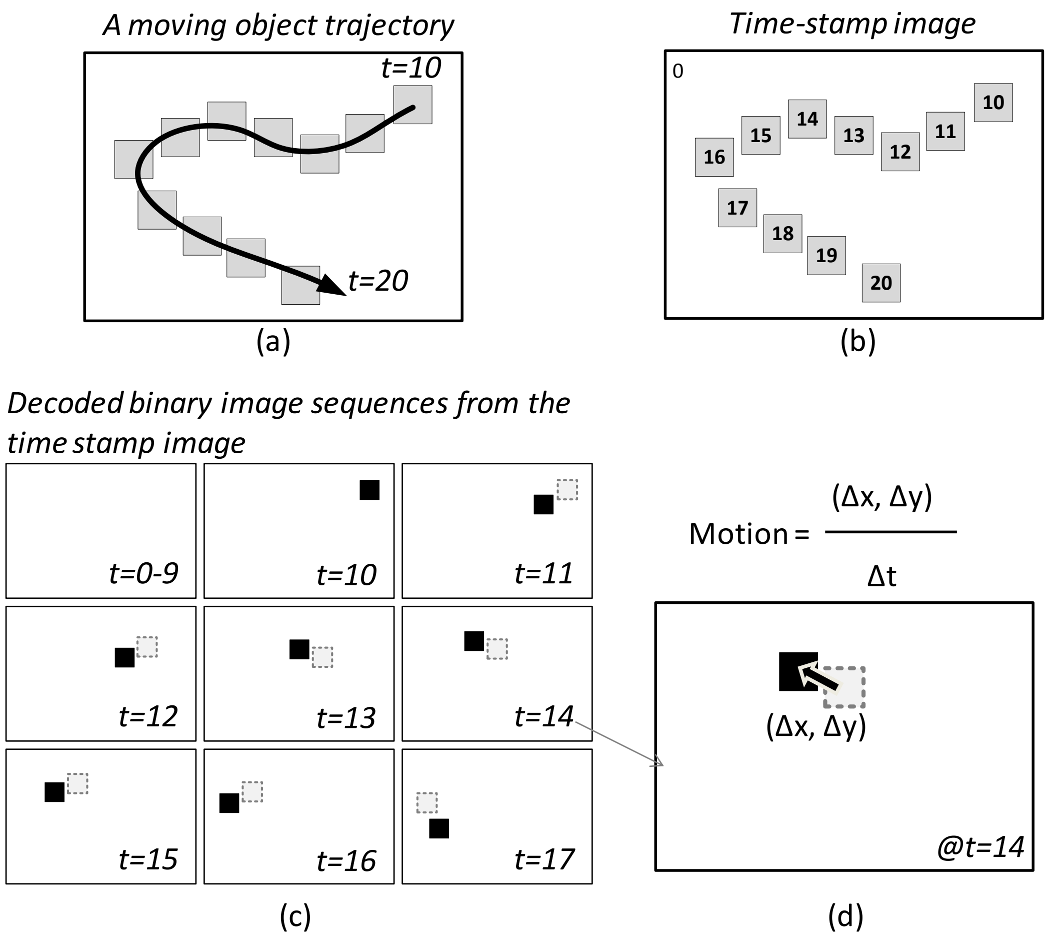


Figure 4-1: General concept of the time-stamp-based motion sensing scheme

4.3 Time-stamp-based 1-D optic flow estimation algorithm

A 1-D optic flow estimation algorithm based on time-stamp information is delineated. The algorithm basically measures time-of-travel at the pixels, where moving objects are newly arrived, and estimates motion velocity by calculating the reciprocal of time-of-travel. Thus, in a normal case, optic flow is sparsely generated at the location of moving features.

4.3.1 Time-of-travel measurement and velocity conversion

The time-stamp based motion sensing scheme can also be applied for time-of-travel optic flow estimation as illustrated in Figure 4-2. An object or a feature is moving from left to right in a 1-D pixel array with a speed gradually increasing from t=10 to t=40, and corresponding 1-D time-stamp image is also shown. The time-of-travel of a moving feature is simply measured by calculating the difference of the neighboring time-stamp values as in equation (1). In the given example, the calculated time-of-travel value of the left most pixel, $T_{time\text{-}of\text{-}travel}(1)$, is 7, and that of the right most pixel, $T_{time\text{-}of\text{-}travel}(9)$, is 1. A slower object travels at a longer time to cross the neighboring pixels than the faster one. Finally, motion information, V(n), can be calculated as in equation (2), which is inversely proportional to the time-of-travel, and its sign corresponds to the direction of motion.

$$T_{time-of-travel}(n) = T_{time-stamp}(n+1) - T_{time-stamp}(n) \text{, } \textit{n is a spatial index} \qquad (1)$$

$$V(n) = \frac{1}{T_{time-of-travel}(n)} \qquad (2)$$

These time-of-travel and velocity conversions are calculated only if a feature is detected at specific time. Therefore, optic flows are sparsely generated for example: at time 7, the optic flow is generated only at the leftmost pixel; at time 40, the optic flow is generated only at the rightmost pixel in the 1-D array.

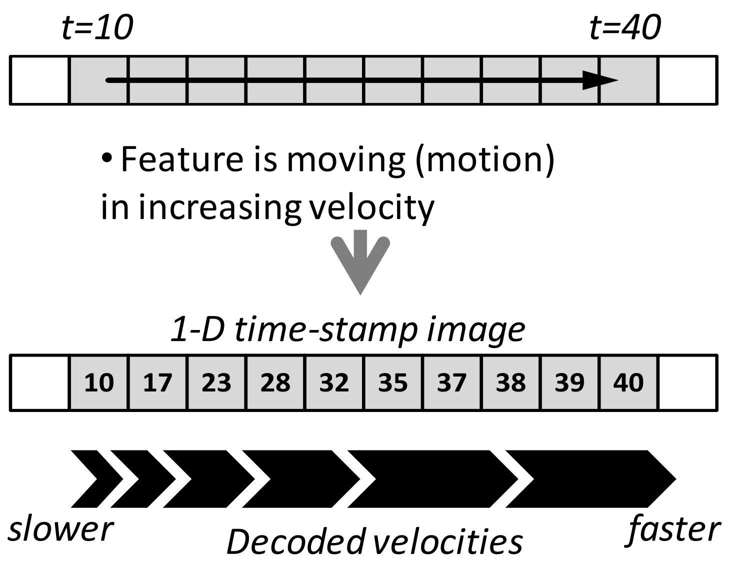


Figure 4-2: Time-stamp-based 1-D time-of-travel measurement

4.3.2 Facilitate-trigger-and-inhibit (FTI) algorithm

Several variations of time-of-travel measurement were reported to increase the algorithm's performance or to implement efficiently on hardware [82]. One of these algorithms, the facilitate-trigger-and-inhibit (FTI), is introduced to increase noise immunity by an additional triggering stage to check a feature's moving history from one more adjacent pixel.

Traditional time-of-travel measurement using two photoreceptors (or pixels) is as shown in Figure 3-5. This algorithm is named "facilitate-and-inhibition" because the time-of-travel is measured by facilitating and inhibiting paired pixels as shown in Figure 3-5 (a) [83]. The timing diagram in Figure 3-5 (b) describes time-of-travel measurement for an object moving to the right. A moving object is represented by a contrast edge, which can be implemented using a continuous-time (CT) high-pass filter (HPF). In the example, a contrast edge moves from a left pixel to right pixel in a certain time interval;

thus a facilitating signal (F1) is generated at the left pixel and an inhibiting signal (I2) is followed from the right pixel. Using the signals (F1 and I2), a pulse is accordingly generated at node R, and its pulse width (Pw) indicates the time-of-travel of the object moving to the right. Finally, using an inversely proportional relationship between the measured time-of-travel and velocity shown in Figure 3-5 (C), motion to the right direction is estimated. Motion to the left direction is estimated by the same procedure with F2 and I1 signals at output node L.

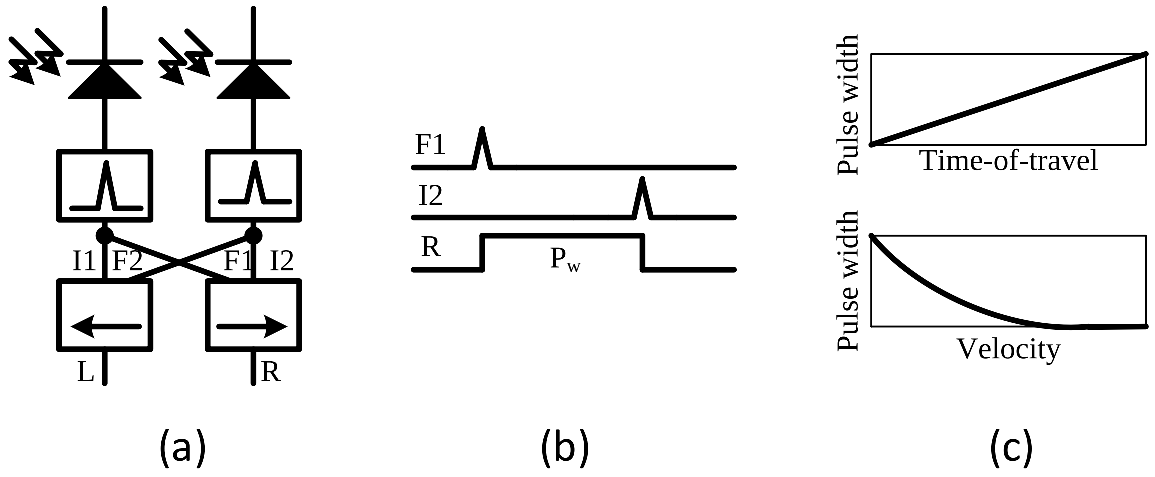


Figure 4-3: Traditional time-of-travel measurement, facilitate-and-inhibit (FI)

Because a falsely detected feature signal at each pixel (F1 or I2) due to noise can directly be reflected to the output velocity, the FT algorithm is sensitive to temporal or spatial noise. To overcome this weakness, the FTI algorithm is reported [11-12]. As shown in Figure 4-4 (a), FTI employs one more pixel and checks one more conditional stage, named a “trigger” stage, to start time-of-travel measurement. For example, to measure time-of-travel on the R node in Figure 4-4 (a), FTI starts measurement if F1 is generated and T2 follows. If this condition is met, FTI triggers the output pulse, R, and the generated pulse lasts until I3 is reached. The timing diagram in Figure 4-4 (b) describes the operation. Since this additional triggering condition monitors the history of a moving feature from an adjacent location, FTI can generate output velocity only for proven moving features. Thus, the FTI algorithm is insensitive to noise introduced from each feature detector; however, it conservatively generates output.

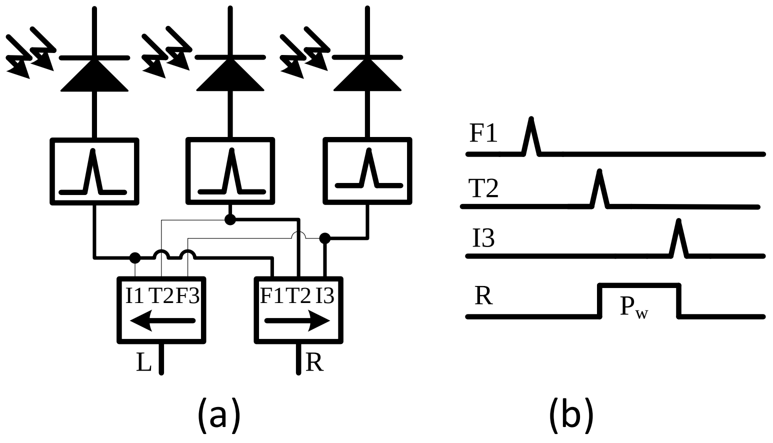


Figure 4-4: Facilitate-trigger-and-inhibition (FTI) time-of-travel measurement

4.4 Time-stamp-based 2-D optic flow estimation algorithm

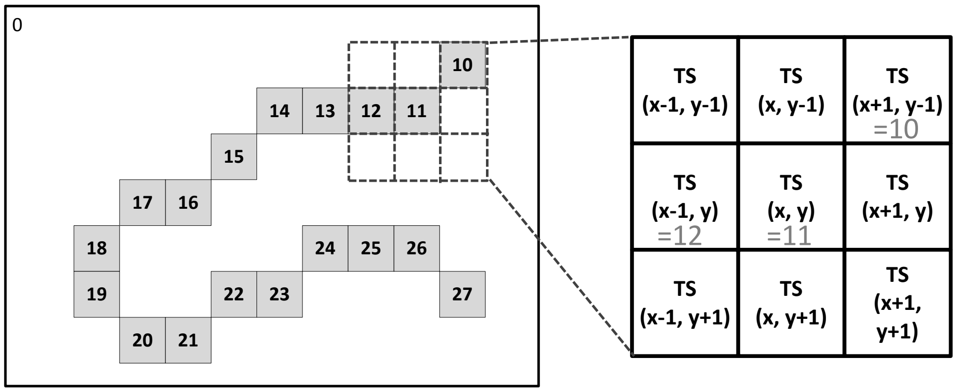
The previously explained 1-D optic flow estimation algorithm is easily applied for 2-D optic flow estimation by measuring multiple time-of-travel values in different directions. An example using a 3x3 spatial window to find 2-D optic flow is illustrated in Figure 4-5. In this example, the 3x3 mask on a time-stamp image in Figure 4-5 (a) is defined to estimate 2-D optic flow for the center position, whose time-stamp value is 11 in the given example. At the mask, four basis velocities, horizontal (V_{WE}), vertical (V_{SN}), and two diagonal motions ($V_{SW\text{-}NE}$, $V_{NW\text{-}SE}$), are calculated using the time-of-travel values measured by gray colored time-stamp information in Figure 4-5 (b). Then, x and y component of motion, (*u*, *v*), at each location can be calculated by equation (3).

$$
\begin{aligned}
u &= V_{WE} + (V_{SW\_NE} + V_{NW\_SE}) / \sqrt{2} \\
v &= V_{SN} + (V_{SW\_NE} - V_{NW\_SE}) / \sqrt{2}
\end{aligned}
\tag{3}
$$

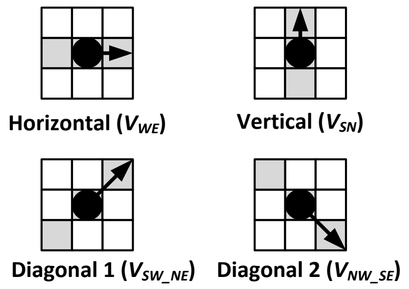
Similar to the 1-D optic flow estimation, these calculations are only performed when a feature is detected at the center pixel of a 3x3 mask. Thus, in the 3x3 mask example in Figure 4-5 (a), the optic flow at the 3x3 window is calculated and generated only when time is 11 because a feature is detected at the time.

A more specific relationship between optic flow generation timing and the 3x3 mask on a time-stamp image is explained by examples in Figure 4-6. At time t=11, a 2-D optic flow is generated at the dotted 3x3 mask because a feature is detected at the center of the mask at that time. In this case, the measured time-of-travel and estimated velocity are both -1 for the direction of diagonal 1 in Figure 4-5 (b), which means V_{SW_NE} is -1 [pixels/frame]. For the other three directions, measured time-of-travel and velocity are all zero. Therefore, from equation (3), the final 2-D optic flow at the dotted 3x3 mask, $(u, v)_{\text{dotted}}$, is $(-\frac{1}{\sqrt{2}}, -\frac{1}{\sqrt{2}})$ [pixels/frame].

However, at time t=12, no feature is detected at the center of the dotted mask; no optic flow is generated at the location: $(u, v)_{\text{dotted}}=0$ at t=12. Instead, a feature is detected at the center of the different 3x3 mask, which is drawn as a solid line; a 2-D optic flow is generated from the solid 3x3 mask at t=12. This case measured time-of-travel and velocity for the horizontal direction is both -1; V_{WE} is -1 [pixels/frame] and $(u, v)_{solid}$ is $(-1, 0)$ at t=12.



(a) A 3x3 mask of the time-stamp image



(b) 4 basis vectors for 2-D optic flow calculation

Figure 4-5: 2-D optic flow estimation example

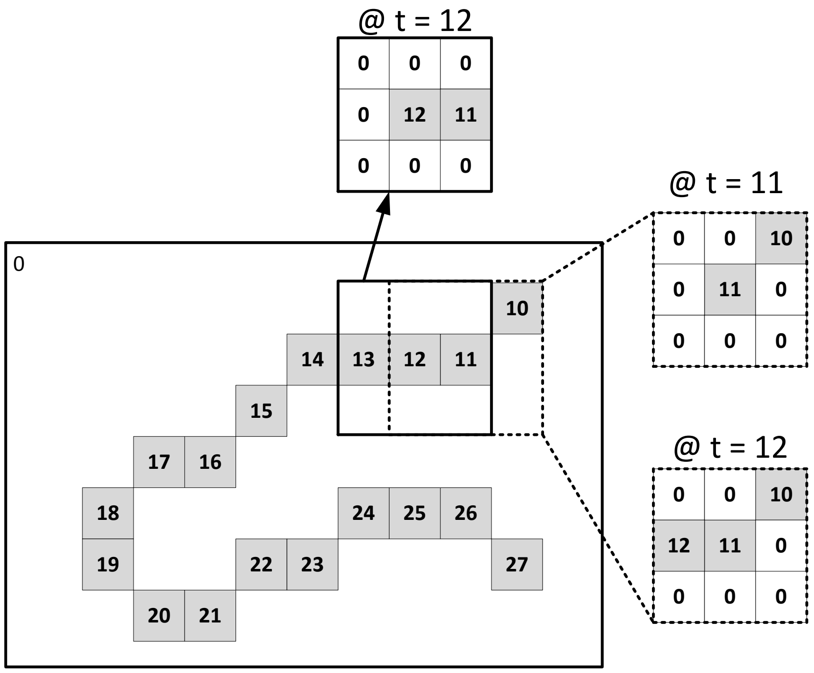


Figure 4-6: 3x3 mask examples at different time and locations

4.5 Performance comparison with respect to MAVs applications

To evaluate if the proposed optic flow algorithm provides enough information for a MAV’s navigation; we set up test environment to measure two metrics for different optic flow algorithms. The first metric is static rotational velocity test to measure the linearity of extracted motion. The second metric is navigation control parameter comparison using computer generated scenes, which are modeled a situation in which a robot is flying in a computer generated virtual arena. Because the generated images have correct optic flows, which are mathematically calculated from physical geometric dimensions [85], correct navigation control parameters can also be found; therefore, control parameters extracted by different optic flow algorithms are compared to the correct parameters.

4.5.1 Optic flow performance evaluation for rotational velocity extraction

4.5.1.1 Test patterns

360 ° panoramic high-resolution (2048x410) images shown in Figure 4-7 are test patterns of our rotational experiment setup. Since those images consist of sufficient amounts of pixels for surrounding environmental changes, we can precisely generate test patterns that can be shifted in any arbitrary rotational angles. By using equations (4) and (5), the amount of pixels to be shifted per frame, or a variable *RotateShift* in equation (5), was calculated from the predetermined parameters: rotational optic flow [rad/sec] and frame rates [frame/sec].

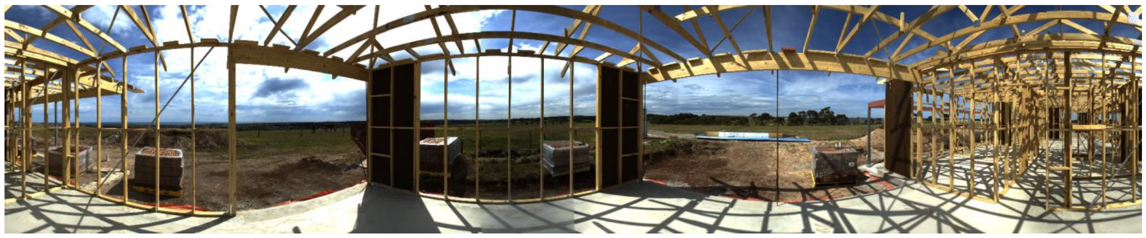


Figure 4-7: A 360 ° wide-field high-resolution imagery example for rotational velocity evaluation

$$FlowPerFrame\ \ [rad/frame] = \frac{Flow\ \ [rad/\sec]}{FrameRate\ \ [frame/\sec]} \tag{4}$$

$$RotateShift\ \ [pixel/frame] = \frac{FlowPerFrame\ \ [rad/frame] * imgWidth\ \ [pixel]}{(2*\pi)\ \ [rad]} \tag{5}$$

This evaluation environment was set up by using Matlab and combining Simulink and C++ algorithm models. Figure 4-8 illustrates the test flow for this evaluation. Simulation starts by reading the panoramic image; vertical averaging follows to reduce the spatial resolution to fit the image into the sensor configuration (4 vertical rings). For this evaluation, we kept the same sensor configuration (4 rings and 96 cells per ring) that was used for performance comparison of MAV navigation. After performing vertical averaging, the amount of horizontal shifts per frame corresponding to the target optic flow velocity and frame rate are precisely calculated in the scale of high resolution image width. Once the rotated pixel positions are determined by the horizontal shifts, horizontal averaging is performed on the updated rotated positions. The horizontally averaged image is plugged into the algorithm models to extract optic flows. If the number of frames reaches the target test frame number, then simulation is completed. Otherwise, the simulation task moves back to the "update rotated positions" and repeats this loop until the target frame number is met.

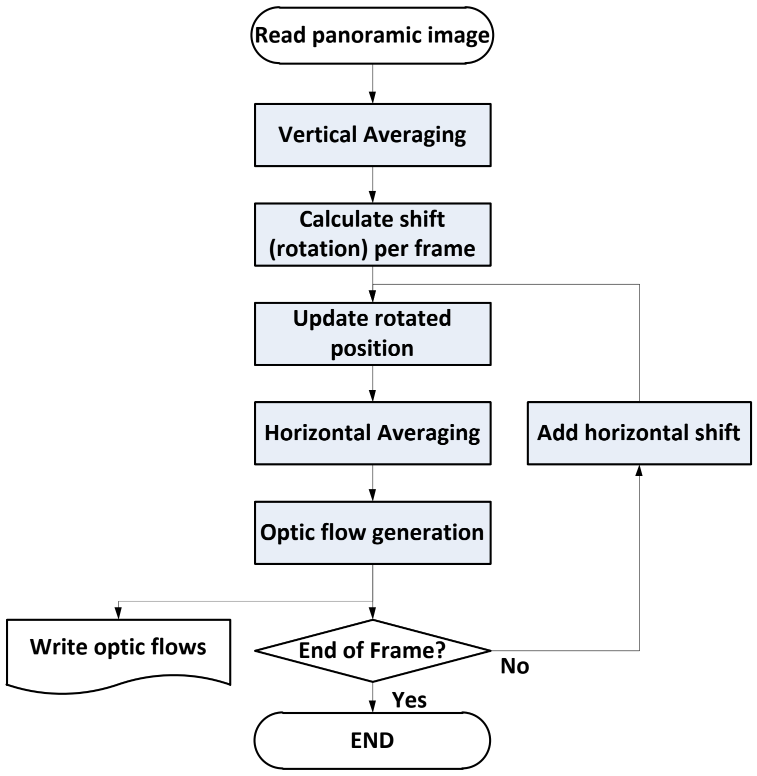


Figure 4-8: Flow chart for evaluation of rotational velocity extraction

4.5.1.2 Theoretical detectable velocity range

Figure 4-1 shows a sensor configuration to analyze theoretical detectable velocity range of the algorithm. If the number of deployed pixels is *nPixels*, the angle between neighboring pixels is equally $\Delta\varphi = 2\pi / nPixels\ [rad]$. The minimum time interval, which the algorithm can distinguish, is determined by a frame rate, and the maximum time interval is determined by both the frame rate and the maximum time-stamp global counter value of the algorithm as described in equation (6) and (7).

- $Min.\ \Delta t = 1/(frame\ rate)\ [sec]$ (6)
- $Max.\ \Delta t = (Max.\ global\ counter\ value)/(frame\ rate)\ [sec]$ (7)

Since the velocity is the ratio of the angle of neighboring pixels to an object's traveling time between the pixels, the equation of the measured velocity can be expressed as in equation (8). Thus, the maximum and minimum detectable velocity ranges are derived as in equation (9) and (10), respectively, from equations (6), (7), and (8).

- $v=\Delta\varphi/\Delta t$ (8)
- $Max\ v = (2\pi / nPixels\ ) * (frame\ rate)$ (9)
- $Min\ v = (2\pi / nPixels\ ) * (frame\ rate/Max.\ time\text{-}stamp\ counter\ value)$ (10)

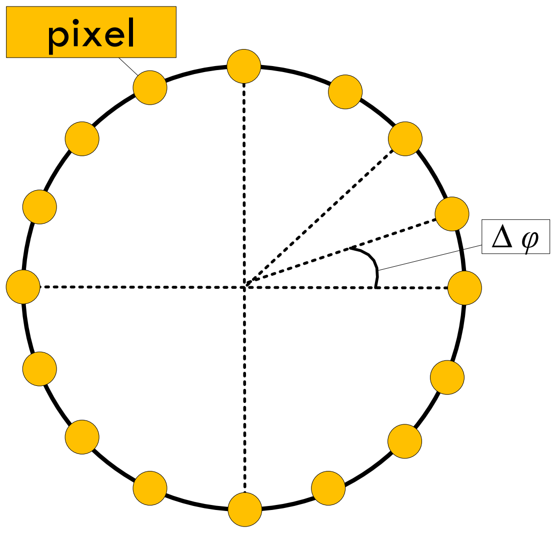


Figure 4-9: Sensor configuration to analyze the detectable velocity range

We performed the simulations for a simple horizontal bar pattern, which is moving from left to right, with a certain velocity and verified the theoretical detectable range. The parameters for the simulation were:

nPixels = 96, frame rate = 125 fps, Max. global counter = 128 (7bit)

From equations (9) and (10), the theoretical maximum and minimum detectable ranges were calculated to be 8.18 [rad/sec] and 0.06 [rad/sec].

