

**Bioinspired Symmetry  
Detection on Resource  
Limited Embedded  
Platforms**

**Alexander William Elliott**

*July 2017*

**Cranfield University  
PhD Thesis**

© Cranfield University 2017. All rights reserved. No part of this publication may be reproduced without the written permission of the copyright owner.

**This page is left intentionally blank.**

## **Abstract**

This work is inspired by the vision of flying insects which enables them to detect and locate a set of relevant objects with remarkable effectiveness despite very limited brainpower. The bioinspired approach worked out here focuses on detection of symmetric objects to be performed by resource-limited embedded platforms such as micro air vehicles. Symmetry detection is posed as a pattern matching problem which is solved by an approach based on the use of composite correlation filters. Two variants of the approach are proposed, analysed and tested in which symmetry detection is cast as 1) static and 2) dynamic pattern matching problems. In the static variant, images of objects are input to two dimensional spatial composite correlation filters. In the dynamic variant, a video (resulting from platform motion) is input to a composite correlation filter of which its peak response is used to define symmetry. In both cases, a novel method is used for designing the composite filter templates for symmetry detection. This method significantly reduces the level of detail which needs to be matched to achieve good detection performance. The resulting performance is systematically quantified using the ROC analysis; it is demonstrated that the bioinspired detection approach is better and with a lower computational cost compared to the best state-of-the-art solution hitherto available.

# **Preface**

## **Motivation and Background**

The appreciable computational demand (and limited performance) of many conventional approaches to visual information processing is the main reason here for seeking alternative algorithms. The need for efficient alternatives, motivating the work presented in this thesis, has arisen in the context of indoor flight of micro air vehicles (MAVs). MAVs are defined as autonomous flying machines with the size of six inches. The primary mission of the MAVs is to detect man-made objects occurring in typical indoor environments; such objects often exhibit symmetries. This indoor reconnaissance mission requires efficient, real-time image processing to be performed with very limited on-board processing capabilities available to a six-inch platform (whose main on-board resources must be devoted to powered flight). MAV design for indoor reconnaissance includes the ability to hover so that the focus here is on the detection of rotational symmetries in computer images without perspective (projective) distortion.

The proficient flight of insects relies largely on their vision. The main aspect of the insects ability to efficiently perceive the visual world is through the integration of both perception of static cues (simple features like edges exhibiting symmetries or orientation), and dynamic perception (tracking objects and estimating self-motion). The key to this efficient processing is due to the highly specialised nature of visual information processing.

Another important aspect learned from insects is their integration of simultaneous static and dynamic patterns with spherical perception. By matching simple patterns and their orientation by comparing the perceived objects and their relative motion with a small number of geometric and optic flow primitives, insects are able to fuse both dynamic and static perception to efficiently locate, track objects while still using visual cues for stabilisation and obstacle avoidance. The static and dynamic primitives are matched locally on small patches of the sphere and then matched globally on overlapping hemispheres so that local and global perception of objects and relative motion is achieved simultaneously. The integration of the local and global overlapping imagery enables robust perception and aids disambiguation of patterns, orientation and motion.

Using the insights from biological studies as a starting point, this thesis uses this inspiration to develop new and efficient approaches for both static and dynamic perception of symmetric objects. Symmetry detection is posed as a pattern matching problem and is solved with an approach based on the use of composite correlation filters. Such filters are designed in a special way so that the patterns to be matched are as efficient as possible. It should also be noted that methods presented in this thesis are not trying to mimic those used by insects, but rather draws inspiration and key principals from them.

## **Thesis Outline**

The focus on this work covers two main topics, that of static perception, and dynamic perception. Chapter 1 provides an overview of the insect visual system which is the motivation



for work presented in this Thesis. Chapter 2 covers the relevant engineering background that the work presented in this thesis that is used in later chapters. Chapter 3 proposes a method of using a pattern based approach for quickly recognising symmetries in images. Chapter 4 extends the pattern based methods to improve symmetry detection through with the use of motion. Later in this chapter a method to describe motion using spatio-temporal filtering is presented along with a pattern based approach to recognise self-motion within video sequences. Chapter 5 provides some closing remarks and discusses possible avenues for future work.

## **Contributions**

- Insect vision review from an engineering perspective (Chapter 1)
- Efficient symmetry detection using composite correlation filters, posed as a pattern matching problem (Section 3.3).
- Composite correlation filter design using training images of varying levels of complexity (Section 3.3).
- Performance comparison between the proposed symmetry detection method to the state-of-the-art feature based symmetry detection algorithm (Section 3.4).
- Symmetry detection based on motion using linear filtering techniques (Section 4.2).
- Motion characterisation for each pixel using spatio-temporal filtering techniques (Section 4.4).
- Ego motion recognition posed as a pattern matching problem using composite correlation filters (Section 4.4).

## **Acknowledgements**

I would like to thank my supervisor Professor Rafal Żbikowski, who provided tremendous guidance and insights during my PhD and the freedom to explore various research avenues.

# Contents

<b>Abstract</b>	<b>i</b>
<b>Preface</b>	<b>ii</b>
Motivation and background . . . . .	ii
Thesis outline . . . . .	ii
Contributions . . . . .	iii
Acknowledgements . . . . .	iii
<b>List of Symbols</b>	<b>vii</b>
<b>1 Biological Background</b>	<b>1</b>
1.1 Insect Visual Sensing . . . . .	1
1.1.1 Brief Overview of the Insect Eye . . . . .	1
1.1.2 Organisation of the Visual System of a Fly . . . . .	4
1.2 Insect Visual Processing . . . . .	9
1.2.1 Insect Static Perception of Symmetries . . . . .	9
1.2.2 Dynamic Perception in Insects . . . . .	15
<b>2 Engineering Background</b>	<b>22</b>
2.1 Loy-Eklundh Symmetry Detector . . . . .	22
2.2 Matched Filters . . . . .	26
2.2.1 Distortion Tolerance . . . . .	27
2.2.2 Convolution and Correlation . . . . .	29
2.3 Composite Correlation Filters . . . . .	29
2.3.1 MACE composite correlation filter . . . . .	31
2.3.2 UMACE composite correlation filter . . . . .	34
2.3.3 Optimal Trade-off MACH composite correlation filter . . . . .	35
2.3.4 3D MACH Filters . . . . .	36
2.4 ROC curves . . . . .	38
2.4.1 Confusion Matrix . . . . .	38
2.4.2 ROC Curve . . . . .	39
2.4.3 Chance Line . . . . .	41
2.4.4 Area Under the ROC Curve (AUC) . . . . .	41
2.5 Optic Flow . . . . .	41
2.5.1 Image Flow Constraint Equation . . . . .	42
2.5.2 Discontinuous Image Flows . . . . .	46
2.6 Spatio-Temporal Filtering . . . . .	48
2.6.1 Gabor Pairs and Spatio-Temporal Energy . . . . .	48
2.6.2 Speeding up the Spatio-Temporal Energy Equations . . . . .	50
2.6.3 Wildes and Bergen Features . . . . .	50

<b>3</b>	<b>Static Patterns</b>	<b>53</b>
3.1	Problem Setting . . . . .	53
3.2	Correlation Filter Training . . . . .	54
3.2.1	Training Templates for Correlation Filters . . . . .	54
3.2.2	Scale, Rotation and Translation Invariance . . . . .	57
3.2.3	Graded-abstraction Dataset for Training and Testing . . . . .	58
3.2.4	Enriched Normalised Dataset . . . . .	58
3.2.5	Actual Dataset Used for Training and Testing . . . . .	59
3.3	Graded-abstraction Testing and Performance Results . . . . .	60
3.3.1	Testing of G-template Design: G-G, G-S, G-E and G-B . . . . .	61
3.3.2	Testing of S-template Design: S-G, S-S, S-E and S-B . . . . .	63
3.3.3	Testing of E-template Design: E-G, E-S, E-E and E-B . . . . .	65
3.3.4	Testing of B-template Design: B-G, B-S, B-E and B-B . . . . .	65
3.3.5	Computational Performance . . . . .	69
3.4	Comparison with Feature Based Symmetry Detection . . . . .	69
3.4.1	Results . . . . .	69
3.5	Conclusions and Interpretation . . . . .	70
<b>4</b>	<b>Dynamic Patterns</b>	<b>73</b>
4.1	Problem Setting . . . . .	73
4.2	Symmetry Based on Motion . . . . .	74
4.2.1	Using Matched Filters to Define Symmetry . . . . .	75
4.2.2	Symmetry Recognition Results . . . . .	78
4.2.3	Conclusions and Interpretation . . . . .	80
4.3	Pattern Based Motion Recognition . . . . .	82
4.3.1	Motion Testing and Training Dataset . . . . .	83
4.3.2	Using Pixel Data for Motion Estimation . . . . .	84
4.4	Spatio-Temporal Pattern Based Motion Detection . . . . .	88
4.4.1	Spatio-temporal Energy Volumes . . . . .	88
4.4.2	Minimising Spatio-Temporal Energies . . . . .	92
4.4.3	Testing and Results . . . . .	95
4.5	Conclusions and Interpretation . . . . .	99
<b>5</b>	<b>Conclusions and Future Work</b>	<b>101</b>
5.1	Future Work . . . . .	101
5.1.1	Static Patterns . . . . .	101
5.1.2	Dynamic Patterns . . . . .	102
5.1.3	Integration . . . . .	103

<b>References</b>	<b>104</b>
-------------------	------------

<b>A</b>	<b>Appendix - Planar and Spherical Optic Flow</b>	<b>113</b>
A.1	Planar Case . . . . .	113
A.1.1	Problem Formulation (planar retina) . . . . .	113
A.1.2	Copoint and Coaxis Vectors . . . . .	117
A.1.3	Motion Recognition . . . . .	121
A.2	Spherical Case . . . . .	121
A.2.1	Problem Formulation (spherical retina) . . . . .	122
A.3	Advantages of a Spherical Retina . . . . .	124
<b>B</b>	<b>Appendix - Franz-Krapp Filter Derivation</b>	<b>129</b>
B.1	Output variance . . . . .	129
B.2	Minimisation of output variance . . . . .	130

## List of Symbols

$\bar{X}$	Diagonal matrix frequency domain average of training samples
$\bar{x}$	Frequency domain average of training samples
$\cdot$	Inner product
$\mathcal{F}$	Discrete Fourier transform
$\mathcal{F}^{-1}$	Inverse discrete Fourier transform
$\nabla E$	Image brightness gradient
$\otimes$	Correlation
$\sigma$	Spread of the spatio-temporal Gaussian window
$\sigma_s$	Scale variation
$\star$	Convolution
$*$	Complex conjugate
$A_{ij}$	Symmetry weighting
$C$	Power spectral density of additive input noise
$D$	Average power spectrum
$d$	Array size
$D_{ij}$	Gaussian weighting function
$E$	Change in brightness between frames
$E_h$	Energy scalar of filter $h$
$E_s$	Energy scalar of test images $s$
$f$	Fourier domain frequency in spatio-temporal volume
$g$	Correlation plane output
$G^\theta$	Gabor filter orientated at $\theta$
$h$	Filter
$I$	Spatio-temporal input volume
$N$	Number of training samples

$R_{ij}$	Rotational symmetry weighting
$S$	Diagonal similarity matrix
$s$	Test input image
$S_{ij}$	Scale weighting
$u, v$	$x, y$ velocity vector components
$w$	Sliding window size
$X_i$	Frequency domain $i$ th training sample

# 1 Biological Background

This first Chapter of this thesis studies the visual system of flying insects. Looking at the biological aspects from an engineering perspective allows for insights to be drawn and later implemented in this thesis. Because computer hardware and flying insects are fundamentally different, the aim here is not to mimic insects but rather collect some key insights. This first chapter is divided into insect visual system from both a sensing and processing which forms a nice parallel to an engineering implementation that relies on both specialist hardware and software.

This chapter is laid out as follows:

- Section 1.1 looks at the visual sensors of flying insects, namely the compound eye. Compound eyes are fundamentally different to how most cameras work.
- Section 1.2 looks at how insects process the visual information for both static and dynamic perception.

## 1.1 Insect Visual Sensing

### 1.1.1 Brief Overview of the Insect Eye

All flying insects have two large compound eyes, that occupy most of their head [1]. Each compound eye is made of many facets (hexagonal lenses) that fit together in a honeycomb-like fashion, as shown in Figures 1 and 2. Light from only a small part of the total scene is admitted by each lens of the eye which focuses light into a small tube underneath each lens. This small tube called the rhabdom is a light-sensitive organ that contains several photosensitive cells [2] and [3].

Each photosensitive cell within the rhabdom shares the same visual field but each cell responds to only certain spectrum of light (such as Ultraviolet, Blue, and Green). Most insects have trichromatic colour vision, compared to human vision, the insect's visible spectrum is shifted more towards the shorter wavelengths.

The collection of lens and rhabdom (along with the crystalline cone) is called ommatidia. Each ommatidium in the compound eye points in a different direction and is arranged in a spherical arrangement. Each ommatidium consists of a lens that focuses light onto the rhabdom inside which contains several photosensitive cells. In many insects each ommatidium is a single sampling element with its own optical axis; only insect eyes with this type of ommatidium will be discussed here. Light absorbed by the rhabdom transmits a signal to the insect's brain through an optic nerve.

Two angles are of prime importance in understanding the function of an insect eye, see Figure 3:

- The first is the angle between the optical axes of two adjacent ommatidia. This is called the inter-ommatidial angle, denoted by  $\Delta\phi$ , and determines how densely the compound eye samples the visual world.

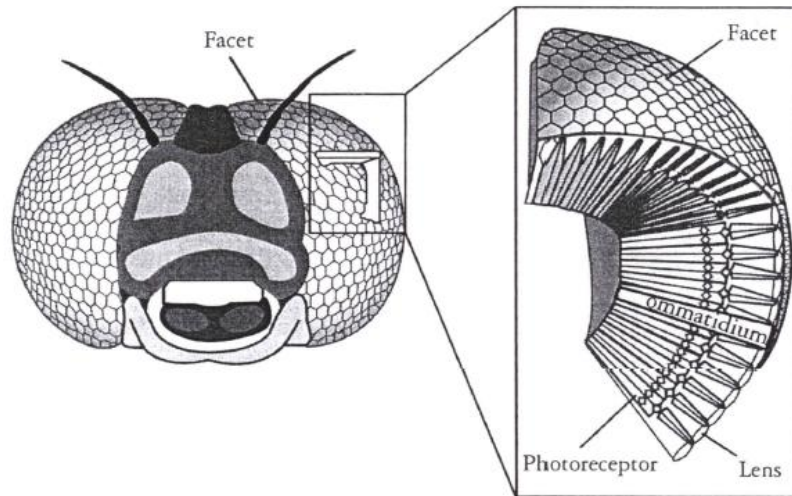


Figure 1: The structure of a typical compound eye of flying insects. This Figure shows how the eyes are made up of many hexagonal facets (lenses). The combination of the lens, photoreceptor and crystalline cone makes up the ommatidium. The eye consists of many ommatidia arranged radially around the eye [2].

- The second is the angle defining the field of view of a single ommatidium, denoted by  $\Delta\rho$ .

The shape of the field of view is in effect the sensitivity of the ommatidium to a point source of light as a function of angle from the optical axis. The sensitivity of the ommatidium is at its greatest along the optical axis and it decreases with the increase of the angle from the axis. Compound eyes employ many apertures to construct a single overall image of the environment.

Compared to the concave structure of single chamber eyes in humans (camera type eyes), compound eyes are located on the outside of an insects head and are convex in structure.

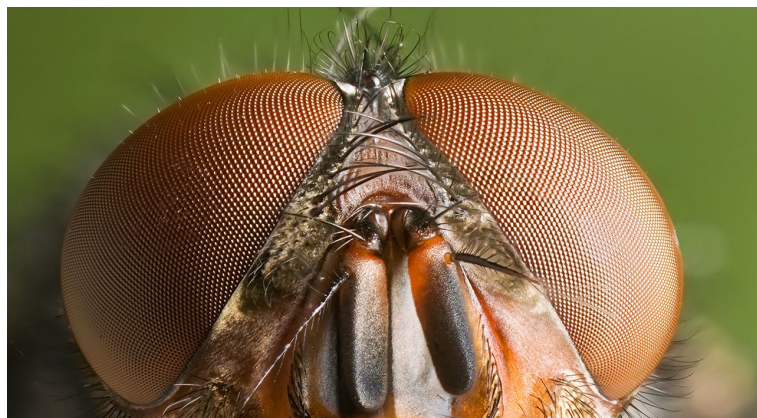


Figure 2: A compound eye of the on a *Calliphora vomitoria* (bluebottle) fly [4].



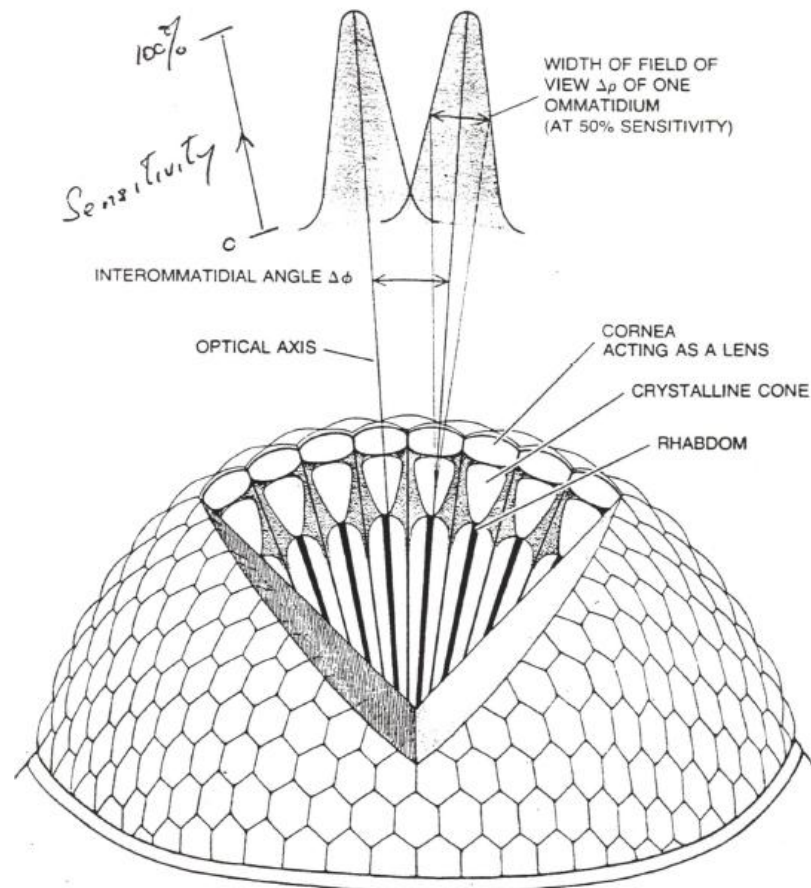


Figure 3: Structure of the insect eye determines how much detail the eye is capable of seeing. Each ommatidium has a cornea that also serves as a lens focusing light through a transparent cone onto a light-sensitive element: the rhabdom. The optical axis of the ommatidia is the line extending through the centre of the lens to the rhabdom. The angle between the optical axes of two adjacent ommatidia and is called the inter-ommatidial angle. The curves above each ommatidium indicate how the sensitivity of the ommatidium decreases with angular distance from the ommatidium optical axis. The field of view of each of the ommatidia is defined as the angle subtended where the sensitivity has fallen to 50 percent of its maximum value [1].

In spite of this major topological difference, the job of the two kinds of eye structure is the same. That is to break up the incoming light according to its direction of origin (see Figure 4). The other difference between the two kinds of eye is that compound eyes employ multiple optical systems. In apposition eyes, such as those of most diurnal insects, each of the lenses form a tiny inverted image, each with a small view of the entire scene (although this is not what the insect actually sees). The similarities between an apposition eye, and a camera type eye is shown in Figure 4.

The function of each ommatidium is to focus the inverted image captured by the lens onto the tip of the rhabdom. The rhabdom acts like a light guide because it has a slightly higher reflective index compared to the surroundings. This means that the rhabdom guides the light to ensure it moves downwards, due to the internal reflections. In this process, due to the multiple internal reflections, any spatial information is lost as it enters the rhabdom. Because of this, the rhabdom acts like a photocell and averages any light that comes in. Geometrically it's field of view of is the angle on the tip that meets at the nodal point of the cornea of the lens, this angle is known as the acceptance angle,  $\Delta\rho$ . The inter-ommatidia angle,  $\Delta\phi$  is defined by the field of view of each ommatidium as shown in Figure 4. In most apposition eyes, the inter-ommatidia angle,  $\Delta\phi$  is about the same as the acceptance angle,  $\Delta\rho$ . Due to the angle being similar and to the placement of the rhabdom and lens, each rhabdom 'opposes' the field of field of its neighbour, which produces an overall erect image. This erect image consists of adjacent fields of view that have been mosaicked together.

There are three primary types of compound insect eyes found in nature, apposition eye, and a neural or optical superposition eye. The fundamentals of each are all similar but the key concept of insect vision is that insects brain process the information for all of the lenses in parallel both simultaneously and sequentially. The simultaneous parallel processing is used to detect simple features, largely based on symmetry while the sequential processing to used for motion detection as a form of optic flow. The fundamentals of insect static and dynamic perception are discussed in the following sections. A more detailed study of the insect vision was conducted and that covers the types of compound eyes and optics in more details is presented in an internal report submitted to the PhD sponsor [6]. This is not included in this thesis as the focus here is not on hardware.

### 1.1.2 Organisation of the Visual System of a Fly

A description of the visual system of the fly can be seen in Figure 5A. The visual system can be broken down into 3 main sections (known as ganglia) of neurons (lamina, medulla and the lobula), that each corresponds to three vision processing centres, and is shown in Figure 5. The three neuron sections are briefly described below.

**The Lamina** The Lamina is the first layer of neurons after the receptor layer in the insect eye. This layer takes the direct outputs from the photo receptors. One of the roles of these neurons is to act as a gain control to quickly adapt to changes in background light intensities. The lamina also acts like a high pass filter which amplifies the temporal changes.

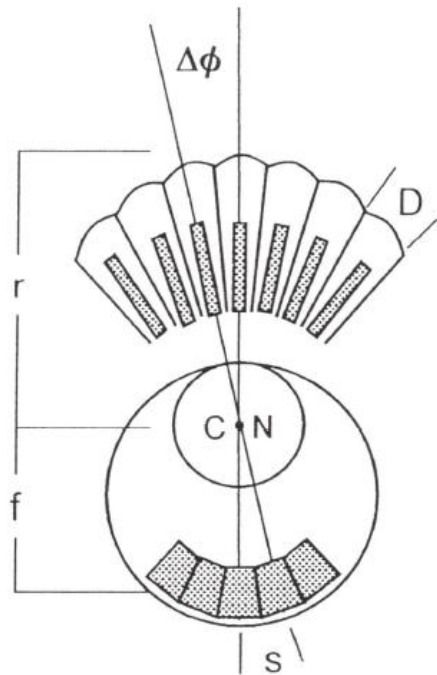


Figure 4: The underlying similarity of function in apposition and simple (camera-type) eyes. The sampling angle  $\Delta\phi$  (inter-ommatidial or inter-receptor angle) is  $D/r$  in an apposition eye and  $s/f$  in a simple eye.  $D$  is the facet diameter,  $r$  the radius of curvature (centre  $C$ ),  $f$  the simple eye focal length and  $s$  the receptor separation.  $N$  is the nodal point of the simple eye [5].

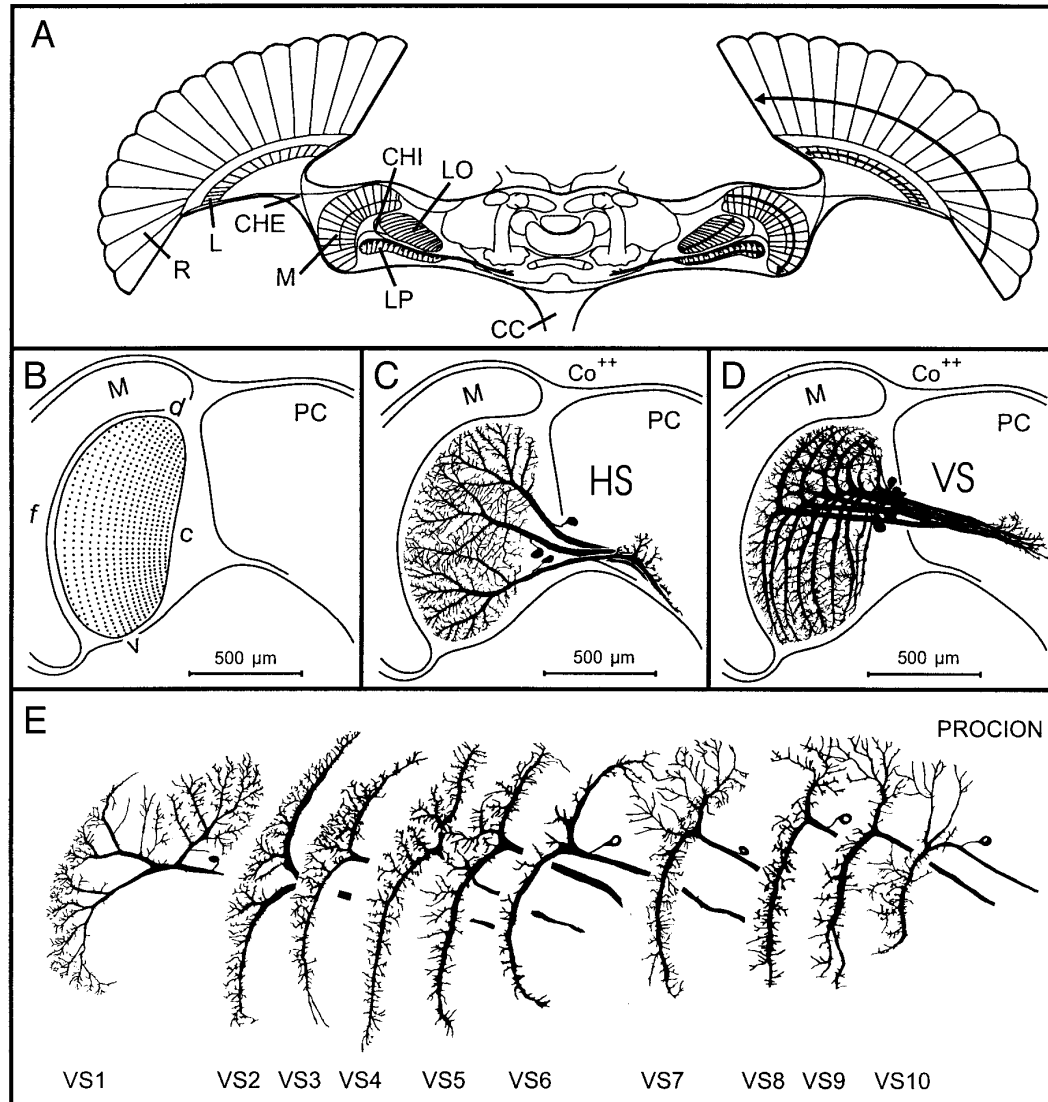


Figure 5: Visual system of the blowfly [7]. A: A horizontal section of the brain showing the retina (R), 3 visual neuropiles: lamina (L), medulla (M), the lobula (LO,LP). B: Schematic of the right visual hemisphere (f: front, c: back, d: top, v: bottom). C: Example of a horizontal system HS neuron (there are three HS neurons out of 60 LPTCs; ‘horizontal’ refers to horizontal stripes of their dendrites);  $\text{Co}^{++}$  means that cobalt staining was used. D: Example of a vertical system VS neuron (there are ten VS neurons out of 60 LPTCs; ‘vertical’ refers to vertical stripes of their dendrites);  $\text{Co}^{++}$  means that cobalt staining was used. E: Dendritic fragments of each of the ten VS neurons; PROCION means that procion yellow injections were used.

**The Medulla** Due to the tiny size of the cells in the medulla, it is difficult to record readings from them. The outputs from the lamina are fed into the medulla, but during this movement between the two ganglia, the image is inverted during projection to the medulla from front to back. The main function of the medulla is to perform local optic flow detection. There are about 50 such neurons for each ommatidium in the eye. The outputs from the medulla are then sent to the lobula.

**The Lobula** The final optic neurons called the lobula, are responsible for combining all the information from many thousands of receptor responses that have been pre-processed by the previous two neurons (lamina and medulla). The lobula plate essentially integrates the directional motion information from all the parts around the eye. In the lobula plate, all of these responses are combined into just 60 cells, known as lobular plate tangential cells (LPTCs). These LPTCs are in the form of broad dendritic trees which receive inputs from very big regions of the medulla, which means that these have very large receptive fields. Further down this information from the LPTCs are projected to other parts of the brain which in turn are used for functions such as flight muscle control further down into the insect's thorax.

Furthermore, the LPTCs can be divided into two systems: A horizontal and a vertical system. There are 10 vertical system cells (VS-cells), and 3 horizontal system cells (HS-cells). The HS-cells combine the visual input from the bottom (HS-South), middle (HS-Equatorial) or top (HS-North) regions of the visual fields of each eye and can be seen in Figure 6). However, it is important to note that the HS-North and HS-South cells also receive extra rotation specific signals from the visual field on the opposite side. As the name suggests, the vertical system cells have the main dendrites orientated vertically.

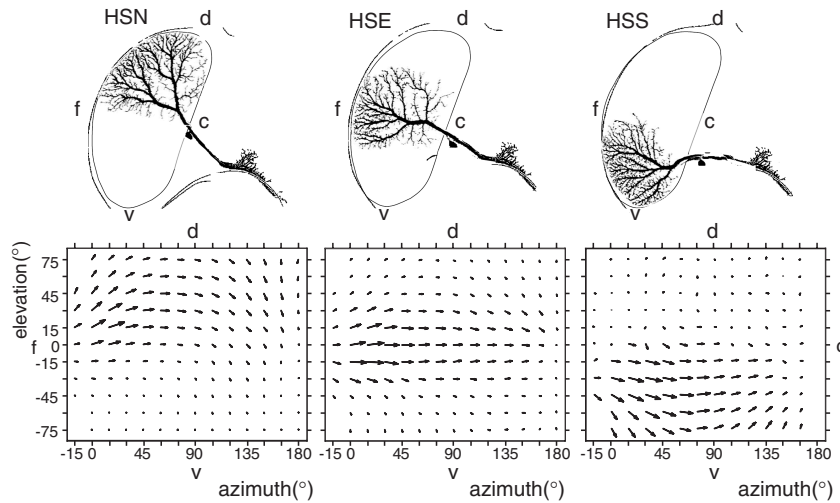


Figure 6: Horizontal system of *Calliphora* LPTCs: HSN, HSE, HSS and optic flow patterns they detect [8]; the patterns are reminiscent of those for pure translation.

Holger Krapp *et al* , conducted experiments by applying visual stimulation to a fly and

the local directional tuning curve of individually identified LPTCs were measured [7]. When their axes of sensitivity are plotted on a cylindrical projection of the visual field, (see Figure 7) the axes of eight of the ten VS-cells (VS3–VS10) fall along a straight line with a slope of about  $12^\circ$  with respect to the horizontal (linear regression:  $p < 0.05$ ,  $R^2 = 0.93$ ). Hence, while the neuron at the start of this line (VS6) is sensitive to pure roll motions, the remaining neurons along the line are sensitive to combinations of roll, pitch, and yaw. The other two VS-cells (VS1–VS2) are each sensitive to rotations about transverse axes tilted slightly with respect to the horizontal, so are sensitive to a combination of pitch, and to a lesser extent, yaw. The structure of this arrangement is striking: rather than being spread out across the visual field so as to sample as wide a range of rotations as possible, the ten VS-neurons are concentrated in a very specific manner. This arrangement may be specialised for a particular manoeuvre.

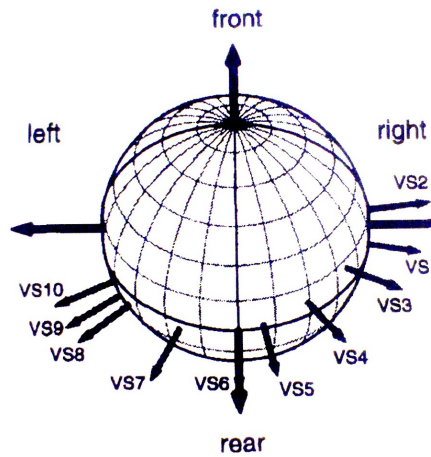


Figure 7: The preferred axis of rotation of the VS neurons. The preferred axis of VS6 coincides with the body axis (roll) and the preferred axis of VS1 and VS2 close below and above the transverse body axis (pitch) [9].

Holger Krapp *et al* [7] showed that the LPTCs do the following:

- Each LPTC covers approximately one hemisphere;
- Each LPTC responds most strongly (is matched to) a specific global pattern of the optic flow on the sphere  $S^2$ ;
- Each of the patterns corresponds either to a pure translation or a pure rotation about an axis;
- Each axis (preferred direction) is different and the axes are arranged non-orthogonally.

To summarise, The VS and HS neurons in the visual system convey key information about vertical and horizontal motion. Rotational components are unavoidable during the flight of an insect. But the rotational optic flow component does not contain any information

about the 3-D environment. The translational component only has information about the 3-D environment. So for tasks such as long or short range distance estimation, an optic flow field that is purely translational is required. Something that could aid this is to reduce the rotational component in the optic flow, which can be achieved by compensating for it by controlling the steering manoeuvres and/or stabilising the head motion of the insect. Evidence of reducing the rotational component is present not only in insects when flying, or vision stabilisation, but also in other animals and humans. Of course, to generate the compensatory actions, the respective rotational self-motion needs to be determined by the sensory systems and transformed into an adequate motor control signal. It is conceivable that HS neurons are utilised to sense both translation and rotation as well.

## 1.2 Insect Visual Processing

### 1.2.1 Insect Static Perception of Symmetries

Honeybees are commercially important pollinators and can be trained in controlled conditions. As a result, they have commonly been used for experiments. Recently, Horridge reappraised the real meaning of honeybee symmetry perception and pointed out its remarkable simplicity [10]. That simple perception is based on detection of elementary geometrical cues together with the spatial placement of the cues and has been shown through several studies and experiments [11], [12]. Based on the accumulated evidence from such carefully-conducted experiments, the current understanding of the honeybee symmetry perception involves three elements:

- feature detectors for edge/contrast which is generated from responses of local receptors;
- cues which are generated from responses of the feature detectors within a local region of the eye;
- landmarks which are generated from relative orientation of the cues. Only the cue angles (not the actual shapes) are used for efficiency and scale invariance.

**Feature Detectors** Bees have trichromatic vision with receptors sensitive to the Blue, Green and Ultraviolet wavelengths. Irrespective of the wavelength, each of the receptors is able to detect edges by detecting changes in intensity across each receptor. Across an edge there is a change/modulation in the receptor response. These responses are fed deeper inside the brain to the lamina of the brain (see Section 1.1.2). The modulated responses are fed into feature detector arrays which are able to detect relative contrast changes at edges but are not affected by overall changes brightness in large areas of the eye. An edge and modulation feature is detected when local neighbouring receptors detect a change in intensity response, an example of this is can be seen in Figure 8.

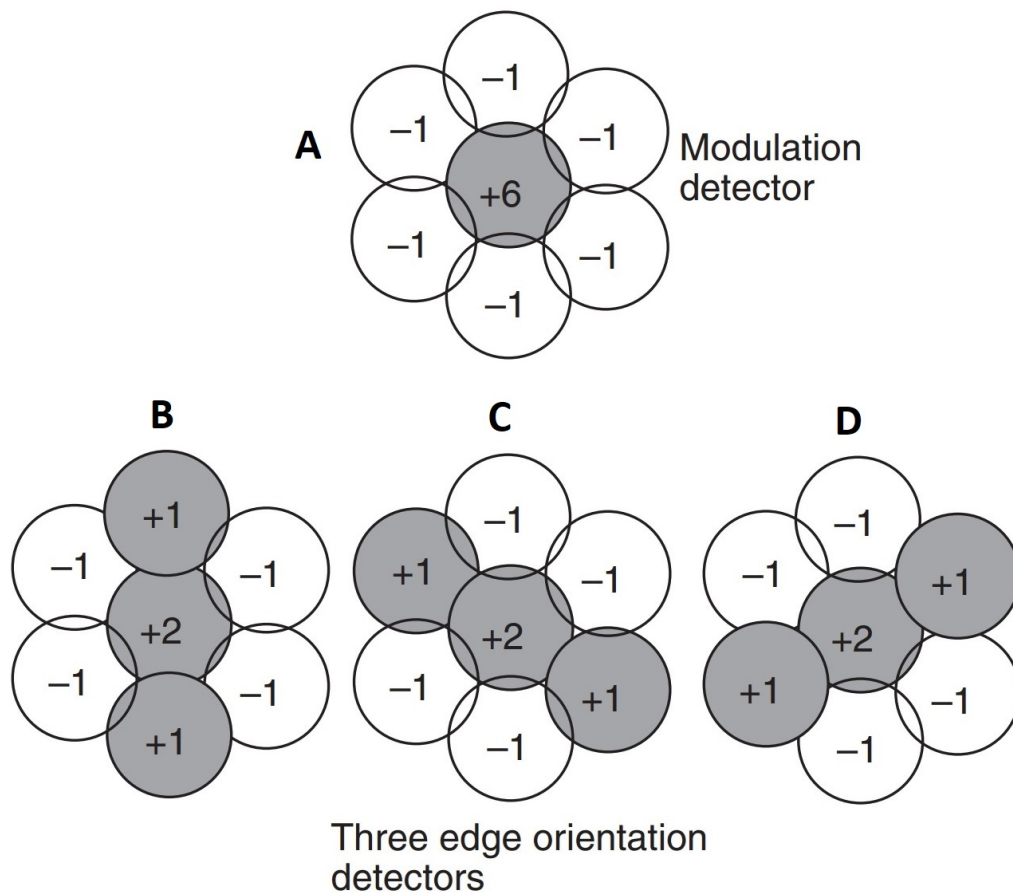


Figure 8: The main types of feature detectors in the honeybee detected by intensity responses from local neighborhoods of 7 receptors (see Figure 1). These responses are not sensitive to widespread intensity changes. **(A)** A modulation detector is a radial symmetry of receptor responses around a central receptor. **(B,C,D)** Show feature detectors with bilateral symmetry can detect edge orientation. The arrangement of the receptors used for this type of feature means that edge orientation detectors are green sensitive and colour blind. The numbers shown represent the relative excitation (+) and inhibition (−) response of the receptors due to light. Image adapted from [11].



**Modulation Detectors** Researchers measured the angular resolution modulation detectors by training bees to tell the difference between black grating patterns that were vertically and horizontally orientated [12]. It was found that the smallest grating period that was able to be detected by the bee was  $2^\circ$ . Because the field of view of a receptor in the eye of the bee is about  $2.5^\circ$ , the modulation detectors in bees have a resolution limit of a single receptor. The modulation detector in bees is able to detect spots in the scene as shown in Figure 8. A single receptor with a strong response is surrounded by neighbouring receptors with little or no response.

**Edge Detectors** Experiments on bees showed that bees use edge detectors in addition to modulation detectors [11]. This was shown in experiments where bees were trained to associate food with certain types of edges (dark edge on a light background, and light edge on dark background). However, experiments showed that bees could not recognise which side of an edge is light or dark, which means that they are able to detect local edge orientations of fuzzy or sharp images based on receptor responses in a local neighbourhood as shown in Figure 8. Each edge detector is symmetrical about the axis of orientation. Another aspect of the bee edge detectors is that they are not like edges detectors of computer vision algorithms (Canny edge detector). Rather than detecting the entire edge, each edge detector of the bee acts independently and does not give a single response along an entire edge, rather at points along it. Furthermore, the edge detectors used by bees can only detect edges in just 3 orientations due to the fact that each edge detector consists of only 7 receptors.

### Cues from Feature Detectors

**Cues from Edge Detectors** The bee forms cues based on the sum of the number of responses from each of its feature detectors within a local region of the eye as shown in Figure 9. This summation of features to form cues has 4 main aspects:

- Edges detected in black areas are summed separately from edges detected in colour regions of the eye.
- Edge detectors consist of a local neighbourhood of receptors and are not able to detect the entire shape of edges. They act independently. So a series of edges along the same direction is indistinguishable to a single continuous edge along the same direction in terms of bee vision.
- Edge detector responses are summed in a specific way that effectively allows the bee to disregard features in a scene that are not relevant. Edge detectors that are at right angles to each other are summed to cancel each other out; this is shown in Figure 9. As an example, bees are not able to tell the difference between a square cross at  $40^\circ$ , and the same square cross that is rotated by  $45^\circ$ . Because of the way the edge features are summed, the orientation will be lost for a  $45^\circ$  line that is made of steps.

- Edge detectors are added together in a special way that allows detection of regions of radial or circular symmetry, or hubs in each local region (Figure 9E,F). An interesting aspect of this type of cue is that the location of the radial pattern/hub is remembered. However, the actual layout of the edge detectors around it are lost when they are all summed together.

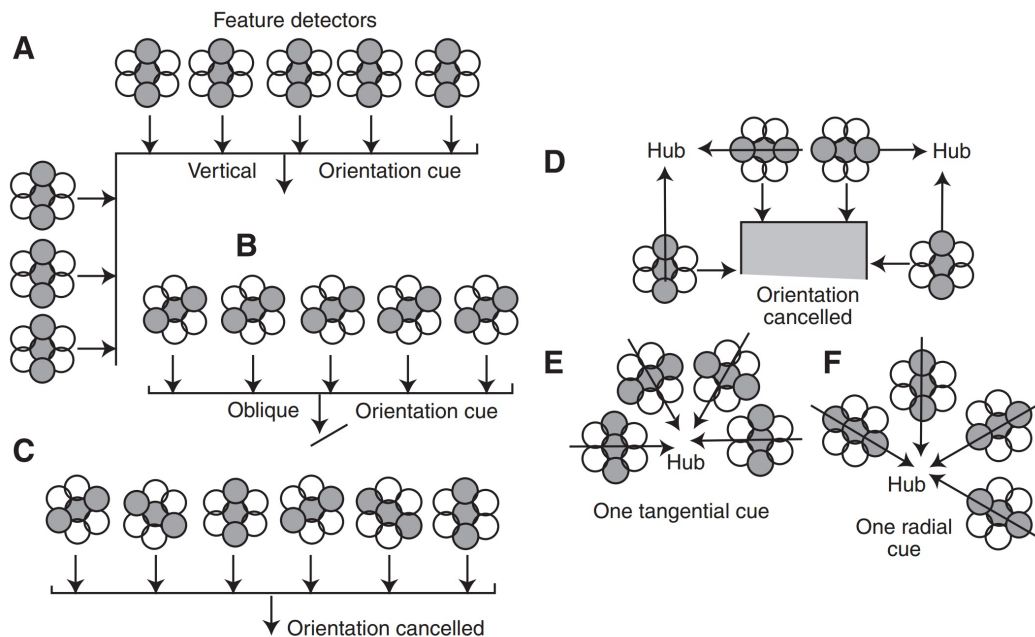


Figure 9: Cues are formed by the summation of edge orientation feature detectors. **(A)** Edge Detectors of vertical axes orientation. **(B)** A line of oblique orientation edge detectors. **(C)** A combination of mixed edge orientations cancel out, but the modulation of the edge is maintained, along with the position. **(D)** Perpendicular edge orientations cancel out along the edges of a square. However, weak hubs are still detected along the corners. **(E,F)** Radial and tangential symmetry cues are formed from hubs of edge detectors [11].

A key to the efficient processing of insect vision is the ability to reduce or filter the information of the scene to only extract what is relevant to the specific task. This is achieved through the special summation of cues to form features as described above. The individual feature detectors are summed together in local regions of the eye that are approximately  $25^\circ$  across, where each region only contains a few cues. Because of this simplicity, bees recognise the cues with no prior information about the pattern that forms them. This results in trained bees recognising wrong patterns that have the same cues the bees were trained on, assuming there are not any unexpected cues in the pattern. Bees are highly generalised in the sense that they learn cues for specific objects, not a library of cues that will describe any object. A summary of the types of cues a bee detects can be shown in Figure 10 in order of preference.















Feature	Weak	Strong
Modulation		
Position of black		
Area		
Radial		
Bilateral symmetry		
Orientation		
Tangential		

Figure 10: This table shows a visual approximation of the 7 types of cues that a bee can detect. The table is listed in order of preference that bees learn/recall cues with most preferable at the top [11].

**Cues from Receptors (Area Cues)** Another important cue that bees detect are area cues. Rather than small groups of 7 neighbouring receptors, area cues are collected over larger regions in the eye that have a similar response in terms of colour, and intensity. Experiments showed that bees learn areas with preference related to the number of receptor responses  $\times$  receptor brightness. The centroid of the area is also stored. Experiments showed that other things like the actual shape of the area are ignored, making this very efficient in terms of computation. Bees do, however, also encode some edge features for smaller areas which allow them to discriminate between some smaller areas, but this is using the edge features and not the actual area cues. If there are two areas that are both small enough to fit into a local region of the eye, the total area will be stored along with the common centre. An example of local areas around the bee vision panorama can be seen in Figure 11.

**Landmarks from Cues** Just as feature detectors are grouped together to form cues, cues are combined in local regions of the eye to form landmark labels as shown in Figure 11. This overall process of how landmarks are formed can be seen in Figure 12. These labels for landmarks are detected whether just one landmark or many are detected in a region of the eye. More specifically, the bees remember the coincidence of the cue responses to form a landmark. Bees are only interested in the angles between each landmark, not the actual shapes they see in the panorama of their vision. The way this is done starts with features, where each one has a position with an associated quality and class. The type of feature could be modulation, vertical edge etc. Cues also have a position, quality and identity, but

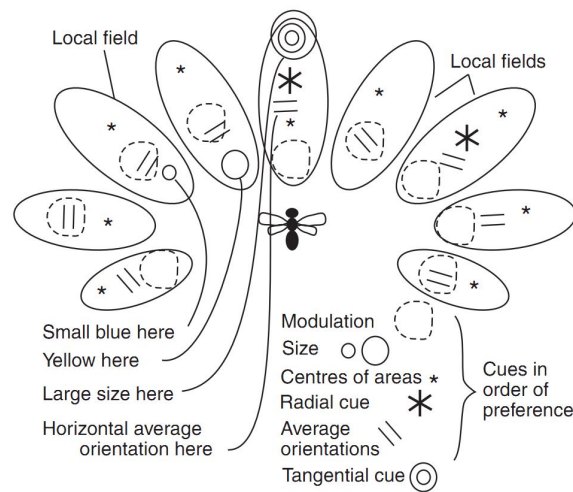


Figure 11: This image shows a representation of the panorama that the bee would detect around its field of view. It is divided into local areas of about  $30^\circ$ . Not more than one cue of each type is detected in each local region. The bee will store the coincidences of cues between each local region in the eye to form landmarks, that allow the bee to recognise familiar locations or targets with a small amount of error [11].

have an added parameter of quantity (sum of receptor responses for each cue). Similarly, landmarks also have a corresponding position and are associated with a particular place the bee needs to remember and is made up of the coincidence of cues. This entire process of how the bee detects landmarks from the receptors is shown in Figure 12. The positions of the features, cues, and landmarks are related to the position of the head and body axis of the bee. Essentially, the bee will look at a scene horizontally while flying and will move its body and head to detect landmarks which will guide it to the goal location.

**Importance of Symmetry** Symmetry as a concept presents itself in a variety of forms in both biological systems, such as flowers, birds, butterflies, humans, and also man-made systems such as architecture, and vehicles. Symmetry has been a design principle for both natural and human design since the beginning of time. Insects use symmetry cues (see Figure 10) as a form of relevance filtering or an attentional cue in their visual system. This allows insects to efficiently filter out any unnecessary visual information and only focus on symmetrical regions that may indicate the presence of a flower, which is vital to honeybees. Furthermore rather than remembering the spatial appearance of landmarks, insects only remember the cues to recognise familiar locations. This allows honeybees to efficiently process the visual information.

Based on this, a bioinspired approach of using pattern matching and relevance filtering of any symmetries was used as the foundation for work conducted in Chapter 3. It should also be noted that the concept of symmetry itself is intrinsically dynamic. In the case of rotational symmetry in images, when part of a symmetric object is moved about the symmetry axis,

the object will repeat itself for each axis of symmetry. This concept that there are both static and dynamic aspects of symmetry is explored in Chapter 4.

### 1.2.2 Dynamic Perception in Insects

**Detection of Local Motion in Insect Eyes** The primary visual cue for insect navigation and control is widely recognised to be the due to optic flow, but the details of mechanisms of the neurons that detect the optic flow inside the brain are not fully understood. Honeybees, being commercially important insects, have been studied to show that they rely on optical flow for landing [13], visual odometry [14], and obstacle avoidance [15]. The best-known neuron model for detecting optic flow is known as the correlation type elementary motion detectors (EMDs). These EMDs correlate changes in intensity between neighbouring ommatidia. This model was first proposed in [16], to account for the optomotor response in the insect which was observed experientially. The optomotor response is linked to the EMDs (optic flow) to help stabilise the insect in response to the changes to an apparent motion in the environment.

The way that correlation EMDs work is based on a multiplication of the two input signals that the EMD receives from two adjacent ommatidia (photoreceptor). See Figure 13. One of the signals that enters the multiplication unit is slightly delayed using a first order low pass temporal filter. The other signal is unchanged. This results in the multiplication unit in the EMD having a preferred direction, dependant on which receptor has the delay. However, by joining two EMDs together that each has an opposite directional sensitivity, they can act together as a bi-directional EMD, where the high or low responses from both are integrated to produce an output. This is a popular model that is successful in emulating the electrophysiological response a due to the visual motion of these EMD tangential cells. In particular the Hassenstein-Reichardt EMD (shown in Figure 13) is the most famous type of correlation-based motion detector.

An important aspect of an EMD is that both the responses are processed through band-pass filters to remove any unsteady illumination (does not contain any motion information). One of these responses is then processed through another first order low-pass filter which exploits the phase lag in this type of filter causing a slight delay. This delayed response is now correlated with a response from a neighbouring receptor that has not been delayed (via a multiplier). This means that as an image moves across the eye, the EMD will produce a strong response as it passes the receptor that has a delay applied to the signal, and then over the receptor with no delay.

The response of the EMD is strongest when the time between the two signals from each receptor is the same as the time delay of the EMD. So this means that in addition to an EMD having a preferred direction, it also has a preferred velocity (dictated by EMD delay), if the image moves faster or slower than this time delay, the response will be weaker. The actual units that cause the delay and correlate the signals are only mildly directionally selective. After the correlation of the two photoreceptor responses, the outputs are then subtracted from one another. This final output is very directionally selective. Using this correlation type EMD model, the final value is positive for motion towards the left and negative for

motion towards to the right.

An important aspect of EMDs is their directional selectivity. This selectivity depends on the spatial arrangement of their sampling bases in the eye as shown in Figure 14. The optical axis (sampling stations) of the ommatidia in the compound eyes are arranged in an orderly hexagonal lattice. Since EMDs receive input from neighbouring sampling stations, there are three directional types with preferred directions. The entire assembly of EMDs is duplicated so that there is a pair of detectors for each of the three directions. The members of each pair have opposite preferred directions. Further to this, studies have shown that EMDs with small sampling bases are dominant in the motion detection system of the fly only in good lighting situations. Under darker situations, there are contributions of additional EMDs with sampling bases up to eight times the interommatidial angle<sup>1</sup> [18]. Hence under scotopic conditions, the sensitivity of the motion detection system is increased at the cost of visual acuity.

Studies have shown that insects use large-field motion for course stabilisation and small field motion for finer tasks like fixation and tracking of objects [18]. For large field motion, a specific combination of detector fields is used to control the activity of particular sub-units of the motor system. The predominant preferred directions of the detector field combinations are generally horizontal and vertical with respect to the eye or head axis. This corresponds to typical flight manoeuvres of flying insects.

**Matched Filters in Insects** Holger Krapp *et al* also proposed an interpretation of the function of LPTCs (see Section 1.1.2) by postulating that the preferred directions are a manifestation of neural versions of matched filters [19]. The derivation of the proposed Franz-Krapp filter [9] is described in Section 1.2.2. Before looking at the derivation, it is worthwhile to examine how matched filters have been employed by biologists, most notably in the influential work by Wehner [20].

The matched filter concept is a well-established topic and was originally proposed in the field of signal processing (detection theory) as discussed later in Section 2.2. The concept of matched filters was later applied in biological systems by Wehner [20]. By correlating a known signal, or pattern (know what you are looking for) with an unknown signal one can find the pattern signal in the presence of noise. A matched filter is the optimal linear filter for maximising the signal to noise ratio (SNR) in the presence of additive stochastic noise as defined in [21].

Wehner used the notion of matched filters as a metaphor, even employing inverted commas (the title of his paper is '*Matched filters*' – *neural models of the external world*) but he offered no mathematical derivations of the filters corresponding to the biological examples he considered. However, his qualitative remarks are insightful and important, because they offer a link between the remarkable efficiency of animal sensor-motor function and a well-understood engineering paradigm.

Wehner emphasised that animals possess highly specialised sensor-motor solutions to motion problems. This is quite different to many engineering systems that are broadly applicable. Indeed, the problems faced by animals are often highly specific and thus admit

---

<sup>1</sup>The interommatidial angle is the angle between each receptor in the insect eye.

one-off solutions where specialisation of the solution entails approximations, short-cuts and simple tricks. Hence, it is important to appreciate not only the utility but also the limitations of such solutions, because animal's specialisation is often restricted to a narrow range of stimuli and situations. Moreover, sensory maps (neural models of the world) are not what we think of as photographs projected onto a special inner screen inside the neurons, but are devices shaped by a special selection of pressures to pre-process sensory information in a way readily translatable into the necessary motor commands.

**Franz-Krapp Filter** While Wehner's examples were qualitative and his invocation of the notion of matched filter was metaphorical, Franz and Krapp [9] assigned a more technical meaning to their filter and expressed it mathematically.

Because the field structure of certain VS and HS tangential cells revealed similarities to some characteristic optic flow vector fields, they proposed that these LPTCs might be extracting certain types of self-motion that result from flight behaviour from the optic flow fields. They interpreted each LPTC as a piecewise linear detector with multiple inputs (the outputs of the EMDs) and one output (the response of the LPTC). The inputs from vectors of the optic flow observed by the spherical eye was compared with a prescribed vector field pattern. This comparison was performed by projecting each input vector on the corresponding vector of the pattern. This is shown in Figure 15. Then a weighted sum of the resulting projections was formed to produce a scalar output.

The constant (non-adaptive) weights were optimised to minimise a constrained quadratic error in the presence of noise and variability of the distance distribution from scene to scene. In order to perform these calculations, knowledge about the EMD noise, self-motion, and distance statistics of the environment was assumed, shown in Figure 16.

It was also noted that a few of the VS neurons are optimised for detecting the direction of rotational optic flow about particular axes, but the neurons are unable to measure the rotational rate. More details including the derivation of the Franz-Krapp Filter is discussed in Appendix B, which relies on an understanding of optic flow discussed in 2.5, with spherical optic flow discussed in Appendix A.

Finally, it should be mentioned that a standard theory of matched filtering was also used in the context of insect vision by Srinivasan *et al* [22]. They investigated whether spatio-temporal filters which detect moving edges or blobs would represent the function of certain neurons in the lamina of the fly brain. The biological evidence for moving edges/blobs detection by the neurons was strong, but not conclusive. For the purposes of the discussion here, the most important aspect was that Srinivasan *et al* did use the standard theory of matched filtering, thus being consistent with both Wehner's metaphor and the engineering understanding of the matched filter paradigm.

**Importance of Motion** We live in a dynamic world so motion is a very important aspect in any visual system. The insect vision system through the EMDs is able to detect motion which is again processed deeper inside the insect's brain. As with static perception, relevance filtering is used, where each EMD response is strongest when the motion corresponds to its preferred direction and speed. Neurons further down the visual system process these

local motion measurements to recognise global motion patterns using a form of matched filtering. By combining both the insect visual system uses the local responses for fixation, or tracking, and large field motions for stabilisation.

Due to the small size of flying insects and current technology, it is relatively difficult to fully understand how the motion fields/patterns are generated. However, in terms of digital applications, optic flow techniques are often employed. These are introduced in Section 2.5, and further discussed in Appendix A. However, other biological studies suggest that many animals employ spatio-temporal filtering as a form of optic flow to detect motion. This is discussed in Section 2.6. Using this bioinspired approach, the problem of self-motion detection is posed as a pattern matching problem in Section 4.4.



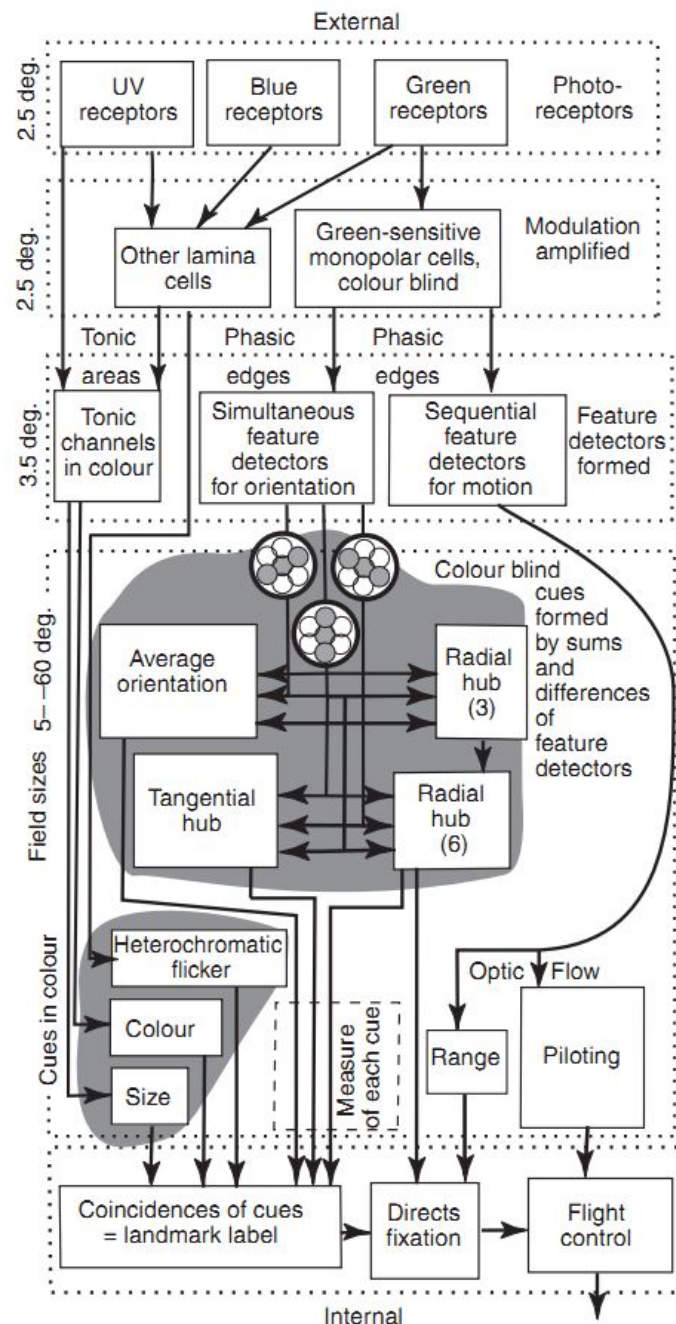


Figure 12: A map of how interactions between different processing elements inside the brain are used to form landmarks labels in each local region of the eye [11]. The entire mechanism of this process is summarised in this diagram, showing how the responses from the receptors are fed through to the lamina to form feature detectors. The responses from the feature detectors are then summed to form different types of cues. A landmark is then formed based on the relative coincidences of the cues. Each local region in the eye is usually used for one task because a single region does not contain enough information to distinguish between all patterns. The approximate fields of view are shown on the left of the diagram.