4.5.1.3 Test results

We performed the simulation with different ranges of target optic flow velocities from 0.2 rad/sec to 2 rad/sec at a frame rate of 125 fps. The results are shown in Figure 4-10. The x-axis specifies target test optic flow velocities from 0.2 rad/sec to 2 rad/sec, and the generated optic flows from different algorithms are shown on the y-axis in the unit of pixels/frame. The black colored plotted points show the pre-calculated correct flows. Blue (labeled 'RMD'), red (labeled 'New'), and green (labeled 'PLK') plots correspond to each of the optic flows extracted from "Basic EMD," "Advanced EMD," which has been proven for a MAV's navigation [85], "Lucas & Kanade," the proposed time-stamp based algorithms, respectively. The error bars among the plotted points indicate a standard deviation of the estimated optic flows. The result shows that "Lucas & Kanade" algorithm outperforms the both neuromorphic algorithms in this rotational velocity evaluation. It should be noted that the "Basic EMD" cannot discriminate rotational velocities at all. However, the "Advanced EMD" gives a linearly increasing tendency in optic flow velocities until the target velocity reaches the sensor limitation though an additional scaling factor should be applied on the result. This limitation comes from EMD's time constants (around 1 rad/sec).

The simulation results of the proposed time-stamp-based algorithm for the given parameters are shown in Figure 4-11, and these results correspond well to their theoretical limit. Note that the estimation of the minimum detectable velocity should be revised. This is because we adopted the FTI time-of-travel method for our algorithm, which utilizes three neighboring pixels as described in the previous quarterly report, and we reduced the maximum time-stamp counter value to a half to save the hardware and power consumption. Thus, the equation for estimating the minimum detectable velocity should be revised as in equation (11), and the actual calculated number is 0.12 [rad/sec] for the given parameters.

$$Min\ v = (2\pi / nPixels) * (frame\ rate/(Max.\ global\ counter\ value/\mathbf{2})) \tag{11}$$

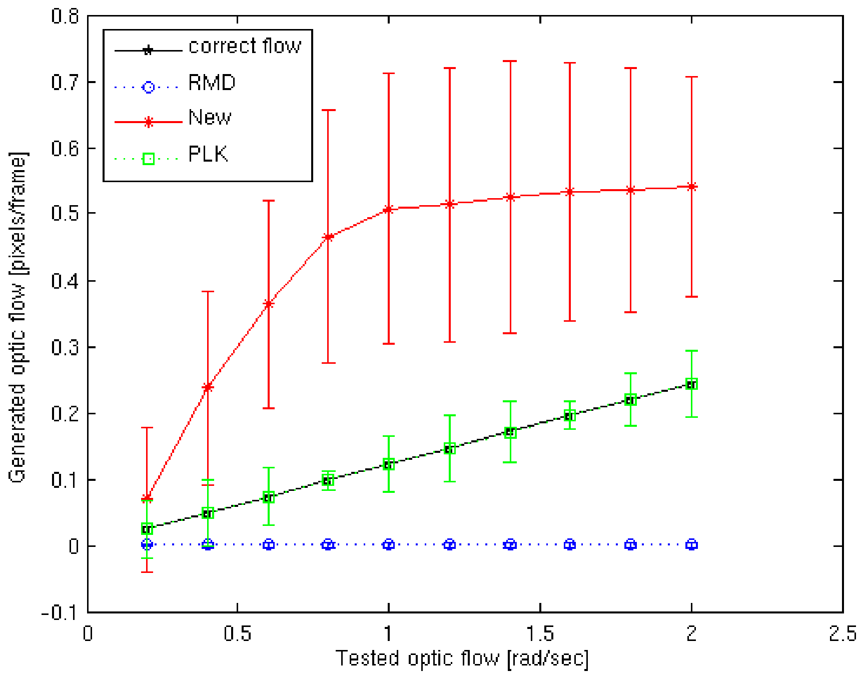


Figure 4-10: Generated optic flows from rotational velocity evaluation

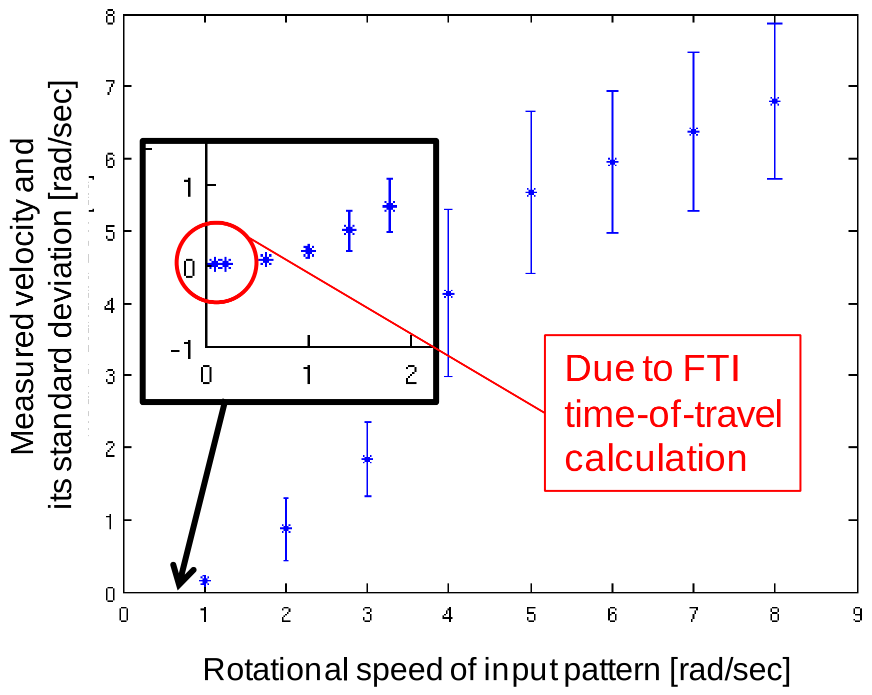


Figure 4-11: Detected velocity plot for different input rotational velocities

4.5.2 Optic flow algorithm performance evaluation for MAV navigation

4.5.2.1 Test patterns and sensor configuration

We evaluated the proposed optic flow algorithm by comparing to gradient-based optic flow algorithms in terms of how well the algorithms can provide information for a navigation control, which is implemented on our target MAV platform (i.e., wide-field integration platform [26]). For this evaluation, methodology should reflect the actual flight simulation of an autonomous MAV. We used ten different test image patterns which were captured while a MAV was moving around a computer-generated 3D virtual arena for 1.2 seconds, as shown in Figure 4-12 [85].

Optic flow sensors were configured to have four vertically stacked rings and the given test images' spatial resolution was fit into this configuration, as shown in Figure 4-13. Each ring consists of 96 photoreceptors (or cells) that capture a 360 ° view in the horizontal plane, and the optic flow algorithm generates four 1-D surrounding optic flows from each ring. Matched filter operations extract control information for MAV's navigation from the optic flows. For example, flight control parameters or spatial Fourier coefficients, A1, A2, B1 and B2, were extracted from the different matched filters, which provide the information of orientation shift, lateral asymmetry, forward velocity, and collision approaching of a flying vehicle, respectively.

The navigation performance evaluation environment is summarized in Figure 4-14. Test images and correct flows for ten flight paths are stored in a desktop, and then using the stored test images each optic flow algorithm is simulated on a computer.At first, ten estimated optic flows are directly compared to correct optic flows in order to find a correlation for optic flows. Next, navigation control parameters are generated using the estimated optic flows. Finally, the control parameters correlation between the estimated optic flows and correct flows are extracted.

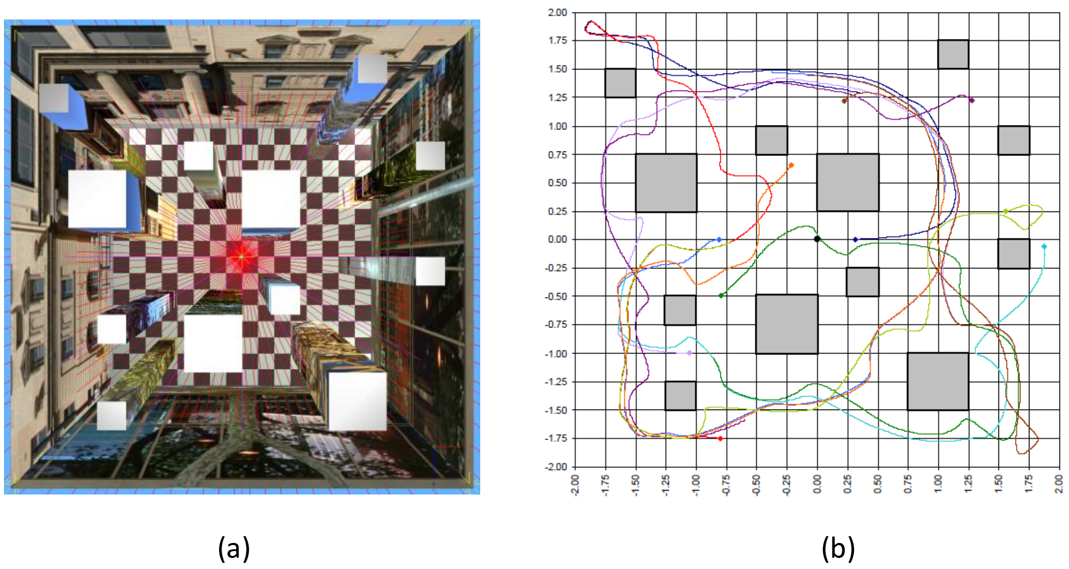


Figure 4-12. A flybot flight simulation: (a) Virtual flight arena, (b) ten fight paths of an MAV [85]

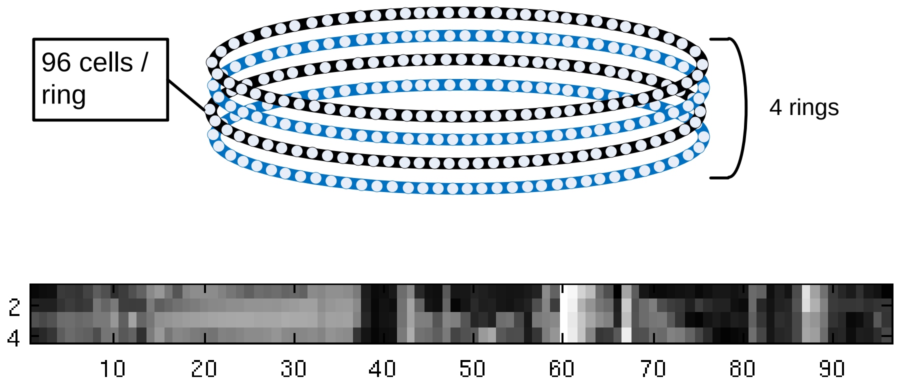


Figure 4-13. Optic flow sensor configuration and a captured image for the performance evaluation.

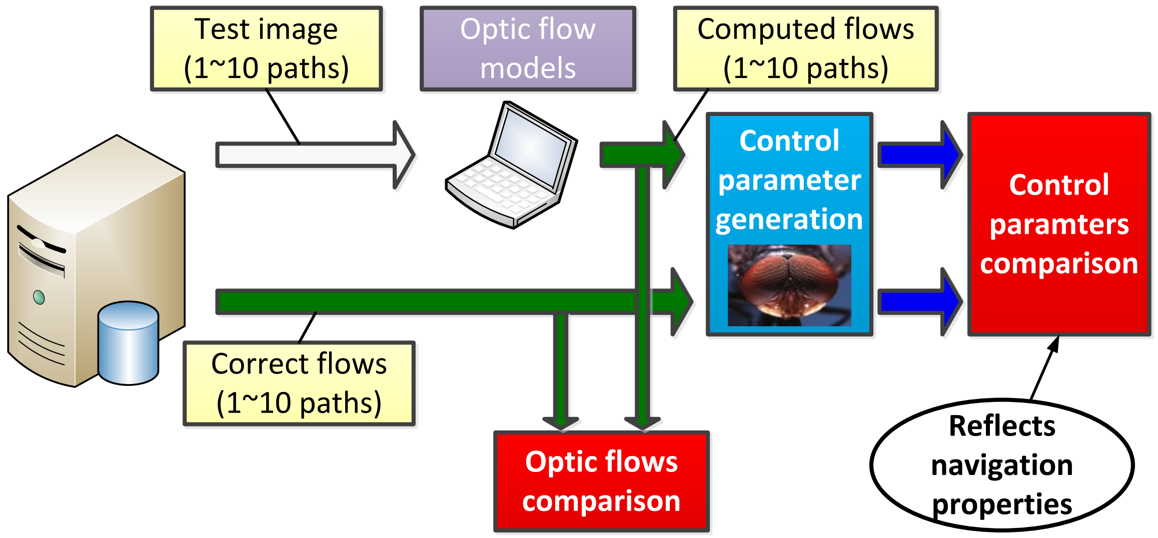


Figure 4-14. Navigation performance evaluation environment setup

4.5.2.2 Evaluation results

To provide a quantitative performance comparison, we examined the correlation between each optic flow algorithm and the correct optic flow over the empirical distribution, treating the two as random variables. Since the correlation is proportional to the similarity of the two random variables, the higher correlation indicates two random variables are closer to each other. In Table 1, the "Optic flow" column shows the correlation between each estimated optic flow and the correct flow, and "A0" to "B1" columns indicate the correlation between spatial Fourier coefficients. The "Basic EMD" algorithm is a simple neuromorphic algorithm which consists of photodetectors, a high-pass filter, and elementary motion detectors (EMDs). The "Minimal Gradient" algorithm is a simpler version of the Lucas & Kanade algorithm that utilizes less spatial window constraints. The "LK" is the state-of-the-art optic flow algorithm, and "Tanner's" algorithm is a proven neuromorphic algorithm for a wide field integration navigation control algorithm. The proposed algorithm provides higher mutual information for all spatial Fourier modes than the proven advanced EMD [85] and minimal gradient, therefore it delivers enough information for an adequate autonomous navigation control.

Table 4-1: Correlation between the optic flow generated from each optic flow algorithm and the correct optic flow using the test images

	Optic Flow	A0	A1	A2	B1	B2f
Proposed	0.159	0.408	0.175	0.189	0.282	0.244
LK	0.414	0.509	0.259	0.442	0.236	0.367
Advanced EMD	0.143	0.249	0.136	0.180	0.244	0.202
Minimal Gradient	0.402	0.361	0.0482	0.0503	0.0587	0.0743

4.6 Summary

The bio-inspired time-stamp-based optic flow estimation algorithm, which borrows two major aspects in an insect's eye: a temporal HPF for feature extraction, and time-of-travel measurement. These two aspects are implemented totally differently as compared to conventional bio-inspired optic flow algorithms in order to provide the optimal A/D partitioning for effective hardware implementation. A discrete-time HPF and 1-b comparator extract a moving feature and deliver digitized 1-b data to the remaining digital blocks. This 1-b A/D conversion is simple and fast; it contributes to reducing conversion power consumption and to increasing temporal resolution. Time-stamp-based time-of-travel measurement is implemented by a digital arithmetic subtractor; therefore, the algorithm provides a optimum optic flow, which is insensitive to temperature and process variations.

CHAPTER 5
BIO-INSPIRED 1D OPTIC FLOW SENSOR AND PHYSICAL VERIFICATION OF TIME-STAMP-BASED OPTIC FLOW ALGORITHM

5.1 Introduction

In the previous chapter, the proposed time-stamp-based optic flow estimation algorithm is introduced and evaluated for low power and payload constrained micro air vehicles (MAVs). This algorithm is integrated on the first prototype optic flow sensor in order to prove feasibility after physical implementation.

The first prototype sensor integrates a temporal high pass filter in a 2D pixel array; a moving feature detector and 8-b time-stamp latch in column-level; and 1-D time-stamp-based optic flow estimation core including time-of-travel measurement and velocity convertor in chip-level. The implemented discrete-time temporal high pass filter measures a temporal contrast change in each pixel. The measured contrast change is compared to a pre-determined threshold value to determine a moving feature existence at each pixel; if the measured contrast change at a certain pixel is bigger than the pre-determined threshold, then a moving feature is considered to be arrived at the pixel location. This feature detection function is implemented in each column as digital circuits using a 1-b comparator. Then, the detected 1-b feature information enables the 8-b time-stamp to latch the frame counter value to record or to update recent time when a moving feature is arrived. Thus, a 1D time-stamp array is generated on column-level 8-b latch array. Using the 1-D time-stamp image, a time-of-travel measurement and velocity conversion is performed in the chip-level digital core. As a result, a pair of 8-b motion information, whose sign value indicates a direction and magnitude specifies speed, is delivered per each pixel. In addition to the 1-D optic flow generation from the integrated core, 2-D optic flows can be generated by the 2-D feature information from the 2-D pixel array and 2-D time-stamp processing from external FPGAs.

This chapter covers from the design to measurement of the first prototype time-stamp-based optic flow sensor. Firstly, the proposed sensor architecture to efficiently implement the time-stamp-based algorithm in analog/digital (A/D) mixed-mode circuits is introduced. Secondly, each block design of the proposed architecture is explained by a signal processing order: a discrete-time temporal high-pass filter; feature detection circuits; time-stamp latch; and digital time-of-travel and velocity calculation circuits. Finally, the measurement results of the fabricated sensor are discussed.

5.2 Chip architecture

The sensor architecture of the first prototype time-stamp-based optic flow sensor, which integrates the 1-D time-stamp-based core, is shown in Figure 5-1. Two main goals to be considered while building this architecture are: carefully partitioning the analog/digital (A/D) domains to balance processing loads, and providing both normal images for micro-ocelli (μ-Ocelli) applications and optic flows for micro-compound eye (μ-Compound Eye) applications.

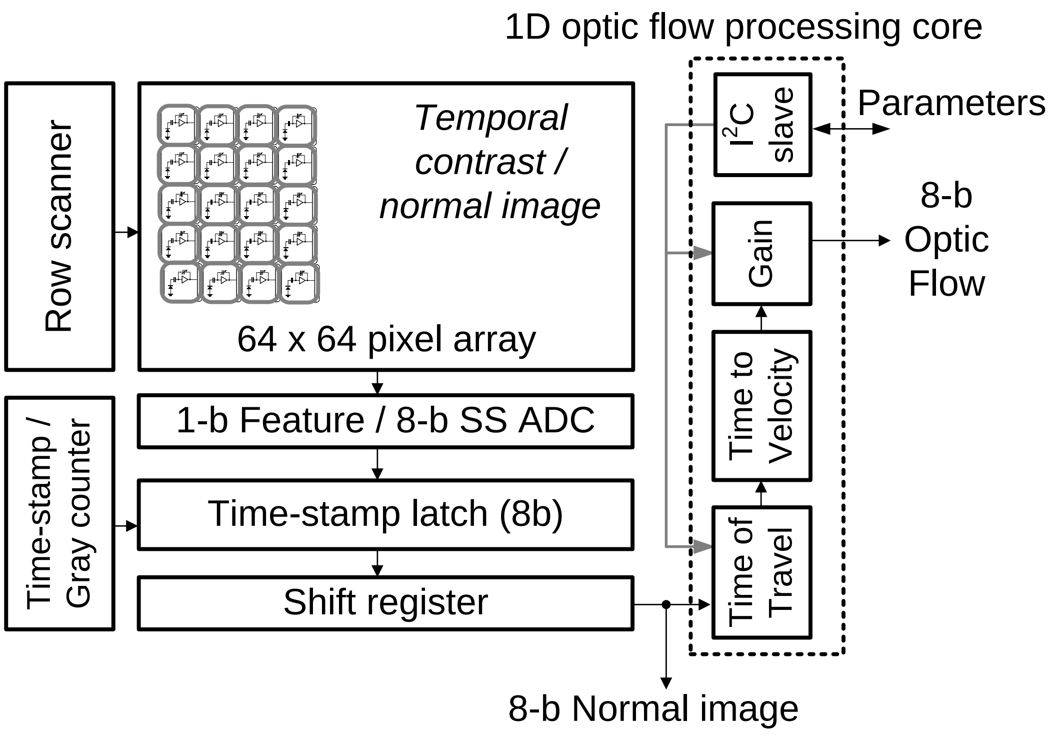


Figure 5-1: First prototype time-stamp-based optic flow sensor architecture

5.2.1 A/D partitioning

For the A/D partitioning boundary, the feature detector block is chosen to be the boundary because the 1-b signal communication between the A/D domains minimizes a chance of information loss from A/D conversion. Also, a temporal high-pass filtering is efficiently implemented in the analog domain to reduce A/D conversion power consumption in an order of magnitude. Table 5-1 compares required memory spaces of a frame memory between analog and digital implementation. In analog implementation, one sampling capacitor (1Cs) per pixel to store illumination information is required. On the contrary, in digital implementation, N-b per pixel memory unit is required and typical N-b values are 12-b, 10-b or 8-b. SRAMs are the only solution to achieve the minimum physical area because DRAMs are difficult to be integrated to the logic process due to fabrication compatibility. In order to implement the temporal high-pass filter in the digital domain, illumination information from a photoreceptor on each pixel must be converted to digital codes and must be stored to SRAMs. This means N-b A/D conversion is required for the digital high-pass filtering operation; Only 1-b A/D conversion is required for the analog high-pass filtering operation. Considering 10-b per pixel A/D conversion, 10 times power saving can be achieved by locating the A/D boundary after the high-pass filter.

Table 5-1: Hardware resource comparison of a discrete temporal high-pass filter implementation

	Analog (Capacitors)	Digital (SRAMs)
Memory space	1Cs per pixel	N-bit per pixel
A/D Sampling frequency	fs	N*fs

5.2.2 Two modes of operation: optic flow generation, and normal image capture

This architecture shown in Figure 5-1 supports both normal image capturing and optic flow sensing modes. In the normal image mode, the sensor generates 8-b digital images using the integrated single slope (SS) ADCs. The row scanner activates one row at a time to readout a conventional 3T CMOS active pixel sensor (APS) through source followers. The readout values are compared to a ramp signal, and latch the 8-b gray counter value in column-parallel for SS ADC operation.

In the optic flow generation mode, the sensor operates as a bio-inspired vision chip by estimating optic flows from the integrated time-stamp-based optic flow algorithm. Firstly, a temporal contrast change is monitored from consecutive frames in the pixel array, operating as a discrete-time (DT) HPF. A 1-b comparator detects the feature when there is a significant change in the temporal contrast beyond a threshold which can be dynamically adjusted. A time-stamp latch records the time when the latest feature arrives in the pixel array. The recorded time-stamp information is kept until the next frame. The information can be moved to a shift register and fed into the chip-level digital optic flow processing unit to decode the time-stamp information into velocity. The I^2C slave interface in the digital core allows on-the-fly setting of conversion gains from an off-chip host.

5.3 Pixel architecture

Figure 5-2 shows the pixel architecture and equivalent circuit for temporal contrast and normal image modes, respectively. The pixel includes a sampling capacitor (C_1) and the dual purpose capacitor (C_2) for setting a programmable gain amplifier (PGA) gain and providing wide dynamic range (WDR) operation. In WDR operation, C_2 is connected to the photodiode in order to increase the well capacity, thus stretching the input signal range. Frame difference is monitored in the temporal contrast mode by sampling the pixel voltage ($V_p(t_1)$) of the previous frame from C1 first, and then applying the current pixel voltage ($V_p(t_2)$) to the PGA. The supply voltage for photodiodes and source followers is 3.3V. The PGA operates at 1.8V and is only enabled during the signal transferring period to reduce static power consumption.

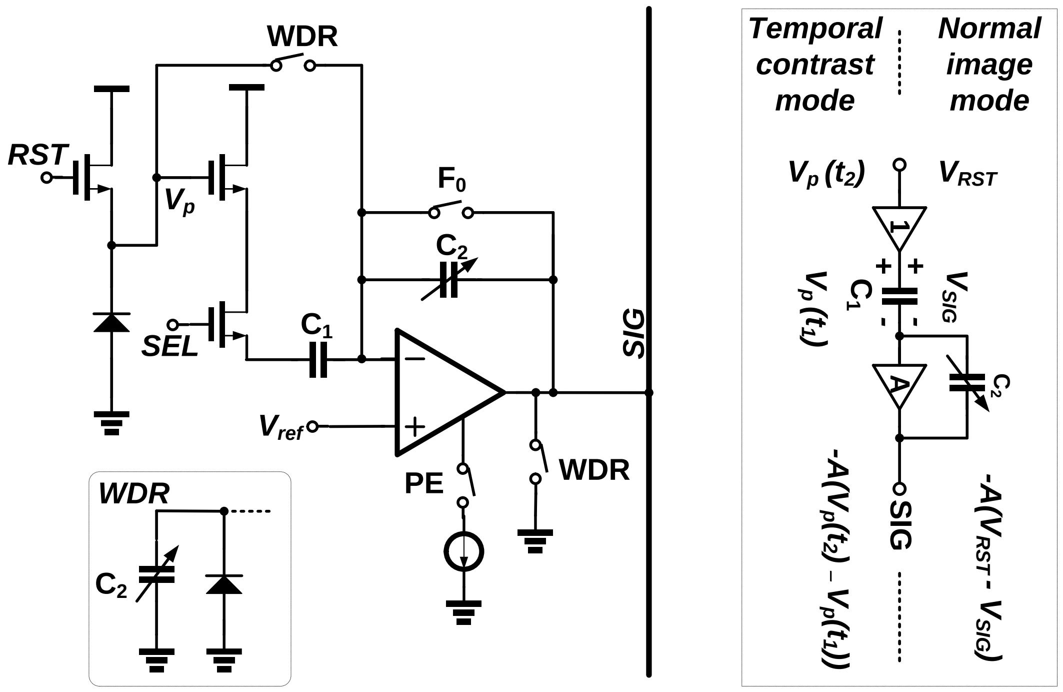


Figure 5-2: Pixel architecture and equivalent circuit

5.4 Column circuits

Figure 5-3 shows the column-parallel circuits implemented to realize the time-stamp based optic flow algorithm in the digital processing core. The column-parallel unit consists of a comparator and an 8-b time-stamp latch. The comparator will generate a 1-b signal in each pixel when the temporal contrast change is larger than a feature threshold. The detected 1-b feature signals in column-parallel enable the time-stamp latches to record the corresponding time using the value of the integrated time-stamp counter, which is implemented as a circulating 8-b frame counter. Therefore, the column-parallel time-stamp latches contain the time information of when the most recent feature has occurred in each pixel. The supply voltage for the comparator and time-stamp latch is

0.9V. In the normal image mode, the column-parallel comparators and latches operate as a SS ADC by applying a ramp signal to the feature threshold node.