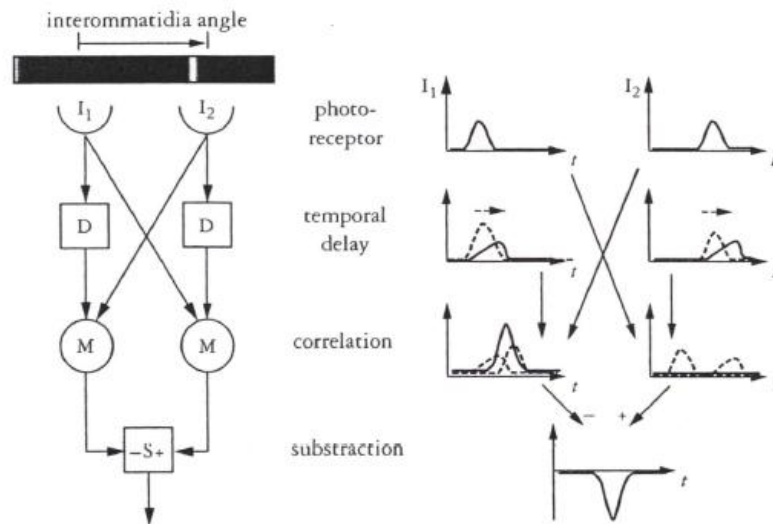


Figure 13: The correlation-type elementary motion detector [17].

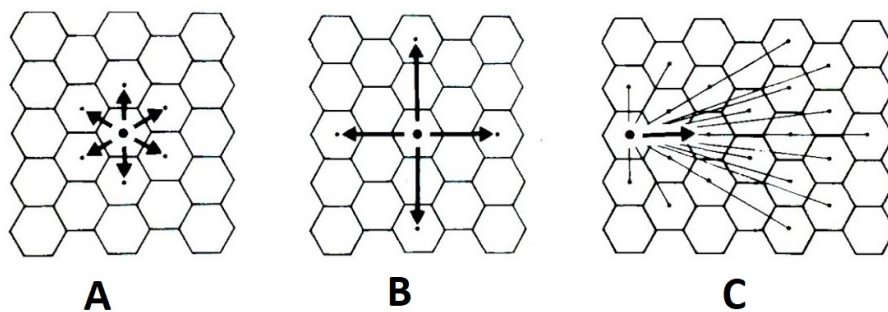


Figure 14: Elementary motion detectors in the visual system of the fly [18].

**A** EMDs of *Drosophila*. The arrangement of the sampling bases of six different directional types of EMDs is shown with respect to the ommatidial mosaic of the eye. Preferred directions are shown by arrows.

**B** EMDs of *Musca* and *Calliphora* with enlarged sampling bases responding selectively to horizontal and vertical motion.

**C** Map of sampling the bases of EMDs of *Calliphora* contributing to the detection of horizontal motion under scotopic (low levels of light) conditions. Only EMDs with the largest contributions to the overall output is shown.

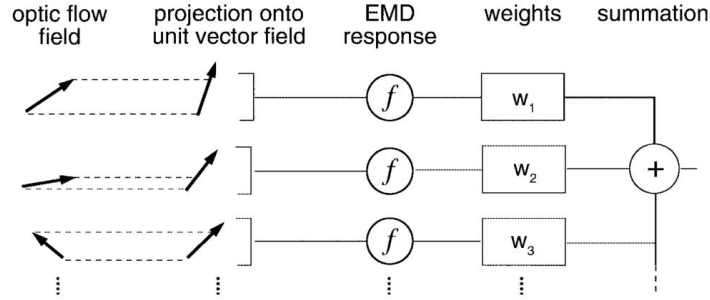


Figure 15: The optic flow vectors (that are noisy) are projected onto a unit vector field ( a pattern that corresponds to a specific LTPC). The response characteristic  $f$  of the EMD distorts the projection, is these distortions are weighted according to local motion sensitivities [9].

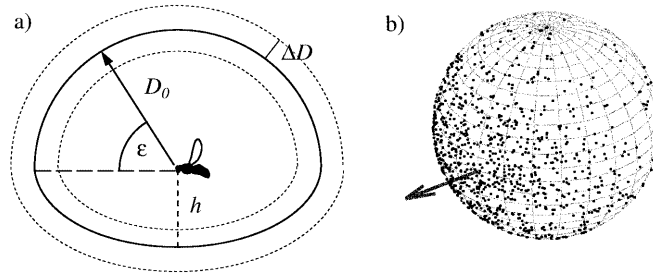


Figure 16: Simplified ‘world model’ of blowfly’s environment with the insect flying at an average height  $h$  over ground [9]. a) Anisotropic distribution of the average distances in the visual field. The mean distance deviation  $\Delta D$  is assumed to be independent of the viewing direction. b) One thousand samples generated by two-dimensional von Mises distribution of the translational directions. The forward direction is indicated by the arrow, and it also represents the distribution maxima.

## 2 Engineering Background

Now that the biological aspects have been studied, this Chapter covers all of the engineering background which provides the foundations of new work presented in this thesis. As discussed in Chapter 1, symmetry is an important concept for insect vision, so the current state of the art research in this area is presented. Matched filtering and its extension is also presented. Other engineering background information is presented here which is referenced later in this thesis such as ROC curve analysis, and optic flow. This chapter is presented as follows:

- Section 2.1 provides an overview on the state of the art symmetry detection method.
- Section 2.2 introduces Matched Filters. The notion of matched filtering in insect vision is discussed in Section 1.2.2
- Section 2.3 extends the principle of Matched Filters to Composite Correlation Filters for better distortion and noise tolerance.
- Section 2.4 provides a brief overview of ROC curve analysis, this is the main performance measure used in this thesis.
- Section 2.5 provides background information on dense optic flow methods along with the main difficulties in solving.
- Section 2.6 introduces the concept of Spatio-temporal filtering, which itself is a bio-inspired approach to motion detection.

### 2.1 Loy-Eklundh Symmetry Detector

Imaging-based automatic analysis of regular objects is a fundamental problem in Machine Vision [23] and such analysis often entails detection of symmetry, defined as object invariance to a rigid-motion transformation [24], [25]. There has been an increasing interest lately in the practical applications and the theoretical challenges of symmetry in image processing [26], [27]. In particular, recent applications of symmetry detection include: engineering analysis with CAD/CAE [28], cytology [29], astronomy [30] and urban planning [31]. The practical significance of considering hierarchies of symmetries, i.e., a nested approach to symmetry, has also been recognised [32].

The increased interest in the symmetry detection problem has led to the development of several relevant image processing algorithms. A comparative evaluation of the algorithms specialised for rotational symmetry detection has been given in [33] with focus on two approaches: 1) the Prasad-Davis approach of [34] based on diffusion and employing gradient-vector flow, and 2) the Loy-Eklundh approach of [35] based on points of interest and employing SIFT [36]. Both the Prasad-Davis and the Loy-Eklundh approaches offer moderate detection performance which is attained with an appreciable computational effort. Other approaches to rotational symmetry detection include the use of the pseudo-polar Fourier transform [37], quadratic binary optimisation [38] and graph matching [39]. These approaches

have computational requirements and performance characteristics similar those ones compared in [33].

In this section, the Loy-Eklundh approach is studied in more details as it is later used to compare with the new approach presented in this thesis (see Section 3.4).

**Detecting and Describing Feature Points** The first step is to detect feature points in a given image. Loy and Eklundh used the SIFT feature point detector which is able to detect distinctive rotation invariant feature points with good repeatability. In the implementation developed in this thesis, the SURF detector was used which outputs similar feature vectors but is significantly more efficient. Since the detector looks at points on the same image, a scale invariant feature detector does not necessarily need to be used since scale does not change in the same image.

For each detected feature point, a vector  $\mathbf{p}_i$  is constructed which stores the  $x, y$  coordinates of them point, the angle  $\phi$  and scale  $s$ , where  $\mathbf{p}_i = (x_i, y_i, \phi_i, s_i)$ . The orientation of this vector is the key value used when determining symmetry.

For each feature point, a descriptor  $\mathbf{k}_i$  is constructed. This descriptor characterises the region around each point. In the case of the SIFT or SURF detectors, the descriptor is extracted around the area after being aligned with the given feature orientation. This descriptor is stored as a  $1 \times n$  vector. In the case of the SIFT descriptor, it is a 128-element vector.

**Detecting Bilateral (Mirror) Symmetry** The concept behind the bilateral symmetry detector is based on trying to see if you can match a given point descriptor,  $\mathbf{k}_i$  to another point in the image that represents a reflection of that point in terms of the mirror descriptor,  $\mathbf{m}_i$ . There are two ways to do this. The first and most efficient method would be to directly modify the descriptors to obtain the equivalent mirror descriptor. An alternative would be to simply mirror the image (the mirror axis does not matter as the descriptor is normalised with respect to the feature orientation) and calculate the descriptor for the corresponding mirror point,  $\mathbf{m}_i$ . Each point with a descriptor  $\mathbf{k}_i$  and its corresponding mirror descriptor,  $\mathbf{m}_i$  are matched to other points form pairs of potentially symmetric features,  $(\mathbf{p}_i, \mathbf{p}_j)$ . For each potential symmetry pair, the symmetry is calculated based upon the orientations of the points, and also the location and scale. The orientation symmetry weighting,  $A_{ij}$  that Loy and Eklundh use is actually adapted from that of Reissfeld [40]. This weighting is given by,

$$A_{ij} = 1 - \cos(\phi_i + \phi_j - 2\theta_{ij}), \quad (1)$$

where  $\theta$  is defined in Figure 17.

The scale weighting  $S_{ij}$  is also computed for the pair of points based on the scales and is defined as

$$S_{ij} = \exp \left( \frac{-|s_i - s_j|}{\sigma_s(s_i + s_j)} \right)^2, \quad (2)$$

where  $\sigma_s$  defines the scale variation accepted. In testing this was set to 1 as used by Loy and Eklundh. The next step, although optional, is to apply a Gaussian distance weighting

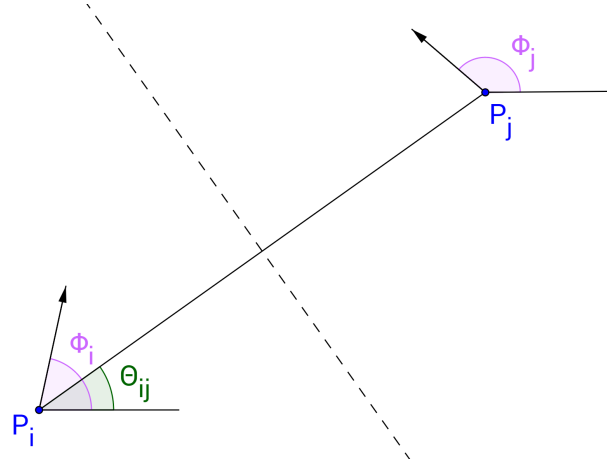


Figure 17: A pair of point vectors  $p_i, p_j$  being compared for mirror symmetry where  $\phi$  is the descriptor orientation, adapted from [35].

function  $D_{ij}$ . This essentially adds extra weighting to matched symmetric pairs that are close to the symmetry axis and is given by

$$D_{ij} = \exp \left( \frac{-d^2}{2\sigma_s^2} \right)^2, \quad (3)$$

where  $d$  is the distance between the matched pair of points and is used to specify the maximum distance between symmetry points. Loy and Eklundh set this parameter to 1 so that there was no constraint on distance.

The next step of the bilateral mirror detector is to calculate the symmetry magnitude  $M_{ij}$  of each pair which is defined as

$$M_{ij} = \begin{cases} A_{ij} S_{ij} D_{ij} & \text{if } A_{ij} > 0 \\ 0 & \text{otherwise} \end{cases} \quad (4)$$

Now that the symmetry magnitude for each symmetrical pair has been calculated, the next step is to determine the dominant symmetries for the given image. This is achieved by accumulating the symmetry magnitudes into a voting space. The axis of symmetry between two matched points is perpendicular to the midpoint of the line that joins the two points ( $\mathbf{p}_i$  and  $\mathbf{p}_j$ ). This is shown in Figure 17. This potential symmetry axis between the two matched points can be represented using polar coordinates ( $r\theta$ ), defined as

$$r_{ij} = x_c \cos \theta_{ij} + y_c \sin \theta_{ij}, \quad (5)$$

where the image centre co-ordinates of the symmetry axis on the midpoint of the line that joins the matched pair is  $(x_c, y_c)$ . The angle of the matched pair line relative to the  $x$ -axis is represented by  $\theta_{ij}$ .

In order to find the major symmetry axis of the given image, the final step is to apply a linear Hough transform. Here, each potential symmetric pair ( $\mathbf{p}_i, \mathbf{p}_j$ ) is weighted based on

the symmetry magnitude  $M_{ij}$  of the point. This weighting represents a vote  $(r_{ij}, \theta_{ij})$  in the Hough voting space. Before the maxima is found, the Hough voting space is first blurred by a Gaussian to smooth the results. The maxima indicates the dominant axis of symmetry. The feature points in areas around the maxima are the points that appear symmetric

**Detecting Rotational Symmetry** Detecting rotational symmetry does not rely on having mirrored descriptors, but is rather based on matching similar features at certain angles (around an arc) in a single image. This means that features,  $\mathbf{k}_i$ , are matched against themselves in the current image. The orientations of each matched pair of features  $(\mathbf{p}_i, \mathbf{p}_j)$  in the image are considered. If the orientations are the same (parallel) then that pair of features will not contribute towards rotational symmetry. However, if they are not the same then there is a chance that they have rotational symmetry about a certain point in the image. This centre of rotation point  $c_{ij}$  is located at some distance  $r$  from the feature points  $(\mathbf{p}_i, \mathbf{p}_j)$ . If the feature point  $\mathbf{p}_i$  was rotated around  $c_{ij}$ , at a certain angle, it would line up with the other matched feature point  $\mathbf{p}_j$  exactly. This idea is shown in Figure 18. The centre of rotation is given by

$$\begin{aligned} c_{ij} = & \begin{pmatrix} x_i \\ y_i \end{pmatrix} + r \begin{pmatrix} \cos(\beta + \gamma) \\ \sin(\beta + \gamma) \end{pmatrix}, \end{aligned} \quad (6)$$

where  $x_i$  and  $y_i$  are the Cartesian coordinates of  $\mathbf{p}_i$ . The angle of the line joining  $\mathbf{p}_i$  with  $\mathbf{p}_j$  is  $\gamma$ .

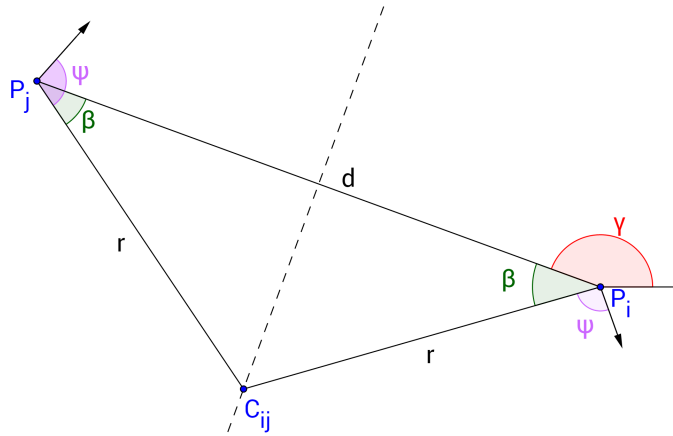


Figure 18: A pair of point vectors  $p_i, p_j$  being compared for rotational symmetry, where the centre of rotation is  $c_{ij}$ , adapted from [35].

Using Pythagoras' theorem, the radius  $r$  to the centre of rotation can be found by

$$\begin{aligned} r^2 &= \left(\frac{d}{2}\right)^2 + \left(\frac{d}{2} \tan \beta\right)^2 \\ r &= \frac{d \sqrt{1 + \tan^2 \beta}}{2}, \end{aligned} \quad (7)$$

Now that the rotation centre of each potential symmetric pair has been found, the rotational symmetry magnitude  $R_{ij}$  must be calculated based on the scale weighting  $S_{ij}$  and the distance weighting  $D_{ij}$  previously defined so that

$$R_{ij} = S_{ij} D_{ij}. \quad (8)$$

Given a collection of potentially symmetric points with a symmetry magnitude, the dominant rotational symmetric points must be found. This is done by combining the rotation centre points  $c_{ij}$  in a voting space where each point is weighted based on its symmetry magnitude  $R_{ij}$  where the maxima would represent the strongest rotational symmetry centres. Again the voting space image is blurred with a Gaussian to reduce noise. Just like with the mirror symmetry detection, the feature points in a neighbourhood around the maxima are the feature points that are associated to that rotational symmetry.

## 2.2 Matched Filters

A matched filter is designed to find a known signal in a measured signal for the measurement corrupted by noise. The matched-filter design [41], [42] is based on maximising the signal-to-noise ratio (SNR) for additive Gaussian white noise; more complex noise models can also be handled [21], [43]. For images, the required SNR maximisation means maximisation of the correlation between the template image (known signal) and the input image (measured signal).

An input image  $s$  is defined by an  $d_1 \times d_2$  array whose entries  $s(m, n)$  are pixel values. The input  $s$  is to be tested for the presence of a known template given by a  $w_1 \times w_2$  array  $h$  with entries  $h(m, n)$ . The template  $h$  may be smaller than the input image,  $w_1 \leq d_1$  and  $w_2 \leq d_2$ , because the template is often present in a subregion, or window, of the input image. Correlation  $g$  of the input image  $s$  with the template image  $h$  is performed by examining all  $w_1 \times w_2$  windows of the  $d_1 \times d_2$  input image  $s$  or a “sliding-window” process. The correlation  $g(m, n)$  between the window and the template  $h(m, n)$  is computed for each pixel  $(m, n)$ , creating the correlation plane or the set  $\{g(m, n) \mid 1 \leq m \leq d_1 \text{ and } 1 \leq n \leq d_2\}$ . See Figure 22. If the correlation value  $g(m, n)$  is above a predefined threshold, then detection of the template  $h(m, n)$  in the  $(m, n)$ -indexed window is declared.

The correlation plane can be obtained from:

$$g(m, n) = \overbrace{s(m, n) \otimes h(m, n)}^{\text{Correlation with } h(m, n)} \quad (9)$$

$$= \sum_{k=1}^{w_1} \sum_{l=1}^{w_2} s(m + k - \lfloor w_1/2 \rfloor, n + l - \lfloor w_2/2 \rfloor) \times h(k, l)$$

$$= \overbrace{s(m, n) \star h(-m, -n)}^{\text{Convolution with } h(-m, -n)} \quad (10)$$



$$= \sum_{k=1}^{w_1} \sum_{l=1}^{w_2} s(m-k + \lfloor w_1/2 \rfloor, n-l + \lfloor w_2/2 \rfloor) \times h(-k, -l)$$

$$\text{for all } m \in \{1, \dots, d_1\} \text{ and } n \in \{1, \dots, d_2\} \\ \text{(where } 1 \leq w_1 \leq d_1 \text{ and } 1 \leq w_2 \leq d_2\text{),}$$

but it is more efficient to implement equation (9) by two-dimensional convolution (equation (10)) with the ‘flipped’ template  $h(-m, -n)$  because linear-filtering theory [44] can then be used. According to that theory, equation (10) can be interpreted as a two-dimensional linear filter designed with the impulse response given by  $h(-m, -n)$ . Moreover, the two-dimensional discrete Fourier transform (DFT) can be applied to equation (10) so that correlation filtering (equation (9)) is efficiently performed in the (two-dimensional) frequency domain [44, Chapter 4]. The frequency-domain equivalent of convolution (equation (10)) is the product of the discrete Fourier transforms of the convolution terms. Defining:

$$\begin{aligned} S(k, l) &= \text{DFT}[s(m, n)] \\ H(k, l) &= \text{DFT}[h(m, n)], \end{aligned} \quad (11)$$

where  $k$  and  $l$  are discrete frequencies, it follows that equation (10) can be equivalently expressed as:

$$\begin{aligned} g(m, n) &= s(m, n) \star h(-m, -n) \\ &\Downarrow \\ G(k, l) &= S(k, l) \times H^*(k, l), \end{aligned} \quad (12)$$

where  $H^*(k, l)$  is the complex-conjugate of  $H(k, l)$  (see equation (11)). Replacing the DFT with the fast Fourier transform (FFT) allows an computationally-efficient implementation of the two-dimensional correlation in equation (9); the issue of zero-aliasing can also be addressed [45] through zero padding to avoid circular correlation.

### 2.2.1 Distortion Tolerance

Matched filters are not a natural solution for target detection in images because matched filters are not robust against target distortions (changes in orientation, illumination, or scale). A robust solution would require a bank of matched filters with each filter corresponding to a particular set of parameters describing predefined distortions, resulting in a rather large filter bank (see Figure 19). The use of such filter bank would also necessitate a procedure to decide which of the individual filters best matches the input (test) image. Such a complex filter bank is not practical for use on a small system with limited processing power like an MAV.

The matched filter is designed to produce a maximum SNR, but another consideration is the peak sharpness (peak correlation energy, or PCE) which helps to discriminate the target from the background. A maximal SNR does not usually produce a sharp peak. Typical images contain most of their energy at low frequencies, and obtaining a filter that gives

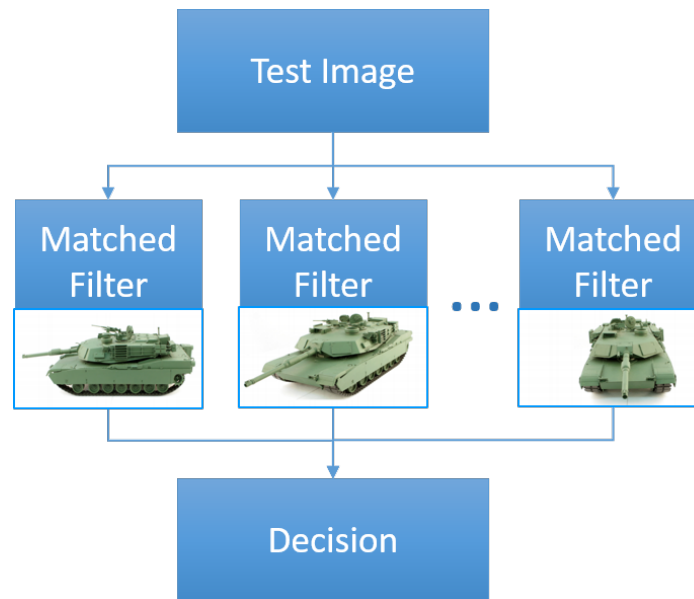


Figure 19: In order to detect a target from an image obtained in realistic conditions, a large bank of matched filters is needed with each matched filter representing only one (of many) possible orientations, illuminations, or scaling of the target.

maximal peak correlation energy (PCE) ends up suppressing the low frequencies and allows the high frequency content which usually contains noise to pass through. Maximising the signal to noise ratio generally suppresses the high frequency region of the image which contains the noise. This is essentially opposite to maximising the peak energy. Maximising both the SNR and PCE are conflicting requirements, so it is not possible to design a single filter that maximises both. However, it is possible to design filters that find the best trade-off between the two parameters, often referred to as composite correlation filters. Some of the key composite correlation filters used in this thesis are discussed in greater detail in Section 2.3.

### 2.2.2 Convolution and Correlation

As discussed the matched filter is implemented using correlation. However, it is also important to understand the difference between correlation and convolution.

- **Convolution:** Many linear image filtering tasks such as image blurring, sharpening, noise removal and edge detection are implemented by convolving a kernel or mask (typically 3x3 pixels) with the image to achieve the desired result (see equation (10)). Image filtering via convolution is a local neighbourhood operation and the result is a new filtered image.
- **Correlation:** On the other hand correlation is used for pattern recognition and is a global operation as it looks for the pattern in the entire image, the size of the correlation mask will depend on the size of the pattern to be found. The result is a correlation map where the peaks indicate regions of greatest similarity between the mask and the image as shown in equation (9).

Both 2D correlation and convolution can become rather computationally expensive when the mask or image becomes large with a computational complexity of  $O(MNmn)$  for an image of size  $(M, N)$  pixels and a mask of size  $(m, n)$  pixels.

For relatively small images or masks it is often more efficient to use separable filters [46], where the computational complexity is reduced to  $O(2MNk)$  where  $k$  is the 1D mask size. This is possible because both convolution and correlation operations have associative properties.

However, for larger images or masks it is far more efficient to perform correlation or convolution in the Fourier domain due to the Fourier convolution theorem. Here convolution becomes a simple multiplication in the Fourier domain. Similarly, correlation becomes a multiplication by the complex conjugate of the mask in the Fourier Domain. Using the Fourier domain for these operations are often far more computationally efficient. However, both the image and the mask will need to be transformed into the Fourier domain. This can be done using the fast Fourier transform (FFT) algorithm. Since the FFT itself has a computational cost of  $O(MN(\log MN))$ , it only becomes more efficient when the image or mask both large. This is because the actual cost of computing the FFT for the mask and image can be more expensive than the direct convolution which has a computational cost of  $O(MNmn)$ .

## 2.3 Composite Correlation Filters

Composite correlation filters are a further generalisation of matched filtering in which a bank of matched filters (see Figure 19) are replaced with a functionally-equivalent single filter. This equivalent filter is designed with a single composite template (replaces a bank of individual templates) and is a correlation filter (simultaneously maximises both SNR and PCE). The composite template combines several versions of the pattern to be detected (Figure 22) so the selection of the training images is an important part of the composite correlation filter design [47].

Correlation filters are typically implemented in either the optical domain, using lenses in an optical correlator and spatial light modulators (SLM), or in the digital domain using computers. Each of these approaches are summarised below.

**Optical Correlation Filters** Optical correlation involves the physical modulation of the image in the Fourier domain. This is done by placing a spherical lens at its focal length away from the image plane. At this point, a scaled version of a 2D Fourier transform is formed. This process means that the Fourier transform is available at the speed of light with no computation. The limiting factor of optical correlations is the speed of the sensing system or photo-detectors which do not operate at the speed of light.

VanderLugt took advantage of this principle in his VanderLugt Correlator [48] as shown in Figure 20. Here the first lens transforms the light of the scene from  $P_1$  and the 2D Fourier transform is captured at the plane  $P_2$ . At plane  $P_2$  a holographic complex transparency image which is the correlation filter in the frequency domain is applied. The light that passes through  $P_2$  is equivalent to applying the correlation between the filter and the test image (see equation (9)). The last reverse lens (again at a focal distance away from the image plane) then transforms the image back into the spatial domain which is captured at plane  $P_3$ , the peaks on this image represent the target location.

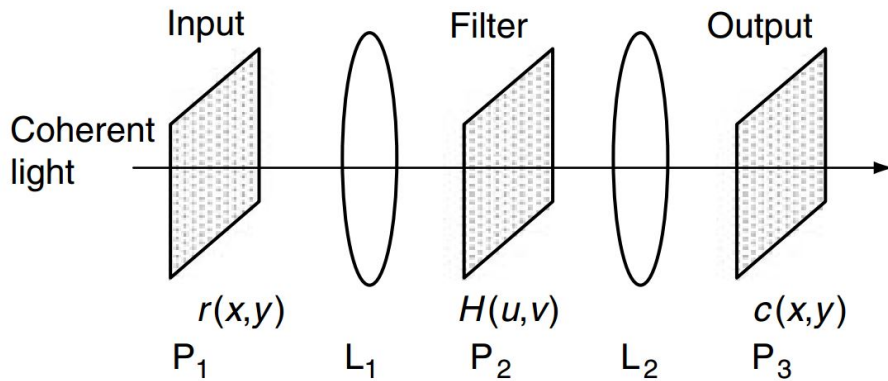


Figure 20: Example schematic of a VanderLugt Correlator [48]

The main advantage of optical correlators is the speed at which they can compute the correlation (at the speed of light). The only limiting factor is the update rate of the SLM and light sensors. This is currently around 1000Hz. However, despite the speed advantages, the cost and size of these devices make them unsuitable for use on-board an MAV. Furthermore, current computational power allows for fast digital implementations of correlation filtering. Because of the size of optical correlators and the fact that there is sufficient computational power available, this PhD project will focus on the digital implementation of correlation filters.

**Digital Correlation Filters** As computation power improved in the 1990s, digital correlation implementation filters became more popular [49]. Digital correlation filtering in-

volves the use of the correlation theorem as discussed previously and shown in equation (12). The main reason for the use of correlation is computational efficiency. The first step with digital correlation is to compute the 2D discrete Fourier Transform of the input image and the filter<sup>2</sup>. The use of the Fast Fourier Transform algorithm is advisable as it speeds up the Fourier Transform computation by several orders of magnitude. Next, multiplying the input image and the conjugate of the filter in the Fourier domain is equivalent to correlation in the spatial domain. The last step is to apply an inverse 2D discrete Fourier transform to get the result in the spatial domain, here the peaks on the correlation output represent the possible target location. It is important to remember to zero pad the two DFT of the target and filter to avoid circular correlation which causes aliasing. More advanced designs have been developed beyond these three and they include: the minimum output sum of squared error (MOSSE) filter, quadratic correlation filters and the maximum margin correlation filter (MMCF). A key feature of the MOSSE filter [50] is that it adapts to real time changes of the object due to motion, scaling, gradual deformation and lighting. Quadratic correlation filters [51] are able to exploit higher order statistics of the data to potentially improve detection performance. The MMCF [52] combines the design principles of linear binary classifiers and correlation filters, outperforming both. Since the focus here is on resource-limited autonomous platforms, these more advanced approaches have not been used as they are more computationally expensive than the standard modern designs based on the maximum average correlation energy (MACE), unconstrained maximum average correlation energy (UMACE), and maximum average correlation height (MACH) filters.