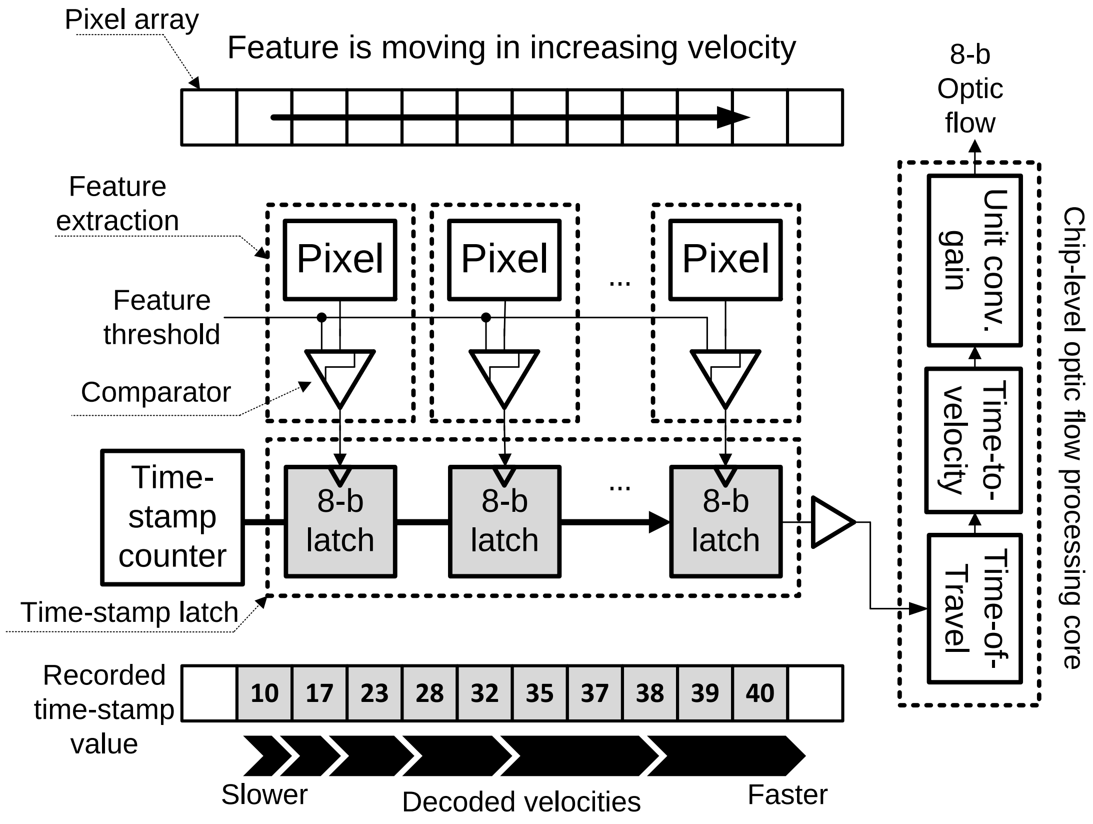


Figure 5-3: Column circuits and implemented 1-D time-stamp-based optic flow core

5.4.1 The circular time-stamp counter and sign conversion circuits

The time-stamp counter is implemented using an 8-b circular counter, which increments for every frame and decreases to the lowest at its maximum value. Because of this circular operation, discontinuous time-stamp values can be obtained for continuous motion, which occurs while the time-stamp counter is circulating. Therefore, the time-stamp value reordering process is required to align the values in a chronological order.

The implemented re-alignment method is illustrated in Figure 5-4. Though the example uses a 3-b counter, the method is easily extended for implementing higher bit-width counters such as 8-b or 10-b. Firstly, zero (0) value is reserved for the no-stamp code, which implies no moving feature is detected on the pixel; thus, if the latched time-stamp value is zero, then it means no time-stamp value is recorded. This no-stamp code is decoded using column-parallel digital circuits as shown in Figure 5-5. The 1-b no-stamp flag is generated when the recorded time-stamp (TS) is equal to zero. This flag is utilized for negating time-of-travel measurement in the chip-level digital core because the recorded time-stamp is invalid.

The lowest circulating counter value is one (1), and the maximum value is 2^n-1, here n is the bit-width of the counter and 7 is for the 3-b counter. In the figure, the circular counter increases in the clockwise direction. The current counter value is indicated by a solid arrow as used on a clock. Therefore, the current time-stamp counter value is 2, which means the recent time is 2; the oldest time-stamp value is 3, whose chronological relationship is illustrated by a dotted line. To recover this relationship, the algorithm subtracts 2^n-1, which is 7 in this example, for bigger recorded time-stamp values than the current time-stamp counter value. As a result, the chronological reordering is automatically accomplished by expressing the reversed events using negative values.

This reordering conversion is implemented in column-parallel digital circuits as shown in Figure 5-6. The recorded time-stamp (TS) is first compared to the current time-stamp counter (TS counter). If the recorded value is bigger than the current counter, then the selection signal of the multiplexer (flagTS_Conv) is set to 1 in order to output the converted value, which is TS-255 in 8-b implementation. Otherwise, the multiplexer selection signal is set to 0 to bypass the recorded time-stamp.

The recorded time-stamp must be cleared to the no-stamp code (0) after one period of the time-stamp counter is passed in order not to use the old time-stamp values for time-of-travel measurement. Because the sensor uses the 8-b time-stamp counter, the recorded time-stamp value is maintained for 255 frames. This clearing circuit is illustrated in Figure 5-7. If a moving feature (Feature) is not detected at the current frame and the recorded time-stamp is equal to the current time-stamp counter, the clear signal (Clear) is set.

The overall column-parallel digital time-stamp circuit is shown in Figure 5-8. The supply voltage is scaled down to 0.9V for low power consumption. The time-stamp latch is cleared to the no-stamp code if the clearing condition in Figure 5-7 is met. If a feature is detected, then the time-stamp latch is updated to the current time-stamp counter value. The latched value is further processed for the no-time stamp decoding and reordering conversion; those signals are moved to shift registers and finally feed into a chip-level digital processing core.

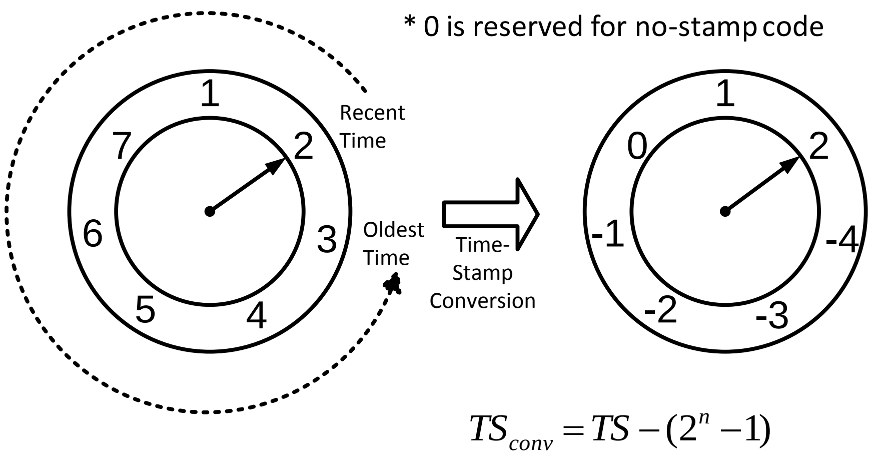

* 0 is reserved for no-stamp code
1
7
2
6
3
5
4
Recent Time
Oldest Time
Time-Stamp Conversion
1
0
2
-1
-4
-2
-3
$TS_{conv} = TS - (2^n - 1)$

Figure 5-4: The implemented circular time-stamp counter and value reordering

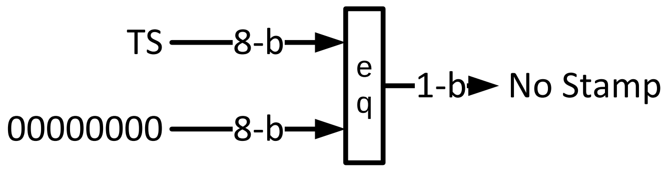


Figure 5-5: Column-parallel no-stamp code decoding logic

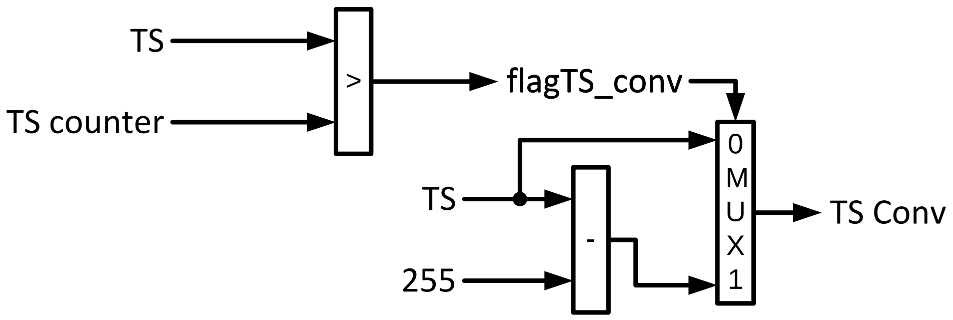


Figure 5-6: Column-parallel time-stamp value reordering logic for the 8-b circular counter

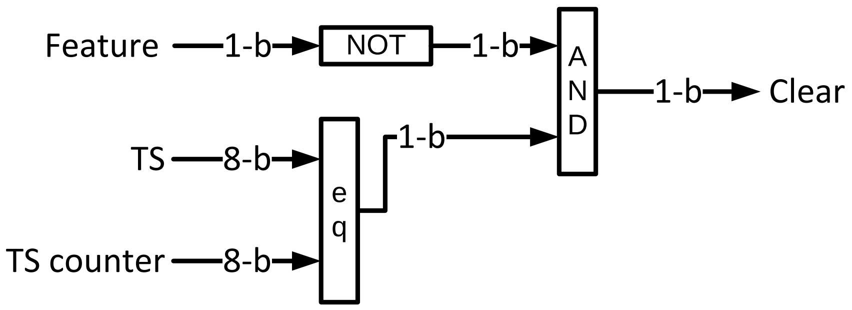


Figure 5-7: Column-parallel recorded time-stamp clearing logic

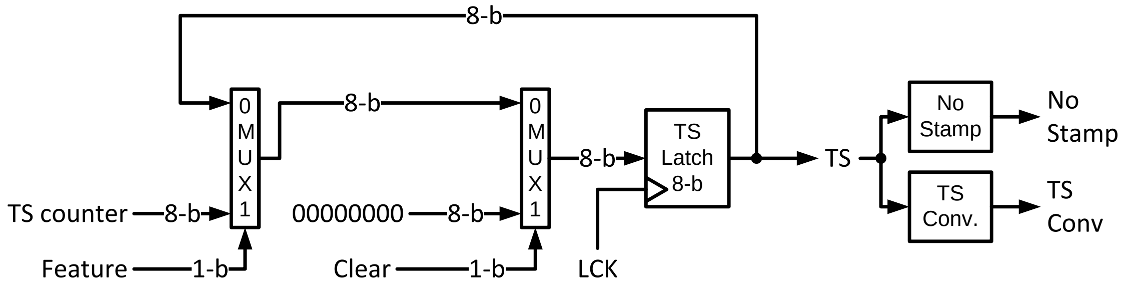


Figure 5-8: Column-parallel digital time-stamp circuits implemented using 0.9V supply

5.4.2 Column-parallel time-stamp sign conversion example from chip measurement

Figure 5-9 shows the details of how the time-stamp optic flow algorithm works. Figure 5-9 (a) shows the actual values recorded in time-stamp latches when two bars are moving horizontally as shown in Figure 5-16. In this example, the current frame number (the time-stamp counter) is 168. The 8-b time-stamp counter is designed to increase by one per each frame, and if the value reaches 255, then it is reset to 1 and continues to increase. Thus, the value (168) in the time-stamp latch in the plot is the most recently updated event, and other values bigger than 168 are old events.

To recover this relationship, we inserted the time-stamp sign conversion function between the time-stamp latch and the shift registers. Figure 8 (b) shows the sign-converted time-stamp values by subtracting 256 for the values bigger than the current time stamp counter (168). By using the converted value, time-of-travel measurement is simply calculated by a subtraction operation. Since the input pattern is horizontally moving bars with 3 different velocities (Figure 5-16), the plot explicitly shows three different slopes (dotted lines), which are inversely proportional to the amplitude of the velocity.

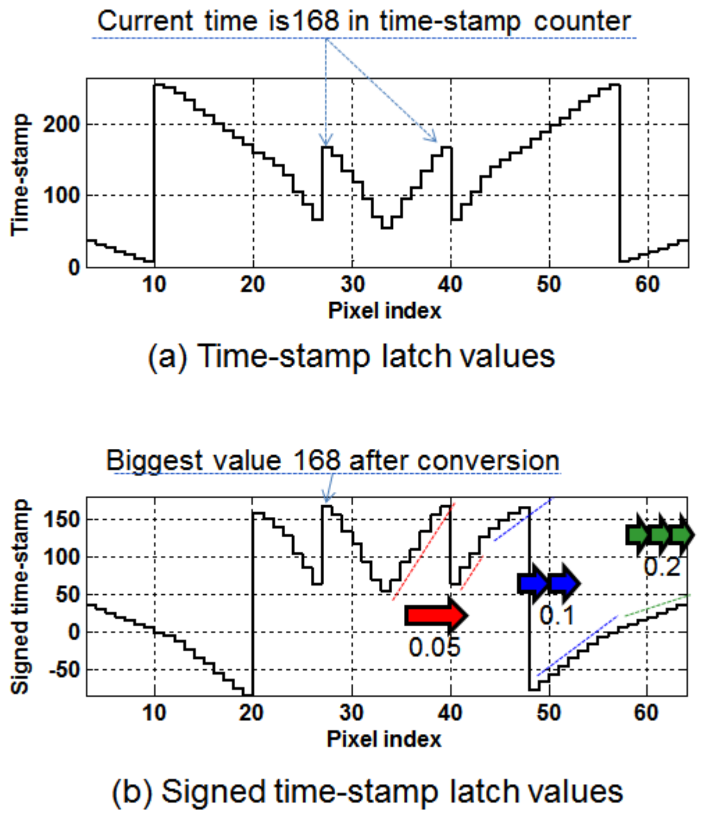


Figure 5-9: Column-level time-stamp sign conversion

5.5 Time-stamp-based optic flow estimation core

The optic flow, the velocity of a moving feature, is estimated by decoding the relationship between spatially neighboring time-stamp values. The velocity is determined by the traveling time between the two neighboring pixels since the moving distance of the feature is fixed to one pixel. As shown in Figure 5-3, the velocity of a moving feature increases, as the difference in the neighboring time-stamp values decreases. This calculation is accomplished by a chip-level digital processing core which consists of three blocks: time-of-travel measurement, time-to-velocity converter, and unit conversion gain blocks. In the time-of-travel measurement block, an 8-b subtractor determines the difference of two neighboring time-stamp values. The inverse of the traveling time is implemented by a look up table (LUT) in the time-to-velocity convertor block. The 8-b multiplier in the unit conversion gain block calculates the final estimated optic flow that will compensate for all the external system parameters such as frame rate, field of view, etc. The 1-D optic flow core is synthesized using standard cells operating at 1.8V.

The chip-level time-stamp processing chain is illustrated in Figure 5-10. The 1-b no-stamp flag (No_Stamp) and converted time-stamp (TS_Conv) are shifted from the column circuits. The shifted data are aligned on five stages D flip-flops; thus, 5-tap spatially neighboring No_Stamp and TS_Conv samples are simultaneously accessed for time-of-travel measurement: TS_Conv[n-2] to TS_Conv[n+2], No_Stamp[n-2] to No_Stamp[n+2], here n is a spatial index. Because the digital FTI algorithm requires three neighboring time-stamps to measure one directional time-of-travel, two FTI cores refer to each half of the aligned TS_Conv and No_Stamp samples to measure two directional time-of-travels, which move from right to left (R2L) and from left to right (L2R).

The implemented FTI logic is shown in Figure 5-11. In the figure, the spatial index n corresponds to the center of the 5 aligned tap in Figure 5-10, and the spatial index n-2 is the right most tap (n+2) for the R2L FTI and the left most tap (n-2) for L2R FTI measurement. The algorithm simultaneously calculates both time-of-travels of the center and neighboring locations, dT[n] and dT[n-1] in Figure 5-11, using two subtractors. Then, the absolute value of the difference between the calculated time-of-travels is compared to the adjustable threshold (FTI_Threshold) for the motion consistency check, which verifies the direction and velocity of extracted motion are in the similar range of the neighboring location. If the motion consistency check is passed, the remaining condition is to find if any no-stamp code is used for the calculation. This condition is implemented using a 3-input NOR gate as illustrated in Figure 5-12. Any no-stamp flag forces the NOR gate's output to zero; as a result, the time-of-travel valid flag (FTI_EN) is also invalid by the AND gate in Figure 5-11. This valid flag signal is registered and bypassed for all remaining pipeline stages to synchronize with calculated data. If the FTI_EN is valid, the measured time-of-travel from the center location, dT[n], is registered via D flip-flops for pipeline processing.

The measured time-of-travel is converted to velocity by a look-up table (LUT) shown in Figure 5-13. The LUT is designed to support 8-b fixed-point data processing; each entry of the LUT is multiplied by 2^8 to the reciprocal (1/x) function. In order to save LUT entries, only unsigned time-of-travels are itemized as a table. Instead, sign extraction and absolute value conversion are performed for input data; thus, only a converted unsigned integer is fed to the table. Then, the original sign is recovered at the end of processing by multiplying previously extracted sign value.

Finally, the converted velocities for opposite directions are merged and multiplied by a user controlled gain, which is used for optic flow unit conversion or additional digital gain control as shown in Figure 5-10. The 8-b fixed point calculation is shifted to the right to align the decimal point. The amount of the right shift is also controlled by a user to provide flexible decimal places: if the amount of the shift is 8, then output data is 8-b integer, if the amount is 6, output data consists of 6-b integer and 2-b below decimal point.

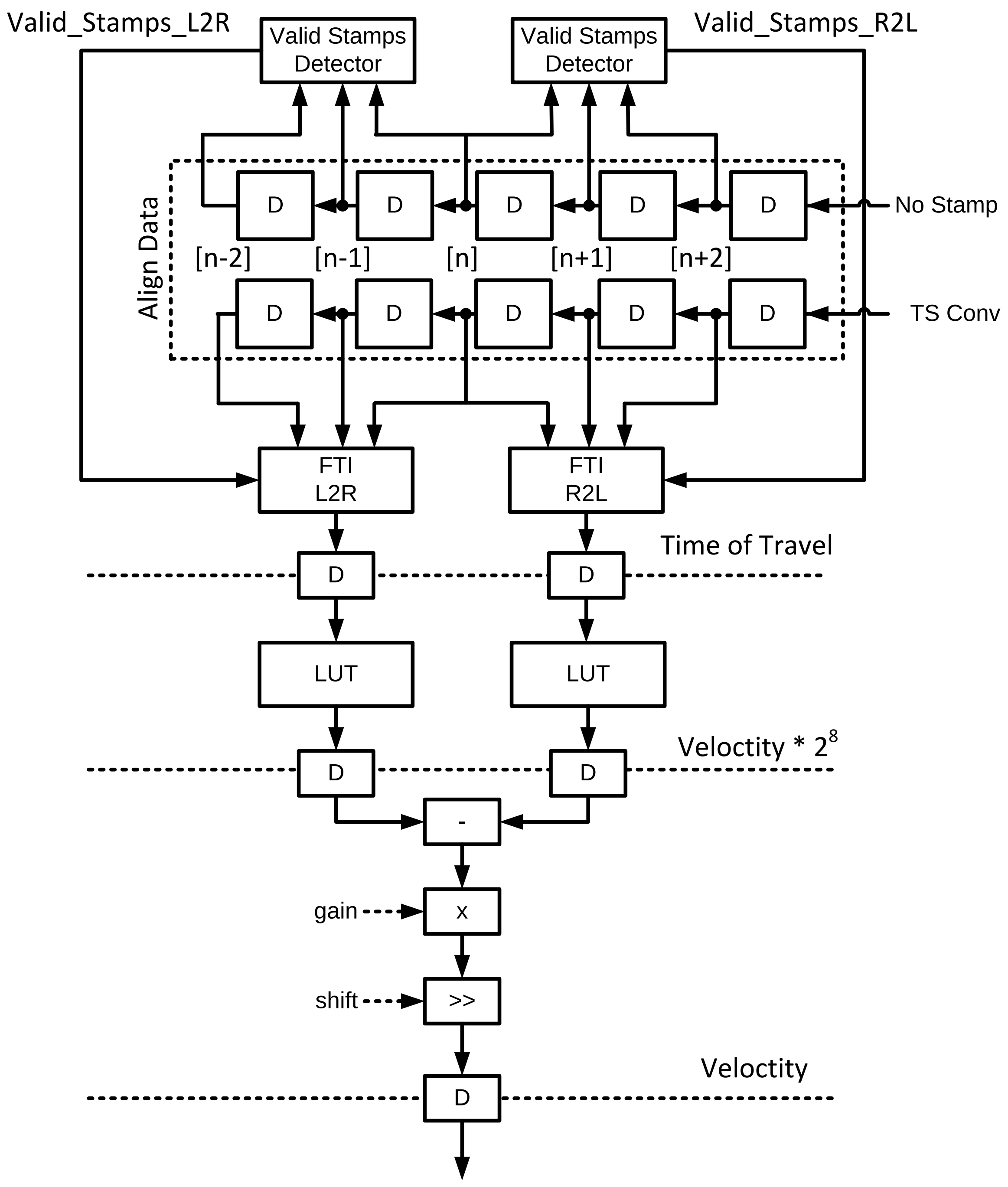


Figure 5-10: Chip-level time-stamp-based 1-D optic flow processing core

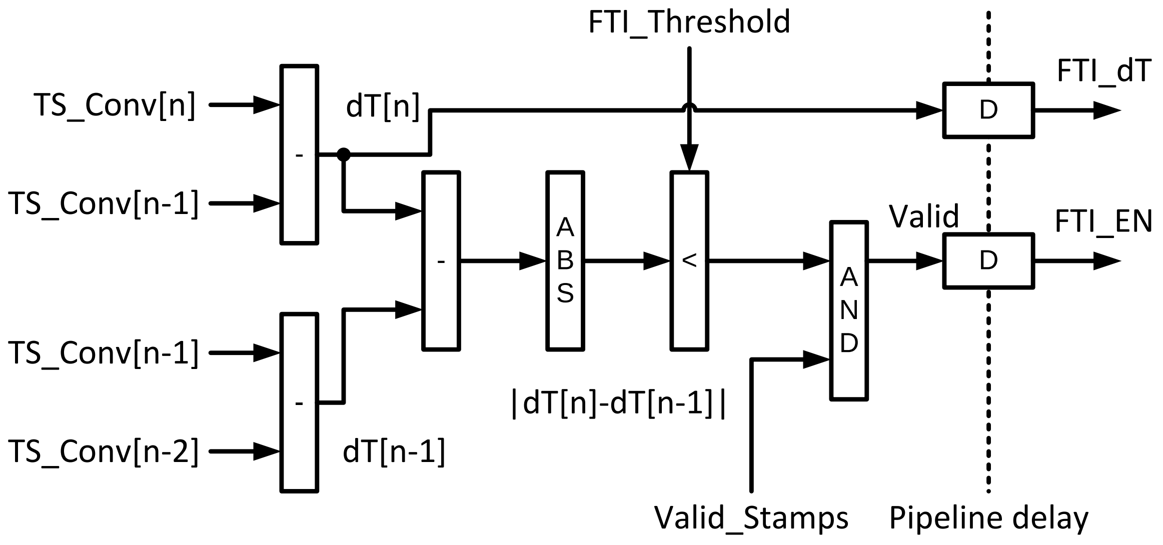


Figure 5-11: Implemented digital facilitate-trigger-and-inhibition (FTI) circuits

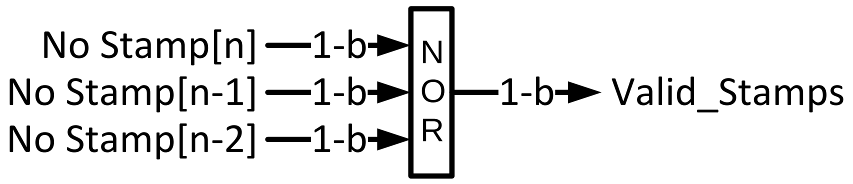


Figure 5-12: Valid stamps detection logic

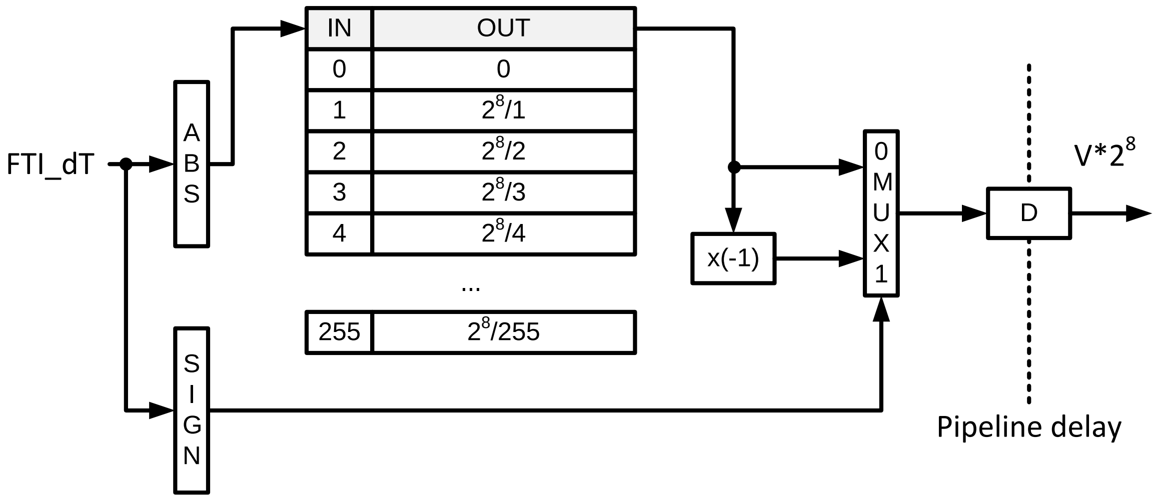


Figure 5-13: A velocity conversion look-up table (LUT)

5.6 Chip characteristics

A prototype chip was fabricated using 0.18μm 1P4M process. A chip micrograph is shown in Figure 5-14. In order to prove the feasibility of a MAV's application, optic flow accuracy and power consumption are mainly characterized. The linearity of generated optic flows is characterized by averaging optic flows from a moving horizontal bar pattern with a constant velocity in pixel/frame.

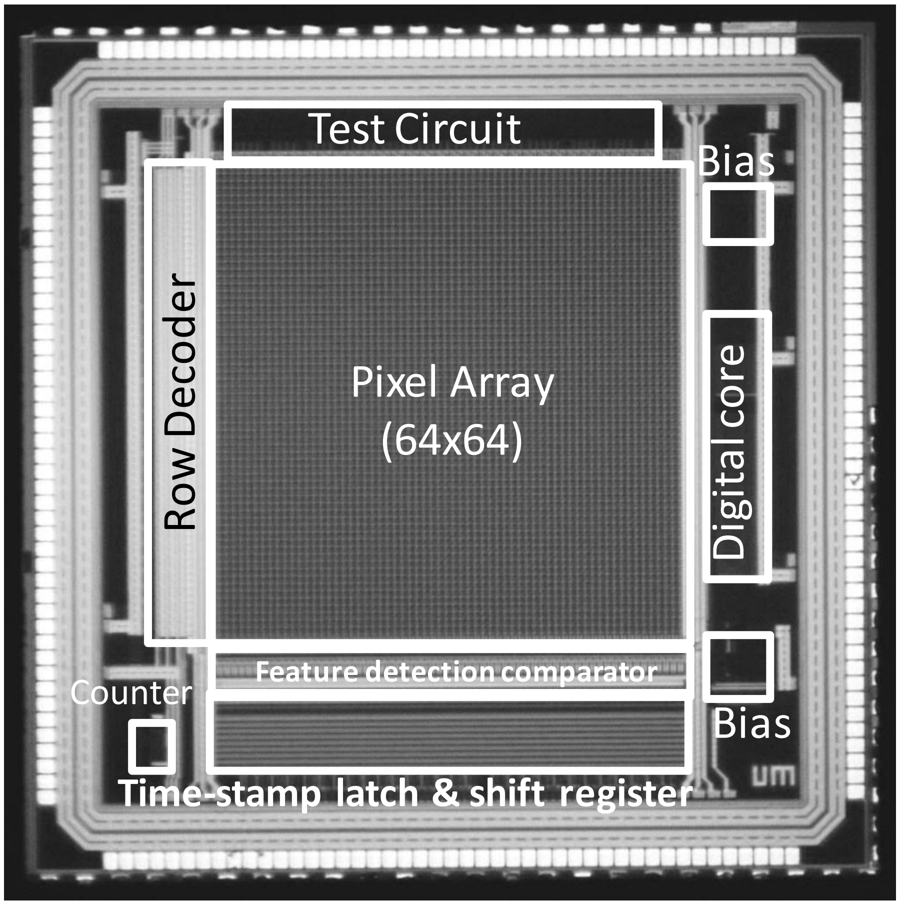


Figure 5-14: Chip micrograph

5.6.1 Measurement setup

The block diagram of the measurement environment for the first prototype chip is shown in Figure 5-15. The prototype chip supports three output modes to generate a captured image, 1D optic flows, and feature information for 2D array. A 3-to-1 multiplexer selects one of the generated outputs from the prototype chip and sends it to a data acquisition (DAQ) system in PC. LabView converts the delivered output data to a proper format in order to display it on a monitor. One FPGA, FPGA1 in Figure 5-15, on the PCB generates all control signals to operate the prototype chip, and the other FPGA, FPGA2 in Figure 5-15, stores time-stamp information for 64x64 2D array and perform an algorithm to generate 2D optic flows by programming FPGA. The PCB design has been completed, and programming for FPGAs and LabView is being performed.

The first prototype chip test is planned as shown in Table 5-1. The operation of the photodiodes will be verified first. With an off-chip ADC, the output of the photodiodes is converted to digital data so that DAQ and LabView can display an image on a display. After the photodiodes verification, the image signal path will be tested in order to check the operation of column-level circuits and latches. Then, the on-chip 1D optic flow digital processing core will be tested by displaying 1D optic flows from the prototype chip. Synthetic image patterns will be utilized to characterize the detectable velocity range of the fabricated chip. A unit of time interval of the time-stamp is controlled by changing the frame rate. The 2D algorithm will be elaborated and demonstrated with an off-chip FPGA that stores time-stamp information of 2D array and processes the 2D algorithm.

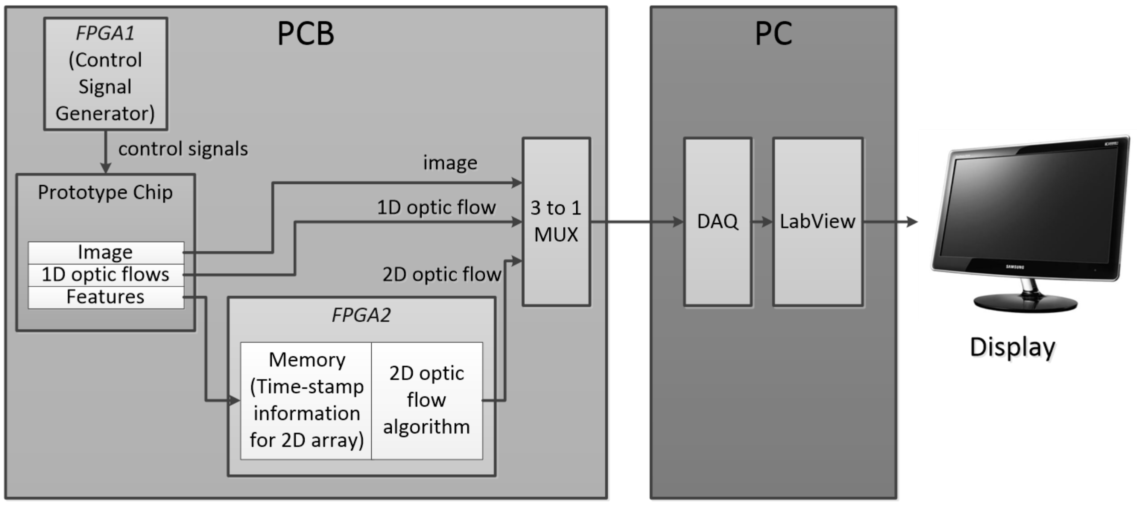


Figure 5-15: Block diagram of measurement environment for the first prototype chip

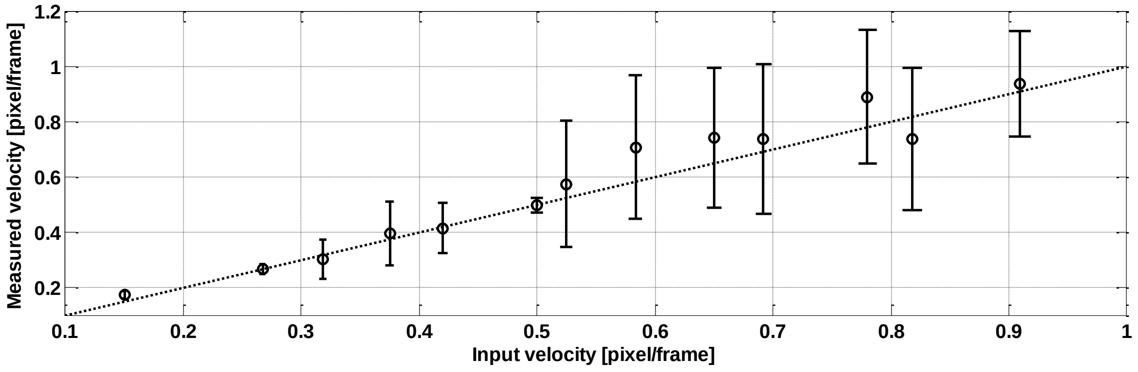
Table 5-2: Measurement plan for the first prototype chip

Test items	Description
Photodiode	With an off-chip ADC, the output of photodiodes is converted to digital data so that DAQ and LabView can display an image on a display.
Image signal path	The image signal path inside the prototype chip is tested by displaying image data, which are directly generated inside the prototype chip. The image signal path includes photodiodes, column circuits, and latches.
1D optic flow signal path	On-chip digital processing unit to perform the time-stamp based 1D optic flow algorithm is tested by getting 1D optic flows from the prototype chip. To characterize optic flow resolution, synthetic input video patterns with different rotational velocities are used. Different frame rate settings control the time interval of the time-stamp unit.
2D optic flow algorithm	With an off-chip FPGA, time-stamp information for 2D array is stored and processed to generate 2D optic flows. This test is for elaborating 2D algorithm and real-time demonstration with real environmental test cases.

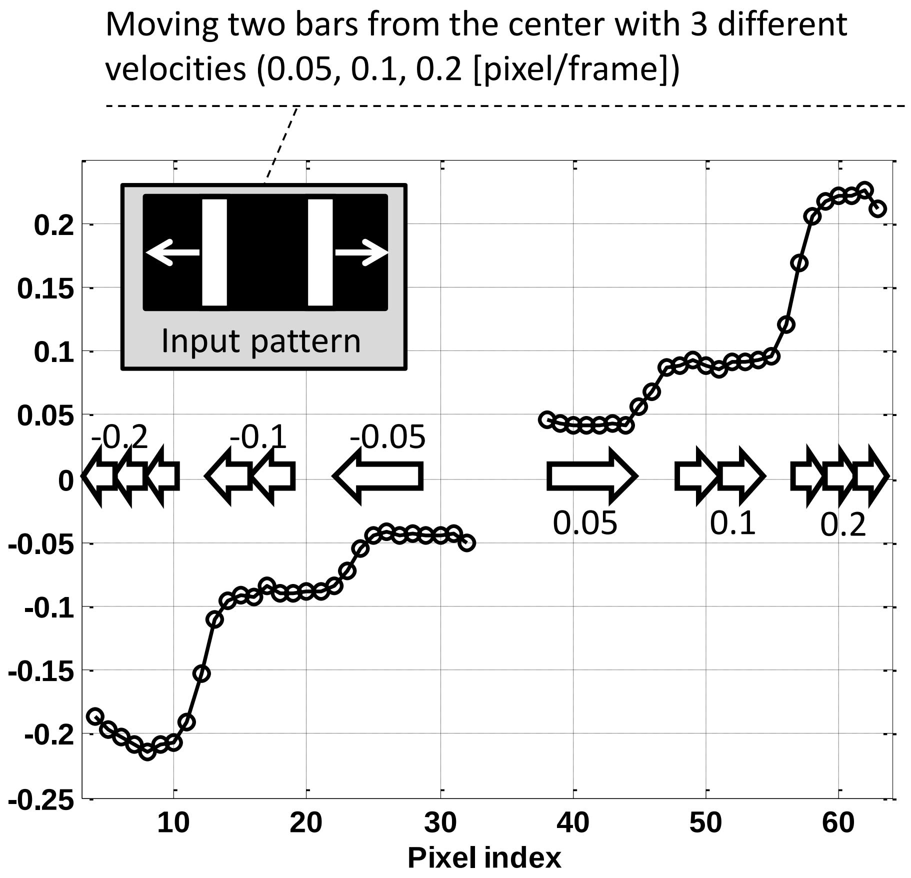
5.6.2 Measurement results

5.6.2.1 1-D optic flow test results

Figure 5-16 shows the performance of the integrated 1-D digital optic flow core. The linearity curve in Figure 5-16 (a) was characterized by averaging the output optic flows while applying fixed velocity input bar patterns with a constant velocity in [pixel/frame]. The patterns were computer-generated and exposed by an LCD monitor screen. Because the patterns displaying on the LCD monitor and the projected images on the sensor were not perfectly matched as expected, the actual input velocity of the pattern visualized at the sensor side was different from the expected input velocity while building the pattern. Thus, the measured steps in the x-axis are not equally spaced. In order to recover the actual input velocity that the sensor captured, we also collected a series of normal images (256 frames) per input velocity and then re-calculated the bar speed in the computer. In Figure 5-16 (a), the circles and error bars indicate the measured mean velocity and its standard deviation. The measured mean velocity linearly follows the input velocity, while the range of standard deviation increases with respect to input velocity as seen in the simulation. We observed that the standard deviation ranges are non-linearly increased by the reciprocal operator converting the time-of-travel to velocity. This reciprocal operator resulted in nonlinear spacing output digital codes, which are precise in lower velocity and sparse in higher velocity. For example, the value of time-of-travel 1 [frame/pixel] maps to 1 [pixel/frame] of velocity; 2 [frame/pixel] time-of-travel maps to 0.5 [pixel/frame]; and 4 [frame/pixel] time-of-travel maps to 0.25 [pixel/frame]. In this non-linear relationship, one code difference in the measured time-of-travel affects a different swing in the converted velocity; thus, a measurement error in time-of-travel causes non-linear ranges in different velocities.



(a) Measured linearity curve from on-chip 1-D optic flow core



(b) Measured multi-velocity curve from two moving bars

Figure 5-16. 1-D optic flow test plots with the on-chip digital core

The multi-velocity test was performed by applying two horizontally moving bars to opposite directions from the center as shown in Figure 5-16 (b). The velocity of the bars is changed in three different segments: 0.05, 0.1, and 0.2 [pixel/frame], respectively. The goal of the test is to verify that the measured 1-D optic flow properly reflected both magnitude and direction of motion. The direction of motion is encoded in the sign of measured optic flow. The negative sign indicates the motion of moving left; the positive sign implies the motion of moving to the right direction. The plot in Figure 5-16 (b) is the accumulated 1-D optic flow for the period while the input bars started to move from the center until reaching to both sides. The x-axis of the plot is the spatial index from 0 to 63; each of the numbers indicates the left most and right most pixel. Thus, the measured plot shows three different plateaus per each direction, which correspond to the velocity of 0.05, 0.1, and 0.2 [pixel/frame]. The plot is also symmetric to the origin because the directions of the patterns were opposite.

5.6.2.2 2-D optic flow test results

Figure 5-17 shows the 2-D optic flow test results constructed from the extracted 1-D features generated from the fabricated chip. The test frame rate was 15fps, and translating, diagonally moving, and rotating input patterns were applied. In addition to the pattern tests, we tested with a moving miniature train in order to emulate the MAV environment.

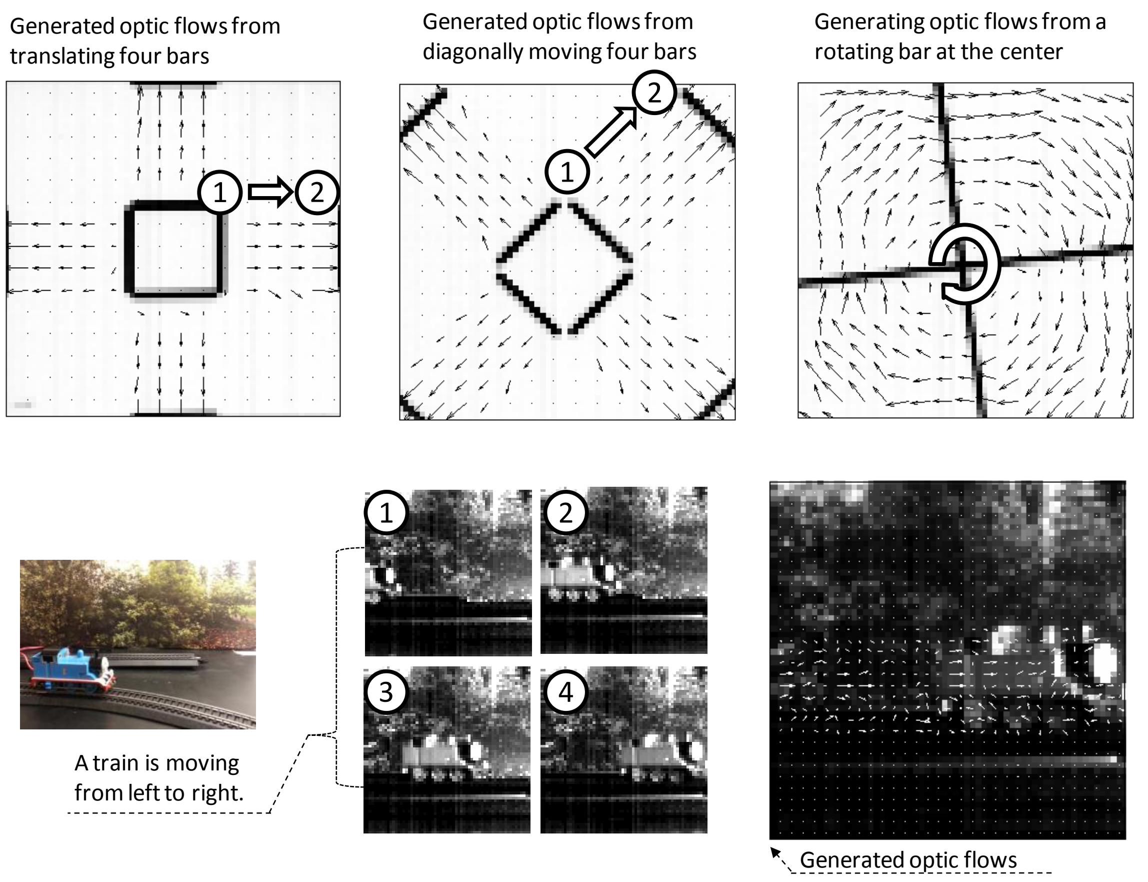

Generated optic flows from translating four bars
Generated optic flows from diagonally moving four bars
Generating optic flows from a rotating bar at the center
1
2
1
2
A train is moving from left to right.
1
2
3
4
Generated optic flows

Figure 5-17. 2-D optic flow test result constructed from 1-b feature information generated by the fabricated chip

5.6.3 Characteristics summary

The performance of the sensor is summarized in Table 5-3. We achieved a 2.38 nW/pixel to estimate 8-b 1-D optic flows. Performance comparison with previously implemented optic flow sensors is presented in Table 5-4.

Table 5-3: Chip characteristics

Process	0.18 µm 1P4M CMOS
Core size	3.26 x 3.26 mm^2
Pixel array	64 x 64
Pixel size	28.8 x 28.8 µm^2
Fill factor	18.32 %
Maximum optic flow	1.96 rad/sec @ 120 fps, FOV 60º
Power (Pixel, 3.3V @ 30 fps)	6.79 nW
Power (ADC analog, 1.8 V, @ 30 fps)	12.96 nW
Power (Digital, 0.9 V, @ 30 fps)	24.03 nW
Power (Digital, 1.8 V, @ 30 fps)	108.72 nW
Total power (@ 30fps)	152.50 nW
Power FOM [pW/pixel·frame]	79.4

Table 5-4: Performance comparison with previous works

	1999 [59]	2001 [86]	2005 [87]	2009 [88]	2011 [89]	**This work**
Technology	1.2 µm CMOS	2 µm Nwell	0.8 µm BiCMOS	-	0.5 µm CMOS	**0.18 µm CMOS**
Array size	6×24	7×7	30×30	40 circular	19×1	**64×64**
Optic flow	1D analog	2D averaging analog	2D analog	1D analog	1D WFI analog	**1D digital**
Pixel size [µm^2]	61×199	70×90	124×124	-	112x257.3	**28.8×28.8**
Fill factor [%]	-	40	-	-	11	**18.32**
Total power [µW]	-	-	-	1000	42.6	**0.152**
Power/pixel [µW]	0.025	350	52	25	2.24	**0.00238 (@30fps)**

5.7 Summary

A bio-inspired A/D mixed-mode optic flow sensor that captures a normal image and also estimates optic flows for vision-based autonomous navigation in micro-air-vehicles is implemented and fabricated. The sensor integrates the 1D time-stamp based optic flow estimation core, which is developed for efficient implementation of bio-inspired time-of-travel measurement in the mixed-mode circuits. The sensor has successfully demonstrated the algorithm by generating 1D optic flows with the maximum frame rate of 120 fps. The feasibility to extend the algorithm for 2D optic flow estimation is also verified using the 1-b feature information acquired from the fabricated sensor.

CHAPTER 6
BIO-INSPIRED TIME-STAMP-BASED 2D OPTIC FLOW SENSOR FOR ARTIFICIAL COMPOUND EYES

This chapter introduces the bio-inspired low-power optic flow sensor, which consists of a 2D time-stamp-based optic flow core and peripheral circuits to accomplish the artificial compound eyes on our 3D semi-hemispherical module platform shown in Figure 6-1.

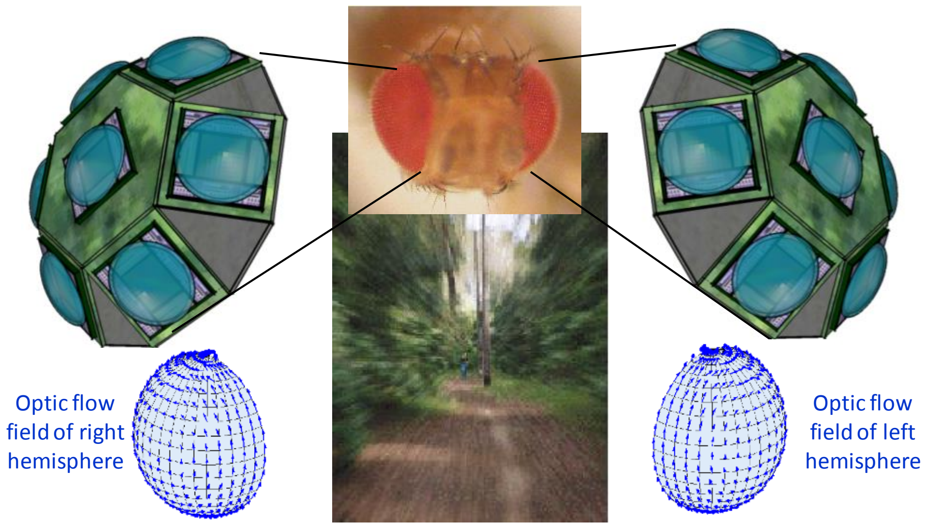

Optic flow field of right hemisphere
Optic flow field of left hemisphere

Figure 6-1: Artificial compound eyes on the 3-D semi-hemispherical module platform

In this work, we have devised and implemented the 2D time-stamp-based optic flow algorithm, which is modified from the conventional EMD algorithm to give an optimum partitioning of hardware blocks in analog and digital domains as well as assign adequate allocation of pixel-level, column-parallel, and chip-level processing. Temporal filtering, which may require huge hardware resources if implemented in digital domain, remains in a pixel-level analog processing unit. Feature detection is implemented using digital circuits in the column-parallel. The embedded digital core decodes the 2D time-stamp information into velocity in chip-level processing. Spatial resolution scaling or down-sampling is optionally supported if users need to reduce the data transmission rate by sacrificing spatial resolution. The estimated 16-b optic flow data are compressed and transmitted to the host through a 4-wired Serial Peripheral Interface (SPI) bus. In addition, to transmit the optic flow data, the embedded 12 matched filters perform the wide-field integration of the measured optic flow to provide the direct input of the bio-inspired navigation systems.

6.1 3D semi-hemispherical module platform for artificial compound eyes

Our 3D semi-hemispherical module platform provides a practical solution for multi-directional optic flow sensing using existing technologies rather than relying on complicated and customized technologies to precisely mimic an insect's compound eye configuration. The conceptual diagram of the 3D module platform is illustrated in Figure 6-2. The multi-directional optic flow sensing is accomplished by 2D optic flow sensors located on top of a flexible PCB, which is bent to any desired inter-sensor angles by origami packaging technique. This sensing scheme is distinguished from that of actual compound eyes.

Several variations of 3D shapes are also possible by folding a base flexible PCB differently. One variation is a uni-axial sensing module as shown in Figure 6-3. In this module, a base flexible PCB is designed as a long band and multiple 2D optic flow sensors are sequentially mounted with equal space between. Then, by bending and connecting both sides, a ring-shaped uni-axial module is formed. This configuration provides 180^{o} wide FoV on a plane so that a 3 degree of freedom (DoF) control is possible for a MAV's autonomous navigation [25], [26].