### 2.3.1 MACE composite correlation filter

As previously mentioned the matched filter does not cope well when the target image is distorted, this could be due to rotation, lighting, or scale changes. In order to deal with this problem, one possible approach would be to have many matched filters to capture the various distortions. These would be implemented into a filter bank, where several matched filters are applied to a given test image and the filter with the strongest response is the selected one. However, in order to capture the full range of distortions (rotation, lighting and scale changes) many matched filters will be required for each target and this can be very expensive in terms of computational resources, as several hundred matched filters will be required to capture the distortions for a given object.

In order to address this issue, developments lead to the composite correlation filter. These filters are able to capture the various distortions that a target would undergo in real world situations into a single filter as shown in Figure 21. A simplistic explanation of composite filters would be to think of them as combining many matched filters into a single composite filter. There are several variants of composite correlation filters starting with the first being the synthetic discrimination function filter (SDF) [53]. For each of the training images, the SDF filter is designed to produce a specific value on the correlation plane. This specific value is usually 1 for target or object training images, and 0 for non-target images.

<sup>2</sup>For improved computational efficiency the DFT of the filter can be pre-computed if the same filter is applied to each image in a video sequence.

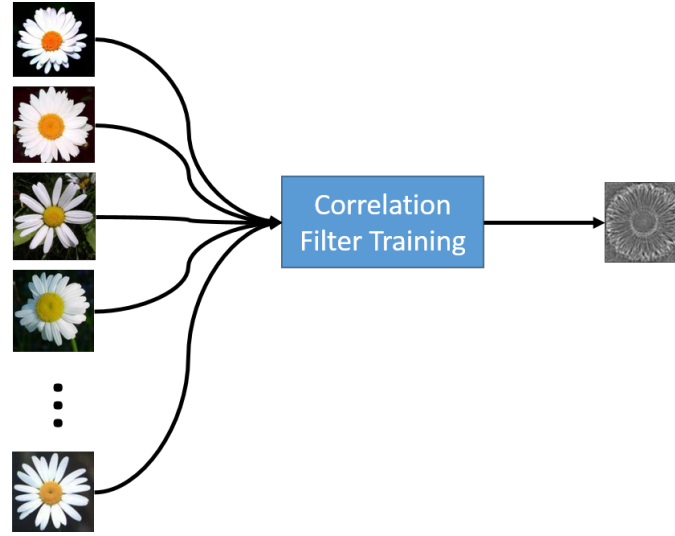


Figure 21: A single composite correlation filter is trained using a collection of training images which represent the object under various distortions due to pose, variation and illumination. This being able to capture all the variations of the training images in a single filter.

The SDF filter was refined to form the MACE composite filter (Minimum Average Correlation Energy) [54]. These composite filters fall into the category of constrained correlation filters where the peak is controlled by hard constraints as shown later in equation (16).

The MACE filter is designed to suppress the side-lobes of the correlation output to form a distinct peak to make the target presence in the test image clear and distinguishable. The MACE filter suppresses the side-lobes by minimising the average correlation energy (ACE) in the correlation plane (see Figure 22) for  $N$  training images  $x_1, x_2, \dots, x_N$  to be combined into a composite template:

$$\text{ACE} = \frac{1}{N} \sum_{i=1}^N \sum_{m=1}^{d_1} \sum_{n=1}^{d_2} |g_i(m, n)|^2, \quad 1 \leq i \leq N, \quad (13)$$

where  $g_i(m, n) = x_i(m, n) \otimes h(m, n)$  is the correlation for the  $i$ th training image, see equation (9), and  $d_1, d_2$  are the dimensions of input images. Here, and in Sections 2.3.2–2.3.3, a full-size window ( $w_1, w_2$ ) is used:  $w_1 = d_1$  and  $w_2 = d_2$ .

In the frequency domain (13) becomes:

$$\begin{aligned} \text{ACE} &= \frac{1}{d \times N} \sum_{i=1}^N \sum_{k=1}^{d_1} \sum_{l=1}^{d_2} |G(k, l)|^2 \quad (\text{with } d = d_1 \times d_2) \\ &= \frac{1}{d \times N} \sum_{i=1}^N \sum_{k=1}^{d_1} \sum_{l=1}^{d_2} |H(k, l)|^2 \times |X_i(k, l)|^2, \end{aligned} \quad (14)$$

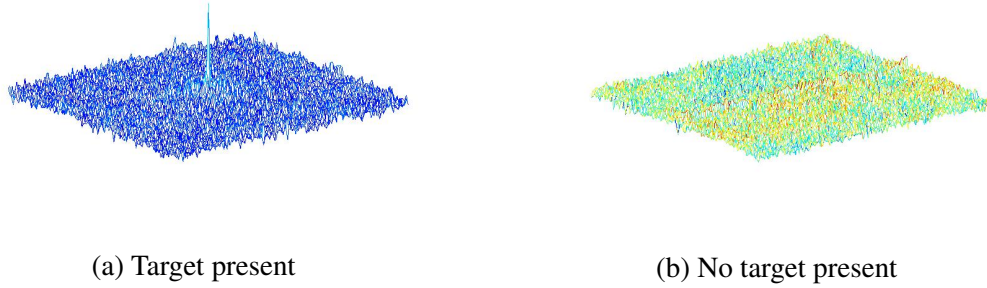


Figure 22: Example of a composite filter that is trained on a collection of training images to produce a sharp peak if the target is present in the test image (a), or no distinct peak if the target is not present in the image (b).

where  $H$  and  $X_i$  are the two-dimensional discrete Fourier transforms of the MACE filter impulse response (still to be determined) and the  $i$ th training image, respectively.

Recalling that  $|H(k, l)|^2 = H(k, l) \times H^*(k, l)$  and also  $|X_i(k, l)|^2 = X_i(k, l) \times X_i^*(k, l)$ , the following conventions are introduced for notational and computational ease. The  $d_1 \times d_2 = d$  terms  $H(k, l)$  in (14) are represented as a  $d \times 1$  column vector  $\mathbf{h}$ . The  $d_1 \times d_2 = d$  terms  $X_i(k, l)$  of  $i$ th training image are expressed as a  $d \times d$  diagonal matrix  $\mathbf{X}_i$ . The  $d$  elements along the main diagonal of  $\mathbf{X}_i$  are terms  $X_i(k, l)$  lexicographically ordered with respect to  $k$  and  $l$ . In this new matrix-vector notation, equation (14) becomes:

$$\begin{aligned}
 \text{ACE} &= \frac{1}{d \times N} \sum_{i=1}^N (\mathbf{h}^* \mathbf{X}_i) (\mathbf{X}_i^* \mathbf{h}) \\
 &= \mathbf{h}^* \underbrace{\left[ \frac{1}{d \times N} \sum_{i=1}^N \mathbf{X}_i \mathbf{X}_i^* \right]}_{\substack{d \times d \text{ diagonal matrix } \mathbf{D} \\ \text{average power spectrum} \\ \text{of all training images}}} \mathbf{h} \\
 &= \mathbf{h}^* \mathbf{D} \mathbf{h},
 \end{aligned} \tag{15}$$

where the average power spectrum of all the training images is represented by a  $d \times d$  diagonal matrix  $\mathbf{D}$ .

The ACE is minimised by applying the constraint

$$\mathbf{X}^* \mathbf{h} = d \times \mathbf{u}, \tag{16}$$

to equation (15), where  $\mathbf{u}$  is a  $1 \times N$  vector containing the desired correlation value for each training image. Finally,  $\mathbf{X}$  is a  $d \times N$  matrix whose columns contain lexicographically-

ordered elements of the discrete Fourier transform of each training image  $X_i$ . An application of the Lagrange multipliers to minimising (15), subject to (16), yields [55] the optimal solution in the form of the discrete Fourier transform of the MACE impulse response:

$$\mathbf{h}_{\text{MACE}} = \mathbf{D}^{-1} \mathbf{X} (\mathbf{X}^* \mathbf{D}^{-1} \mathbf{X})^{-1} \mathbf{u}. \quad (17)$$

Vector  $\mathbf{h}_{\text{MACE}}$  contains  $d$  terms of the filter impulse response in the frequency domain and can be re-arranged into a  $d_1 \times d_2$  matrix with entries  $H_{\text{MACE}}(k, l)$  to be used in equation (12).

### 2.3.2 UMACe composite correlation filter

The Unconstrained Minimum Average Correlation Energy composite filter (UMACE) is a simplified [51] version of the MACE filter which also minimises the average correlation energy defined in equation (15), in a similar way to MACE. Unlike the MACE approach, the UMACe design does not constrain the correlation outputs at the origin to a specific value (see equation (16)) but only maximises the peak height at the origin. Such a constraint-free approach is not only computationally simpler but can exhibit better robustness against noise [56]. However, a disadvantage of the UMACe filter is that the correlation-plane peak may be less sharp (lower peak-to-sidelobe ratio) than in the MACE case because the UMACe design relies only on the frequency-domain average of the training images. See equation (19) below.

Computationally, the UMACe filter requires the average power spectrum  $\mathbf{D}$  of the training images:

$$\mathbf{D} = \frac{1}{d \times N} \sum_{i=1}^N \mathbf{X}_i \mathbf{X}_i^*, \quad (18)$$

and the vector average of the training images in the frequency domain:

$$\bar{\mathbf{x}} = \frac{1}{N} \sum_{i=1}^N \mathbf{x}_i, \quad (19)$$

where each  $\mathbf{x}_i$  is a lexicographically-ordered  $d \times 1$  vector obtained from the DFT of the training image  $x_i$  so that  $\bar{\mathbf{x}}$  effectively is the frequency-domain average of all training images expressed in a vector form. Then the discrete Fourier transform of the MACE impulse response is:

$$\mathbf{h}_{\text{UMACE}} = \mathbf{D}^{-1} \bar{\mathbf{x}}, \quad (20)$$

so that vector  $\mathbf{h}_{\text{UMACE}}$  contains  $d$  elements of the UMACe filter impulse response in the frequency domain. The elements of vector  $\mathbf{h}_{\text{UMACE}}$  can be re-arranged into a  $d_1 \times d_2$  matrix with entries  $H_{\text{UMACE}}(k, l)$  to be used in equation (12).

Comparing equation (20) with equation (17) shows that the UMACe filter is computationally simpler than the MACE filter.



### 2.3.3 Optimal Trade-off MACH composite correlation filter

The Maximum Average Correlation Height (MACH) filter [51], [57], was introduced to achieve a greater degree of noise and distortion tolerance than MACE and UMACE filters. The MACE and UMACE filters are designed only with respect to one performance measure, namely the average correlation energy (ACE). The MACH filter not only minimises ACE, but also minimises the average similarity measure (ASM) and the output noise variance (ONV), while maximising the average correlation height (ACH). An optimal trade-off approach is used in order to best satisfy each of these three additional design criteria simultaneously.

The first optimisation criterion is the output noise variance (ONV) which is a measure of the correlation-plane variance  $\sigma^2$  due to additive noise. The ONV is defined as:

$$\text{ONV} = \sigma^2 = \mathbf{h}^* \mathbf{C} \mathbf{h}, \quad (21)$$

where  $\mathbf{h}$  is a  $d \times 1$  vector as in (15) and  $\mathbf{C}$  is the power spectral density of the additive input noise. Gaussian white noise with unit variance is usually assumed so that  $\mathbf{C} = (1/d)\mathbf{I}$ , where  $\mathbf{I}$  is the identity matrix.

The second optimisation criterion is the average similarity measure (ASM). This is an average measure of the variation (or similarity) across the training images:

$$\begin{aligned} \text{ASM} &= \frac{1}{N} \sum_{i=1}^N \sum_{m=1}^{d_1} \sum_{n=1}^{d_2} |g_i(m, n) - \bar{g}(m, n)|^2 \\ &= \mathbf{h}^* \underbrace{\left( \frac{1}{d \times N} \sum_{i=1}^N (\mathbf{X}_i - \bar{\mathbf{X}})(\mathbf{X}_i - \bar{\mathbf{X}})^* \right)}_{d \times d \text{ diagonal similarity matrix } \mathbf{S}} \mathbf{h} \\ &= \mathbf{h}^* \mathbf{S} \mathbf{h}, \end{aligned} \quad (22)$$

where  $\bar{g}(m, n) = (1/N) \sum_{i=1}^N g_i(m, n)$  is the average across the correlation plane,  $\mathbf{h}$  and  $\mathbf{X}_i$  have previously been defined in equation (15), and  $\bar{\mathbf{X}} = (1/N) \sum_{i=1}^N \mathbf{X}_i$  is the frequency-domain average training image expressed as a  $d \times d$  diagonal matrix. See also equation (19).

The third optimisation criterion is the average correlation height (ACH) whose maximisation ensures that the filter produces a strong peak (on average) for all the training images but without imposing a constraint as in the MACE filter design, (see equation (16)). The ACH is given by:

$$\text{ACH} = \frac{1}{N} \sum_{i=1}^N \mathbf{x}_i^* \mathbf{h} = \bar{\mathbf{x}}^* \mathbf{h}, \quad (23)$$

where  $\bar{\mathbf{x}}^*$  is the conjugate transpose of the average training image  $\bar{\mathbf{x}}$  defined in equation (19).

Using equation (15) and equations (21)–(23), the combined performance criterion to be minimised is:

$$\begin{aligned} E(\mathbf{h}) &= \alpha \cdot \text{ONV} + \beta \cdot \text{ACE} + \gamma \cdot \text{ASM} - \delta \cdot \text{ACH} \\ &= \alpha \cdot \mathbf{h}^* \mathbf{C} \mathbf{h} + \beta \cdot \mathbf{h}^* \mathbf{D} \mathbf{h} + \gamma \cdot \mathbf{h}^* \mathbf{S} \mathbf{h} - \delta \cdot \bar{\mathbf{x}}^* \mathbf{h}, \end{aligned} \quad (24)$$

where user-provided design weights  $\alpha$ ,  $\beta$ ,  $\gamma$  and  $\delta$  are non-negative and normalised according to:

$$\alpha^2 + \beta^2 + \gamma^2 + \delta^2 = 1 \quad (25)$$

so that systematic trade-offs can be made for the four performance measures. Since (25) means that not all weights are independent, minimisation of (24) results in a solution involving three of the weights and the standard choice is:

$$\mathbf{h}_{\text{MACH}} = (\alpha \mathbf{C} + \beta \mathbf{D} + \gamma \mathbf{S})^{-1} \bar{\mathbf{x}}. \quad (26)$$

Vector  $\mathbf{h}_{\text{MACH}}$  has  $d$  elements of the filter impulse response in the frequency domain which can be re-arranged into a  $d_1 \times d_2$  matrix with entries  $H_{\text{MACH}}(k, l)$  to be used in (12).

Choosing specific values for  $\alpha$ ,  $\beta$  and  $\gamma$  in (26) means tuning the MACH filter to provide the desired balance of noise tolerance, peak sharpness and distortion [51], [58]. For example, a MACH filter can behave like a MACE filter (maximise peak sharpness but sensitive to distortion) if  $\alpha$  and  $\gamma$  are set to 0.

### 2.3.4 3D MACH Filters

In order to utilise a pattern based recognition approach to dynamic scenes (as used later in Chapter 4), it is useful to extend the correlation filters to three dimensions. This enables the use of pattern matching within a video sequence. Training a 3D correlation filter is very similar to that of the 2D training process which is extended to three dimensions. The main difference is the initial processing steps of converting the spatial 3D spatio-temporal volume into the frequency domain as discussed in [59]. Rather than training on image data, video data is used. Each training video sequence is stored as a spatio-temporal volume. Because the MACH correlation filter is trained in the Fourier domain, a three dimensional fast Fourier transform (3D FFT) is used:

$$F(u, v, w) = \sum_{t=0}^{N-1} \sum_{y=0}^{M-1} \sum_{x=0}^{L-1} f(x, y, t) \exp \left( -j2\pi \left( \frac{ux}{L} + \frac{vy}{M} + \frac{wt}{N} \right) \right) \quad (27)$$

where  $f(x, y, t)$  is the spatio-temporal volume in the spatial domain,  $F(u, v, w)$  is the same spatio-temporal volume represented in the Fourier domain.  $L$ ,  $M$  is the number of columns and rows respectively in pixels.  $N$  is the number of frames or slices in the video sequence.

The separability of the Fourier transform is utilised to greatly increase efficiency of this operation by computing the 1D FFT of each row for every slice in the volume  $h(x, y, t)$ , along the  $x$ ,  $y$ , and  $t$  axis sequentially as given in equations (28)–(30).

$$F(u, y, t) = \sum_{x=0}^{L-1} f(x, y, t) \exp \left( -j 2\pi \left( \frac{ux}{L} \right) \right) \quad (28)$$

$$F(u, v, t) = \sum_{y=0}^{M-1} F(u, y, t) \exp \left( -j 2\pi \left( \frac{vy}{M} \right) \right) \quad (29)$$

$$F(u, v, w) = \sum_{t=0}^{N-1} F(u, v, t) \exp \left( -j 2\pi \left( \frac{wt}{N} \right) \right) \quad (30)$$

Each scalar video volume is Lexicographically ordered into  $\mathbf{x}_i$  1D column vectors of size  $1 \times d$  where  $d = L \times M \times N$ . The number of training video sequences is represented by subscript  $i^3$ . Now that the column vector representing each training sequence volume has been obtained, this is the same input as with the 2D correlation filter, (all be it a bigger 1D column vector). Equation (26) is used to build the MACH filter, minimising the average similarity measure, average correlation energy and the output noise variance while simultaneously maximising the average correlation height. The final step is to transform the 1D MACH vector back into a volume to form  $h_{\text{MACH3D}}$ .

**Tuning Parameters of MACH** As discussed in Section 2.3.3, there are a few performance metrics that can be tuned during the training phase. In the case of self-motion estimation, it is more important to classify the type of motion. The  $\alpha$  (ONV),  $\gamma$  (ASM) coefficients were tuned to be higher to make the filters more tolerant to noise and distortion. This is desirable in this case due to the variation of pixel intensity across varying scenes. The primary interest is in the patterns formed along the  $(y, t)$ -plane and  $(x, t)$ -plane. Lowering these two coefficients would give the filter a greater discrimination ability (causing the filter to behave more like a MACE filter). This would be more useful in object classification tasks such as recognising a specific type of flower opposed to if an object is a flower or not.

Once a 3D MACH filter for a specific motion has been trained, it can be used to detect the same motion in video sequence via a 3D correlation function as shown.

$$g(x, y, t) = \sum_{n=0}^{N-1} \sum_{m=0}^{M-1} \sum_{l=0}^{L-1} s(x+l, y+m, t+n) h(l, m, n) \quad (31)$$

where  $s$ , is the test image. The MACH filter  $h$  is of size  $L$  columns,  $M$  rows, and  $N$  slices. The range of the 3D sliding window along each axis is  $x = 0, 1, \dots, X-L$ ,  $y = 0, 1, \dots, Y-M$ , and  $t = 0, 1, \dots, T-N$ . This results in a correlation plane output  $c$  of size  $(X-L+1) \times (Y-M+1) \times (T-N+1)$  which assumes that the motion is contained fully within the test video sequence.

The 2D correlation theorem for discrete Fourier transforms [60] can still be exploited for increased efficiency. By extending to the 3D case:

<sup>3</sup>All the training video volumes must be all the same size in terms of  $L, M, N$

$$g(x, y, t) = \mathcal{F}^{-1} \left[ \mathcal{F}[s(x, y, t)] \mathcal{F}^*[h(l, m, n)] \right], \quad (32)$$

where  $\mathcal{F}$  represents the 3D discrete Fourier transform (DFT) as defined in equation (27) and  $\mathcal{F}^{-1}$  is the inverse discrete Fourier transform (IDFT). The complex conjugate is denoted as superscript  $*$ .

To avoid noise and changing lighting conditions between the training and testing data interfering with the correlation output, the correlation output is normalised.

$$g_{\text{norm}}(x, y, t) = \frac{g(x, y, t)}{\sqrt{E_h E_s(x, y, t)}} \quad (33)$$

where  $g(x, y, t)$  is defined in equation (31). The energy scalar of the 3D MACH filter,  $E_h$ , is defined in equation (34). The energy scalar of the test images,  $E_s$  is defined in equation (35).

$$E_h = \sum_{n=0}^{N-1} \sum_{m=0}^{M-1} \sum_{l=0}^{L-1} h^2(l, m, n) \quad (34)$$

$$E_s(x, y, t) = \sum_{n=0}^{N-1} \sum_{m=0}^{M-1} \sum_{l=0}^{L-1} s^2(x + l, y + m, t + n) \quad (35)$$

This normalises the range of correlation response to be between 0 and 1. The existence of the motion is detected in a test sequence if the peak value is above a threshold. Furthermore, the correlation value at each peak also provides the certainty of the detection, with stronger peaks being of a higher certainty. This value can be useful when combining two or more motion detectors together. The threshold for each 3D MACH filter can be automatically calculated during the training process by finding the minimum peak value when correlating the 3D MACH filter with each of the training sequences whilst returning no false positives.

## 2.4 ROC curves

It is important to first define some performance measures in order to assess the performance of matched filters, and other object classification algorithms. The Receiver operating characteristic (ROC) is a very useful performance measure that can be used to compare and visualise the performance of various classifiers. ROC curves are an effective performance measure that is used to visualise the trade-off between false alarm rates and true detection rates. Spackman [61] was one of the first to demonstrate the usefulness of ROC analysis to the area of machine learning relative to simple accuracy measures. Since then, ROC analysis has become increasingly popular in the machine learning community.

### 2.4.1 Confusion Matrix

A classification problem is usually considered as a two class problem, a true class, and a false class. Most classifiers fall into two instances, some that directly produce a binary label

to indicate which class the test image belongs to, either the true class or false class. But other classifiers (like correlation filters) produce a continuous output in the form of a score. A threshold is applied to the score to define a true or false class membership.

Irrespective of the type of classifier, there are 4 outcomes that can be measured:

**True Positive** If the test image contains the target, and is classified correctly as positive.

**True Negative** If the test image does not contain the target and is classified correctly as being negative.

**False Positive** If the test image does not contain the target and is misclassified as being positive.

**False Negative** If the test image contains the target, and is misclassified as negative.

These 4 possible outcomes are tallied into a confusion matrix, an example of which is shown in Figure 23 where the correct classification results are represented across the main diagonal.

		True class			
		p	n		
Hypothesized class	Y	True Positives	False Positives	fp rate = $\frac{FP}{N}$	tp rate = $\frac{TP}{P}$
	N	False Negatives	True Negatives	precision = $\frac{TP}{TP+FP}$	recall = $\frac{TP}{P}$
Column totals:		P	N	accuracy = $\frac{TP+TN}{P+N}$	
				F-measure = $\frac{2}{1/\text{precision}+1/\text{recall}}$	

Figure 23: Confusion Matrix along with performance metrics calculated using it [62].

### 2.4.2 ROC Curve

The values from the confusion matrices for different thresholds can be used to graph the ROC curve. The ROC curves are represented by 2D graphs which indicate the true positive rate (shown in equation (36)) on the y-axis, the false positive rate (shown in equation (37)) along the x-axis. The ROC curve graphically depicts the tradeoff between the accuracy (true positive rate) and the cost of this accuracy (false positive rate). These two parameters are discussed below.

**True Positive Rate (tp rate)** The true positive rate or tp rate, (also known as the recall, or hit rate) can be calculated as:

$$\text{True positive rate} = \frac{\text{Positives correctly classified}}{\text{Total positives}} \quad (36)$$

**False Positive Rate (fp rate)** The false positive rate or fp rate, (also known as false alarm rate) can be estimated base on the number of negative test images which are incorrectly classified as being part of the true class and is given as:

$$\text{False positive rate} = \frac{\text{Negatives incorrectly classified}}{\text{Total negatives}} \quad (37)$$

The false positive rate and true positive rate are plotted against one another to form a curve<sup>4</sup> as shown in Figure 24.

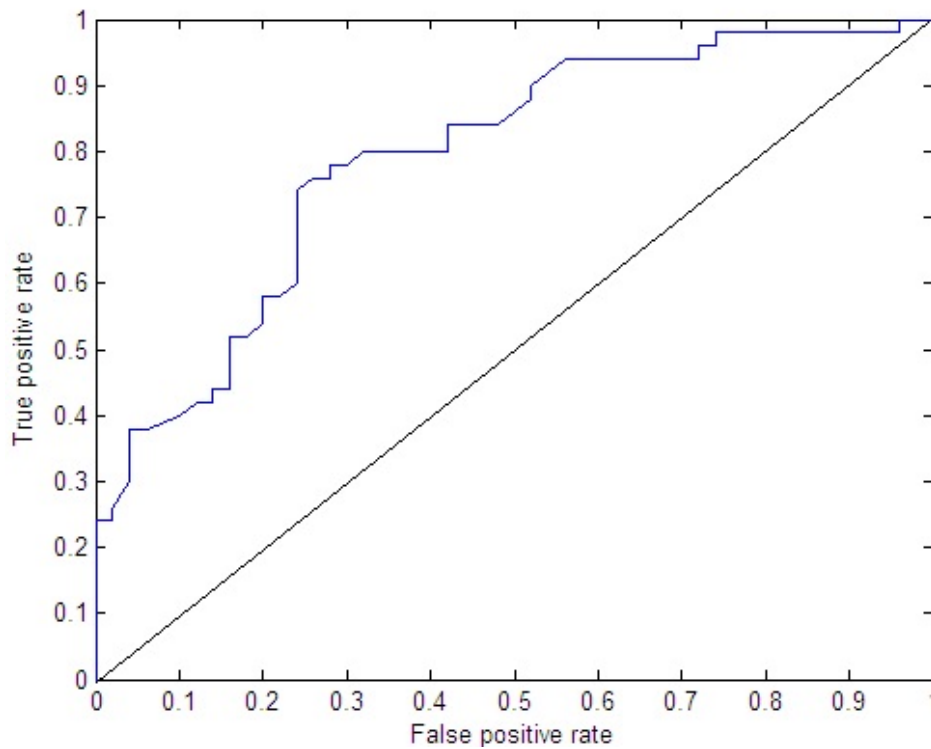


Figure 24: An example ROC curve for a classifier. The True positive rate is plotted on the Y-axis, and the false positive rate is plotted on the X-axis.

There are certain regions that are important to note on the ROC curve. The top left region indicates the classifier performs almost perfectly, with the point (0, 1) representing a perfect

<sup>4</sup>It should be noted that binary classifiers do not produce an ROC curve, but just a point on the ROC space. Several runs a binary classifier using various parameters can be used to compare their performance by the points on the ROC space with the top left region being the most desirable.

classifier that is able to detect all of the targets with no misclassifications. The lower left region means that the classifier does not have many false alarms, but does not detect very many of the targets either. Regions close to the right are not desirable as this indicates a high false alarm rate irrespective of the number of correctly classified targets.

### 2.4.3 Chance Line

A diagonal line of  $y = x$  is usually plotted on the ROC curve which indicates the performance if the labels were chosen by random chance. Anything above this line (in the upper left triangle) means that the classifier is able to use the data to perform better than randomly guessing. Anything below this line (lower right triangle) suggests that the classifier performs worse than randomly. If the graph is located in the lower right triangle, the criteria of the classifier could be reversed to improve performance, (negatives become positives) and it would then appear above the chance line. There are however some cases where a single ROC curve can appear both above and below the chance line. In this case it is not possible to change the criteria to improve performance. In such situations by simply reversing the output of the classifier for regions when it is below the chance line, it can still provide useful information.

### 2.4.4 Area Under the ROC Curve (AUC)

Another characteristic to consider for a given ROC curve is the area underneath it, known as the AUC. The area under the curve allows us to summarise the performance of a classifier by a single scalar value [62]. The AUC is calculated as a portion of the unit square area and is a value between 0 and 1. A classifier that has a curve that is along the chance line, would have an AUC of 0.5, and the closer to 1 the AUC is, the better classifier performance is. The AUC can also be thought of as a value that is equivalent to the probability of the classifier ranking a target image above a negative image. It is also important to note that although one classifier could have a greater AUC than another, it does not necessarily mean it will outperform a classifier with a lower AUC for specific ROC regions. However, as a baseline, classifiers with an AUC higher than 0.8 are considered good [62].

## 2.5 Optic Flow

Biological studies suggest that insects dynamic perception is largely due to a form of optic flow as discussed in Section 1.2.2. The visual processing inside an insect's brain is tuned to recognise specific patterns in the optic flow fields. It is important to be clear on what is meant by optic flow (optical flow) and motion field (velocity field). Optic flow is defined as the apparent motion of image intensities (brightness). In other words, optic flow is the velocity field that represents the motion between two frames (images) in a video sequence. However, the motion field is defined as the 2D projection of the relative 3D motion of scene points onto a retina (or image sensor) and is a geometrical concept. The motion field is a projection of the 3D motion vectors onto the image. It is also important to note that the optic flow and the motion field are not always the same. An example of this could occur when

regions have a lack of texture. Computing optic flow enables us to estimate the motion fields which use the information for useful tasks such as navigation and obstacle avoidance.

There are two main directions for bioinspired optic flow implementations on hardware [63]. The first uses standard high resolution camera hardware which is likened to single chamber eyes on humans and mammals. The second uses simulated compound eyes as found in flying insects. Here specialist custom camera hardware is used to simulate the response of a single neuron sensor to motion following the EMD response models discussed previously in Section 1.2.2.

In this thesis, differential methods are considered as a starting point as they are similar to how the elementary motion detectors in insects function. Within the differential based methods, there are two main approaches, local and global methods. However, all of the algorithms share three core assumptions:

- **Constant Brightness:** The intensity of each pixel in the video frame is assumed to remain constant from one frame to another
- **Small Motion between frames:** It is important that the sampling rate of the camera is fast enough so that the motion of each pixel between frames is small.
- **Smooth Reflectance:** The motion of local neighbourhood of pixels between frames is assumed to be constant.

The most prevalent local method is the Lokas-Kanade approach [64]. This method optimises a local energy function. Such local methods have an advantage of being more robust to noise, however, the flow fields are sparse.

Global approaches, first proposed by Berthold Horn [65] attempts to minimise a global energy function. These dense methods provide an advantage of providing a fully dense flow field but are sensitive to noise. In this work, dense flow fields are considered as having a dense flow field is important for locating object boundaries which can aid in the detection of objects. Also, the optic flow fields obtained by flying insects is considered to be a dense flow field [9].

Foundational papers published by Berthold Horn, and Brian Schunck [65][66] looked at solving dense optic flow. Their methods introduce constraints that are not always realistic on real world situations, but this work is still useful as a starting point and it also paved the way for a variational approach to machine vision. Variational approaches are not feature based and can take all parts of an image into account where feature based methods only look at isolated pixels.

### 2.5.1 Image Flow Constraint Equation

Constraints are required in order to compute optic flow at a point in an image independently to the neighbouring points. This situation arises as there are 2 motion components (motion in  $x$ , and  $y$ ) for a point in the velocity field of the image, whereas the actual change of intensity (brightness) at that same point due to motion is represented as a single scalar value. Therefore it is necessary to introduce extra constraints for this ill posed problem. These



constraints were introduced by Horn in the image flow constraint equation are defined as follows:

### Constant Brightness

The brightness at a given point in the image plane is assumed to be constant. The only change in terms brightness is only caused by the motion of the pixels.

### Smooth Reflectance

As the pattern of image brightness translates across the image plane, it does so without any distortion to the brightness pattern. The brightness pattern on a surface should be smooth and constant, except at a finite number of discontinuities (along the edges).

The change of the image brightness is only due to motion,  $E(x, y, t)$  at a point  $(x, y)$  at time  $t$  is captured in the image flow constraint equation. This image flow constraint equation is derived by looking at a section of the image brightness pattern after it has been moved by  $\delta y$  in the  $y$  direction and by  $\delta x$  in the  $x$  at a time  $\delta t$ . Due to constraints mentioned before we assume that the image brightness remains unchanged between displacements (brightness at a particular point is constant) so:

$$E(x, y, t) = E(x + \delta x, y + \delta y, t + \delta t) \quad (38)$$

Using a Taylor Series we can expand the right-hand side of the equation about the point  $(x, y, t)$  to become:

$$E(x + \delta x, y + \delta y, t + \delta t) = E(x, y, t) + \frac{\partial E}{\partial x} \delta x + \frac{\partial E}{\partial y} \delta y + \frac{\partial E}{\partial t} \delta t + \epsilon \quad (39)$$

where the the second order and higher order terms in  $\delta x$ ,  $\delta y$  and  $\delta t$  are contained in  $\epsilon$ . If we subtract  $E(x, y, t)$  from both sides of equation (39) and then divide the result by  $\delta t$  we get the image constraint equation:

$$\frac{\partial E}{\partial x} \frac{\delta x}{\delta t} + \frac{\partial E}{\partial y} \frac{\delta y}{\delta t} + \frac{\partial E}{\partial t} = A(\delta t) \quad (40)$$

where  $A(\delta t)$  contains the first order and higher variations of  $x$  and  $y$  that depend on  $\delta t$ . In the limit of  $\delta t \rightarrow 0$  the equation becomes the more common form of the image flow constraint equation as shown below.

$$E_x \mathbf{u} + E_y \mathbf{v} + E_t = 0 \quad (41)$$

where  $E_x = \partial E / \partial x$ ,  $E_y = \partial E / \partial y$  and  $E_t = \partial E / \partial t$  are the partial derivatives of image brightness with respect to  $x$ ,  $y$  and  $t$ , respectively, while  $\mathbf{u} = dx/dt$  and  $\mathbf{v} = dy/dt$ .

A line in the velocity space satisfying the image flow constraint equation at a given point on the image plane at a specific time instant is shown in Figure 25.

The change in image brightness is a 1D value, but velocity field is 2D ( $x$  and  $y$  components), this causes a problem when trying to find the velocity field as only the component of the velocity field along the gradient direction can be computed. The component of the motion in the direction of the brightness gradient  $\nabla E$  is:

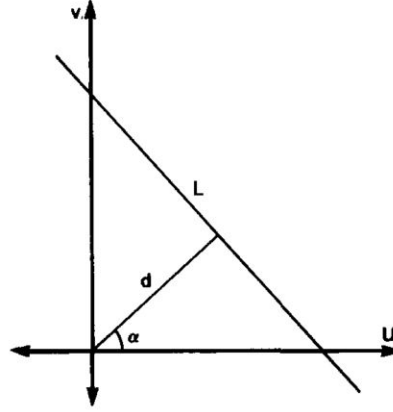


Figure 25: Image Constraint Line

The displacement of the constraint line from the origin is represented by  $d$ ; The orientation of the constraint line is represented by  $\alpha$  [66]

$$\nabla E = \frac{-E_t}{\sqrt{E_x^2 + E_y^2}} \quad (42)$$

However, we can only recover component of movement along the brightness gradient locally without introducing extra constraints. See Figure 26.

For the condition of constant brightness

$$\frac{dE}{dt} = E_t + \nabla E \cdot \mathbf{v} = 0 \quad (43)$$

which can be rewritten as

$$E_t + \|\nabla E\| \mathbf{v}_\perp = 0, \quad (44)$$

where  $\mathbf{v}_\perp$  is the norm of  $\mathbf{v}$  component of the motion field  $\mathbf{v}$  along the direction of  $\nabla E$ . Since the brightness is constant (see equation (43)), but the brightness gradient is not constant  $\|\nabla E\| \neq 0$ ,

$$\mathbf{v}_\perp = -\overbrace{\frac{E_t}{\|\nabla E\|}}^{v_\perp \text{ from (44)}} \underbrace{\frac{\nabla E}{\|\nabla E\|}}_{\text{unit gradient}}, \quad (45)$$

where  $\mathbf{v}_\perp$  is the vector whose length is  $v_\perp$ . Therefore, if equation (43) holds, the component of the motion field along the direction of the gradient of the image brightness  $\mathbf{v}_\perp$  can be written in terms of derivatives of  $E$  (which can be computed). Equation (45) can be interpreted as an instance of the well-known aperture problem for the unknown  $\mathbf{v}$ : that is, the information available at each point of a sequence of frames is only the component of the motion field along the direction of  $\nabla E$  (brightness gradient) not along the brightness contour.

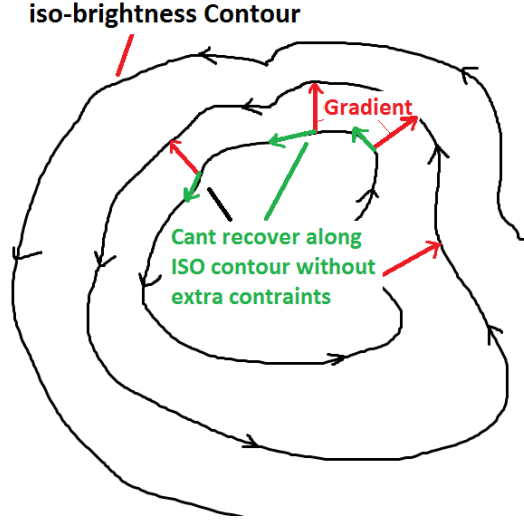


Figure 26: Brightness contours

It is only possible to recover the component of movement along the brightness gradient (red), It is not possible to recover movement along the iso-brightness contours without additional constraints

In order to estimate the partial derivatives, Horn and Schunck introduced a smoothness constraint to pose solving the derivatives as an optimisation problem. The assumption was based on the fact that small regions of an object would move together since objects are rigid. This would mean that neighbouring regions would have similar velocities. The optimisation problem consists of two terms:

- A penalty on the deviation of the estimated velocity field from the image flow constraint equation.
- A penalty on the deviation of the velocity field component from that for a smooth surface.

The optimisation criterion is given as:

$$\int \int (E_x u + E_y v + E_t)^2 + \kappa^2 \left( \left( \frac{\partial u}{\partial x} \right)^2 + \left( \frac{\partial u}{\partial y} \right)^2 + \left( \frac{\partial v}{\partial x} \right)^2 + \left( \frac{\partial v}{\partial y} \right)^2 \right) dx dy \quad (46)$$

where the relative cost of deviations from velocity field of the image flow constraint and the smoothness constraint are controlled by the term,  $\kappa^2$ . A coupled pair of partial differential equations can be found using the calculus of variations:

$$\Delta u = \frac{E_x}{\kappa^2} (E_x u + E_y v + E_t) \quad (47a)$$

$$\Delta v = \frac{E_y}{\kappa^2} (E_x u + E_y v + E_t) \quad (47b)$$

Horn and Schunck were able to estimate the image flow field by iteratively solving a pair of partial differential equations for  $u$  and  $v$  as shown in equation 47. Horn and Schunck found that this algorithm could only handle simple image flows, that did not have any discontinuities in the brightness or velocity field. This method was not ideal as it blurs sudden or sharp changes in the velocity field and can not be used beyond simple flows with no discontinuities in brightness or velocity.

### 2.5.2 Discontinuous Image Flows

Brian Schunck took the work of the image flow constraint equation further by dealing with discontinuities through the introduction of an alternative derivation of the image flow constraint equation. Such discontinuities arise when a scene is mapped onto the image plane. These discontinuities separate the various surfaces in the scene. This means that in terms of the velocity field, the projected surface on the image plane is represented by areas of smooth motion, with boundaries along the edges. The previous image flow constraint equation (shown in equation (41)) can not handle discontinuities as they are blurred by the smoothing function. An alternative image constraint equation was presented by Schunck which includes  $\delta$ -functions.

**Image Brightness Discontinuities** Figure 27 shows a step change in image brightness. The image is moving with a velocity, with brightness of  $I_0$  at point  $P_0$ , and point  $P_1$  has a brightness of  $I_1$ .

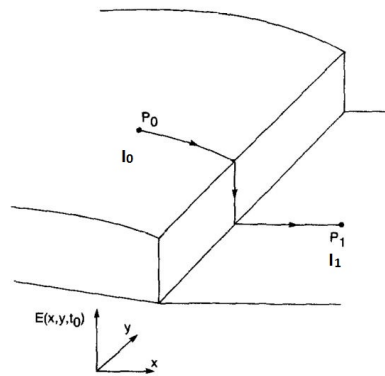


Figure 27: Step change in image brightness [66].

Two points  $P_1, P_0$  in the image plane on either side of a step change in brightness.

The line integral is used to calculate the brightness change along the line from  $P_0$  to  $P_1$ , and is given as:

$$I_1 = \int_{P_0}^{P_1} \Delta E \cdot dl + I_0 \quad (48)$$

This equation is valid only if the gradient is evaluated across a step change in image brightness since the integral must surround any  $\delta$ -functions.

The integral that represents the change in brightness due to motion over point  $p_1$  is given as:

$$I_0 = \int_{t_0}^{t_1} \frac{\partial E}{\partial t} dt + I_1 \quad (49)$$

where the patch of brightness moves from its initial position,  $P_0$  at  $t_0$  to point  $P_1$  at time  $t_1$ . The overall change in brightness must be the same for both cases  $I_1 - I_0 = 0$

$$- \int_{t_0}^{t_1} \frac{\partial E}{\partial t} dt = \int_{P_0}^{P_1} \Delta E \cdot dl \quad (50)$$

This equation (50) means that the change in brightness when you move from  $P_0$  to  $P_1$  not considering time (right hand side) will equal the change in brightness if you stay at point  $P_1$  and the surface moves with time ending at point  $P_0$  (left hand side). The right hand side of equation (50) is transformed to units of time. If  $(x, y) = (u, v)t$ , then:

$$- \int_{t_0}^{t_1} \frac{\partial E}{\partial t} dt = \int_{t_0}^{t_1} \Delta E \cdot (u, v) dt \quad (51)$$

Since this equation is true for arbitrary units of time, and arbitrary points both sides of the argument are equal, allowing us to obtain an alternative image flow constraint equation.

$$\Delta E \cdot (u \cdot v) + \frac{\partial E}{\partial t} = E_x u + E_y v + E_t = 0 \quad (52)$$

Essentially equation (52) means that at a fixed point, the change over time must be the same as the change over space at a fixed time. This means that the brightness change detected over time at a point  $P$  in the image must equal the change in brightness detected in the image by following the path the point would follow across the image. This assumes that the motion is the only factory causing the brightness to change.

This alternative image flow constraint (see equation (52)) derivation is useful as it valid for discontinuities in image brightness. It relies on restricted problem domains with limited surface reflectance, constant illumination. The aperture problem cannot be avoided with a limited field of view camera. Some of the assumptions mentioned at the start of this section are not applicable to realistic situations, in particular constant brightness. The work in this thesis ultimately makes use of an alternative approach to solving optic flow through the use of spatio-temporal filtering (see Section 4.4) which is introduced in Section 2.6. However, a detailed study was conducted on research to look at solving dense optic flow from a global pattern like approach. This is discussed in Appendix A for both planar and spherical systems. An important differentiator between common camera sensors and insect eyes is that insect eyes are spherical, as discussed in Appendix A. This much wider field of view reduces some ambiguities particularly the aperture problem, especially when coupled with IMU sensor information.

## 2.6 Spatio-Temporal Filtering

As discussed in Section 2.5, classical dense optic flow is rather difficult to establish unambiguously, even without the presence of noise.

An alternative approach to dynamic pattern recognition is through the use of spatio-temporal filtering which is discussed in this section. Most of the research in this area is based on the human visual system, but some research suggests the same thing happens in insect vision to some extent [22]. The difficulty in studying this in insects is mainly a function of current technology and the comparatively small size of an insects brain.

### 2.6.1 Gabor Pairs and Spatio-Temporal Energy

In 1985, a new bio inspired modelling approach to velocity computation in the Human Visual System (HVS) was proposed in two papers: [67] and [68]. It was postulated that HVS works in the spatio-temporal domain, i.e., that the images registered on the eye's retina in the spatial coordinates  $(x, y)$  are stacked over time  $t$  in the short-term memory to form a spatio-temporal volume represented by the coordinate triple  $(x, y, t)$ . Motion patterns are detected by convolution (linear filtering) of the spatio-temporal volume with directional filters. For example, a dot moving at a uniform speed from left to right in the  $(x, y)$ -plane will result in a line in the  $(x, y, t)$ -volume and the slope of that line will correspond to the speed of the dot. Similarly, a rectangular bar moving at a uniform speed from left to right in the  $(x, y)$ -plane will result in a plane in the  $(x, y, t)$ -volume and the slope of that plane will correspond to the speed of the bar. An example of this is shown in Figure 28.

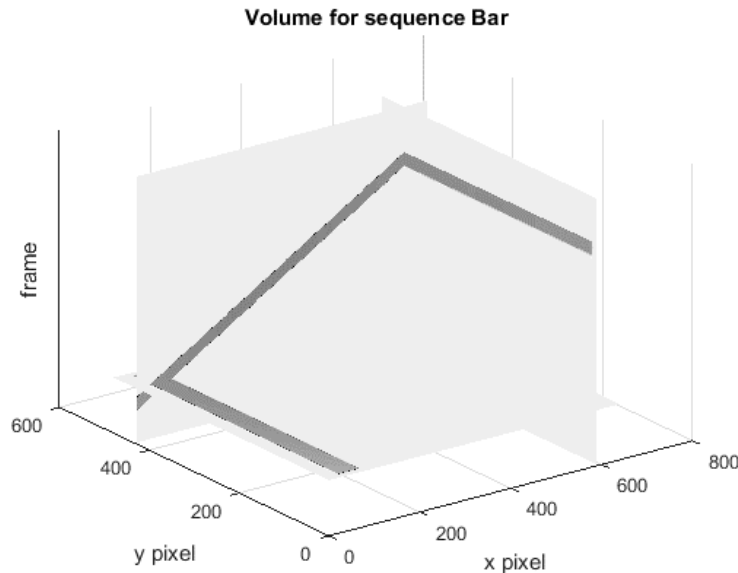


Figure 28: A rectangular bar moving at a uniform speed from left to right in the  $(x, y)$ -plane will result in a plane in the  $(x, y, t)$ -volume and the slope of that plane will correspond to the speed of the bar.

Hence, using a filter bank of directional filters, working in the spatio-temporal domain (or the  $(x, y, t)$ -volume), uniform motions of simple objects can be detected at various speeds by examining the outputs of the filters. At the same time, for a given time  $t$ , the position  $(x, y)$  of a given object on the retina is also available, thus giving an efficient solution to both spatial localisation and also the speed of the observed objects. Finally, the processing based on a bank of directional, spatio-temporal filters is not limited to uniformly moving, simple objects, as shown in [69].

Subsequent investigations [70] confirmed physiological plausibility of the spatio-temporal processing model proposed in [67] and [68] and led to significant refinements of the model. In particular, it was shown that the bank of directional, spatio-temporal filters is realised by the use of three-dimensional (3-D) Gabor functions [71]. These 3-D Gabor functions optimally [72] (in terms of the uncertainty principle) balance the extent of their action (concentration) in the spatio-temporal domain  $(x, y, t)$  and the corresponding frequency (Fourier) domain  $(f_x, f_y, f_t)$ . Three-dimensional spatio-temporal Gabor functions are given [71], [73] in even/odd pairs:

$$G_{\text{even}}(x, y, t) = \frac{1}{\sqrt{2\pi^3}\sigma_x\sigma_y\sigma_t} \times \exp\left[-\left(\frac{x^2}{2\sigma_x^2} + \frac{y^2}{2\sigma_y^2} + \frac{t^2}{2\sigma_t^2}\right)\right] \times \cos[2\pi(f_{x_0}x + f_{y_0}y + f_{t_0}t)] \quad (53)$$

$$G_{\text{odd}}(x, y, t) = \frac{1}{\sqrt{2\pi^3}\sigma_x\sigma_y\sigma_t} \times \exp\left[-\left(\frac{x^2}{2\sigma_x^2} + \frac{y^2}{2\sigma_y^2} + \frac{t^2}{2\sigma_t^2}\right)\right] \times \sin[2\pi(f_{x_0}x + f_{y_0}y + f_{t_0}t)], \quad (54)$$

where  $(f_{x_0}, f_{y_0}, f_{t_0})$  is the centre frequency (the spatial and temporal frequency for which  $G_{\text{even}}$  gives its greatest output) and  $(\sigma_x, \sigma_y, \sigma_t)$  defines the spread of the spatio-temporal Gaussian window. The directionality of the Gabor pair (see equation (53) and equation (54)) expresses itself in the preferred spatio-temporal direction:

$$f_{x_0}x + f_{y_0}y + f_{t_0}t = 0. \quad (55)$$

Along the line (equation (55)) in the spatio-temporal volume, the response of the  $G_{\text{even}}$  filter in equation (53) is strongest; its response is zero in the direction orthogonal to equation (55) and intermediate in-between. It is worth noting that the 3-D Gabor functions in equation (53) and equation (54) form a family of multi-resolution steerable filters [46], [74].

A remarkable property of the Gabor pair (see equation (53) and equation (54)) is that they form a quadrature pair [75], i.e.,  $G_{\text{even}}$  and  $G_{\text{odd}}$  contribute orthogonally<sup>5</sup> to the oriented spatio-temporal energy:

$$E(x, y, t) = [G_{\text{even}}(x, y, t) \star I(x, y, t)]^2 + [G_{\text{odd}}(x, y, t) \star I(x, y, t)]^2, \quad (56)$$

<sup>5</sup>Note that  $G_{\text{even}}^2(x, y, t) = G_{\text{odd}}^2(x, y, t)$  for all  $(x, y, t)$ .

where  $I$  is the input video in the form of a spatio-temporal volume and  $\star$  stands for three-dimensional convolution. There seems to be plausible evidence [70] that the human visual system indeed computes the motion energy according to equation (56). Furthermore, this motion energy is simultaneously computed for two quadrature pairs, with the pairs tuned to mutually orthogonal directions. These two motion energies are then subtracted from each other to achieve noise reduction.

### 2.6.2 Speeding up the Spatio-Temporal Energy Equations

The spatio-temporal Gabor functions (given in equations (53)–(54)) that are used for the motion calculations are not computationally efficient when they need to be applied at various combinations of angles in the spatio-temporal volume. The human visual system as it is designed for this as is able to this efficiently. The first step to speed this up is to take advantage of the separability of 3D Gabor functions and implement it as a series of 1D functions. To further increase the computational efficiency an approximation of the Gabor functions [76] can be used. An example of this is shown in Figure 29. By approximating the Gabor wavelet with rectangular regions of constant value, authors in [76] reported a 4.4 times speed improvement with similar levels of performance.

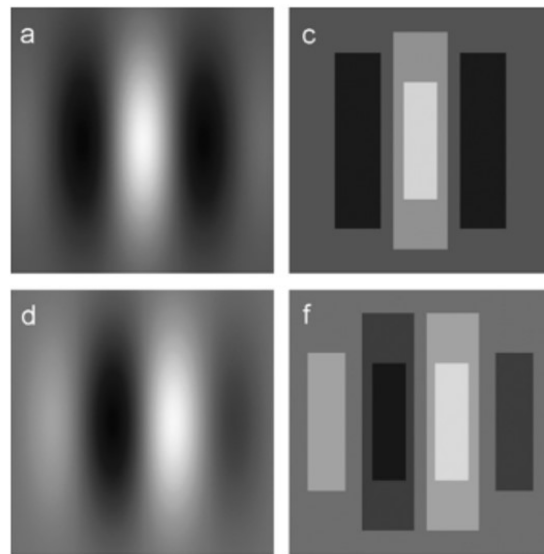


Figure 29: A 2D Gabor wavelet (left) is approximated by rectangular regions of constant value (right) [76].

### 2.6.3 Wildes and Bergen Features

Wildes and Bergen made a fruitful observation [77] that the oriented spatio-temporal energy (equation (56)) can be used to classify motion types by simple arithmetic combinations of



only four filter responses. This approach was later extended to be able to classify surprisingly complex motions [78]. They considered the  $(x, t)$ -slice and  $(y, t)$ -slice in the spatio-temporal volume corresponding to, respectively, horizontal and vertical motion. They found that, both for the horizontal  $(x, t)$ -slice and vertical  $(y, t)$ -slice, it is sufficient to perform motion-energy detection along four Gabor directions:  $0^\circ$ ,  $45^\circ$ ,  $90^\circ$  and  $135^\circ$  to cover basic types of motion. In this work, the angle calculation convention follows the original definition from [46] and is shown in Figure 30. This four-angle filtering can be done for different scales, thus realizing multi-resolution analysis of oriented spatio-temporal energy.

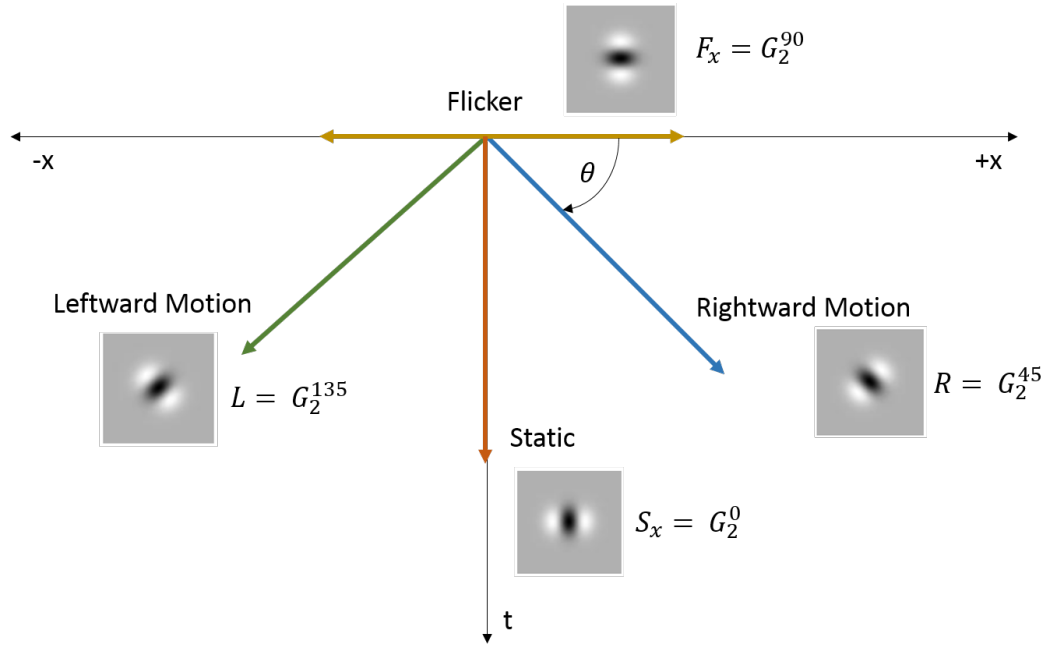


Figure 30: Vectors of the basic motion types from the  $(x, y)$ -plane projected onto the  $(x, t)$ -plane of a spatio-temporal volume. The corresponding second order Gaussian filter that is used to detect each vector is also shown.

Each of these Gabor filters are oriented to detect a certain direction of motion along the  $(x, t)$ -plane and  $(y, t)$ -plane. Wildes and Bergen used four filters on each plane to detect the following basic motion types:

- $F_x$ , Flicker - such as sudden illumination changes.
- $R$ , Rightward Motion on  $(x, t)$  ( $U$ , Upward Motion on  $(y, t)$ ).
- $L$ , Leftward Motion on  $(x, t)$  ( $D$ , Downward Motion on  $(y, t)$ ).
- $S_x$ , Static.

Using these four basic features, the six main types of motion can be found in a spatio-temporal volume as shown in Table 1. The top two rows show the types of motion on real

	Unstructured	Static	Flicker	Coherent Motion	Incoherent Motion	Scintillation
$ R - L $	0	0	0	++	0	0
$R + L$	0	++	++	++	++++	++
$S_x$	0	++	0	+	+	+
$F_x$	0	0	++	+	+	+

Table 1: The six classifications of motion in a spatio-temporal volume as proposed by Wildes and Bergen [77]. For simplicity this is shown for a  $(x, t)$ -slice of the spatio-temporal volume.

world footage for a  $(x, y)$ -slice and  $(x, t)$ -slice. The third row then shows an idealised pattern that each of these motion types would produce. When looking at the Fourier domain indicated in the fourth row, these patterns focus energy at specific regions in the frequency domain. The last four rows show that by using just approximate energy combinations of the Gabor filter responses, the type of motion can be classified. By combining the various energy responses, Wildes and Bergen were able to qualitatively recognise video regions exhibiting these motions. For example, coherent motion occurs when regions contain a moderate response from the difference of Gaussian  $|R - L|$  and  $R + L$ , and a low response of static and flicker filters.

Table 1 provides great insight into the spatio-temporal motion classifications and has lead to several applications extending segmentation of video sequences [78], and for background subtraction for CCTV surveillance videos [79]. However, none of these approaches truly use three dimensions as only 2D Gabor filters are used along the  $(x, t)$ -plane and  $(y, t)$ -plane across the volume. Furthermore, if the Gabor filter angles are spaced too far apart, situations can occur where this approach will label regions as having both coherent and incoherent motion coherent motions.

### 3 Symmetry Detection for Static Patterns

In this chapter, symmetry detection is posed as a static pattern recognition problem. This allows for the efficient detection of symmetries using composite correlation filters which is one of the contributions of this thesis. Furthermore, methods discussed in this section are shown to be considerably faster when compared to state-of-the-art symmetry detection methods, this shown in Section 3.4. This chapter is presented in the following manner:

- Section 3.1 defines the problem setting of the work conducted in this chapter, this relies on biological background information discussed previously in Chapter 1.
- Section 3.2 relies on engineering background in Sections 2.2 and 2.3 to implement the pattern based symmetry detection method. This section also introduces the primary, and abstracted datasets used for training and testing.
- Section 3.3 discusses the testing and ROC analysis of the symmetry detection method. An understanding of ROC analysis discussed in Section 2.4 is useful here.
- Section 3.4 compares the pattern based symmetry detection method introduced in this chapter to the current state-of-the-art method (see Section 2.1).
- Section 3.5 interprets the results and summarises the conclusions of this chapter.

#### 3.1 Problem Setting

Detection of objects with regularities is successfully performed by flying insects which possess very limited brainpower and devote much of their neural processing to flight control [3]. In particular, pollinating insects like honeybees are very effective in detecting flower symmetries, a surprisingly challenging problem due to the great variety of such symmetries [80], [81]. Hence, the work presented here is a bio-inspired approach to rotational symmetry detection, motivated by the ability for honeybees to efficiently detect symmetry patterns with limited processing power.

Horridge [82] has shown that honeybees can be readily trained to recall rotationally symmetric patterns. Figure 31, left, shows the results of one of Horridge's experiments in which more than 80% of the trained honeybees successfully recognised a rotationally-symmetric pattern. In the same experiment, when honeybees were trained on a pattern with no symmetric properties (Figure 31, right), very few were able to correctly recognise it, showing their clear preference for symmetric structures in the environment.