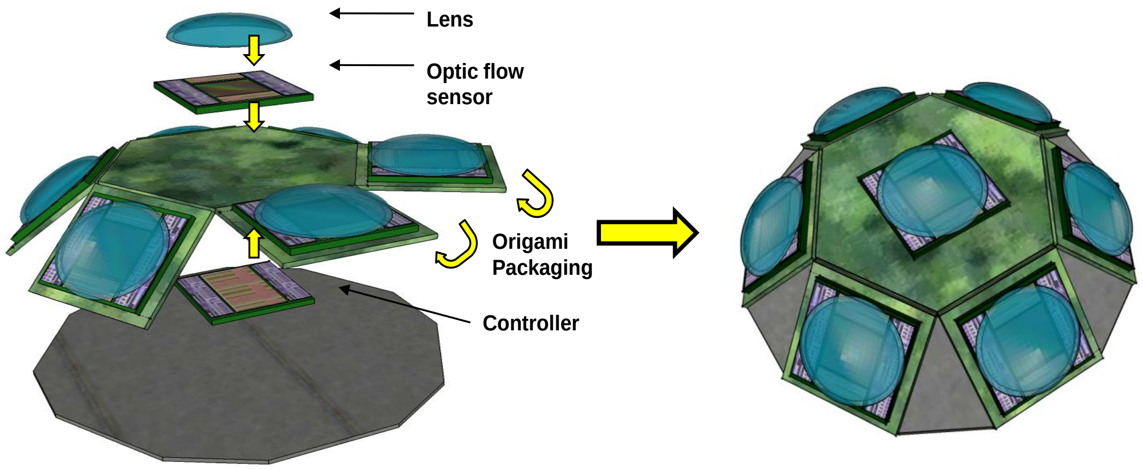


Figure 6-2: The conceptual diagram of 3D semi-hemispherical module platform

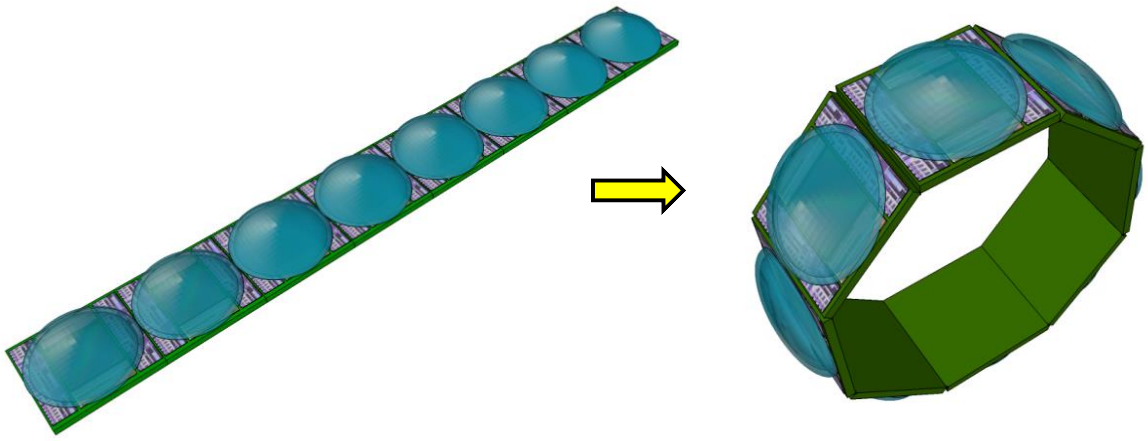


Figure 6-3: The uni-axial sensing module of the 3D semi-hemispherical module platform

A variety of 3D shaped modules can be built depending on a user's different purpose; however, the requirements of 2D optic flow sensors on the modules are straightforward.

Low power consumption and a small form factor are the highest concerns from a MAV's limited power source and size. In addition, a simple data-line connection between the sensors and secure data bandwidth are issued due to its multiple sensors on a module platform. Those issues are discussed in the following sections.

6.2 System requirements of 2D optic flow sensors on the 3D module platform

The 3D module platform that mounts multiple sensors on a single module requires two major design constraints: minimal electrical connection and on-demand data transmission. The minimal connection between the sensors on the module enables assembling the module in a small form factor because the signal routing area on a PCB is minimized; on-demand data transmission allows the host to acquire surrounding optic flow data on time without any loss from each sensor. To resolve both issues, we researched available bus protocols in a MAV system.

Table 6-1: Serial bus protocols used in MAV systems

	Connections [wire]	Bandwidth [kB/s]	Supported sensors @100fps, full resolution (64x64) [EA]
I2C	2	40	0.048
UART	2	92	0.11
SPI	4	3,500	4.3

I2C, UART, and SPI are widely used serial bus protocols in the MAV system as summarized in Table 1-1. I2C and UART supports a 2-wired interface; SPI supports a 4-wired interface. Though I^{2}C and UART provide the minimum electrical connectivity (two wires), data bandwidth is limited only up to 40kB/s for I^{2}C and 92kB/s for UART. These bandwidth specifications are not enough to transmit 64x64 full resolution of 2D optic flow data (16-b per pixel and 8-b for x and y components), especially considering multiple sensors are transmitting data concurrently. However, 3MB/s SPI supports up to 4 sensors to transmit full resolution data at 100fps. Although this capability still does not fully satisfy our requirement using 7 sensors for the 3D semi-hemispherical form and 8 sensors for the uni-axial form, SPI is chosen because it is the only serialize data bus protocol that can handle full resolution data. Therefore, further efforts on reducing data bandwidth at the sensor side should be elaborated. A down-sampler, a data compressor, and wide-field integration blocks are designed to contribute to the data reduction.

A remaining issue is how the host acquires optic flow data from multiple sensors in the module. To resolve this problem, sensors are operating as slaves on the SPI bus as illustrated in Figure 6-4. The data transmission from one sensor is invoked when the host activates the sensor and requests optic flow data for a predetermined period. Therefore, the data transmission is accomplished in such a time division multiplexing way from multiple sensors. In order to support this data transmission operation, each sensor must contain at least the amount of one frame of recently generated optic flow data so that the sensor can immediately send data according to the host's request.

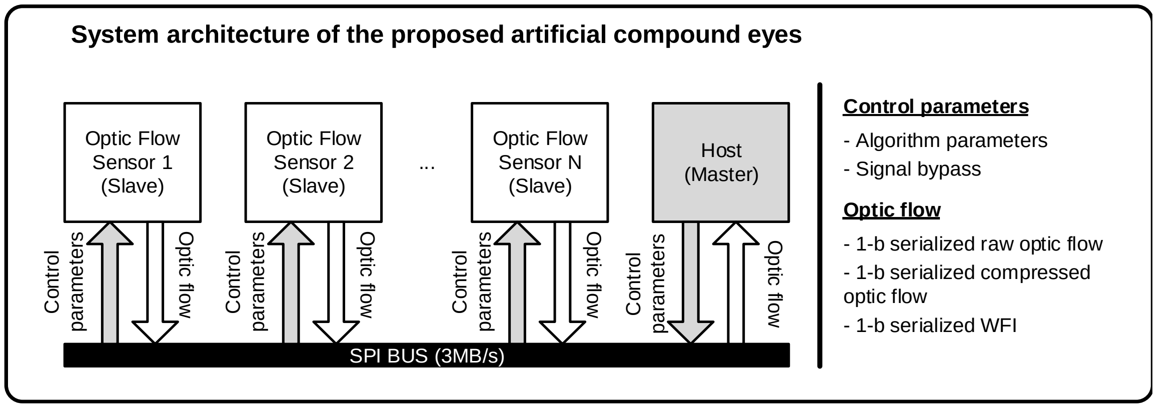

System architecture of the proposed artificial compound eyes
Optic Flow Sensor 1 (Slave)
Optic Flow Sensor 2 (Slave)
...
Optic Flow Sensor N (Slave)
Host (Master)
Control parameters
Optic flow
SPI BUS (3MB/s)
Control parameters
- Algorithm parameters
- Signal bypass
Optic flow
- 1-b serialized raw optic flow
- 1-b serialized compressed optic flow
- 1-b serialized WFI

Figure 6-4: System architecture of the artificial compound eye module

6.3 Implemented bio-inspired 2D time-stamp-based optic flow algorithm

Bio-inspired optic flow estimation algorithms and sensors have been widely investigated for MAV autonomous navigation. Most previous works implemented an insect's motion estimation in aVLSI circuits. We adopted this bio-inspired algorithm for low-power calculation of optic flow estimation utilizing the time-of-travel estimation of a moving feature. However, we implemented the algorithm in analog/digital mixed-mode circuits instead of pure analog neuromorphic circuits. This mixed-mode circuit approach can address the drawbacks in pure analog signal processing and estimate accurate optic flows with high noise immunity from process and temperature variations.

In order to efficiently implement the bio-inspired time-of-travel measurement in mixed-mode circuits, we have introduced the time-stamp imaging. We defined the time-stamp information as the time of the most recent event for a moving feature's arriving into a pixel. Therefore, the time-stamp information is stored in a two dimensional array in our sensor; every pixel keeps updating the information whenever a moving feature arrival is detected at the location. The basic idea of the proposed imaging is illustrated in Figure 6-5.

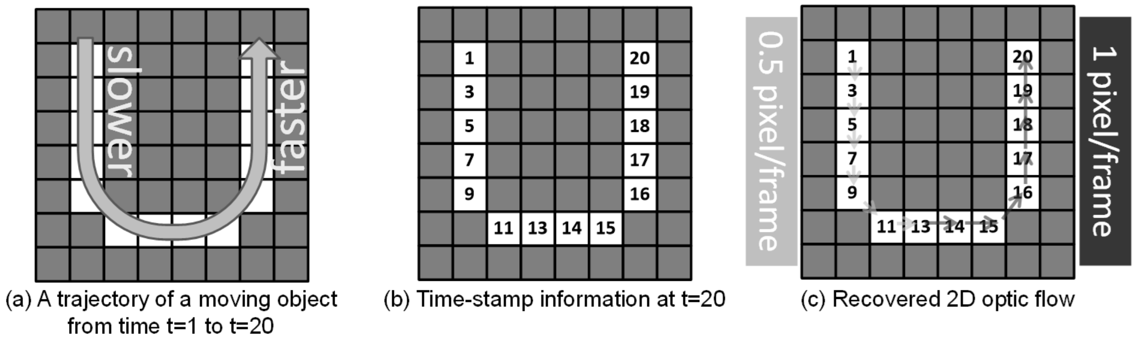

slower
faster
1
3
5
7
9
11
13
14
15
16
17
18
19
20
0.5 pixel/frame
1 pixel/frame
(a) A trajectory of a moving object from time t=1 to t=20
(b) Time-stamp information at t=20
(c) Recovered 2D optic flow

Figure 6-5: Concept diagram of 2D time-stamp information

In the example, an object is moving on the sensor plane by following the trajectory from the time t=1 to t=20. The object moves with the speed of 0.5pixel/frame for the first half of the trajectory, and it speeds up twice (1pixel/frame) for the remaining movement. As a result, at time t=20, the 2D time-stamp array records the object's arrival time as specified in Figure 6-5 (b). Once the time-stamp array is updated with the latest information, the 2D optic flow is decoded from the information at every pixel. The direction of motion is estimated by finding the direction of the increasing time-stamp information in neighboring pixels; it starts from the pixel of lower time-stamp value and ends at the pixel of the higher one. The speed of the flow is inversely proportional to the time-of-travel of the direction; the time-of-travel can be simply calculated by subtracting the recorded two time-stamp values of the corresponding locations. The example in Figure 6-5 (c) shows the recovered 2D optic flow. We have developed the 2D time-stamp-based optic flow algorithm that is able to automatically update the time-stamp array and recover the optic flow by recovering the aforementioned relationship.

The moving object arrival detection in each pixel enables the algorithm to automatically update the time-stamp information. We mimic the moving feature detection scheme inside an insect's visual signal pathway that utilizes the temporal contrast edge as an object's arrival or departure at each photo-recepting unit called an ommatidium. The detection scheme in an insect's eyes is illustrated in Figure 6-6. Two simplified cases in the figure are: (a) a brighter object than background arrived at time t=1 and left the location at t=2; (b) a darker object than the background arrived at t=1 and left at t=2. The temporal contrast is measured by a temporal high pass filter (HPF). The positive polarity of the measured temporal is named on-edge polarity. In the same manner, the negative temporal contrast is named off-edge polarity [73]. This on-edge (or also off-edge) occurs when a moving object arrived to or departed from the pixel. The on-edge (off-edge) is caused by the arrival (departure) if relative brightness of the object is brighter than the background; the on-edge (off-edge) is caused by the departure (arrival) if relative brightness of the object is darker than the background. As an example in Figure 6-6 (a), the on-edge contrast change represents the arrival; the off-edge indicates the departure for a brighter object. In Figure 6-6 (b) due to the opposite brightness relationship of the object and background, the on-edge (off-edge) represents the departure (arrival). No matter what the on-edge or off-edge represent either arrival or departure, we can measure the time-of-travel time passing on the pixel by considering only one polarity. For example, if considering only on-edge polarity to measure the traveling time of a moving object that passes two neighboring pixels as shown in Figure 6-6 (c), the measurement is actually the results of the arrival time difference between the pixels for a bright moving object, which corresponds to the time difference in Figure 6-6 (d). In an actual insect's visual system, the on-edge and off-edge polarities are processed in parallel in two different pathways for the independent time-of-travel measurements [73]. However, our algorithm processes only one polarity to save hardware resources, which are mainly reduced by allocating one 64x64x8-b 2D time-stamp array, not two arrays for both on-edge and off-edge information. This algorithm reduction is possible due to the property that is explained from the examples of Figure 6-6 (a) and (b): the on-edge (off-edge) of the contrast change always traces the off-edge (on-edge) contrast change if an object is passing on the pixel. This property implies even though only a single pathway is implemented, there is no chance to miss any motion information passing on the pixel.

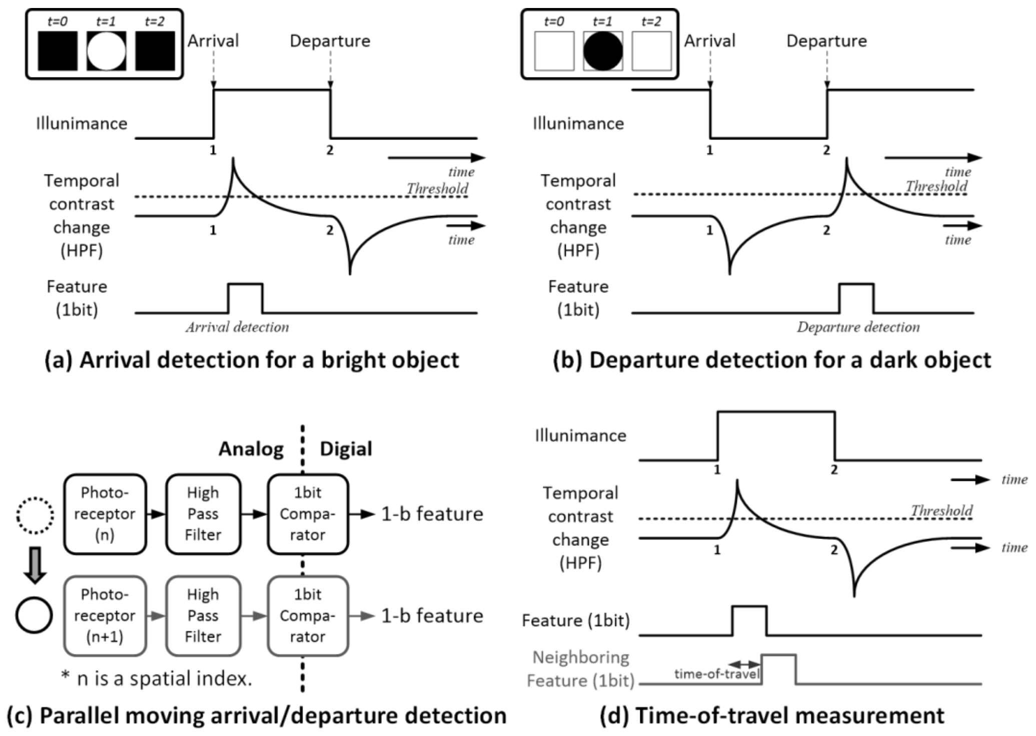


Figure 6-6: A bio-inspired moving object arrival/departure detection

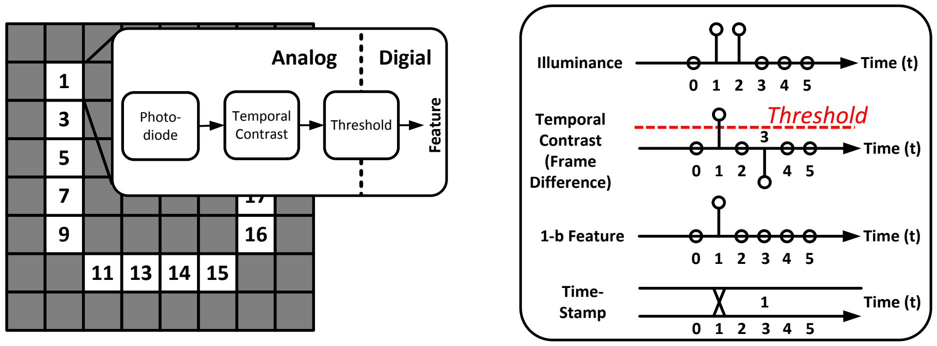
In the proposed algorithm, we adopted a simplified version of a discrete-time high-pass filter (DT-HPF) implemented using A/D mixed-mode circuits to perform the function describe in eq. (1). The algorithm only requires 1-b digital feature information at the A/D boundary defined as eq. (2). This information is required to stamp the corresponding time. Thus, the algorithm requires only 1-b ADCs which can be implemented using 1-b comparators with a feature threshold (*Th_{feature}*).

$$HPF_t(n) = I_t(n) - I_{t-1}(n) \tag{1}$$

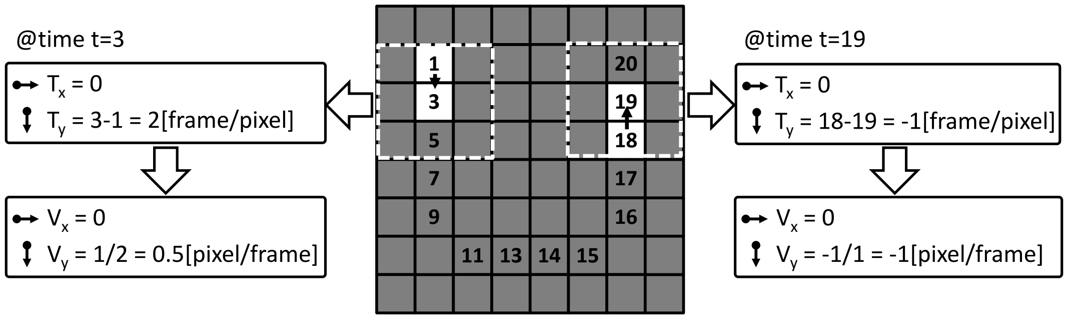
$$Feature(n) = \begin{cases} 1, & if \quad HPF(n) \geq Th_{feature} \\ 0, & if \quad HPF(n) < Th_{feature} \end{cases} \tag{2}$$

Where $I_t(n)$ and $I_{t-1}(n)$ are the measured illuminances at the current time t and previous time t-1 in the spatial location of n; and $Th_{feature}$ is the feature threshold which is an adjustable reference value to determine the moving feature.

Figure 6-7 summarizes the implemented time-stamp-based 2D optic flow algorithm. The pixel-level moving feature detection in Figure 6-7 (a) consists of the photodiode, DT-HPF for temporal contrast measurement, and thresholding comparator. Because the implemented algorithm operates in the discrete-time domain, the signals in the example timing diagram are sampled. The sampling frequency corresponds to the frame rate. In the timing diagram, on-edge contrast change is quantized using an adjustable threshold; as a result, 1-b moving feature information is detected. This feature information updates the time-stamp information of the pixel by latching the global frame counter value. The time-of-travel measurement and velocity conversion using the updated 2D time-stamp array is described in Figure 6-7 (b). As shown in the figure, the algorithm generates the 2D optic flow in every pixel by referring to 8 neighboring time-stamp pieces of information which are a 3x3 masking operation. The calculation is triggered only when the center pixel detects the moving feature. If a feature is detected, the updated time-stamp information is the same as the current global counter value at the location (pixel). For example, at time t=3 only at the pixel in the left dotted box, the calculation is performed; in the same manner, at time t=19 only at the location of the right dotted box in the figure is the optic flow estimated. The estimation is sequentially performed by measuring time-of-travel values of x and y directions, (T_x, T_y). This measurement is simply accomplished by subtracting time-stamp information of neighboring pixels from that of the center. The optic flow (V_x, V_y) is inversely proportional to the measured time-of-travel (T_x, T_y) which is implemented using look up tables. In the given examples in the dotted boxes, are the direction and velocity of 2D vector (V_x, V_y).



(a) The implemented moving feature detection algorithm



@time t=3
$T_x = 0$
$T_y = 3-1 = 2$[frame/pixel]
$V_x = 0$
$V_y = 1/2 = 0.5$[pixel/frame]
@time t=19
$T_x = 0$
$T_y = 18-19 = -1$[frame/pixel]
$V_x = 0$
$V_y = -1/1 = -1$[pixel/frame]
1
3
5
7
9
11
13
14
15
16
17
18
19
20

(b) A simplified 3x3 masking operation for optic flow measurement

Figure 6-7: Implemented time-stamp-based 2D optic flow estimation algorithm

6.4 Sensor architecture

The prototype sensor consists of the 2D pixel array which integrates the frame difference measurement circuit as a simplified DT HPF for the temporal contrast measurement in each pixel; the moving feature arrival/departure detection comparator in the column circuits; the chip-level digital time-stamp-based 2D optic flow computation core; and the digital peripheral circuits to provide a modular expandability for the multiple sensors in a module platform as shown in Figure 6-8.

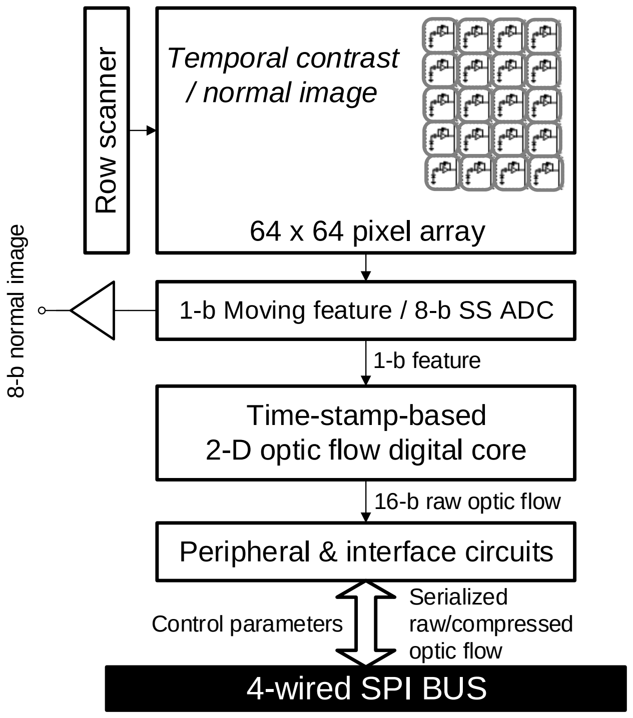

Row scanner
Temporal contrast / normal image
64 x 64 pixel array
8-b normal image
1-b Moving feature / 8-b SS ADC
1-b feature
Time-stamp-based 2-D optic flow digital core
16-b raw optic flow
Peripheral & interface circuits
Control parameters
Serialized raw/compressed optic flow
4-wired SPI BUS

Figure 6-8: Prototype time-stamp-based 2D optic flow sensor architecture

This architecture is similar to that of normal CMOS imagers integrating column-parallel single slope ADCs (SS ADCs) in order to support both normal image generation and optic flow estimation modes. The major difference from the normal imagers is that the pixel integrates DT HPF to measure the temporal contrast using switched capacitor circuits in the optic flow mode. In the optic flow mode, the comparator in the SS ADC is used for a thresholding circuit. The threshold value ($Th_{feature}$) is an adjustable voltage and is to be connected to another input of the comparator. Therefore, the 1-b moving feature arrival or departure signal is generated if the measured positive temporal contrast (on-edge) is bigger than $Th_{feature}$. This location is the boundary between the analog and digital domains, and only 1-b quantization is required for A/D conversion. The 1-b A/D conversion can provide a robust signal interface between the domains due to a large margin from the noise floor to the measured temporal contrast signal to perform the 1-b quantization. These column feature detection circuits are implemented to use 0.9V supply for low power consumption.

After performing the 1-b conversion in the column circuits, the digital 1-b moving feature signal flows into the integrated digital 2D time-stamp-based optic flow estimation core in the raster scan order. In the digital core, the delivered 1-b feature updates the 8-b time-stamp information of the corresponding pixel if a moving feature is detected. The updated time-stamp information is aligned for a 3x3 masking operation in the two-line buffer which stores the upper two rows of updated time-stamp information. Then, using the updated 3x3 time-stamp information, the core measures four directional time-of-travel values with respect to horizontal, vertical, and two diagonal axes. Thus, the implemented hardware considers two more directional motion pieces of information than the aforementioned simple algorithm for accurate estimation. The time-of-travel measurement is implemented using digital arithmetic subractors. The measured time-of-travel values in four directions are converted to velocities from four look-up tables. Finally, the four converted velocities are projected to x and y axes to find 16-b 2D optic flow (V_x, V_y). Peripheral circuits further process the estimated 16-b raw optic flow. The main reason is to reduce data bandwidth to satisfy the system requirement of the multiple sensors since they share a single serialize bus. Thus, we developed and implemented a data compression algorithm that is customized for optic flow data. The algorithm can be selected to operate as a lossless or lossy compressor based on parameter settings. The peripheral circuit is able to further reduce data bandwidth using spatial down-sampling raw optic flow down to 32x32, 16x16, and 8x8. In addition to provide optic flow, the peripheral circuit supports the processed wide field integration (WFI) output which is the front-end information of a bio-inspired autonomous navigation algorithm for MAVs [25]–[27].

6.5 Circuit implementation and proposed sensor

6.5.1 Pixel architecture

The pixel architecture and layout for the optic flow and normal image modes are shown in Figure 6-9. In the optic flow mode, the pixel measures frame difference. The pixel includes a sampling capacitor (C_1) and the capacitor (C_2) for setting a programmable gain amplifier (PGA) gain. Frame difference is monitored by sampling the pixel voltage ($V_p(t_1)$) of the previous frame from C_1 first, and then applying the current pixel voltage ($V_p(t_2)$) to the PGA. In both normal image and frame difference modes, the PGA supports x1, x2, x4, and x8 gains by connecting more unit capacitors of C_2 in parallel. The supply voltage for photodiodes and source followers is 3.3V. The PGA operates at 1.8V and is only enabled during the signal transferring period to reduce static power consumption. The layout of the pixel is shown in Figure 6-9 (b). The pixel size is 28.8x28.8μm^2. The laid out photodiode achieved 18% of fill factor. The capacitor array consists of metal-insulator-metal (MIM) capacitors occupying about half of the pixel area. The remaining area is for the PGA and switches and the source follower transistors. The signal lines are vertically passing through the center between the photodiode and capacitor array and also horizontally passing at the top and bottom of the pixel.