Flowers are food sources for honeybees and floral symmetry is an indicator of the quality and/or quantity of nutrition [81]. Detection of symmetries in a feature-rich environment is an efficient way of achieving relevance filtering, or focusing only on the relevant information in the environment (e.g., nutrition potential) whilst ignoring the rest. The relevance filtering principle of disregarding all but the absolutely essential details is of significant interest for autonomous systems because it offers a potentially efficient use of Machine Vision in order to achieve well-defined situational awareness. In particular, small—and thus

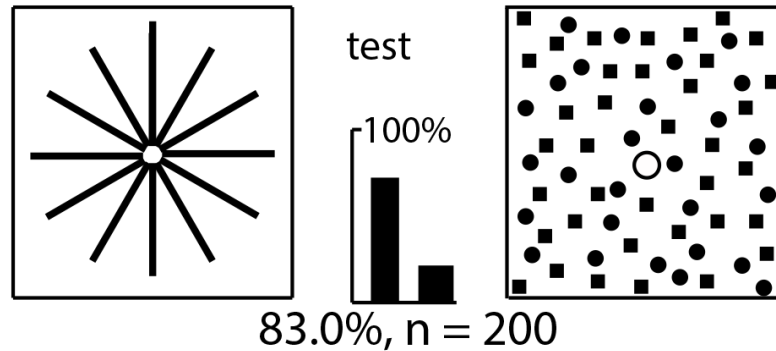


Figure 31: Honeybees have a clear preference for symmetric structures in their environment (redrawn from [82]). After being trained on two kinds of new patterns (shown on the left and right above), 83% of trained honeybees chose the symmetric pattern on the left as shown by the experimental results from Horridge [82].

resource-limited—autonomous platforms (e.g., MAVs) would benefit from the inexpensive processing of imagery in order to foster an artificial perception of their external environment.

The relevance filtering used by honeybees is illustrated in Figure 32 which shows, on the left, simplified (abstracted) representations of flower symmetry proposed by Horridge [82]. These Horridge patterns are efficient abstractions of complex floral symmetries [80] which represent the relevant floral symmetry structure and ignore the rest of the flower detail. In Horridge’s experiments, rather than remembering the details such as texture, the number of petals and species of flower, honeybees learnt a simplified (abstracted) representation of a generic flower based on rotational symmetry. When presented with an unknown flower, the visual system of a honeybee can be envisaged as performing relevance filtering by testing the flower image for the symmetry type (and regularity) to inform a decision whether to choose or ignore the flower.

The example Horridge patterns, shown on the left of Figure 32, are non-unique abstractions of rotational symmetry of certain flowers from the Anthemideae tribe. This non-uniqueness gives rise to two questions: 1) which of the patterns is the most efficient abstraction? and 2) which of the patterns leads to better detection performance? The first of these questions needs a definition of image complexity and the second question requires defining a detection performance measure. Here, abstraction efficiency is defined through the Haralick image correlation [85], consistently with the use of correlation filters for detection.

## 3.2 Correlation Filter Training

### 3.2.1 Training Templates for Correlation Filters

The training image dataset used for synthesising composite correlation filters is an important aspect which can impact the performance of the filter. In order to synthesise a plane rotation invariant correlation filter, training images of the target at various rotations are used. Similarly, for perspective distortion, the data should include images taken at various elevation

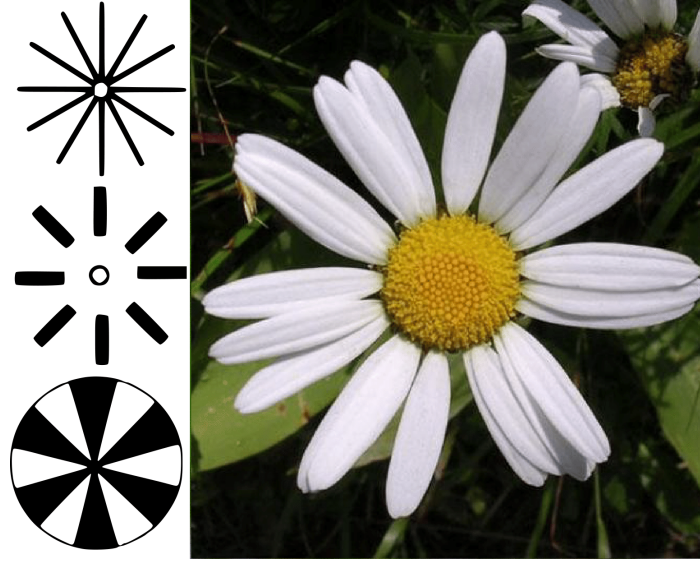


Figure 32: Three images on the left show examples of symmetric patterns due to Horridge (redrawn from [82]) which honeybees learn readily (see also Figure 31). These Horridge patterns efficiently capture the high contrast between the petal boundaries in the Chrysanthemum (*image\_06206* from [83]) on the right. Such simplified (abstracted) patterns are compared by the honeybee visual system with the flower image thus enabling efficient detection of floral symmetry. This pattern-based symmetry detection is applicable not only to this example Chrysanthemum from the Anthemideae tribe, but also to other Anthemideae flowers with a similar symmetry structure, e.g., *Osmitopsis asteriscoides* or *Ismelia carinata*, see [84].

angles. For each elevation angle, a series of rotated versions of that object are used for training. One would assume that to train the best rotation invariant filter would require many rotated versions of the object.

However, this is not the case, as the number of training images used has a direct impact on the filters SNR performance in [86]. Increasing the number of training image rotations reduces the best case SNR (poorer discrimination) while increasing the worst case SNR (better distortion tolerance). There is a point at which adding more training images will not increase the worst case SNR which is often dependant on the actual training data and object properties.

This is shown in Figure 33, where the signal to noise ratio of five MACH correlation filters were compared, each of which were trained using a different number of training images. A single flower image was used which was transformed into a new image for each angle between  $0^\circ$  and  $179^\circ$ . The five filters were trained by synthesising a MACH filter at various angular steps.

The first filter was trained using rotations at every  $90^\circ$ , so only two images were used to synthesise this filter. As shown by the green line, there is a clear best case SNR peak at  $0^\circ$  and  $90^\circ$ , which corresponds to the rotations of the two training images.

As the number of training image rotations is increased, the angle spacing reduced between

them, the best case SNR reduces while the worst case SNR increases. As indicated by a blue line for a MACH filter trained on every rotated image, there is no clear best or worst case SNR peaks, however, the overall worst case SNR is the highest compared to the other filters, indicating good distortion tolerance. However, using training images spaced at approximately  $5^\circ$  to  $15^\circ$  (red and orange lines respectively) results in a higher best case SNR while still achieving an acceptable worst case SNR. This worst case SNR is still high enough to be able to easily recognise the target against background noise in the scene.

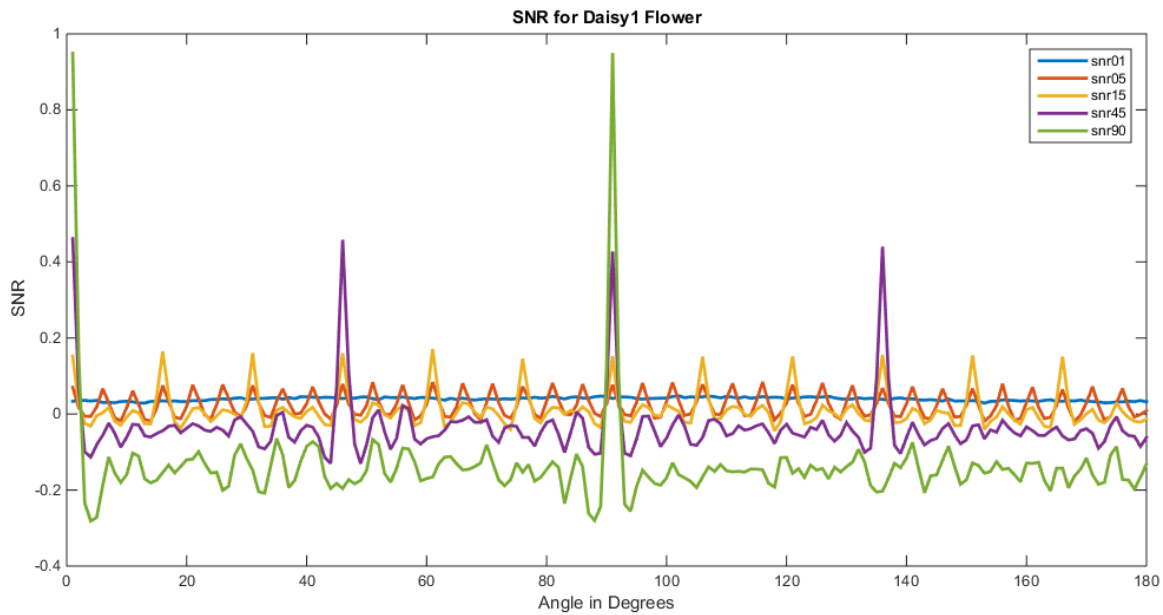


Figure 33: Signal to Noise comparison between different number of training images and plane rotation angles used to train a MACH filter. Five correlation filters were compared, where each uses a different number of rotations of a flower. When fewer rotations of the training images are used, the maximum peak SNR is higher, with very low SNR at other rotations, resulting in a filter that has poor distortion tolerance. When more training images are used the peak SNR is lower, with an overall higher worst case SNR, resulting in better distortion tolerance at the cost of poorer discrimination ability.

If too many training images are used, the best case SNR reduces, which could result in the filter misclassifying similar looking objects rather than being able to discriminate between them. Depending on the application it may be desirable to find an exact species of flower, rather than detecting if an object is just a flower.

This agrees with [86] as there is a relationship between the number of training images used and the discrimination and distortion tolerance of the correlation filter. For a given training dataset, there is an ideal number of training images to give a good balance between best and worst case SNR. This can also be used for efficient computation since only the required number of training images need to be processed.

Additional research into the training dataset quality has also shown that detection results are further improved when the centre of the target in all of the training images are exactly

the same [87].

### 3.2.2 Scale, Rotation and Translation Invariance

As mentioned in Section 3.2.1, the type of training image used can synthesis a correlation filter that is in-plane rotation invariant if a sufficient quantity of rotated versions of the target are used. This can be extended to perspective effects by using a dataset of training images of each object at a variety of azimuth elevations for all rotations. Then a bank of correlation filters of the same object at different azimuth elevation angles will be correlated with the test image to find the object. The correlation results would indicate the presence of the target.

The issue of scale is often resolved through the use of multiscale image pyramids as shown in Figure 34. A multilevel Gaussian image pyramid can be used to resample the test image from a top down approach, with each level gaining more details. However, an alternative method would be to simply synthesise a correlation filter at multiple scales, where each filter size will be dependant on the operating environment and the type of object.

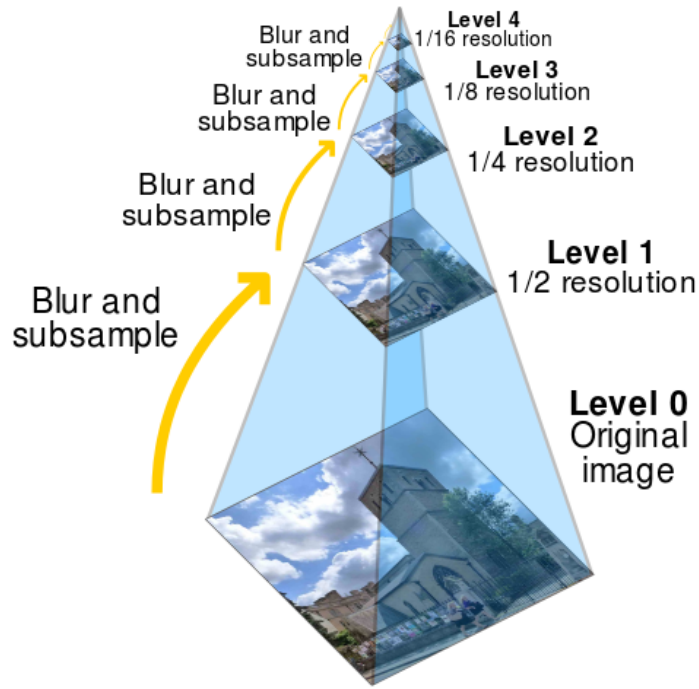


Figure 34: Visual representation of an image pyramid with 5 levels [88].

The work on correlation filters conducted in this thesis, in-plane rotation invariance for a given object is achieved through using multiple representations at each orientation during the training phase. Scale invariance was achieved through the use of a 3 level Gaussian pyramid. Translation invariance is solved through the use of the correlation sliding window function as shown in equation (9). The issue of perspective changes or other distortions were not studied in this work.



### 3.2.3 Graded-abstraction Dataset for Training and Testing

The proposed approach to rotational symmetry detection in flowers has two elements. Both are inspired by the honeybee vision. Firstly, symmetry detection is cast as a pattern matching problem to be solved with modern composite correlation filters (see Section 2.3). Secondly, the composite patterns to be detected are abstracted simplifications of the flowers to be recognised in accordance with the relevance filtering principle of disregarding the inessential details (see Figure 32).

The actual dataset used here was derived from a publicly-available Oxford flower dataset (see Section 3.2.4). Modified images from the Oxford flower dataset were progressively simplified to generate the graded-abstraction dataset actually used for training and testing (see Section 3.2.5).



Figure 35: Example images derived from [83].

### 3.2.4 Enriched Normalised Dataset

The starting point for the dataset used in this work was the publicly-available Oxford flower dataset [83] which contains 17 categories of flowers typically found in Europe. For consistency, the Oxford flower dataset was normalised so that all the images contain only a single flower in the centre without perspective distortion, (see Figure 35). All the images were further resized to a  $128 \times 128$ -pixel size so that  $d_1 = d_2 = w_1 = w_2 = 128$ .

Additional images rotated by  $90^\circ$ ,  $180^\circ$  and  $270^\circ$  were generated in order to enrich the dataset with more symmetry examples. The images in the enriched normalised dataset were randomly split into two subsets: a training dataset (709 images) and a test dataset (105 images).



### 3.2.5 Actual Dataset Used for Training and Testing

The actual realisation of the relevance filtering principle was achieved by generation of training images with a varying degree of complexity. These progressively-simplified training images were derived from the images in the enriched normalised dataset, defined in Section 3.2.4 above.

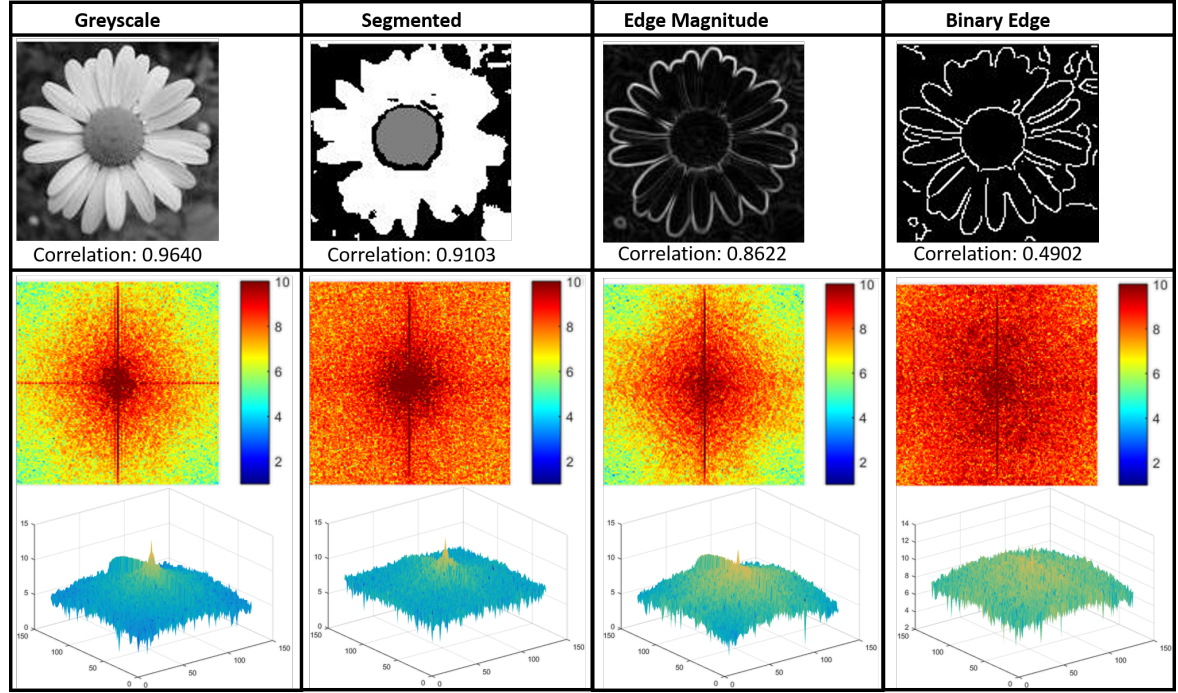


Table 2: Top row: Example flower image with detail reduction measured by correlation (see equation (57)). Bottom row: Amplitude distribution for the DFT of the top row; note diminishing amplitude variety consistent with the ordering  $G \succcurlyeq S \succcurlyeq E \succcurlyeq B$  and the correlation value in the top row.

The initial step for generation of progressively-simplified training images was to convert all images of the enriched normalised dataset from colour to greyscale, (see the top left of Table 2). For the thus-obtained full-detail (G) greyscale images, progressive detail reduction was then performed. The degree of progressive detail reduction for image  $x$  was measured by the Haralick image correlation [85]:

$$\rho = \frac{1}{\sigma^2} \sum_{m=1}^{d_1} \sum_{n=1}^{d_2} (m - \mu)(n - \mu)x(m, n), \quad (57)$$

where  $\mu$  and  $\sigma$  are the image mean and the standard deviation. The Haralick correlation  $\rho$  in equation (57) is a measure of image complexity arising in the context of co-occurrence statistics applied to image texture analysis [60, Chapter 16].

The first abstraction was reduction of each full-detail (G) greyscale image to its segmented (S) greyscale version; for the reference example in Table 2 this resulted in reduction

from  $\rho_G = 0.9640$  to  $\rho_S = 0.9103$ . The next abstracted category had only the greyscale edge-magnitude (E) information resulting in complexity reduction from  $\rho_S = 0.9103$  to  $\rho_E = 0.8622$  for the reference example. The last category was a binary (B) image with edges only, having correlation  $\rho_E = 0.4902$ , thus reducing complexity by a factor of two when compared to the full-detail image with  $\rho_G = 0.9640$ .

The graded-abstraction approach  $G \succcurlyeq S \succcurlyeq E \succcurlyeq B$  allowed training the filters on varying levels of detail with the richness of detail expressed by progressive reduction  $\rho_G \geq \rho_S \geq \rho_E \geq \rho_B$ . An additional illustration of image-complexity reduction is provided by the DFT of the  $G \succcurlyeq S \succcurlyeq E \succcurlyeq B$  image sequence, (see the bottom row of Table 2) which shows clear gradation of the DFT amplitude variety.

### 3.3 Graded-abstraction Testing and Performance Results

The experiments performed in this work used the graded-abstraction dataset described in Section 3.2.5 both for the design and testing of the three kinds (MACE, UMACe or MACH) of composite correlation filters defined in Section 2.3.

Four designs were made for a given kind of composite correlation filter (MACE, UMACe or MACH), each design resulting in a different composite template for the given kind of filter. Each of the composite templates, generated in the design, was based on the same abstraction level (grade) of the images from the training dataset. Hence, the design based on full-detail (G) greyscale training images generated the G-template, segmented (S) greyscale training images generated the S-template, greyscale edge-magnitude (E) training images generated the E-template and, finally, the binary (B) edge training images generated the B-template. As a result of this design process, twelve filter designs were obtained: three filter kinds (MACE, UMACe or MACH) with four templates (G, S, E and B) each.

The graded-abstraction filter design allowed, for each filter kind, testing the graded-abstraction designs with respect to their generalisation and specialisation performance. In generalisation tests, the filters trained on low-grade data were given high-grade input, detecting symmetry of complex objects despite using simple templates; in specialisation tests, the grading was reversed. This training/testing set-up is illustrated in the table shown in Table 3 where the main diagonal (blue) corresponds to the default training/testing set-up used in correlation filter theory and practice [89]. The lower triangle of the table (green) represents the generalisation region in which the correlation filter performance is assessed for the ability to detect targets which are the more complex than the training data. Good generalisation capabilities are attractive due to the resulting design simplicity and operational efficiency. The ideal situation is the B-G scenario: training on binary edge images (B-template) and detection of the flower in the full-detail (G) greyscale test image (see Table 3). The upper triangle of the table (pink) represents the specialisation region where the training data contain more information than the testing data. Specialisation capabilities are less attractive in practice but were also considered in this work for completeness.

Recalling from Section 3.2.4 that all test images have a flower centred in the middle of the image, a correct detection was declared if the correlation peak was located within the central region of the test image. In the remainder of this section, performance results of

Test \ Train	Grey		Segment		Edge Mag		Bin Edge	
	Train	Test	Train	Test	Train	Test	Train	Test
<b>Grey</b>	G	→ G	G	→ S	G	→ E	G	→ B
<b>Segment</b>	S	→ G	S	→ S	S	→ E	S	→ B
<b>Edge Mag</b>	E	→ G	E	→ S	E	→ E	E	→ B
<b>Bin Edge</b>	B	→ G	B	→ S	B	→ E	B	→ B

Generalisation

Specialisation

Table 3: Graded-abstraction testing set-up. For each of the three composite correlation filters (MACE, UMACe or MACH), four composite templates were designed based on the graded-abstraction training images with image-complexity gradation as in Table 2. Generalisation testing (green) examined detection of targets which are more complex than the training data; specialisation testing (pink) was the converse, with more image complexity in the training data than the target.

the generalisation and specialisation experiments are presented using ROC curves and area-under-curve (AUC) values [62]. The results are grouped by the type of training images used, i.e., following the rows of the table in Table 3.

### 3.3.1 Testing of G-template Design: G-G, G-S, G-E and G-B

The results presented in Figure 36 correspond to the first row of the table in Table 3. The corresponding area under curve table is shown in Table 4. Filter design was done using the full-detail (G) greyscale training images only, generating G-templates for each of the three composite correlation filters: MACE, UMACe or MACH. Testing of the resulting G-template designs was done on test images with all complexity levels: full-detail (G) greyscale, segmented (S) greyscale, greyscale edge-magnitude (E) and binary (B) edge.

In the G-G test, the performance of all three composite correlation filters was high. This means that all G-template designs are able to detect a large majority of the flowers under default testing conditions (training and testing images of the same complexity). The MACH and UMACe filters outperform the MACE filter, this is due to the performance limitation imposed by the MACE constraint equation (16).

In the G-S test, despite the lower amount of information in segmented test images, all three filters performed well. Although large regions of the image have a constant value, the segmented region boundaries still coincide with the original images so the correlation filters are able to detect the flowers. Interestingly, the MACH filter has the best performance ( $AUC \approx 0.92$ ). Due to the lower contrast between regions of the segmented regions, the MACE filter has the worst performance (see Section 3.3.2).

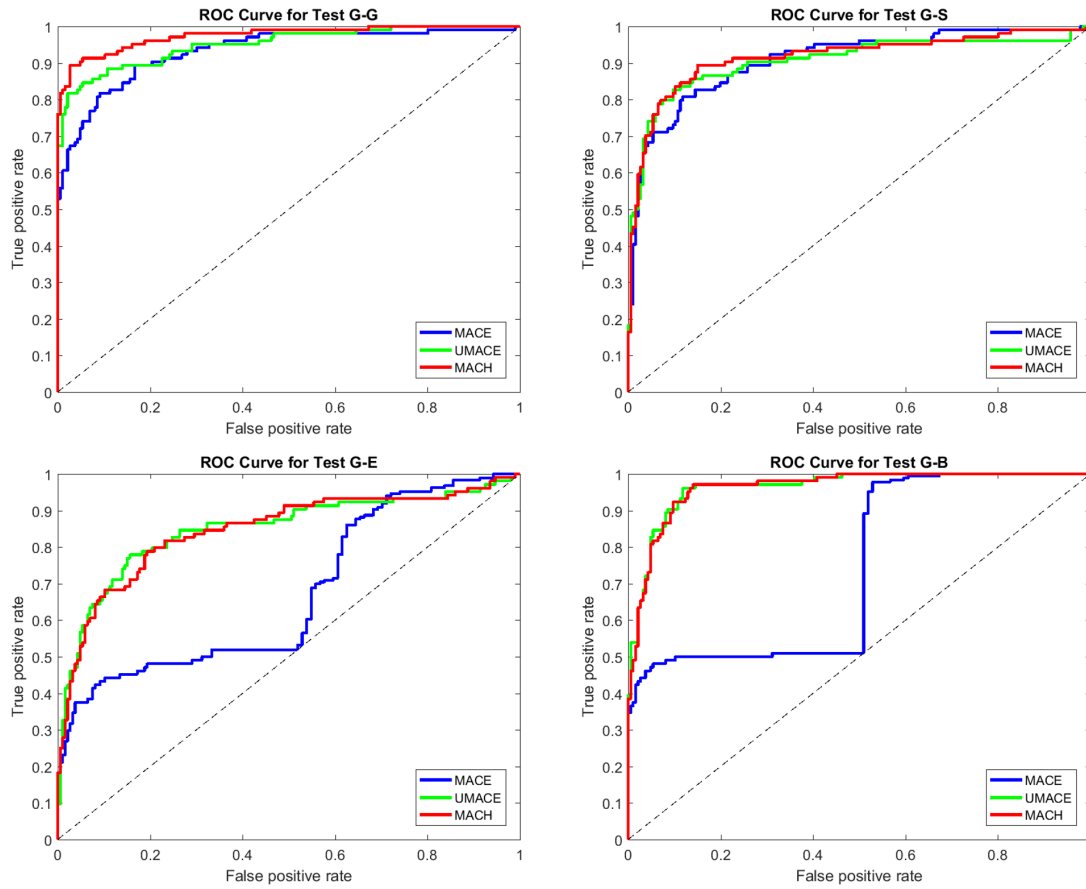


Figure 36: ROC curves using Greyscale training images for MACE, UMAC and MACH filters. Tests G-S, G-E, G-B are examples of generalisation tests where the training data contains more information than the testing data.

In the G-E test, the overall performance of all three filters is mediocre. Despite the edge-magnitude images still containing all of the key edges in a flower image, some of the weaker edges have little contrast. The limits of the MACE filter, due to equation (16), are clearly evident in the results (blue curve). Interestingly at a false positive rate of  $\approx 0.55$  the ROC curve falls below the chance line. At this point, although the filter is actually performing worse than random chance, it is still providing valuable feedback. By simply reversing the output at this point, useful feedback can still be obtained since the MACE filter is more often than not reporting the target to be present when it is actually not present. This is shown in Figure 36 where the MACE ROC curve touches the chance line, the response is reversed, effectively mirroring the ROC curve about the chance line.

In the G-B test, the ultimate specialisation case was examined. The G-B test results in Figure 36 show that the performance for both the MACH and UMAC ( $AUC \approx 0.96$ ) filters are very close to that of the G-G tests with the same filters ( $AUC \approx 0.98$ ). As with the other tests, the MACE filter did not perform very well as shown by its peak true-positive rate of about 0.5 until the false positive rate of  $\approx 0.5$ . This response from the MACE filter

is mainly due to the fact that the key flower characteristics usually manifest themselves in high-frequency regions corresponding to edges (present in both the greyscale and binary-edge images). The AUC of the MACH and UMACE filters are similar in both the G-B and G-G tests but the 100% true-positive rate for the G-B case occurs at a lower false-alarm rate ( $\approx 0.5$ ) in the G-B tests, compared to ( $\approx 0.7$ ) the G-G test case. This improvement occurs due to complexity reduction in binary (B) edge images (keeping the strongest edges only), resulting in only the essential high-frequency content remaining in the image.

	G-G	G-S	G-E	G-B
MACE	0.93	0.91	0.54	0.51
UMACE	0.95	0.91	0.85	0.96
MACH	0.98	0.92	0.81	0.96

Table 4: AUC table for the MACE, UMACE and MACH correlation filters trained using greyscale training images.

### 3.3.2 Testing of S-template Design: S-G, S-S, S-E and S-B

The results presented in Figure 37 correspond to the second row of the table in Table 3. The AUC is presented in Table 5. Filter design was done using the segmented (S) greyscale training images only, generating S-templates for each of the three composite correlation filters: MACE, UMACE or MACH. Testing of the resulting S-template designs was done on test images with all complexity levels: full-detail (G) greyscale, segmented (S) greyscale, greyscale edge-magnitude (E) and binary (B) edge.

Most of the tests which involve using segmented images seem to have poor performance in general, including the S-S case for the default training/testing set-up. The ROC curves for all the segmentation tests are shown in Figure 37.

For the S-G test, there is a noticeable discrepancy between the MACH, UMACE, and MACE filters. The MACH filter has the best performance ( $\text{AUC} \approx 0.96$ ), while the MACE filter only achieved  $\text{AUC} \approx 0.78$ . MACE filters are sensitive to noise and were trained on segmented images, a greyscale test image can be considered to have much more noise while still containing the same major edge information. Because of the extra noise due to the greater pixel intensity variation in a greyscale image, the MACE filter performs poorly. However, the unconstrained correlation filters (UMACE, MACH), are able to cope with noise much more effectively (see Section 2.3), as evident from the better results.

One might expect the S-S test to produce the best results out of the four segmented data tests as the training and test data is of the same complexity. The results are reasonable but the MACH filter performed much better in the S-G and S-B tests. The main reason for that better performance is due to the energy distribution in the testing images: when designing/training the composite correlation filter, the high-frequency content is key for

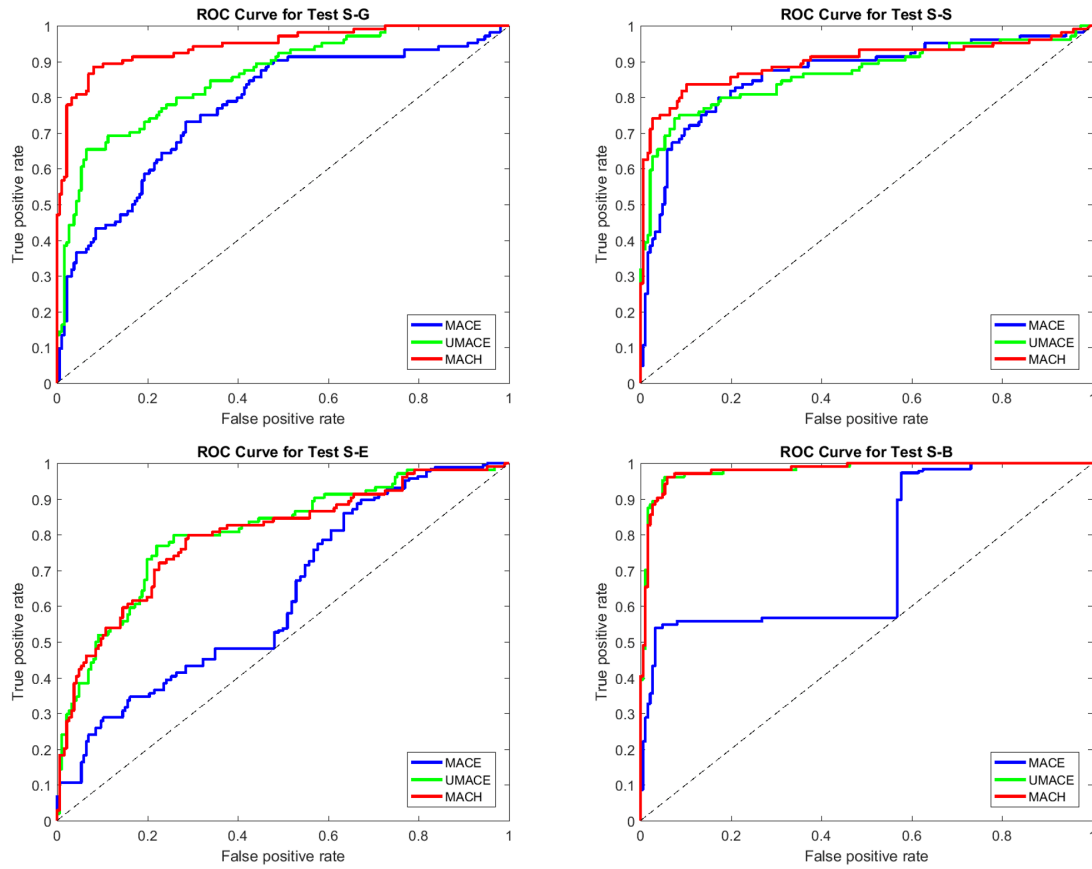


Figure 37: ROC curves using Segmented training images for MACE, UMAC and MACH filters. Test S-G is a specialisation test, whereas S-E, and S-B are generalisation tests.

discrimination. During the segmentation preprocessing, the test flower image is segmented into regions in only three colours. Sometimes the contrast between these regions is not very high. This is more noticeable when testing on segmented images as there is not much high-frequency content for the correlation filter to compare with. Hence, when the filter is applied to the image, the resulting peak and clear discrimination is difficult. Compared to the greyscale and binary testing images of flowers, most of the segmented images have high-frequency content around the edges of petals.

In the S-E tests, the MACE filter performed poorly ( $AUC \approx 0.49$ ), touching the chance line, causing the filter response to be reversed due to inadequate noise tolerance, as discussed previously. The MACH, UMAC filters performed better ( $AUC \approx 0.8$ ), but the same high-frequency content in the segmented training images is not always present in the edge-magnitude images.

Surprisingly, the S-B tests for the unconstrained correlation filters perform very well ( $AUC \approx 0.98$ ) for both MACH, and UMAC. Although both the binary and segmented images have the same high-frequency content, the binary edges are always at maximum contrast (including some extra noise), whereas the segmented image boundaries do not al-



ways have a high contrast, as discussed previously. The MACH and UMACE filters show high performance but the MACE filter performs the worst out of these S-template tests ( $AUC \approx 0.56$ ) due to poor noise tolerance.

	<b>S-G</b>	<b>S-S</b>	<b>S-E</b>	<b>S-B</b>
<b>MACE</b>	0.78	<b>0.86</b>	0.49	0.56
<b>UMACE</b>	0.86	<b>0.86</b>	0.8	0.98
<b>MACH</b>	0.96	<b>0.90</b>	0.80	0.98

Table 5: AUC table for the MACE, UMACE and MACH correlation filters trained using segmented training images.

### 3.3.3 Testing of E-template Design: E-G, E-S, E-E and E-B

The results presented in Figure 38 correspond to the third row of the table in Table 3. Filter design was done using the greyscale edge-magnitude (E) training images only, generating E-templates for each of the three composite correlation filters: MACE, UMACE or MACH. Testing of the resulting E-template designs was done on test images with all complexity levels: full-detail (G) greyscale, segmented (S) greyscale, greyscale edge-magnitude (E) and binary (B) edge. The corresponding AUC table for these tests is shown in Table 6.

As shown in the ROC curve for the E-G case in Figure 38, the performance of training the correlation filter based on edge-magnitude training images is relatively good ( $AUC \approx 0.91$ ). The reason for that good performance is that the greyscale test images still have all the high-frequency content, particularly at the stronger edges.

The ROC curve for the E-S case is mediocre for similar reasons to those discussed in Section 3.3.2. Since this is a reversed situation of the S-E test, the MACE filter still performs well as the training edge-magnitude images only emphasise the strongest of edges.

In the E-B test, the MACE filter has very poor performance ( $AUC \approx 0.63$ ), however, the UMACE, and MACH filters have relatively good performance ( $AUC \approx 0.98$ ). This is due to some of the weaker edges in the magnitude image appearing as strong edges in the binary edge image. Since the MACE filter is trained on the edge magnitude images, only the strongest of edges are incorporated into the filter. When testing on the binary edge images, some extra strong edges appear due to the thresholding (some of which are not necessarily part of the flower) which confuses the MACE filter due to the constraint in equation (16). The unconstrained filters (UMACE, and MACH) do not have this problem and are able to handle extra noise and distortions.

### 3.3.4 Testing of B-template Design: B-G, B-S, B-E and B-B

The results presented in Figure 39 correspond to the last row of the table in Table 3 with corresponding AUC in Table 7. Filter design was done using the binary (B) edge train-

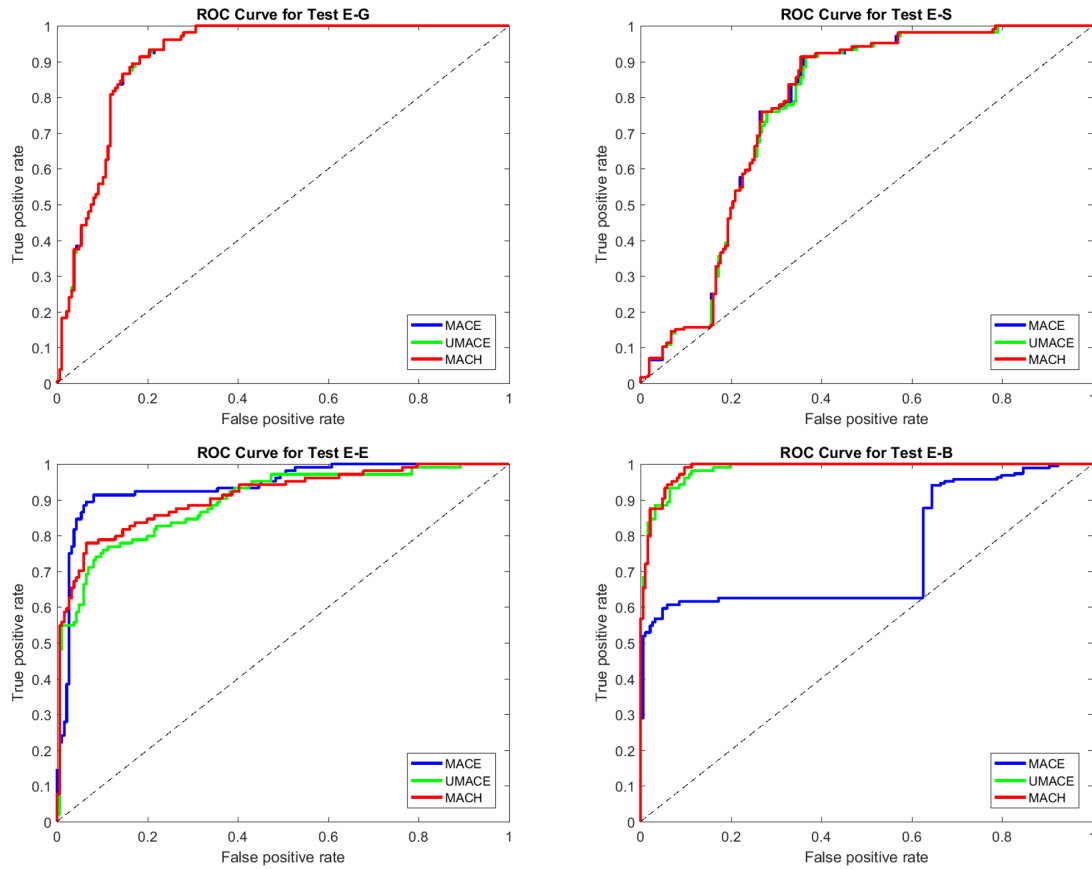


Figure 38: ROC curves using edge-magnitude training images for MACE, UMACE and MACH filters. Tests E-G, and E-S are specialisation tests. Test E-B is a generalisation test.

ing images only, generating B-templates for each of the three composite correlation filters: MACE, UMACE or MACH. Testing of the resulting B-template designs was done on test images with all complexity levels: full-detail (G) greyscale, segmented (S) greyscale, greyscale edge-magnitude (E) and binary (B) edge.

In the B-G test, all three filters showed very similar performance with an  $AUC \approx 0.91$ , see Figure 39. This test represents the greatest abstraction jump in the training/testing set-up. The filter is trained using the simplest form of data (edges-only, binary image), and is still able to detect flowers for the most complex input (full-detail, 8-bit greyscale image). This is consistent with the research by Horridge [82] which showed that honeybees are able to discriminate well using simplified patterns. These patterns can be thought of as skeletal representations of flowers which honeybees store in memory (relevance filtering), as shown in Figure 32.

In the B-S test, the ROC curves are similar ( $AUC \approx 0.75$ ), see Figure 39. The reason for the poor performance is the lower edge contrast in the segmented images.

In the B-E test, the MACE filter outperformed the MACH and UMACH filters. This is because the binary training images usually have more high-energy content than the edge-



	<b>E-G</b>	<b>E-S</b>	<b>E-E</b>	<b>E-B</b>
<b>MACE</b>	0.91	0.76	<b>0.94</b>	0.63
<b>UMACE</b>	0.91	0.76	<b>0.89</b>	0.98
<b>MACH</b>	0.91	0.76	<b>0.91</b>	0.98

Table 6: AUC table for the MACE, UMACE and MACH correlation filters trained using greyscale edge-magnitude training images.

magnitude images. During training, the MACE filter puts a lot more emphasis is on the high-frequency content, most of which is still present in the edge-magnitude image (the binary images are just threshold edge-magnitude images). Overall, the filters have reasonable performance with an AUC of 0.91 for the MACE filter, and about 0.82 for the MACH filter.

In the B-B test, all three filters perform well, with unconstrained filters demonstrating superior noise tolerance. The discrimination power of the MACE filter leads to better results with a higher true positive rate at false positive rates less than 0.1; however, the true positive rate is lower than the UMACE, and MACH for a false positive rate that plateaus between 0.1 and 0.8 with a true positive rate just under 0.88.

	<b>B-G</b>	<b>B-S</b>	<b>B-E</b>	<b>B-B</b>
<b>MACE</b>	0.91	0.75	0.91	<b>0.88</b>
<b>UMACE</b>	0.91	0.75	0.81	<b>0.96</b>
<b>MACH</b>	0.91	0.75	0.82	<b>0.97</b>

Table 7: AUC table for the MACE, UMACE and MACH correlation filters trained using binary edge training images.

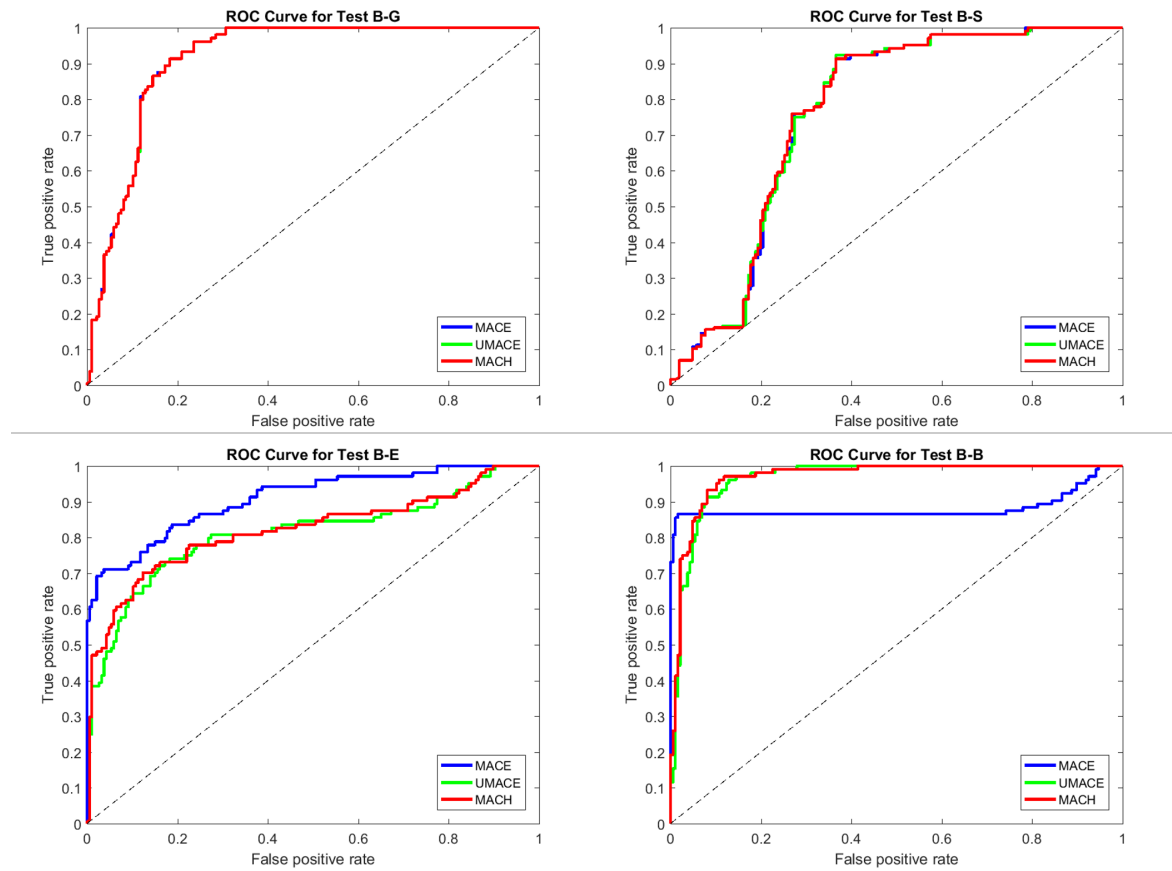


Figure 39: ROC curves using binary edge training images for MACE, UMACE and MACH filters. Tests B-G, B-S and B-E are specialisation tests, where the training data contains less information content than the testing data.

Method	Accuracy		Average Time (128x128 image)	Speed Improvement
	TP	FP		
Loy-Eklund	80%	17%	1.6 frames per second	baseline
Composite Correlation Filter	90%	5%	55.4 frames per second	34x faster

Table 8: Average performance comparison between the Loy-Eklundh method and the composite correlation filter approach on the (normalised and enriched) dataset from Section 3.2.4. For detecting rotational symmetry in flowers the composite correlation filter approach achieves better accuracy at a much lower computational cost.

### 3.3.5 Computational Performance

Efficient processing is a key advantage of using correlation filters. Digital correlation filters for object detection have been shown to run very quickly, even as fast as 600 frames per second [90] with optimisations. The tests performed in this thesis did not use any code optimisations other than those already built into standard Matlab functions. Since the filter is precomputed in the Fourier domain, the major computation required is to transform the input image into the Fourier domain. Despite this, each filter took just 2 milliseconds on average to run on each image on a laptop with an Intel core i7-4500U and 8GB of RAM, utilising all four cores. This average time of 2 ms includes multiplication between the DFT of the 128 square pixel image and the filter of the same size, and the IDFT back to the spatial domain (using built-in Matlab functions) as well as locating the peak.

The attained speed of 2 ms is only representative of the overall system performance, given that several filters and some other minor processing of the results is required in the final system. Compared to other state of the art flower-detection methods based on features (points of interest) (see Section 3.4), the correlation filter system proposed in this thesis is an order of magnitude faster as shown in Table 8.

## 3.4 Comparison with Feature Based Symmetry Detection

The Loy-Eklundh symmetry detector [35] is among the current state-of-the-art methods to detect symmetry in images [33] and is based on matching and grouping local feature points, as opposed to the global pattern matching framework used in the correlation filter based approach. Once a series of SIFT features [36] has been found in a given image, the algorithm searches through them and matches them based on symmetrical properties in order to find the dominant symmetries. The key attribute for matching symmetric features is the dominant feature orientation.

### 3.4.1 Results

In the work presented here, the Loy-Eklundh symmetry detector was implemented in Matlab in two variants. The first variant corresponds to the original implementation in [35] which

used SIFT features. The second variant is a faster version of the original algorithm in which the SIFT part of the algorithm is replaced with the simpler SURF features [91] resulting in an order-of-magnitude speed improvement with similar performance. In software implementation, the freely available OpenSURF library [92] was used as it contains a Matlab implementation of the SURF feature detector. Since the input to the Loy-Eklund symmetry detector requires a feature point vector, the SURF feature vector was arranged to be in the required form and processed the same way as SIFT features in the original algorithm.

A few examples of the results obtained for the symmetry detector are shown in Figure 40 showing that the Loy-Eklund algorithm is able to detect objects with bilateral or rotational symmetry on a variety of flowers including simplified representations of these flowers.

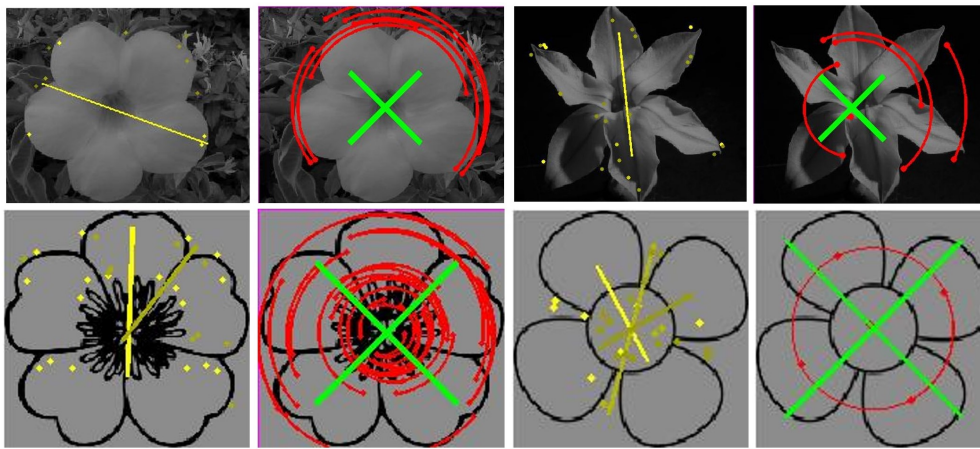


Figure 40: Loy-Eklund symmetry detection results for example flower images and also simplified representations of these flowers.

In order to directly compare the proposed composite correlation filter approach to the Loy-Eklund method, both algorithms were tested on the same (normalised and enriched) flower dataset, defined in Section 3.2.4. The average accuracy in terms of true and false detection rates over the whole dataset are presented in Table 8.

The results in Table 8 show that the composite correlation filter approach to rotational symmetry detection is considerably faster, achieving an average of 55 frames per second, compared to just 1.6 for the Loy-Eklundh method.

### 3.5 Conclusions and Interpretation

The main purpose of this work on static perception was to develop a fast and effective algorithm for rotational symmetry detection, suitable for realistic implementation and use on resource-limited platforms, especially micro air vehicles (MAVs). MAVs are autonomous flying machines, six inches in size, designed to detect symmetrical objects in indoor environments and cannot use computationally-intensive approaches. The bio-inspired algorithm proposed here is based on a pattern-recognition approach to symmetry detection, realised with composite correlation filters. Compared with a leading state-of-the-art symmetry de-

tection method based on points of interest, the proposed approach shows better accuracy in the ROC analysis sense and also results in an order of magnitude better computational efficiency, (see Table 8).

The bio-inspired character of the proposed approach was motivated by honeybee vision and has two aspects: 1) symmetry detection as pattern matching (detection with training examples), and 2) the use of simplified training examples (relevance filtering). The first aspect led to the use of composite correlation filters whose design entails learning by example: the gist of all training examples is expressed as a single composite template, and only this template is matched against test inputs. The second aspect was a realisation of the relevance filtering principle by giving the pattern-matching filters a generalisation capability at the design stage through a systematic choice of the training examples. The generalisation capability was defined here as the ability to detect a complex pattern in the test input after learning a less complex version of that pattern in training. Since the filters used here perform pattern matching by maximising the input/template correlation, a natural measure of complexity was the Haralick image correlation.

The algorithms defining composite correlation filters are optimised in the frequency domain and hence the analysis of the generalisation performance is most transparent in the frequency domain as well. In particular, two questions were posed at the beginning of this chapter: 1) which of the training patterns is the best abstraction of the full-complexity patterns? and 2) which of the simplified training patterns leads to better generalisation performance?

As for the first question, the answer is the binary (B) image with edges only as the most efficient abstraction of the full-detail original greyscale (G) image. The filters are trained with only the key information but are still able to reliably detect the targets in the full greyscale images. The best results were obtained for the case with the strongest contrast content corresponding to the binary edge training tests. See Section 3.3.4, especially the B-G test.

As for the second question, the answer is less clear cut but there is a trend pointing to the importance of the high-frequency content in the training and testing images. When there is a low amount of high-frequency content in either the training or testing image, the performance is reduced. This is clearly visible in all the filters that are tested on segmented (S) images which have the weakest high-frequency content. Hence, it is important for the high-frequency content to be available in the testing images, more so (to some extent) than the training images. However, noise is present in the high-frequency content of images as well and hence a design trade-off must be performed for which the MACH filter is well suited. See Section 2.3.3 and equation (24) and equation (25) in particular.

In Section 3.4, the bioinspired pattern matching approach proposed in this work was compared with a leading state-of-the-art symmetry detection method. The Loy-Eklundh method is fundamentally different as it is a point of interest based method. Although robust points of interest are used that are scale and translation invariant, the computational requirements for finding, and matching all of these features might not be suitable for a resource limited embedded system. On the other hand, the proposed pattern matching method using composite correlation filters are able to recognise the same symmetric objects with just a

single correlation operation, which offers a significant computational advantage (see Table 8).

Another advantage of the bioinspired approach is the existence of guidelines (see Section 3.2.1) on how many training images are required in order to build a correlation filter able to adequately identify regions in an image. This was extended to the context of information content in each image. By contrast, it is not clear how many local points of interest are required in conventional symmetry detection methods employing SIFT/SURF features like the Loy-Eklundh method discussed in Section 2.1.

## 4 Dynamic Pattern Recognition

In this chapter, a pattern based approach is applied to the problem of visual dynamic perception. The first contribution in this chapter looks at defining and recognising symmetry using motion. By inferring motion on patterns, it is possible to detect and categorise the symmetry in a given pattern. This method is shown to improve the detection performance of the static pattern based symmetry detection method proposed in Chapter 3.

The second contribution in this chapter introduces a dynamic pattern based method for recognising self-motion in a given video sequence. Using linear spatio-temporal filtering techniques, a method for defining motion in a video sequence is presented. This method had an advantage that it does not require explicit calculation of the optic flow field. Furthermore, the results from the spatio-temporal filtering techniques are then combined with similar pattern based methods (introduced in Chapter 3) to recognise self-motion in the video. This chapter is laid out as follows:

- Section 4.1 introduces the problem setting for this chapter.
- Section 4.2 uses motion to recognise and define symmetry using linear filtering techniques. This relies on an understanding of matched filters (see Section 2.2), and is an extension of the pattern based symmetry detection method used in Section 3.3. This method is also compared to a feature based symmetry detection method discussed in Section 2.1.
- Section 4.3 extends the correlation filter pattern matching approach to three dimensions in able to use video data. Instead of recognising objects, this method finds motion patterns in a video sequence. This section relies on concepts introduced in Chapter 3 and background information from Section 2.3.
- Section 4.4 utilises spatio-temporal filtering methods to describe motion in a video sequence that does not require direct computation of optic flow. This section relies of background information discussed in Section 2.5 and Section 2.6. Using three dimensional correlation filters, it is shown that simply motion types can be detected in a video sequence using a pattern based approach.
- Section 4.5 interprets the results and summarises the conclusions of this chapter.

### 4.1 Problem Setting

Symmetry as a concept is an intrinsically dynamic concept. For the case of rotational symmetry, as regions will match with one another as they are rotated about the axis of rotation. Similarly, for bilateral symmetry, symmetric regions will line up with one another when flipped about the mirror axis. The first section of this chapter looks at using motion to define symmetry of a given object to improve detection performance of the pattern matching based symmetry detection method used in Chapter 3. In this chapter, motion is applied to test patterns to recognise and define symmetry for each pattern.

Visual dynamic perception is a very important aspect for both flying insects and mobile robotics. Visual dynamic perception facilitates navigation, obstacle avoidance and closed loop motion control. Many visual dynamic perception problems are studied using differential optic flow techniques based on the temporal and spatial derivatives [63]. These methods are divided into local or global approaches, where local methods such as the Lucas - Kanade method [64] have the advantage of being robust to noise. Global approaches such as the Horn and Schunck method [65] provide a dense flow field which where discontinuities can be used to locate object boundaries. However these global methods are known to be sensitive to noise [93]. The second part of this chapter proposes an alternative pattern based approach that uses spatio-temporal filtering techniques to estimate the optic flow field (see Section 2.6). This method also has an advantage of providing a quality measure of each flow vector. Furthermore, correlation filter techniques are used to recognise specific motion patterns (see Section 4.3).

## 4.2 Symmetry Based on Motion

Approaches to symmetry detection for static patterns (as discussed in Section 3.4) be they feature based or correlation based, are still limited in that they only look at a single frame, however, the concept of symmetry itself is intrinsically dynamic. By considering the problem of detecting symmetry to be related to a systematic sequence of frames, where the object under test is rotated, or flipped about its centre in each frame, not only can the symmetry of an object be detected, but also defined. The approach presented in this section is able to recognise both rotational and bilateral symmetry using a two-step matched filtering approach.

The core assumption of this approach is that the region under test needs to be known prior. This assumption is met by using composite correlation filters (as discussed in Section 2.3) to locate regions of interest (due to symmetry) in a given scene that correlate highly to the filter. However, any other object detection method could be used to locate a potential region.

This approach for symmetry detection can improve the performance of a target detection system, assuming the targets to be identified exhibit symmetrical properties. This assumption is not unreasonable since symmetry is present in most objects, both natural and man-made. By adding the symmetry detection step to an object detection method, the accuracy can be improved as symmetry can be used as a sanity check to reduce false alarms. By lowering the thresholds of an object detection method the chance of missing an object is reduced as the cost of more false alarms. But by testing each positive detection for symmetric properties some of these regions can be ruled out (if no symmetry is present) reducing the false alarm rate of the detector. The true positive results are unlikely to be filtered out by this step since they will already exhibit strong symmetry based on the correlation filtering step. This is achieved using some known symmetric properties of the object to be found. As an example, if we are looking for dandelion flowers, but a sunflower region is returned by the detector, a rotational symmetry test will show that there are many more axis of rotational symmetry than one would expect for a dandelion (a dandelion typically has 6 petals whereas a sunflower has approximately 34 petals).



### 4.2.1 Using Matched Filters to Define Symmetry

Once a potential region containing the object has been located using a correlation filter, the region around the object is cropped to be the new test image,  $x$ . The size of this test image will be the same size as the correlation filter template that was used to detect it. A series of geometric transformations are applied to the test image to form a series of templates,  $h$ . Transformations include rotation for rotational symmetry detection (detailed in Section 4.2.1), and a combination of rotation and flipping for bilateral symmetry detection (detailed in Section 4.2.1). By finding the correlation peak between the test image and each template, a correlation vector is used to compare the normalised correlation energy peak across the various transformations. This allows the symmetric properties of the test region to be detected efficiently. An illustration of this process can be found in Figure 41. There will always be a peak at  $0^\circ$  since the image and template will be identical, so this will always be ignored. It should also be noted that in this work the motion is artificiality produced via geometric image transformations. A similar approach can be implemented with the motion of the camera itself. However, in this case, aspects such as illumination changes, or camera perspective distortions will need to be addressed.