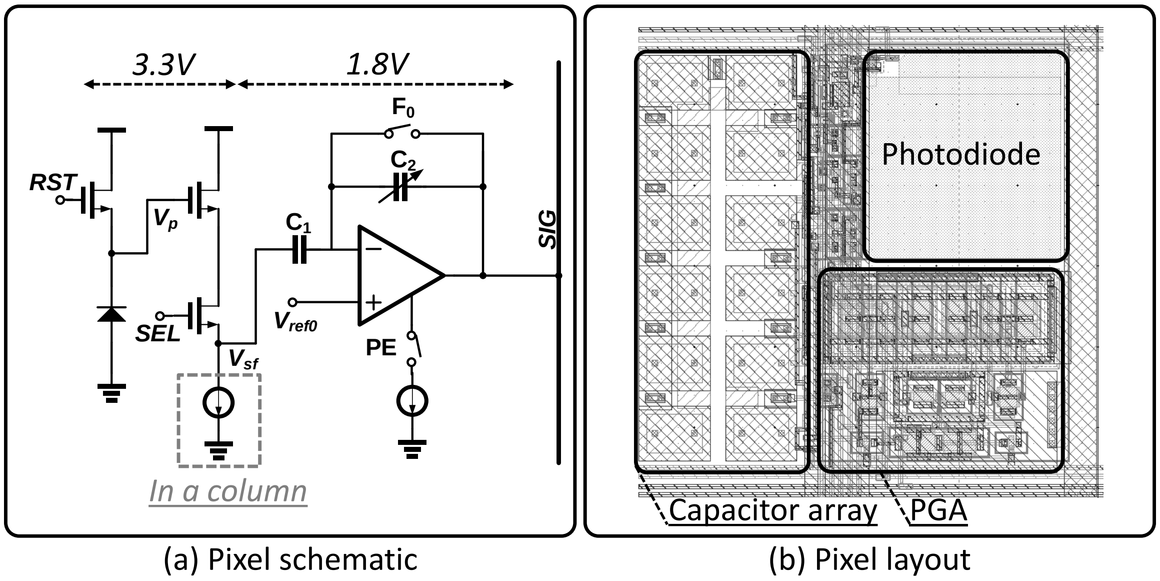


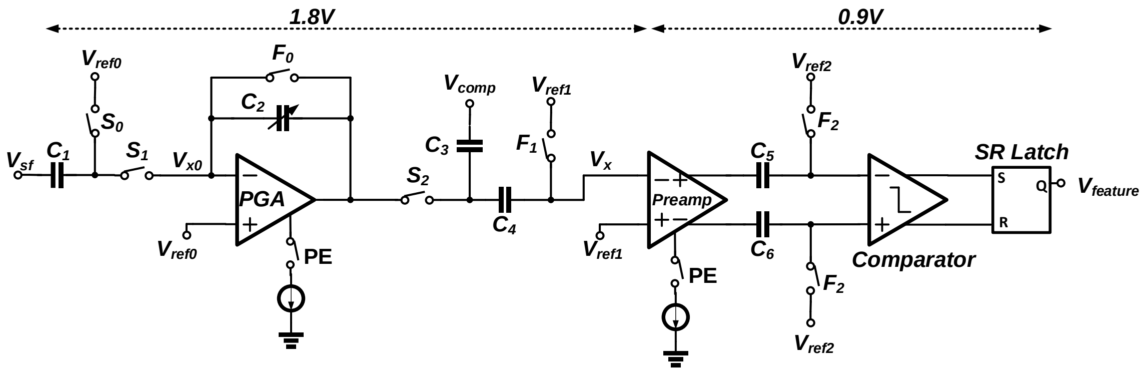
Figure 6-9: Pixel architecture: schematic and layout

6.5.2 Column-parallel feature detection circuits

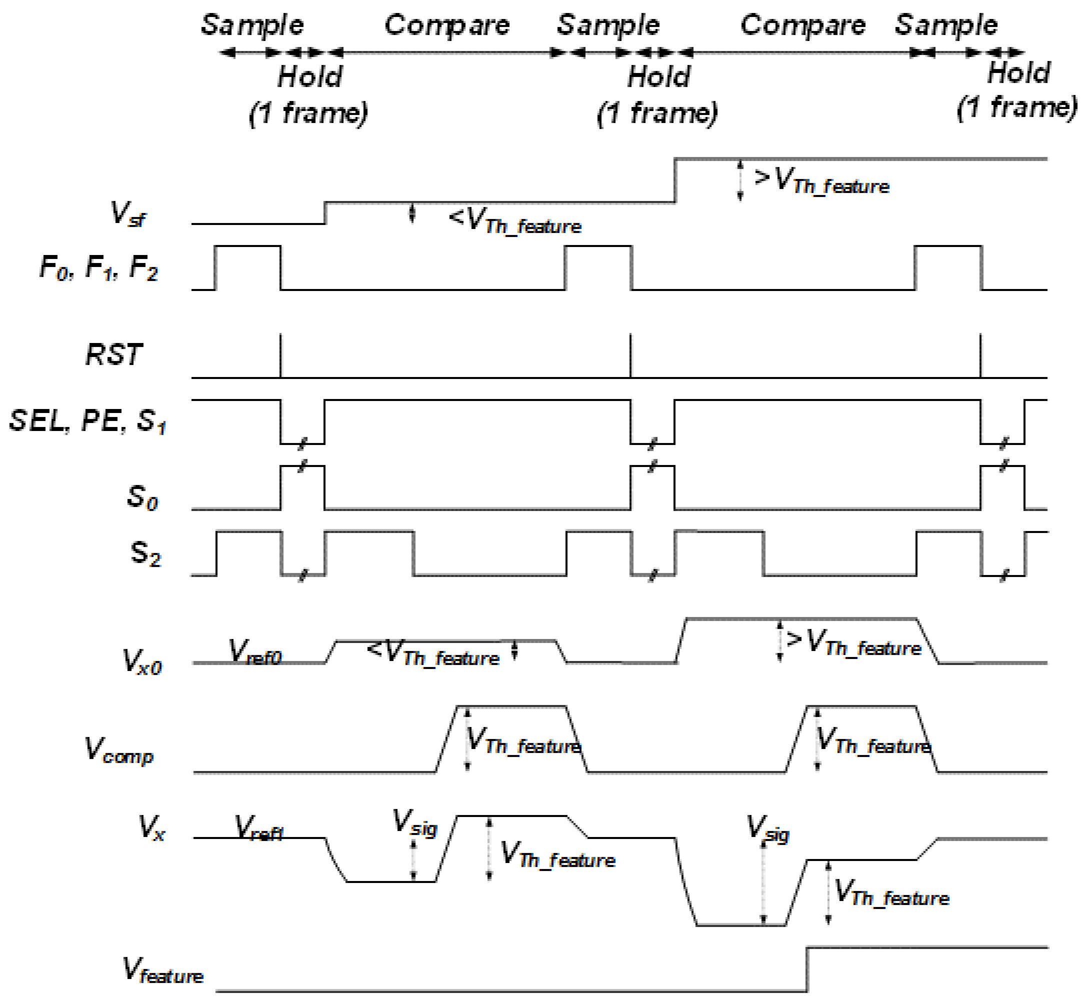
Figure 6-10 shows the moving feature arrival/departure detection circuits which are implemented in column-parallel combining with the in-pixel PGA. The schematic of the full feature detection signal path is shown in Figure 6-10 (a). The boundary of in-pixel PGA and the remaining column-level circuits is at the S_2 switch. The implemented circuits compare the measured frame difference to the adjustable feature threshold, $V_{Th\_feature}$; then, generate 1-b feature information if the measured value is higher than the threshold. We designed the circuits to detect only the off-edges of the temporal contrast; as a result, the circuit can use smaller voltage headroom comparably to support both edges. Thus, the power supply is reduced down to 1.8V for low power consumption. In addition, by only using the off-edges we also reduced the digital hardware resource as mentioned in the algorithm description, in that only one 2D time-stamp buffer (64x64x8b) is required. In Figure 6-10 (b), an example timing diagram is illustrated to explain the feature detection operation. In the example, the gain of the PGA is set to 1 to make the example simple. The operation is composed of three phases: sample, hold, and compare. The example shows three different levels of the source follower output voltage (V_{sf}) in time to demonstrate two frame difference measurements. The source follower voltage level in the example is described as flat over one frame for simplification; however, the actual voltage is reset to 3.3V when the reset signal (RST) is asserted and decreases to the steady level over integration. The hold phase is expressed for a short duration in the timing diagram, but the actual time lasts as long as almost one frame. The example starts from the sample phase of the V_{sf} by closing all F_0, F_1, and F_2 switches. At this phase, the SEL, PE, and S1, S2 switches are also turned on in order for V_{sf} to be stored at C_1. The next hold phase is for the pixel to hold value for the integration time by disconnecting S_2 from column circuit so that another row can utilize the column detector circuits. For this hold phase, the S_0 and S_1 are controlled to keep one terminal of C_1 to the reference voltage of the PGA. After one frame of the integration time is passed, the operation moves to the first compare phase. The V_{sf} is changed after the integration time, but the difference from the previous frame is not higher than the threshold voltage ($V_{Th\_feature}$). In the compare phase, the V_{sf} of the current frame is applied by closing the S_1 and S_2 switches; as a result, the frame difference is reflected by decreasing voltage at the node V_x as shown in the timing diagram. The measured frame difference (V_{sig}) is expressed as the Eq (3).

$$V_{sig} = -A(V_{sf}(t_n) - V_{sf}(t_{n-1})) \tag{3}$$

Where, $V_{sf}(t_n)$ is the source follower voltage of the current frame, $V_{sf}(t_{n-1})$ is the source follower voltage of the previous frame, and A is the PGA gain. In the given timing diagram, A is set to 1. Then, the signal path from the PGA is isolated by opening S_2. The comparison to the $V_{Th\_freature}$ is performed by applying the threshold voltage to the V_{comp} terminal. At this comparison, because the V_{sig} is smaller than $V_{Th\_freature}$, V_x becomes bigger than V_{ref1} of the pre-amplifier. Therefore, S and R inputs of the SR latch are configured to reset by the comparator output. As a result, the output of SR latch ($V_{feature}$) is zero, which means no feature is detected. After the first comparison phase, the current frame voltage is sampled at C_1 in the second sample phase and held for the next integration time in the second hold phase. In the second compare phase, V_{sf} difference (V_{sig}) is bigger than $V_{Th\_freature}$; thus, the V_x finally becomes lower than V_{ref1}, which triggers the SR latch to be set. As a result, the 1-b feature information ($V_{feature}$) is detected.



(a) Feature detection circuit schematic



(b) Feature detection circuit timing diagram example

Figure 6-10: Feature detection circuits: schematic and timing diagram

6.5.3 Digital time-stamp-based optic flow estimation core

The block diagram of the implemented chip-level digital circuits is shown in Figure 6-11. The embedded digital processing circuits mainly consist of two parts: the time-stamp-based optic flow estimation core that generates 16-b 2D raw optic flow from neighboring 3x3 time-stamp information, and the peripheral circuits including SPI interface and post processors, which reduce the data bandwidth (down-sampler and compressor) as well as extract wide field integration (WFI) control parameters for a MAV's autonomous navigation.

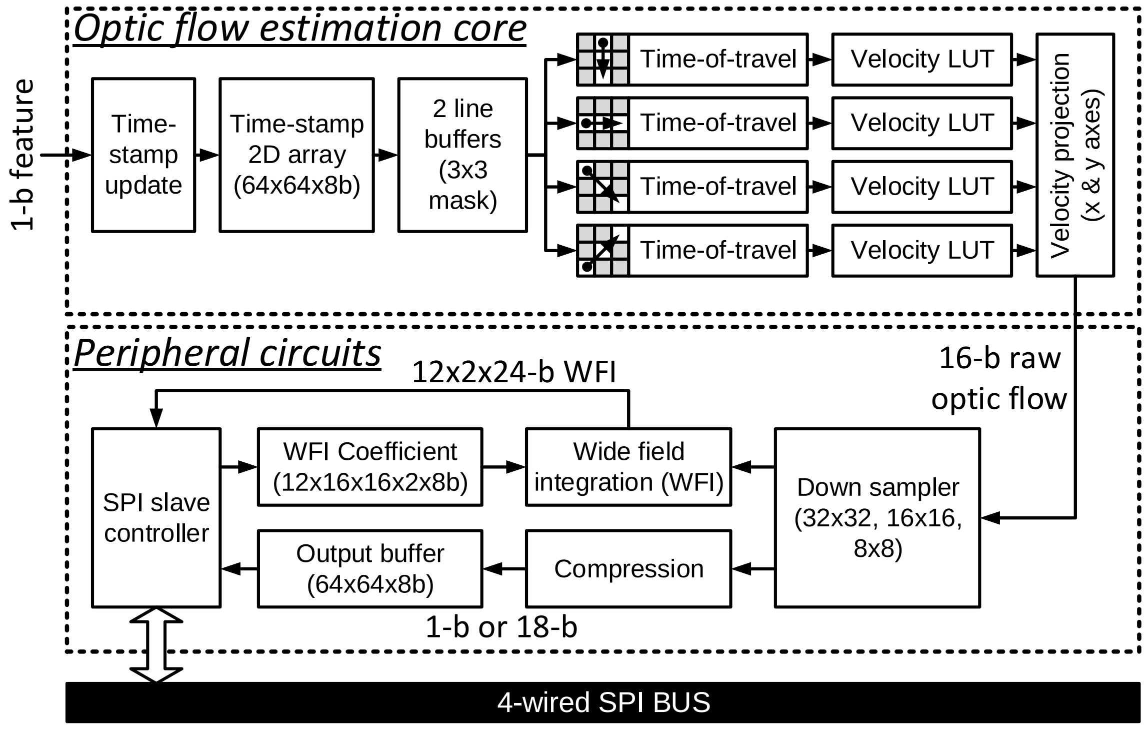

Optic flow estimation core
1-b feature
Time-stamp update
Time-stamp 2D array (64x64x8b)
2 line buffers (3x3 mask)
Time-of-travel
Velocity LUT
Velocity projection (x & y axes)
Peripheral circuits
12x2x24-b WFI
16-b raw optic flow
SPI slave controller
WFI Coefficient (12x16x16x2x8b)
Wide field integration (WFI)
Down sampler (32x32, 16x16, 8x8)
Output buffer (64x64x8b)
Compression
1-b or 18-b
4-wired SPI BUS

Figure 6-11: Digital time-stamp-based optic flow estimation core and peripheral circuits

The time-stamp-based optic flow core manages the 2D time-stamp array. The first block of the diagram, which is the time-stamp update block, updates the 2D array for every frame using the 1-b feature information from the column-level feature detector circuits. Also, the block chronologically re-orders the time-stamp information. The implemented re-alignment algorithm is illustrated in Figure 6-12. Though the example uses a 3-b counter, this method is simply extended for higher bit-width counters such as 8-b or 10-b. Before considering the reordering algorithm, note the zero time-stamp values are reserved for the no-stamp code, which implies no moving feature is detected on the corresponding pixel. Thus, if the latched time-stamp value is zero, further velocity calculation processing must be suspended since no useful time-stamp is recorded at the location. Because the no-stamp code reserves zero, the lowest circulating counter value is one (1), and the maximum value is 2^n-1 (7 for a 3-b counter and 255 for an 8-b counter). As shown in Figure 6-12, the circular counter increases in the clockwise direction. The current counter value is indicated by a solid arrow as on a clock; thus, the current time-stamp counter is 2. This implies that among the recorded time-stamp values in the latches the most recent event is the counter value of 2, and the oldest is 3. This chronological relationship is illustrated by a dotted line. To automatically recover this order, the algorithm subtracts 2^n-1 (7 for this example) only for the time-stamp values that are bigger than the current time-stamp counter (>2). As a result, the chronological order is automatically recovered by expressing the older events rather than the current time stamp counter using negative values as shown in Figure 6-12. The time-stamp update block in Figure 6-11 also performs to clear the recorded time-stamp information by writing the no stamp code (0) if it lasted for one period of the frame counter. The updated time-stamp value is recoded to the 2D time-stamp array and then read out in the raster scan order.

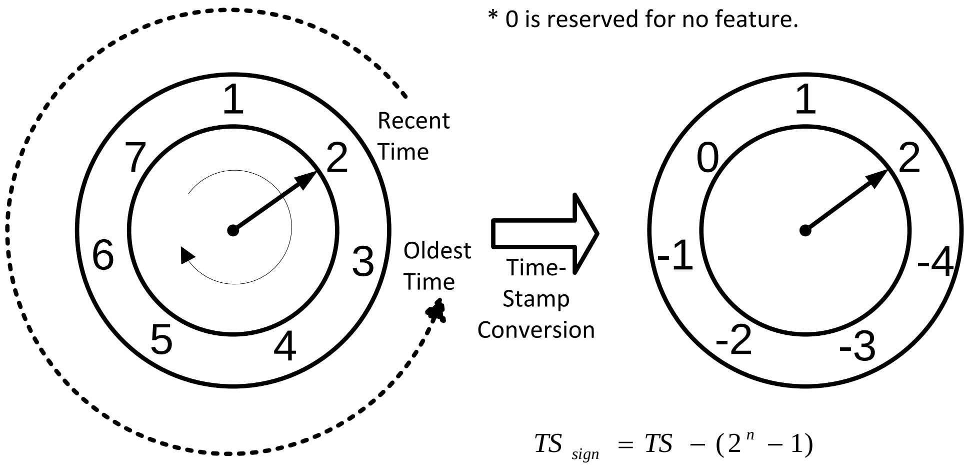


Figure 6-12: The implemented circular time-stamp counter and value reordering

A two-line buffer aligns the updated time-stamp information for the 3x3 masking operation. The 3x3 time-stamp information is required because the implemented core finds the velocity of a moving feature in four directions, which are horizontal, vertical, and two diagonals. In addition, four 1D time-stamp-based optic flow estimation units are implemented to find 1D optic flow for each direction using the equations in Table 6-2. Then, velocity projection block merges the four 1D optic flow values to generate 2D optic flow information by projecting the measured flows onto the x and y axes using Eq. (4).

$$
\begin{aligned}
u &= V_{WE} + (V_{SW\_NE} + V_{NW\_SE})/W_d \\
v &= V_{SN} + (V_{SW\_NE} - V_{NW\_SE})/W_d
\end{aligned}
\tag{4}
$$

Where, (u, v) is the x and y directional 2D optic flow components. V_{WE}, V_{SN}, V_{SW_NE}, and V_{NW_SE} are the estimated velocities for horizontal, vertical, and two diagonal directions. W_d is an adjustable weight factor for the diagonal flows onto the perpendicular axes, $\sqrt{2}$ is used for the measurement. The measured u and v are 8-b each. Thus, total 16-b raw 2D optic flow is generated from the core.

Table 6-2: Equations for 2D time-stamp-based optic flow estimation

Time to move from the center	Time of Travel	Velocity
$dT_N = TS(x, y) - TS(x, y-1)$ $dT_S = TS(x, y) - TS(x, y+1)$ $dT_W = TS(x, y) - TS(x-1, y)$ $dT_E = TS(x, y) - TS(x+1, y)$ $dT_{NE} = TS(x, y) - TS(x+1, y-1)$ $dT_{NW} = TS(x, y) - TS(x-1, y-1)$ $dT_{SE} = TS(x, y) - TS(x+1, y+1)$ $dT_{SW} = TS(x, y) - TS(x-1, y+1)$	$T_{WE} = dT_W - dT_E$ $T_{SN} = dT_S - dT_N$ $T_{SW\_NE} = dT_{SW} - dT_{NE}$ $T_{NW\_SE} = dT_{NW} - dT_{SE}$	$V_{WE} = 1 / T_{WE}$ $V_{SN} = 1 / T_{SN}$ $V_{SW\_NE} = 1 / T_{SW\_NE}$ $V_{NW\_SE} = 1 / T_{NW\_SE}$

- TS(x, y) means the updated time-stamp information at the spatial location of (x, y)
- dT means the time-stamp difference between the center to neighbors.
- N, S, W, and E mean North, South, West, and East directions.
- T means the measured time-of-travel for each 1D axis.
- V means the converted velocity for each 1D direction.

6.5.4 Digital peripheral circuits: SPI controller, compressor, down-sampler, and WFI

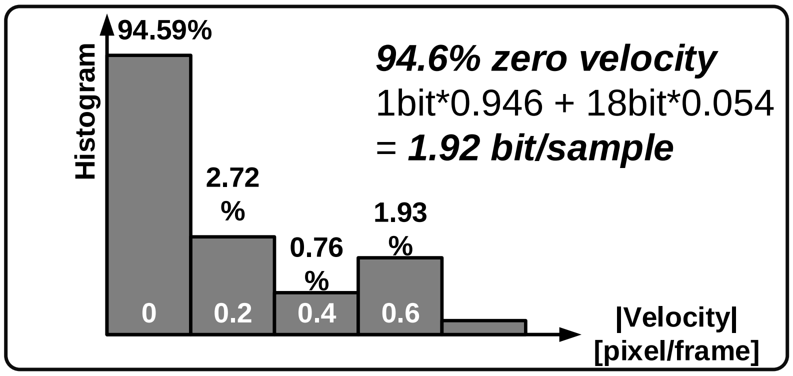
The peripheral circuits were implemented to support our modular-based artificial compound eyes platform. The 4-wired SPI was chosen to secure high data bandwidth among currently available serial bus interfaces utilized in MAVs because the compound eyes platform is mountable with multiple sensors on the system. Though the integrated

3MB/sec SPI is the fastest option, only three 2D optic flow sensors can send data at full resolution, 64x64x16-b at 120fps. Thus, the sensor integrates additional data reduction blocks to meet the system requirements. The optic flow data compression algorithm can support both lossy and lossless compression and is optimized for simple hardware implementation. Using the lossless algorithm, up to 25 sensors are able to transfer the full resolution data on the same 3MB/sec SPI bus. In addition to the compression core, the integrated down-sampler can further reduce the data by spatially resampling the raw flow data down to 32x32, 16x16, and 8x8 with the fixed scaling ratios. Finally, the wide field integration (WFI) block performs 2D matched filtering operation using the down-sampled optic flow and user defined coefficients. The WFI outputs can be directly used as control parameters by the bio-inspired MAVs for autonomous navigation.

6.5.5 Optic flow data compression core

The implemented compression algorithm is described in Figure 6-13. The algorithm utilizes the sparsity of generated optic flow in a scene because scenes in nature are mostly composed of low spatial and temporal frequency components. Specifically, the sparsity is maximized in such bio-inspired algorithms in which both temporal and spatial edges are considered to estimate the optic flow. For example, our algorithm only generates optic flow at the locations where spatial edges are moving. Thus, in a rotating fan scene, the flow is generated only at the edges of the vanes. Based on the observation, we simulated using image sets from a robot simulator to verify that the sparsity is consistent in the moving robot’s vision [85]. We tested with ten thousand scenes which were captured from the robot simulator while the robot was flying the five trajectories. The result in

Figure 6-13 (a) shows optic flow was generated at only about 5.4% of total measurement sites; 94.59% of locations has zero optic flow. Thus, the sparsity property is still valid in a mobile robot situation. This distribution of the histogram was inspired to apply entropy data compression technique for the optic flow data: the shortest code is allocated for zero optic flow data; the longer codes are allocated with respect to the frequency of occurrence. To simplify hardware implementation, we categorized the estimated optic flow data to only two types of the data set: non-flow data and valid optic flow data. Then, 1-b '1' code is reserved for the majority. For the detected optic flow, 2-b header ('01') plus 16-b raw optic flow is sent. Therefore, based on the probability distribution, 16-b/sample raw optic flow is compressed to only 1.92 bit/sample in average without any loss.



(a) Optic flow magnitude histogram in one frame

```
If  (|u|<=Threshold) & (|v|<=Threshold)
{
    Send 1-b '1' code;  % non-flow
}
else
{
    Send 2-b header ('01') + 16-b raw optic flow;
}
```

(b) Compression algorithm pseudo code

Figure 6-13: Implemented 2D optic flow data compression algorithm

We introduced a threshold to determine the non-flow data for preventing negligible small flows which degrade the compression performance. The sources of the small flows can be a continuous weak vibration of MAVs as they constantly work to maintain balance. As shown in Figure 6-13 (b), the criteria of determining non-flow is the absolute optic flow in both the x and y directions which must be less than the threshold. Thus, if the threshold is zero, the algorithm performs lossless compression because all measured flow is considered to be valid; if the threshold is above zero, then the algorithm performs lossy compression which abandons smaller measured flows than the threshold for compensation of a noisy environment.

6.5.6 Down-sampler and WFI

The sensor integrated the digital wide field integration (WFI) function. The WFI preforms the 2D matched filtering operation to extract clues of self-motion information by parsing the measured surrounding optic flow. This mechanism is inspired by the tangential cells in a flying insect's compound eyes, which are directionally selective cells for detecting motions from its surroundings [53]. This behavior is modeled by the 2D matched filter that measures the correlation between the optic flow and the predetermined coefficient sets. The filter operation is an inner product between the surrounding 2D optic flow and the user-defined 2D coefficients. Bio-inspired autonomous navigation control theories utilizing the calculated WFI as front-end information were successfully demonstrated on MAVs and ground robots [25], [26].

The integrated WFI supports up to 12 matched filter coefficient sets that are configured through the SPI. The memory space to store the coefficients is not trivial. If the filter supported up to the full 64x64 spatial resolution, the required memory space would be the amount of storing several frame buffers assuming 8-b per each coefficient and 12 sets. Therefore, the embedded spatial down-sampler reduces the spatial resolution down to 32x32, 16x16, and 8x8. Then, the coefficients can be configured either 16x16 or 8x8 for both the x and y optic flow components. Each coefficient is 8-b; thus, in total 12x16x16x2x8-b memory space is required. We shared the space with the frame buffer SRAM; thus, in the WFI mode, the SRAM works as a coefficients set memory, and in the optic flow mode, the SRAM is used for the frame buffer to provide the data in accordance to the host's requests.

6.6 Experimental results

A prototype chip was fabricated using 0.18μm 1P4M process and has been fully characterized. A chip micrograph is shown in Figure 6-15. The total chip size 3.09mm×4.18mm including I/O pads. The chip contains a 64×64 pixel array, column-level feature extraction circuits, and 8-b single-slope ADCs for the normal image mode. The ramp generator is not implemented in the sensor, and an external DAC is used to generate the ramp signal for SS ADCs and feature threshold. The digital processing core is designed using standard cells. Two SRAMs are integrated for the 2D time-stamp array and output frame buffer/WFI coefficient's storage. The test system contains a printed circuit board (PCB) to integrate the sensor, ramp generator, and an FPGA to generate input control signals. The measurement environment is shown in Figure 6-14, whose signal flow is described in Figure 5-15. Input motion is generated in front of the sensor by moving objects or on an LCD screen. The output data from the sensor are transmitted to PCs through DAQ and LabView. For a test purpose, the 8-b normal image output path is implemented using bi-directional pads so that the digital core is able to directly get input data from the on-board FPGA. As a result, characterizing the digital core separately from the analog parts is possible if required.

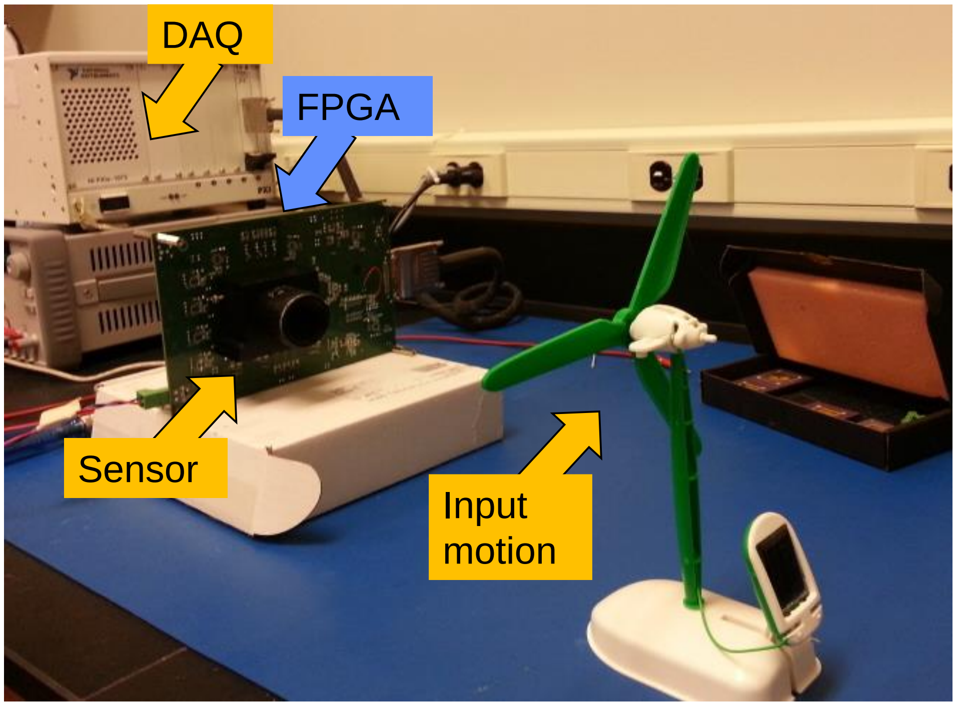


Figure 6-14: Measurement setup

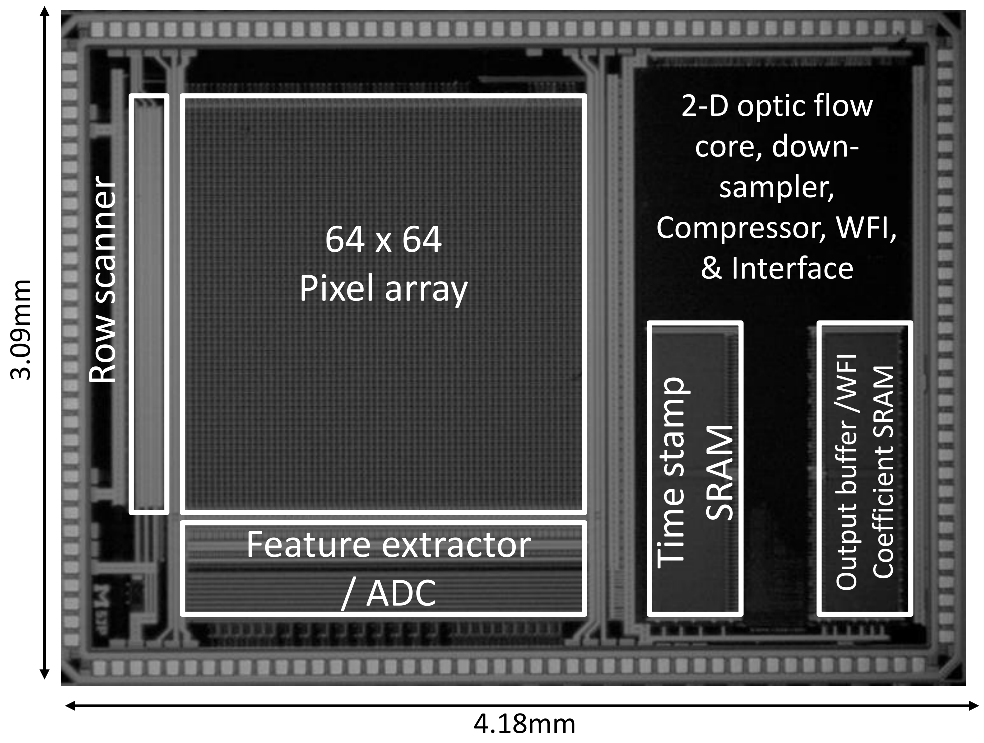
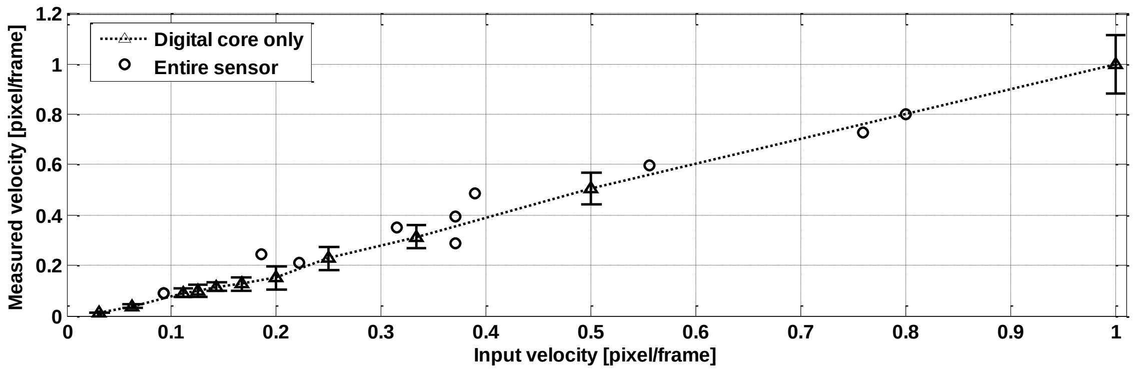

3.09mm
Row scanner
64 x 64 Pixel array
Feature extractor / ADC
2-D optic flow core, down-sampler, Compressor, WFI, & Interface
Time stamp SRAM
Output buffer /WFI Coefficient SRAM
4.18mm

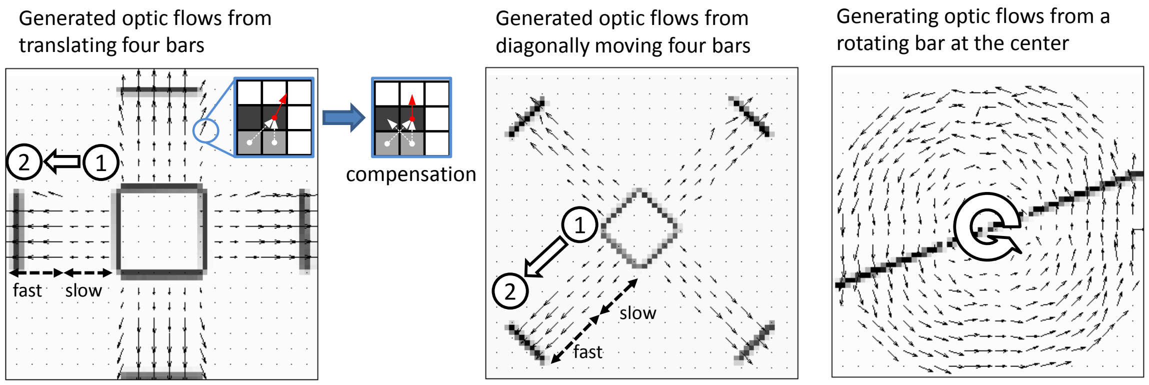
Figure 6-15: Chip micrograph

6.6.1 2D optic flow performance

The linearity test was performed by applying a horizontally moving bar pattern on the sensor. The measured linearity curve in Figure 6-16 (a) shows both results from the pure digital core and entire signal chain. The dotted triangle line with an error bar is characterized by the pure digital core. The linear performance of the pure digital optic flow core was characterized by directly applying the digital pattern through the test ports. The test pattern was the white bar on the black background. The pattern was controlled to move 1 pixel per n-th frame; thus, 1/n [pixel/frame] velocities were tested. The triangle and error bar in the plot are the mean and standard deviation of the measured optic flow. The result curve shows the average measured flow which linearly follows the input velocity. In addition, the measured error bar shows the increasing tendency by applying fast patterns. This is explained by the reciprocal operator that converts the time-of-travel to the velocity specified in Table 6-2. The reciprocal operator non-linearly converts the measured time-of-travel values. The 1-code difference in time-of-travel values is mapped differently to the velocity: 1-code difference in small time-of-travel values, for example 0 and 1frame/pixel, maps to large range of velocity, for example 1frame/pixel; 1-code difference in large time-of-travel values, for example 254 and 255frame/pixel, maps to small range of velocity, for example 1/255frame/pixel to larger range of the converted velocity. Thus, the fast motion is sensitive to the error introduced by the time-of-travel measurement as specified in the error bars of the plot. The circles in the plot were measured through the entire signal path starting from the pixel. To characterize the sensor, a horizontally moving bar pattern was applied by an LCD monitor; then, the projected pattern on the sensor was characterized by averaging all the measured flows. We also tested simple computer-generated patterns: translating, diagonally moving, and rotating patterns at 30fps as shown in Figure 6-16 (b). The translating and diagonal patterns were applied to verify the four axes of time-of-travel measurement were properly operating. The patterns consist of four bars moving from the center projectin out to the end of the corners with two different velocities. The bars were moving slowly for the half of the distance and sped up for the remaining half. Those velocity differences are reflected in the lengths of the arrows in the figure. The indicating error off the horizontal and vertical axes was observed in the translating pattern at the boundary of the bars. The error can be mainly categorized to an aperture problem, which is caused from the fact that the algorithm considers only 3x3 pixels to estimate the motion. The problematic case occurs when the 3x3 window is applied on the corner of the translating bar. For example, as specified in the circle in the figure, the corner of the pattern is passing the given window. The bright gray color of the window indicates the pattern located at the previous frame; the dark gray color describes the pattern has moved to the current location. Because the implemented algorithm calculates the 1D optic flow in four horizontal, vertical, and two diagonal directions, the problematic case causes the generation of motion in two axes, the vertical and one diagonal directions specified by the white arrows. As a result, the merged final 2D optic flow is off to the vertical axis as indicated by the red arrow. One solution to handle this type of error is to use additional measurement axes in diagonal directions so that the unwanted diagonal component can be cancelled out, which is described in the same figure. However, the additional measurement units increase hardware resources and power consumption. Thus, the optimal number of the measurement axes must be determined by considering the design constraints and applications. The accumulated 2D optic flow from the rotating bar test pattern is also shown in Figure 6-16.



(a) Linearity test result



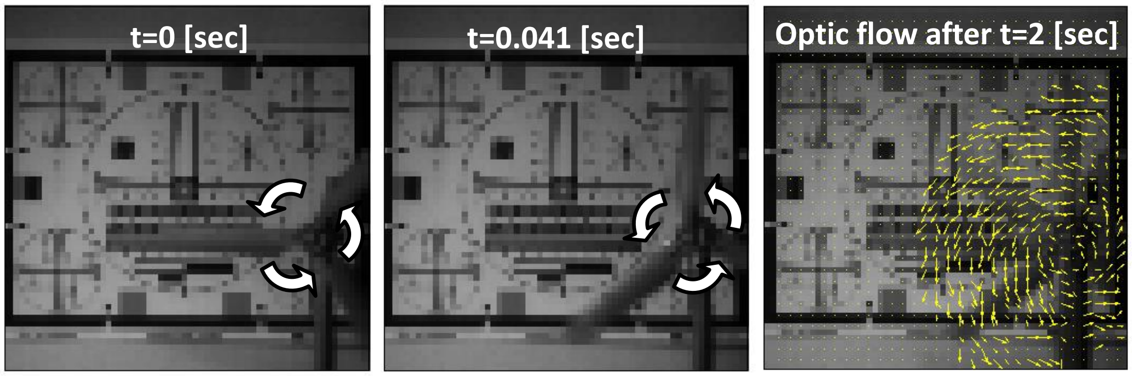
Generated optic flows from translating four bars
2
1
fast
slow
compensation
Generated optic flows from diagonally moving four bars
1
2
slow
fast
Generating optic flows from a rotating bar at the center

(b) Computer generated pattern test results

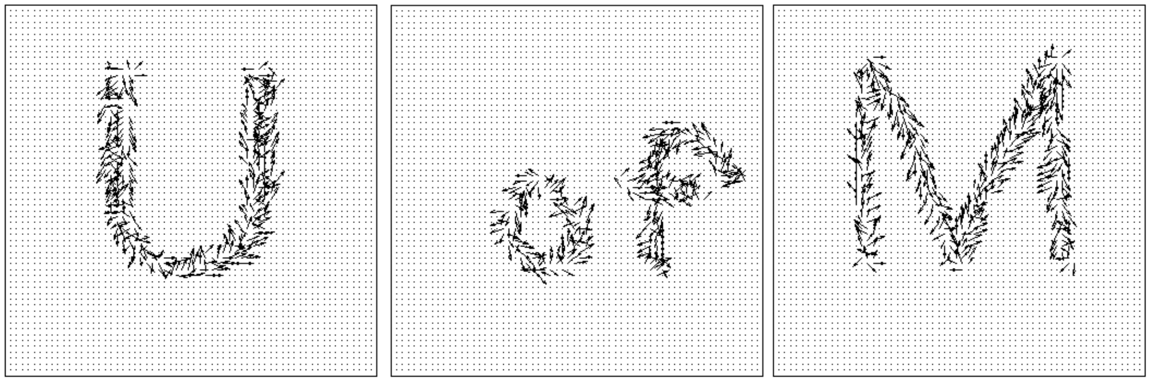
Figure 6-16: Characterized 2D Optic flow performance

6.6.2 Sample images and optic flow from real objects

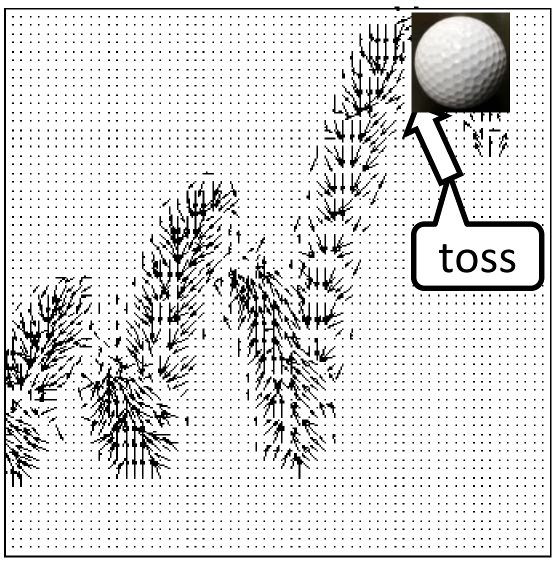
Sample images and captured 2D optic flows from real moving objects are shown in Figure 6-17. A rotating fan, a laser pointer, and a bouncing golf ball were used as the moving objects. The rotating fan was set up in front of the resolution chart in order to verify if the sensor was able to properly capture the optic flow under the condition of complicated background patterns. The fan was rotating with the speed of 160rpm; the sensor was capturing the flow at the frame rate of 120fps. The result shown in Figure 6-17 (a) is the accumulated optic flow for 2 seconds. Comparing to the result of a rotating bar pattern in Figure 6-17 (b), the captured optic flow is distorted near the complex background spatial edges. However, the global pattern of the measured flow clearly shows the rotation of the fan, which verifies the feasibility of extracting global self-motion of a MAV. A laser pointer was also used to test if the sensor can track the movement of a point source. We sequentially wrote the letters U, of, and M on a plane board. As shown in the Figure 6-17 (b), the sensor successfully captured the optic flow from the point source. Finally, the trajectory of a bouncing golf ball was captured by the sensor in Figure 6-17 (c). We tossed the ball to the highest spot of the trajectory; then, the ball freely dropped and bounced by following the trajectory. The sensor was operating at 120fps to capture the fast motion.



(a) 2D optic flow from a rotating fan for 2 seconds, captured at 120fps.



(b) Letters by tracing a laser pointer source captured at 60fps.

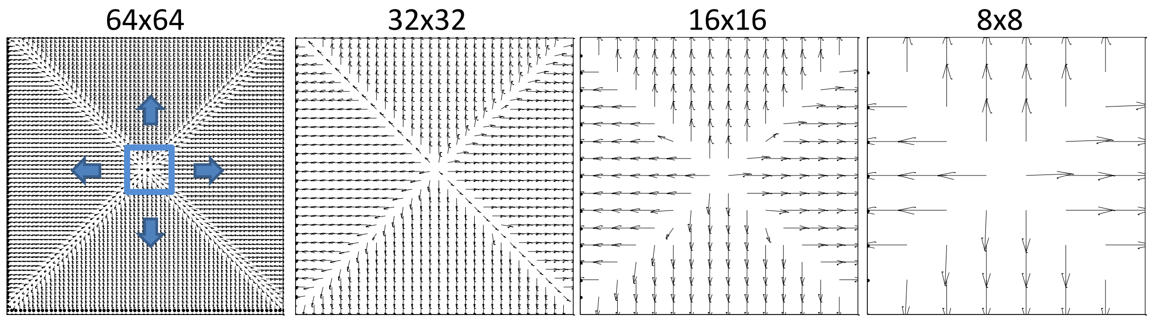


(c) Bouncing ball captured at 120fps.

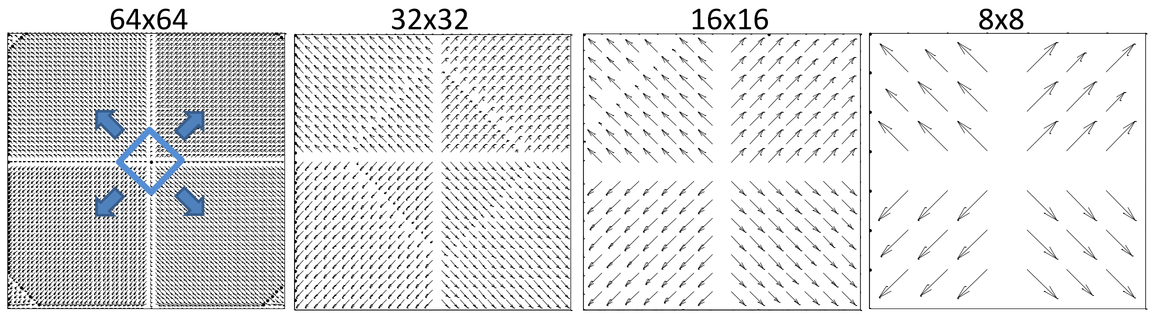
Figure 6-17: Sample images and captured optic flow from real moving objects

6.6.3 Down-sampler and WFI

The integrated down-sampler and WFI cores were characterized by applying patterns through the digital test ports. The patterns were generated in the external FPGA. The generated patterns of the expanding square and rhombus for the down-sampler tests are superimposed on the captured 64x64 flows in Figure 6-18. As shown in the figure, the down-sampler finds one vector that represents 2x2, 4x4, and 8x8 windows for the down-sampling ratios of x2, x4, and x8.



(a) Horizontally and vertically expanding square pattern



(b) Diagonally expanding rhombus pattern

Figure 6-18: Down-sampler performance

The WFI core was tested under the condition to perform 8x8 2D matched filtering with 6 coefficient sets. The coefficients were set to be sensitive to 6 different self-motions as illustrated in Figure 6-19 [66]. Four test patterns were applied: horizontally moving bar, vertically moving bar, horizontally and vertically expanding square, and diagonally expanding rhombus. The square and rhombus patterns were the same as those for down-sampling tests. The plots in the figure show the WFI (matched filtering) output in time. The unit of the x-axis is the frame index from 0 to 60frame; the unit of the y-axis is an arbitrary unit. For each test, two WFI outputs from the sensitive coefficient sets of the applied motions are shown. The other WFI outputs from remaining coefficient sets were fluctuated near zero for the test period. The WFI plots of the horizontally moving bar are from two coefficient sets: a) horizontal translation, and b) horizontal expansion. Because the bar is moving in one direction, the plot is all positive for the translation filter. However, for the expansion filter, the result is opposite in sign for half of the scene. The WFI plots of the vertically moving bar are also explained in the same way, which is sensitive to two coefficient sets: d) vertical translation, and e) vertical expansion. Both expanding square and rhombus patterns are sensitive to b) horizontal expansion and e) vertical expansion coefficients. The WFI plots of the expanding square pattern monotonically increase since the applied square pattern is continuously expanding. Thus, the number of locations generating optic flow increases. However, the WFI of the diagonal rhombus is symmetric to the middle of the applied period (near 40th frame) because the rhombus pattern started to break after half of the period. Then, the broken bars were shrinking for the remaining of the period.

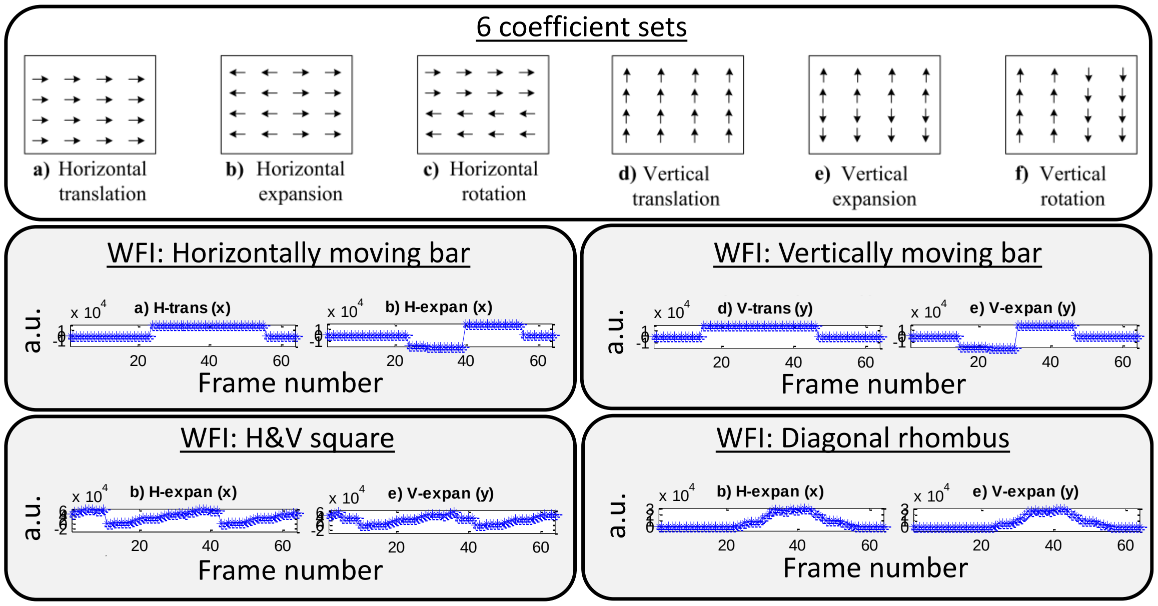


Figure 6-19: WFI core performance

6.6.4 Performance summary and comparison

The test configuration and the measured parameters are summarized in Table 6-3. The key parameters are compared with those of the state-of-the-art optic flow sensors and systems available in the literature in Table 6-4. As can be seen, the proposed sensor achieved the smallest power consumption per pixel to estimate 2D optic flow in the condition of 30fps operation. The digital optic flow sensing approaches require separate CPUs or FPGAs for the intensive digital computation in addition to normal image sensors [26], [37]. Thus, these systems are not optimized for a low power solution. The pure analog approaches are advantageous in regards to bandwidth and power consumption, especially subthreshold circuit design technique is applied. However, these approaches process the estimation in pixel circuits, which must be laid out large enough to minimize matching issues. Therefore, power consumption and pixel scaling are in a trade off relationship. The prototype sensor achieved the optimum power and area consumption by adopting the A/D mixed-mode approach. In addition, the sensor integrated for the first time the digital 2D WFI in the sensor, while the analog 1D WFI was implemented and demonstrated robot navigation in [89].

Table 6-3: Chip characteristics

Process	0.18 µm 1P4M CMOS		
Chip size	4.18 x 3.09 mm^2		
Pixel array	64 x 64		
Pixel size	28.8 x 28.8 µm^2		
Fill factor	18.32 %		
Maximum optic flow	1.96 rad/sec @ 120 fps, FOV 60º		
Power (Pixel, 3.3V @ 30 fps)	0.57 µW	2-D optic flow estimation	29.9 µW
Power (ADC 1.8 V analog, 0.9V digital @ 30 fps)	5.30 µW		
Power (2-D optic flow core, 1.8 V, @ 30 fps)	24.03 µW		
Power (Compressor & interface, 1.8 V, @ 30 fps)	79.48 µW		
Total power (@ 30fps)	109.38 µW		
Optic flow data rate after compression (16-b/sample raw data)	Average: 1.92-b/sample Peak: 4.84-b/sample (< 2.77% occurrence frequency)		

Table 6-4: Performance comparison of optic flow sensors and systems

	[26]	[37]
Image Sensor	OV7725 (120mW*)	N/A
Processor	BlackFin (64mW**)	Vertex Pro2 (50mW**)
Optic flow	2D digital (160x120)	2D digital (150x150)
Total power [mW]	184@55fps	>50mW@30fps
FOM [nJ/pixel]	173	73

- From datasheet, ** estimated

	[90]		[87]	[89]	This work	
Technology	0.5 μm CMOS		0.8 μm BiCMOS	0.5 μm CMOS	0.18 μm CMOS	
Array size	17x17		30x30	19x1	64x64	
Optic flow	1D analog		2D analog	1D WFI analog	2D digital 2D WFI digital	
Pixel size [μm^2]	61x199		124x124	112x257.3	28.8x28.8	
Total power [μW]	140 @70Hz	Core 5.2@70Hz	52/pixel @1kHz	42.6 @1kHz	109 @30Hz	Core 29.9@30Hz
FOM [nJ/pixel]	6.9	0.257	52	2.2	0.89	0.243

6.7 Conclusions

A bio-inspired A/D mixed-mode 2D optic flow sensor is implemented and fabricated for artificial compound eyes' applications. The sensor integrates the 2D time-stamp-based optic flow estimation core, which is developed for efficient implementation of bio-inspired time-of-travel measurement in the mixed-mode circuits. The sensor also integrates peripheral circuits to provide modular capability, which is required for our proposed multiple sensors on a module style artificial compound eyes' system. The integrated optic flow data compressor reduces the full resolution 2D raw optic flow down to 12.0% of the total data on average without any loss. Thus, more than 25 sensors can be mounted on the same 3MB/sec SPI bus. The integrated 2D digital WFI supports up to 12 matched filters for the down-sampled 2D optic flows (16x16 and 8x8). The sensor has successfully demonstrated to deliver all types of data at the maximum frame rate of 120fps through the 4 wired-SPI.

CHAPTER 7
CONCLUSIONS, CONTRIBUTIONS, AND FUTURE WORK

Miniaturized artificial compound eyes are one of the key components in MAVs. Multi-directional sensing and motion estimation capabilities can give wide FoV optic flows up to 360° of solid angle. By integrating the wide FoV optic flows, relevant information on the self-status of flight is parsed and utilized for flight commands generation. The proposed pseudo-hemispherical artificial compound eye system provides a simple and practical solution for multi-directional sensing. The main technical challenge for the successful realization of the system is to customize 2D optic flow sensors to meet design constraints inherited by MAV systems. The multiple-sensors-in-a-module fashioned system requires extremely low power optic flow core and expandable interface protocol. This work presents a bio-inspired time-stamp-based optic flow algorithm to accomplish a low power estimation core. In addition, integrated peripheral circuits enable a 4-wired interface between the sensors on the same module.

To achieve the aims above, the proposed bio-inspired optic flow sensors have introduced the following contributions/innovations:

- *Bio-inspired time-stamp-based optic flow estimation algorithm:* in this work, we introduced time-stamp-based optic flow sensing which robustly extracts velocity information from time-stamp image array.

- *A/D mixed-mode implementation of the time-stamp-based optic flow core:* analog temporal contrast measurement, 1-b feature A/D conversion, and digital arithmetic time-of-travel measurement accomplish optimal balancing in terms of hardware resources and computational power consumption.
- *Integrated circuit optimization for low power consumption:* supply voltage scaling, power gating of analog amplifiers, and simple optic flow computation reduce total power consumption.
- *Serialize data communication protocol for the multiple-sensors-in-a-module platform:* SPI connects all the sensors on 4 wires; data access protocols for raw optic flow, compressed flow, and WFI are implemented.
- *Lossless/lossy optic flow data compression:* entropy coding utilizing the sparsity property of optic flow data reduces data rate down to 1.92-b/sample so that 25 sensors can transmit full resolution data on a single bus.
- *Integrated digital WFI for robot command generation:* WFI function is implemented by supporting 12 user-defined matched filters configured through SPI bus.
- *Modular expandable sensor system:* the modular system can be expanded to cover wide FoV of surroundings and also to be independently configured to perform a variety of mission scenarios.
- *Dual-mode optic flow/normal image sensing:* normal image and optic flow sensing modes are supported in the same architecture to provide surrounding scene information for navigation and surveillance.

- *Dynamic configuration:* control parameters in the sensors are dynamically configured to support real-time adaptation; feature threshold, compressor threshold, down-sampling ratio, and WFI coefficients are configured on the fly.

7.1 Suggestions for future work

In this work, we introduced the bio-inspired optic flow sensors embedding the customized low power optic flow core and peripheral blocks for system integration. For further investigation and future work, the following research topics are suggested:

- Single chip stand-alone optic flow sensing system: the integration of a timing generator, a ramp generator, and bias circuits enables complete 4-wire interface on a miniaturized module; only 4 interface and power pins need to be bonded.
- Hybrid coarse and fine optic flow estimation: to overcome 1pixel/frame maximum detectable velocity inherited by bio-inspired EMDs, hybrid estimation is suggested; coarse flow is measured by displacement search, and fine optic flow is measured by time-of-travel.
- Contrast adaptation: contrast adaptation discussed in chapter 1 can be implemented by supporting a feedback path; the feature threshold is dynamically adjusted by monitored contrast statistics in the previous frame.
- Pixel scaling and high spatial resolution for dense flow sensing: high spatial resolution increases flow density; pixel scaling by relocating frame difference circuits from in-pixel increases spatial resolution.

- Pixel performance improvement: a variety of pixel architecture can be applied to improve pixel performance in terms of sensitivity, dynamic range, and area optimization.
- Lens customization: compact size, low payload, and a wide FoV lens design is a challenging research topic to allow the building of the artificial compound eye module.
- Flexible and expandable PCB design and origami packaging: the proposed artificial compound eye platform can be finally accomplish by foldable flexible PCB design and origami packaging; 3D configuration to achieve desired inter-sensor's angle requires a robust assembly process.

BIBLIOGRAPY

[1] D. Floreano, R. Pericet-Camara, S. Viollet, F. Ruffier, A. Brückner, R. Leitel, W. Buss, M. Menouni, F. Expert, R. Juston, M. K. Dobrzynski, G. L'eplattenier, F. Recktenwald, H. a Mallot, and N. Franceschini, "Miniature curved artificial compound eyes.," *Proc. Natl. Acad. Sci. U. S. A.*, pp. 1–6, May 2013.

[2] Y. M. Song, Y. Xie, V. Malyarchuk, J. Xiao, I. Jung, K.-J. Choi, Z. Liu, H. Park, C. Lu, R.-H. Kim, R. Li, K. B. Crozier, Y. Huang, and J. A. Rogers, "Digital cameras with designs inspired by the arthropod eye.," *Nature*, vol. 497, no. 7447, pp. 95–9, May 2013.

[3] M. K. Dobrzynski, R. Pericet-Camara, and D. Floreano, "Vision Tape—A Flexible Compound Vision Sensor for Motion Detection and Proximity Estimation," *IEEE Sens. J.*, vol. 12, no. 5, pp. 1131–1139, May 2012.

[4] M. O. Franz and H. G. Krapp, "Wide-field, motion-sensitive neurons and matched filters for optic flow fields," *Biol. Cybern.*, vol. 83, no. 3, pp. 185–197, Aug. 2000.

[5] J. Conroy, G. Gremillion, B. Ranganathan, and J. S. Humbert, "Implementation of wide-field integration of optic flow for autonomous quadrotor navigation," *Auton. Robots*, vol. 27, no. 3, pp. 189–198, Aug. 2009.

[6] F. Kendoul, I. Fantoni, and K. Nonami, "Optic flow-based vision system for autonomous 3D localization and control of small aerial vehicles," *Rob. Auton. Syst.*, vol. 57, no. 6–7, pp. 591–602, Jun. 2009.

[7] J.-C. Zufferey, A. Klaptocz, A. Beyeler, J.-D. Nicoud, and D. Floreano, "A 10-gram vision-based flying robot," *Adv. Robot.*, vol. 21, no. 14, pp. 1671–1684, Jan. 2007.

[8] F. Ruffier and N. Franceschini, "Visually guided micro-aerial vehicle: automatic take off, terrain following, landing and wind reaction," in *IEEE International Conference on Robotics and Automation, 2004. Proceedings. ICRA '04. 2004*, 2004, vol. 3, pp. 2339–2346 Vol.3.

[9] J. Zufferey, A. Klaptocz, A. Beyeler, J. Nicoud, and D. Floreano, "A 10-gram Microflyer for Vision-based Indoor Navigation," in *2006 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2006, pp. 3267–3272.

[10] A. Beyeler, J.-C. Zufferey, and D. Floreano, "3D Vision-based Navigation for Indoor Microflyers," in *Proceedings 2007 IEEE International Conference on Robotics and Automation*, 2007, pp. 1336–1341.

[11] F. Ruffier and N. Franceschini, "Aerial robot piloted in steep relief by optic flow sensors," in *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2008, pp. 1266–1273.

[12] S. Zingg, D. Scaramuzza, S. Weiss, and R. Siegwart, "MAV navigation through indoor corridors using optical flow," in *2010 IEEE International Conference on Robotics and Automation*, 2010, pp. 3361–3368.

[13] Q. Chu, B. Mulder, D. Choukroun, E.-J. Kampen, C. Visser, and G. Looye, Eds., *Advances in Aerospace Guidance, Navigation and Control*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013.

[14] P.-E. J. Duhamel, C. O. Perez-Arancibia, G. L. Barrows, and R. J. Wood, "Biologically Inspired Optical-Flow Sensing for Altitude Control of Flapping-Wing Microrobots," *IEEE/ASME Trans. Mechatronics*, vol. 18, no. 2, pp. 556–568, Apr. 2013.

[15] F. L. Roubieu, J. R. Serres, F. Colonnier, N. Franceschini, S. Viollet, and F. Ruffier, "A biomimetic vision-based hovercraft accounts for bees' complex behaviour in various corridors.," *Bioinspir. Biomim.*, vol. 9, no. 3, p. 036003, Mar. 2014.

[16] W. Buss, F. Kraze, A. Bruckner, R. Leitel, A. Mann, and A. Brauer, "Assembly of high-aspect ratio optoelectronic sensor arrays on flexible substrates." pp. 1–6, 2013.

[17] M. K. Dobrzynski, H. Vanderparre, R. Pericet-Camara, G. L'Eplattenier, S. Lacour, and D. Floreano, "Hyper-flexible 1-D shape sensor," in *2013 Transducers & Eurosensors XXVII: The 17th International Conference on Solid-State Sensors, Actuators and Microsystems (TRANSDUCERS & EUROSENSORS XXVII)*, 2013, pp. 1899–1902.

[18] F. L. Roubieu, F. Expert, G. Sabiron, and F. Ruffier, "Two-Directional 1-g Visual Motion Sensor Inspired by the Fly's Eye," *IEEE Sens. J.*, vol. 13, no. 3, pp. 1025–1035, Mar. 2013.

[19] "Micro Autonomous Systems and Technology (MAST)." [Online]. Available: http://www.arl.army.mil/www/default.cfm?page=332.

[20] W. R. Davis, B. B. Kosicki, D. M. Boroson, and D. F. Kostishack, "Micro Air Vehicles for Optical Surveillance," *Micro*, vol. 9, no. 2, pp. 197–214, 1996.

[21] B. M. Finio, S. Member, J. K. Shang, and R. J. Wood, "Body torque modulation for a microrobotic fly," pp. 3449–3456, 2009.

[22] B. B. B. Sc, "Improving flight performance of DelFly II in hover by improving wing design and driving mechanism," 2010.

[23] P. Zdunich, D. Bilyk, M. MacMaster, D. Loewen, J. DeLaurier, R. Kornbluh, T. Low, S. Stanford, and D. Holeman, "Development and Testing of the Mentor

Flapping-Wing Micro Air Vehicle," *J. Aircr.*, vol. 44, no. 5, pp. 1701–1711, Sep. 2007.

[24] R. J. Wood, "Design, fabrication, and analysis of a 3DOF, 3cm flapping-wing MAV," *2007 IEEE/RSJ Int. Conf. Intell. Robot. Syst.*, pp. 1576–1581, Oct. 2007.

[25] J. Humbert and A. Hyslop, "Bioinspired visuomotor convergence," *Robot. IEEE Trans.*, vol. 26, no. 1, pp. 121–130, 2010.

[26] J. Conroy, G. Gremillion, B. Ranganathan, and J. S. Humbert, "Implementation of wide-field integration of optic flow for autonomous quadrotor navigation," *Auton. Robots*, vol. 27, no. 3, pp. 189–198, Aug. 2009.

[27] A. M. Hyslop and J. S. Humbert, "Autonomous Navigation in Three-Dimensional Urban Environments Using Wide-Field Integration of Optic Flow," *J. Guid. Control. Dyn.*, vol. 33, no. 1, pp. 147–159, Jan. 2010.

[28] J. Zufferey, A. Klaptocz, A. Beyeler, J. Nicoud, and D. Floreano, "A 10-gram Vision-based Flying Robot," *Adv. Robot.*, vol. 21, no. 14, pp. 1671–1684, 2007.

[29] N. Franceschini, J. M. Pichon, C. Blanes, and J. M. Brady, "From Insect Vision to Robot Vision [and Discussion]," *Philos. Trans. R. Soc. B Biol. Sci.*, vol. 337, no. 1281, pp. 283–294, Sep. 1992.

[30] P. A. Shoemaker, "Insect-based visual motion detection with contrast adaptation," in *Proceedings of SPIE*, 2005, vol. 5783, pp. 292–303.

[31] F. Expert, F. L. Roubieu, and F. Ruffier, "Interpolation based 'time of travel' scheme in a Visual Motion Sensor using a small 2D retina," in *2012 IEEE Sensors*, 2012, pp. 1–4.

[32] J. Kramer, R. Sarpeshkar, and C. Koch, "Pulse-Based Analog VLSI Velocity Sensors," *Inf. Sci. (Ny).*, vol. 44, no. 2, pp. 86–101, 1997.

[33] B. N. Ranganathan, K. D. Dimble, J. M. Faddy, and J. S. Humbert, "Underwater navigation behaviors using Wide-Field Integration methods," in *2013 IEEE International Conference on Robotics and Automation*, 2013, pp. 4147–4152.

[34] J. Ratti and G. Vachtsevanos, "A Biologically-Inspired Micro Aerial Vehicle," *J. Intell. Robot. Syst.*, vol. 60, no. 1, pp. 153–178, Apr. 2010.

[35] F. Ruffier, S. Viollet, S. Amic, and N. Franceschini, "Bio-inspired optical flow circuits for the visual guidance of micro air vehicles," *Proc. 2003 Int. Symp. Circuits Syst. 2003. ISCAS '03.*, vol. 3, pp. III–846–III–849, 2003.

[36] B. D. Lucas, "An Iterative Image Registration Technique with an Application to Stereo Vision," *Proc. Imaging Underst. Work.*, vol. 130, pp. 121–130, 1981.

[37] V. Mahalingam, K. Bhattacharya, N. Ranganathan, H. Chakravarthula, R. R. Murphy, and K. S. Pratt, "A VLSI Architecture and Algorithm for Lucas–Kanade-Based optical flow computation," vol. 18, no. 1, pp. 29–38, 2010.

[38] Micron, "MT9M001: 1/2-Inch Megapixel CMOS Digital Image Sensor Datasheet," 2004.

[39] J. B., "Estimating Power for ADSP-BF534/BF536/BF537 Blackfin® Processors," 2007.

[40] Altera Corporation, "Stratix II and Virtex-4 Power Comparison." [Online]. Available: http://www.altera.com/devices/fpga/stratix-fpgas/stratix-ii/stratix-ii/features/st2-competitive.html.

[41] R. F. Chapman, *The Insects: Structure and Function*, vol. 1998. Cambridge University Press, 1998, p. 587.

[42] A. W. Snyder and R. Menzel, Eds., *Photoreceptor Optics*. Berlin, Heidelberg: Springer Berlin Heidelberg, 1975.

[43] M. V Srinivasan, M. Poteser, and K. Kral, "Motion detection in insect orientation and navigation.," *Vision Res.*, vol. 39, no. 16, pp. 2749–66, Aug. 1999.

[44] R. Dudley, *The biomechanics of insect flight: form, function, evolution*. Princeton University Press, 2002, p. 206.

[45] R. R. Harrison, "An Analgo VLSI Motion Sensor Based on the Fly Visual System," 2000.

[46] J.-C. Zufferey, *Bio-inspired Flying Robots: Experimental Synthesis of Autonomous Indoor Flyers (Google eBook)*. CRC Press, 2008, p. 250.

[47] C. Gilbert, "Visual neuroscience: hypercomplex cells in the arthropod visual system.," *Curr. Biol.*, vol. 17, no. 11, pp. R412–4, Jun. 2007.

[48] D. G. Stavenga and R. C. Hardie, Eds., *Facets of Vision*. Berlin, Heidelberg: Springer Berlin Heidelberg, 1989.

[49] W. Hassenstein, B., Reichardt, "Systemtheoretische analyse der Zeit-, Reihenfolgen-, und Vorseichenauswertung bei der Berwegungsperzeption des Rüsselkäfers Chlorophanus," *Zeitschrift für Naturforsch.*, vol. 11b, pp. 513–524, 1956.

[50] E. Buchner, “Elementary movement detectors in an insect visual system,” *Biol. Cybern.*, vol. 24, no. 2, pp. 85–101, 1976.

[51] M. Potters and W. Bialek, “Statistical mechanics and visual signal processing,” *J. Phys. I Fr.*, vol. Volume 4,, no. November 1994, pp. 1755 – 1775, 1994.

[52] H. G. Krapp, “Sensory integration: neuronal adaptations for robust visual self-motion estimation.,” *Curr. Biol.*, vol. 19, no. 10, pp. R413–6, May 2009.

[53] H. G. Krapp, B. Hengstenberg, and R. Hengstenberg, “Dendritic Structure and Receptive-Field Organization of Optic Flow Processing Interneurons in the Fly,” *J Neurophysiol*, vol. 79, no. 4, pp. 1902–1917, Apr. 1998.

[54] “Optic flows.” [Online]. Available: http://en.wikipedia.org/wiki/Optical_flow.

[55] J. L. Barron, D. J. Fleet, S. S. Beauchemin, and T. A. Burkitt, “Performance of optical flow techniques,” in *Proceedings 1992 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 236–242.

[56] J. Bouguet, “Pyramidal Implementation of the Lucas Kanade Feature Tracker Description of the algorithm,” vol. 1, no. 2, pp. 1–9.

[57] T. Delbruck and C. A. Mead, “Adaptive photoreceptor with wide dynamic range,” in *Proceedings of IEEE International Symposium on Circuits and Systems - ISCAS ’94*, 1994, vol. 4, pp. 339–342.

[58] B. Gilbert, “A precise four-quadrant multiplier with subnanosecond response,” *IEEE J. Solid-State Circuits*, vol. 3, no. 4, pp. 365–373, Dec. 1968.

[59] R. R. Harrison and C. Koch, “A Robust Analog VLSI Reichardt Motion Sensor,” *Analog Integr. Circuits Signal Process.*, vol. 24, no. 3, pp. 213–229, Sep. 2000.

[60] F. Ruffier, S. Viollet, S. Amic, and N. Franceschini, “Bio-inspired optical flow circuits for the visual guidance of micro air vehicles,” in *Proceedings of the 2003 International Symposium on Circuits and Systems, 2003. ISCAS ’03.*, 2003, vol. 3, pp. III–846–III–849.

[61] J. Krammer and C. Koch, “Pulse-based analog VLSI velocity sensors,” *IEEE Trans. Circuits Syst. II Analog Digit. Signal Process.*, vol. 44, no. 2, pp. 86–101, 1997.

[62] T. Köhler, F. Röchter, J. P. Lindemann, and R. Möller, “Bio-inspired motion detection in an FPGA-based smart camera module.,” *Bioinspir. Biomim.*, vol. 4, no. 1, p. 015008, Mar. 2009.

[63] R. Sarpeshkar, W. Bair, and C. Koch, “Visual Motion Computation in Analog VLSI using Pulses,” in *in Advances in Neural Information Processing Systems 5*, 1993, pp. 781–788.

[64] C. K. Rainer A. Deutschmann, “Compact Analog VLSI 2-D Velocity Sensor,” in *IEEE International Conference on Intelligent Vehicles*, 1998, pp. 359–364.

[65] R. Moeckel and S.-C. Liu, “Motion Detection Chips for Robotic Platform,” *Fly. Insects Robot.*, 2009.

[66] T. Zhang, H. Wu, A. Borst, K. Kuhnlenz, and M. Buss, “An FPGA implementation of insect-inspired motion detector for high-speed vision systems,” in *2008 IEEE International Conference on Robotics and Automation*, 2008, pp. 335–340.

[67] F. Aubepart, M. El Farji, and N. Franceschini, “FPGA implementation of elementary motion detectors for the visual guidance of micro-air-vehicles,” in *2004 IEEE International Symposium on Industrial Electronics*, 2004, vol. 1, pp. 71–76 vol. 1.

[68] F. Aubépart and N. Franceschini, “Bio-inspired optic flow sensors based on FPGA: Application to Micro-Air-Vehicles,” *Microprocess. Microsyst.*, vol. 31, no. 6, pp. 408–419, Sep. 2007.

[69] “FPGA Design and Codesign - FPGA and ASIC Design with HDL Coder and HDL Verifier.” [Online]. Available: http://www.mathworks.com/fpga-design/solutions.html. [Accessed: 08-Apr-2014].

[70] A. Borst, M. Egelhaaf, and J. Haag, “Mechanisms of dendritic integration underlying gain control in fly motion-sensitive interneurons,” *J. Comput. Neurosci.*, vol. 2, no. 1, pp. 5–18, Mar. 1995.

[71] R. A. Harris, D. C. O’Carroll, and S. B. Laughlin, “Adaptation and the temporal delay filter of fly motion detectors,” *Vision Res.*, vol. 39, no. 16, pp. 2603–2613, Aug. 1999.

[72] R. A. Harris, D. C. O’Carroll, and S. B. Laughlin, “Contrast Gain Reduction in Fly Motion Adaptation,” *Neuron*, vol. 28, no. 2, pp. 595–606, Nov. 2000.

[73] M. Joesch, B. Schnell, S. V. Raghu, D. F. Reiff, and A. Borst, “ON and OFF pathways in Drosophila motion vision.,” *Nature*, vol. 468, no. 7321, pp. 300–4, Nov. 2010.

[74] P. H. Schiller, J. H. Sandell, and J. H. Maunsell, “Functions of the ON and OFF channels of the visual system.,” *Nature*, vol. 322, no. 6082, pp. 824–5, Jan. 1986.

[75] H. Wässle, "Parallel processing in the mammalian retina.," *Nat. Rev. Neurosci.*, vol. 5, no. 10, pp. 747–57, Oct. 2004.

[76] S. Liu, "A Neuromorphic aVLSI Model of Global Motion Processing in the Fly," vol. 47, no. 12, pp. 1458–1467, 2000.

[77] H. Meng, K. Appiah, S. Yue, A. Hunter, M. Hobden, N. Priestley, P. Hobden, and C. Pettit, "A modified model for the Lobula Giant Movement Detector and its FPGA implementation," *Comput. Vis. Image Underst.*, vol. 114, no. 11, pp. 1238–1247, Nov. 2010.

[78] J. Choi, S. Park, J. Cho, and E. Yoon, "A 1.36μW adaptive CMOS image sensor with reconfigurable modes of operation from available energy/illumination for distributed wireless sensor network," in *2012 IEEE International Solid-State Circuits Conference*, 2012, pp. 112–114.

[79] J. Choi, S. Park, J. Cho, and E. Yoon, "A 3.4μW CMOS image sensor with embedded feature-extraction algorithm for motion-triggered object-of-interest imaging," in *2013 IEEE International Solid-State Circuits Conference Digest of Technical Papers*, 2013, pp. 478–479.

[80] G. Kim, M. Barangi, Z. Foo, N. Pinckney, S. Bang, D. Blaauw, and D. Sylvester, "A 467nW CMOS visual motion sensor with temporal averaging and pixel aggregation," in *2013 IEEE International Solid-State Circuits Conference Digest of Technical Papers*, 2013, pp. 480–481.

[81] E. Culurciello, R. Etienne-Cummings, and K. A. Boahen, "A biomorphic digital image sensor," *IEEE J. Solid-State Circuits*, vol. 38, no. 2, pp. 281–294, Feb. 2003.

[82] R. Moeckel and S. Liu, "Flying Insects and Robots," *Flying*, pp. 101–114, 2010.

[83] T. D. Ronald G. Benson, "Direction selective silicon retina that uses null inhibition," *Adv. Neural Inf. Process. Syst.*, vol. 4, pp. 756–763, 1992.

[84] J. Kramer, "Compact integrated motion sensor with three-pixel interaction," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 18, no. 4, pp. 455–460, Apr. 1996.

[85] P. A. Shoemaker, A. M. Hyslop, and J. S. Humbert, "Optic flow estimation on trajectories generated by bio-inspired closed-loop flight.," *Biol. Cybern.*, vol. 104, no. 4–5, pp. 339–50, May 2011.

[86] R. Etienne-Cummings, "Biologically Inspired Visual Motion Detection in VLSI," *Int. J. Comput. Vis.*, vol. 44, no. 3, pp. 175–198, Sep. 2001.

[87] A. A. Stocker, "Analog Integrated 2-D Optical Flow Sensor," *Analog Integr. Circuits Signal Process.*, vol. 46, no. 2, pp. 121–138, Dec. 2005.

[88] R. S. A. Brinkworth, P. A. Shoemaker, and D. C. O’Carroll, “Characterization of a neuromorphic motion detection chip based on insect visual system,” in *2009 International Conference on Intelligent Sensors, Sensor Networks and Information Processing (ISSNIP)*, 2009, pp. 289–294.

[89] P. Xu, J. S. Humbert, and P. Abshire, “Analog VLSI Implementation of Wide-field Integration Methods,” *J. Intell. Robot. Syst.*, vol. 64, no. 3–4, pp. 465–487, Feb. 2011.

[90] R. R. Harrison, “A biologically inspired analog IC for visual collision detection,” *IEEE Trans. Circuits Syst. I Regul. Pap.*, vol. 52, no. 11, pp. 2308–2318, Nov. 2005.