**Rotational Symmetry** By rotating the template,  $h$ , between  $0^\circ - 180^\circ$  at a designed angle step, a collection of  $h_i$  templates of the same size<sup>6</sup> at angles between  $0^\circ - 180^\circ$  are created. The test image  $x$  is correlated with each template  $h_i$ , and recording the centre correlation value for each angle in a  $1 \times i$  dimensional correlation vector. By looking along this correlation vector a clear pattern of a stronger correlation values can be seen when the rotated templates align best with the original image. The number of local peaks along this single dimensional vector represents the order of rotational symmetry. Similarly, the angle separation between the peaks indicate the angle separation between each axis of radial symmetry of the test image. An illustration of this process is shown in Figure 41, where the correlation vector is plotted for each angle of rotation, the angle range has also been extended between the range of  $0^\circ - 360^\circ$  for illustration purposes.

**Centre Cropped Average Correlation Energy** When looking at the correlation values, ambiguous situations occur when either the test object circular, or exhibits no symmetrical properties. If only looking at the value associated with the correlation, both the circular object, an a non-symmetric object will have no clear peaks (other than at  $0^\circ$ ) in the one dimensional correlation value. This is particularly true for non-symmetric objects that have large regions of smooth texture.

To obtain reliable results for symmetry detection, it is important to only consider the maxima in the central region of the correlation plane. This is particularly applicable for distinguishing between a circular object such as a flower, and an object with no symmetrical properties. An example image where the object is not symmetric, but still possesses a large average correlation energy is shown in Figure 42. If the maximum correlation value on the

<sup>6</sup>In the case of rotating the image, the image will be cropped to be the same size as the original image  $x$ . To maintain the same image size, at certain rotations, extra zero values pixels are added

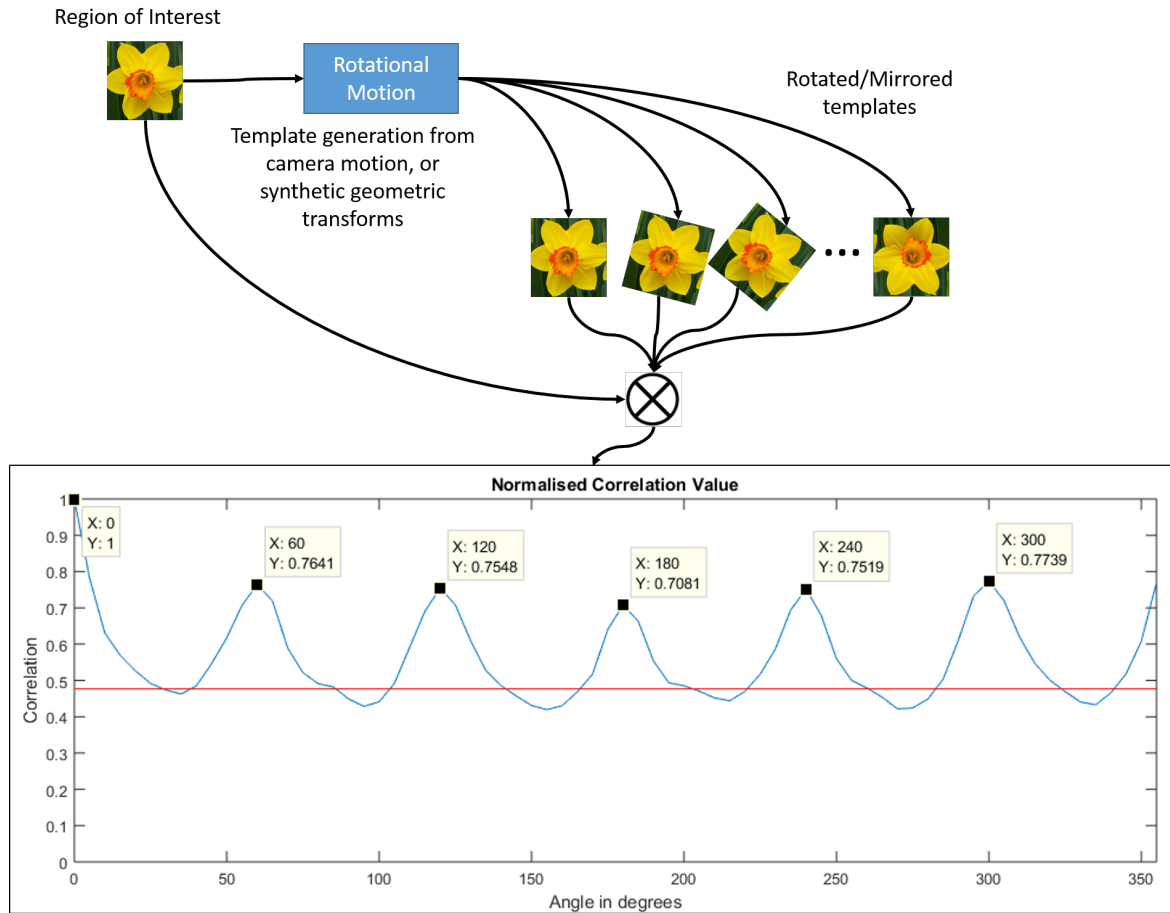


Figure 41: Illustration of a symmetry test, where the region of interest is extracted, and transformed to form new templates. Maximum correlation peaks are obtained when the rotated/mirrored image closely matches the original image, indicating an axis of symmetry. The total number of peaks in  $0^\circ - 360^\circ$  indicate the order of rotational symmetry the object exhibits.

entire correlation plane is considered there are some cases when certain regions of the template will correlate with regions in the test image, giving false positive readings (see Figure 42 (left)). Similarly, depending on the object, the overall amount of energy in the correlation plane might also be very high when large regions in the image and rotated template are similar. By simply recording only the maxima located to be within the centre of the correlation plane, a peak is only obtained when the object exhibits symmetrical patterns (see Figure 42 (right)).

It is also important to consider the correlation energy, or relative height of each of the normalised correlation peaks to avoid false detections. In the case of rotationally symmetrical images, all the correctly identified peaks (representing an axis of symmetry) have the strongest peaks with a similar correlation value. It was found that counting peaks within 20 percent of the maximum peak produced acceptable results. This value was obtained in an

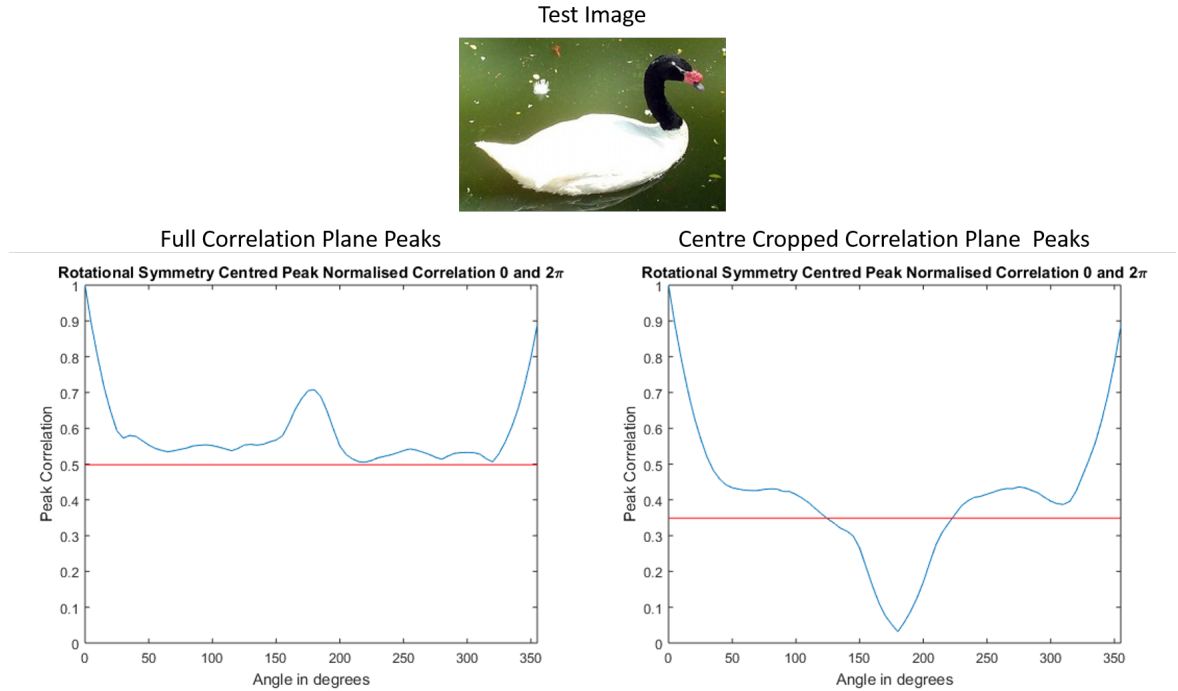


Figure 42: An example image where a false detection takes place because of the large regions of smooth texture (left). No peak (other than at  $0^\circ, 360^\circ$ ) is formed when looking only at the cropped central correlation plane indicating no rotational symmetry (right).

ad-hoc manner, based on testing on the flower dataset (see Section 3.2.4). The first peak (at  $0^\circ$ ) in the correlation vector is not included since this will always be high as the template and test image are identical.

**Bilateral Symmetry** The process for finding bilateral symmetry is very similar to the previous method discussed for rotation symmetry. For each angle rotation, an additional step is to horizontally flip/mirror the template array before correlating it with the test image. Assuming an object that exhibits bilateral symmetry is centred in the image, there will be a specific angle (reflection axis) between the rotations around  $0^\circ, 180^\circ$  where the mirrored version of the original image will produce a correlation peak. An example of this for an image with bilateral symmetry is shown in Figure 43. In this figure, the maximum correlation value is plotted at each rotation step of the template. The angle on this plot is not the actual angle of the axis of reflection, but rather the angle of the rotated template. To obtain the axis of reflection this angle is subtracted from  $180^\circ$  (because the template is mirrored).

For reasons explained previously it is important to only look at the energy in a central region in the correlation plane. This avoids some ambiguities when an image has large regions of smooth texture. Again it is important to also consider the strength of the peak in correlation to avoid false positives. Because the template is always mirrored, it is not possible to achieve a perfect correlation value of 1 unless the object is perfectly symmetrical. As the template is rotated, there will be a template angle that closely matches the test image,

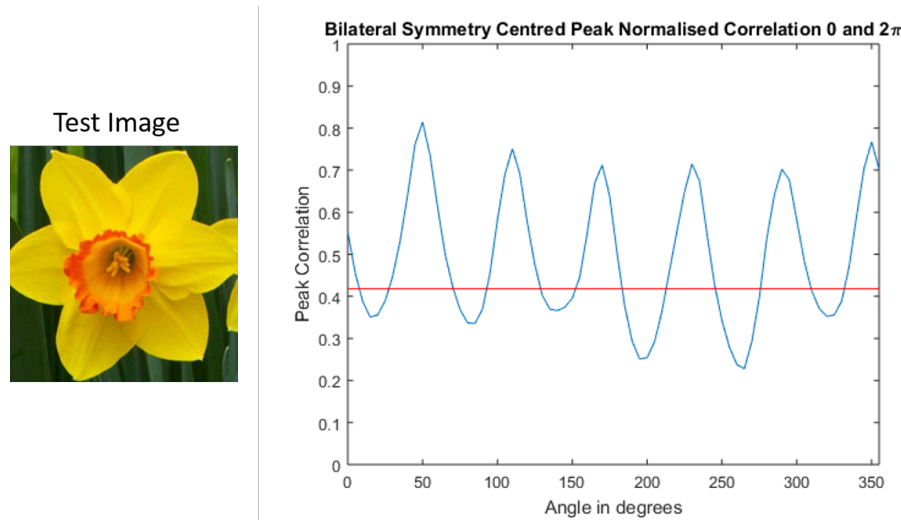


Figure 43: An example of the correlation peak vector for a bilateral symmetry test. Each peak represents a possible axis of reflection. Notice that the peaks all have approximately the same amplitude. Note that the y axis represents the angle of the template, not the angle of the axis of reflection.

represented by a peak. This would indicate a possible axis of reflection. However it is also important to consider the relative energy, or value of each correlation peak as some objects might have partial symmetries as shown in Figure 44.

In Figure 44 the main peak is clearly visible at a template rotation of approximately  $190^\circ$ , which would correspond to an axis of symmetry at  $10^\circ$ . Two weaker peaks are found at  $90^\circ$  and  $170^\circ$  due to the low aspect ratio of the wings, and the fuselage of the aircraft causes the bilateral symmetry test produce these extra peaks. Across the tests it was found that these weaker symmetries have significantly lower peaks. Likewise, all of the major bilateral symmetry axes have the same stronger peaks on the correlation vector, so it is easy to distinguish between them. As a general guide (based on ad-hoc testing), if the correlation peaks are not within 30% of the strongest peak, they do not count towards a bilateral symmetry axis.

#### 4.2.2 Symmetry Recognition Results

In order to quantify and understand the performance of this symmetry detection method a test was conducted using the flower dataset described in Section 3.2.3 which is a collection of flower images with no perspective distortions. 100 of the images from this dataset were manually selected as true class images as the flowers exhibit both approximate rotational and bilateral symmetry. These images also represent the type of regions the flower symmetry composite correlation filters are expected to extract. 50 negative class images were obtained from various public domain sources which do not exhibit any rotational symmetry in the scene. These images were used for the rotational symmetry test. For the bilateral symmetry test another 50 negative class images that exhibited no symmetry were also obtained

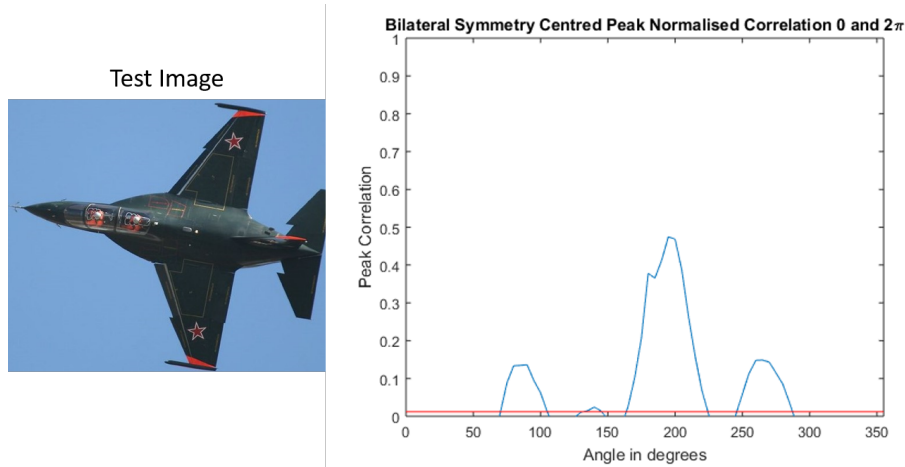


Figure 44: An example of the correlation peak vector for a bilateral symmetry test. Here due to the weak symmetrical properties of the aircraft, low aspect ratio wings cause weak peaks to form on the correlation vector.

from public domain sources. The number of true class and negative class images used was chosen based on the available images at the time of the test. The results in terms of true detection rates and false alarms are shown for both methods in Table 9.

When animals (including humans) look for a specific object in a scene, the first process is to have a quick scan for potential matches before focusing on a specific area to decide if it is the actual target object [94], [95]. By doing this, computation is focused on relevant areas only. It is not important that the exact symmetry/symmetries are detected, but rather that an object shows an appropriate level of symmetry or not. So the output is binary to test if symmetry exists for a given object.

Method	Accuracy	
	TP	FP
Rotational Symmetry	100%	2%
Bilateral Symmetry	100%	11%

Table 9: True positive (TP) and false positive (FP) detection rate for the rotational and bilateral symmetry detector

As can be seen in Table 9, this simple test shows that this method is able to recognise rotational symmetry with a very low false alarm rate. The bilateral symmetry detector was still able to correctly identify when there was bilateral symmetry in an image, however it is more susceptible to false alarms. The reason high true detection rate is that this test was not conducted to test if this detector would exactly locate all axis of symmetry, but rather if the object exhibited symmetry or not. This is a more lenient test requirement and is also closer related to biological methods [95].

**Improving correlation filter performance** As mentioned at the start of this section, a question is asked if this method of using motion to recognise symmetries can be used in combination with an object detection method to increase performance.

This is based on the assumption that the objects to be located exhibit symmetrical properties. To answer this, the flower detection results from the MACH correlation filter (as used in 3.3.1) are further processed by the symmetry recognition from motion method proposed in this section (see Section 4.2.1).

Once a region has been found using the MACH correlation filter, this region is then passed through the symmetry recognition test. If no symmetry is found for that region it is no longer regarded as a true detection. This way the second step can potential prune out any false alarms. The results are compared with the MACH filter on its own, and with symmetry recognition step in Figure 45.

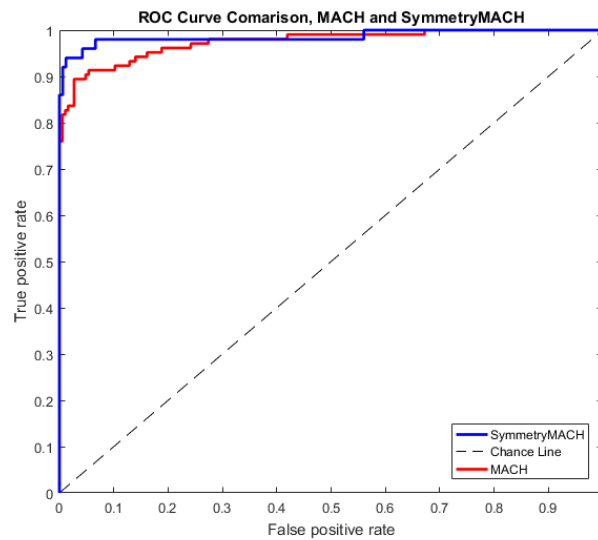


Figure 45: ROC Curve comparing a MACH correlation filter with and without symmetry recognition post processing. The symmetry recognition post processing is shown to increase detection performance.

As can be seen in Figure 45, adding the extra symmetry recognition step is shown to increase overall performance. The AUC for the standard MACH filter is 0.975, compared to 0.986 after the symmetry recognition step. The symmetry recognition post processing step increases the accuracy to  $\approx 0.98$  opposed to  $\approx 0.90$  at a false positive rate of less than 0.1. At lower false positive rates the improvement is

#### 4.2.3 Conclusions and Interpretation

This section has shown that the simple paradigm of symmetry being dynamic, it is possible to recognise both bilateral and rotational symmetry in objects using simple linear filtering techniques. Although both methods are able to recognise if symmetry is present, the bilateral symmetry detector still needs some refinements to decrease the false positive rate.

**Integration** It has been demonstrated that this symmetry recognition test can be used as a to improve detection performance. Since the type of object the correlation filter is looking for is known (from training) the symmetrical properties of that object can be inferred. Once the composite correlation filter finds a potential match, that region is further processed with this symmetry recognition to rule out false positive results as shown in Figure 45.

Let us consider the example of detecting daffodil flowers which have clear rotational symmetry via the petals distributed around the centre. The composite correlation filter will locate symmetric regions that are likely to contain the flower, along with a certainty metric (the value of the normalised correlation peak). Passing the less certain regions through the rotational symmetry test will either confirm or deny the location as being a flower based on its symmetry, improving the accuracy of the detection. As shown in Figure 46, the same two images are considered where one of them has been blurred. From the correlation filter point of view the second blurry image produces a weaker correlation filter response, so is more uncertainty if that region is a flower or not. However, when it is passed through the rotational symmetry recognition test, the blurry image is still recognised as exhibiting symmetry. This allows us to use a lower threshold for object detection via the correlation filter to obtain a higher true detection ratio while pruning the false alarms through additional processing.

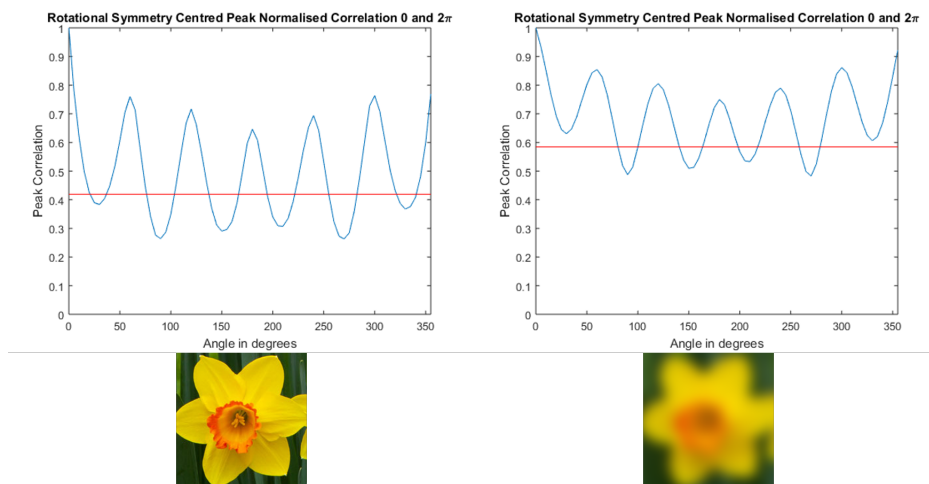


Figure 46: An example of two regions detected by a composite correlation filter. The first region (left) has a higher detection certainty than the second region (right), however, the rotational symmetry test still produces the same peaks, allowing us to still confirm the second region as being a correct detection.

Furthermore, from a biological point of view, the healthiest flowers typically exhibit the highest levels of symmetry as no petals are missing or deformed. When bees react strongest to symmetric objects, this is instinctively a way to collect pollen from the best flowers. A honeybee does not necessarily care about what order of symmetry an object possesses, rather if it has strong symmetrical properties or not. The symmetry recognition method proposed in this section follows a similar principle of being able to recognise if a region exhibits strong symmetry through the correlation value.

**Self Motion Corrections** In addition to just detecting symmetries, this method can be used to help improve self-motion estimations. As discussed in Section 2.5, a major difficulty with optic flow is disentangling translational and rotational motion. By tracking a stationary object and its symmetrical properties the problem can be resolved. Once a symmetric object has been found, the peak correlation values from the rotational symmetry test can be tracked. By comparing the angles between each peak, and the axis of symmetry of the object as the camera (vehicle) rotates above it, the rotational motion of the camera (vehicle) in a single direction can be estimated. However, this might not be entirely feasible to disentangle self-motion from independent object motion.

**Limitations** The main limitation of this approach is perspective distortions have not been explicitly addressed in this work. However given that if a specific composite correlation filter has been trained to recognise an object at a specific azimuth elevation range, standard perspective mapping and correction techniques can be used to correct this before passing the image through the symmetry detection process. However further experimentation is required to validate the situations when this is possible.

Due to the limited processing capacity of insects, it is doubtful that this approach is fully utilised by insects to use a template for each possible angle. The test patterns used to study symmetry in honeybee static perception was largely conducted using symmetries aligned to the horizon [10], and discussed in Section 1.2.1. This could suggest that honeybees are only able to recognise symmetrical axis at limited angles. In reality this is a reasonable assumption, as in the case of bilateral symmetries the mirror axis is mostly aligned to perpendicular to the horizon in the case of natural and man-made objects. So rather than using all possible rotations to find the bilateral symmetry axis, the computations could be drastically reduced to only look at an axis perpendicular to horizon, or along the direction of flight.

### 4.3 Pattern Based Motion Recognition

In this section, a pattern based method is proposed to classify self-motion using spatio-temporal features. Extending the approach as used by Wildes and Bergen a new method is for recognising types of motion in the spatio-temporal volume. This method does not require the direct computation of optic flow as it uses linear spatio-temporal filtering techniques on motion energy volumes to recognise motion. This approach is motivated by how the human visual system recognises motion (see Section 2.6).

In Section 3.3, it was shown that composite correlation filters are effective at accurately finding objects in a scene very quickly and efficiently. By training these filters to store an abstracted representation of the desired object to lead to increased efficiency (see Section 3.3), motivated by how honeybees recognise patterns. However, the methods only consider a single frame. This section investigates if these concepts can be used to detect dynamic patterns for self-motion estimation, by extending correlation filters to the three dimensions of the spatio-temporal volume.

In order to extend correlation filters to detect dynamic patterns, these correlation filters are applied to the spatio-temporal volume. As discussed in Section 2.3.3, a MACH



correlation filter is trained from many instances of an object to create a single filter which optimises various performance metrics (ACH, ACE, ASM and ONV). However, this creates a two dimensional filter which, when correlated with a test image, will output a correlation plane where the peaks indicate potential occurrences of the trained object in the given scene. This concept can be extended to video sequences by treating the required motion to be detected as a sequence of 2D correlations filters that occur one after another. This concept was first used in [59], for human action recognition, and has proven to be an effective method using composite correlation filters to recognise human actions in video sequences. It has since been extended to make use of integral images for real time computation [96]. This approach for human action recognition on the spatio-temporal volume was motivated by [97], which uses optic flow vectors to detect certain actions. However, by using correlation filters, the limitation of requiring direct computation of the optic flow field can be addressed.

**A Bioinspired Approach** This idea of self-motion pattern recognition follows the concept that insects use a form of matched filtering for motion detection from their receptors response (EMD's). Deeper inside the vision system, neurons are tuned to respond sequentially to specific patterns to recognise specific types motions (as discussed in Section 1.2.2). 3D correlation filters applied to video sequences can be likened to this concept where they are tuned to respond strongly (through a strong correlation peak) when exposed to certain patterns the filter has been trained on. In this case, the filter will be trained to recognise specific motions, such as translational self-motion along a specific direction. This approach is highly specialised, as the insect detects only certain types of motions that correspond to the typical flight manoeuvres the insect will perform.

The problem of self-motion estimation is a widely studied field with a strong focus on optic flow calculations [98], of which most rely on optic flow calculations [99]. There has also been recent bio-inspired work on creating hardware that performs in a similar way to how insect eyes work, with multiple lenses and motion detectors [100] [101]. However, in this section, the problem is posed as utilising data from standard hardware cameras using the spatio-temporal volume.

#### 4.3.1 Motion Testing and Training Dataset

In order to train and test the composite correlation filters, a library needs to be constructed of self-motion video sequences. To test this concept, a set of simplified motions were recorded from a canonical top down view at constant altitude (1 meter above ground level) perpendicular to the ground. These motion sequences are similar to what downwards looking stabilised MAV flight would look like. This data requirement is easily achievable with a gimbal, or optically stabilised camera. Video sequences include translational motion along 8 cardinal directions at slow, and fast speeds. The slow speed corresponds to  $1\text{ms}^{-1}$  which is a slow walking speed for most people, and the fast speed corresponds to  $3\text{ms}^{-1}$ , which is similar to a jogging speed. Both of these correspond to typical speeds a small indoor MAV would travel at. Depending on the application a separate training dataset might be required to closer match the dynamics of the vehicle. Similarly, a sequence of rotational motion se-

quences were constructed for slow and fast rotations. Both clockwise and counter clockwise rotations were considered. For both rotational and translational motions a variety of patterns and textures were used. These included grass, sand, asphalt, concrete, leaves and others as shown in Figure 47.

Obtaining controlled real world test footage can be difficult as controlling a flying vehicle to fly at an exact altitude and speed can be challenging. So a computer game simulation engine was used to generate the required training dataset of a camera flying through along an exact path in 3D space. The ARMA3 game engine was used and a script was written to move the camera at the pre-defined motion paths at specific speeds and distances. Using these sequences 10 main motion categories were obtained, 8 for the cardinal directions, and 1 of each rotational direction. Each of these motion categories is further divided into fast and slow motions.

Before the training the correlation filters, the videos were split into a set number of frames. The length of the frame chunk can be adjusted depending on the type motion the filter is being trained on. Within each category, the number of individual training clips is dependent on how frames are used for each chunk. For each of the 10 motion categories, at least 15 motion sequences at varying lengths were used.

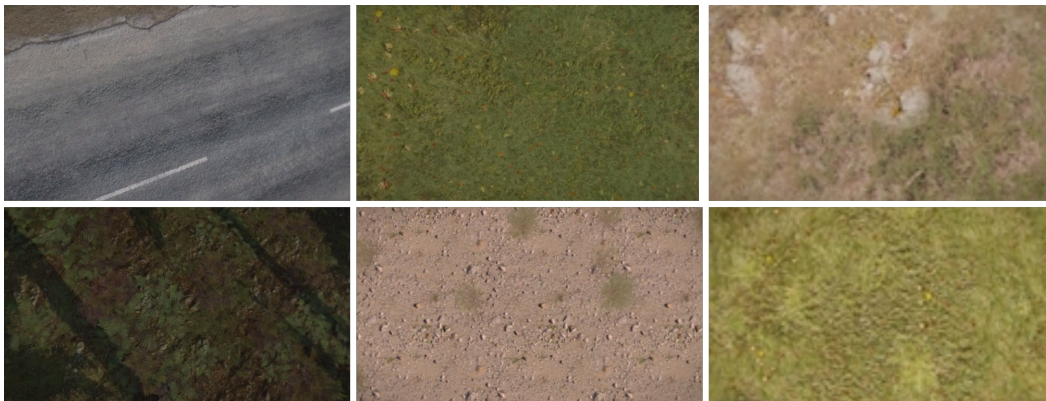


Figure 47: Single frames of selected top down motion training sequences.

**Sequence Resolution and Duration** Because the composite correlation filter is implemented as a sliding window function for translation invariance, the dimensions of the filter are important factors to consider. However, for motion estimation, the  $(x, y)$  resolution of the filter will be the same for both the training video and testing video as global motion needs to be detected. The filter resolution could be reduced to detect specific motion patterns in local regions of a video. This could be used to detect specific regions of rotational motion, indicating a potential area of interest.

#### 4.3.2 Using Pixel Data for Motion Estimation

For action recognition, using scalar (grey-scale pixel) data has been shown to work well [59] as each action can be broken down into a set of sequential poses of an object that needs to oc-

cur in a certain order. This takes into account the spatial appearance of the object, and when the correlation filter is trained using many training poses, it learns the spatial variations. The reason for this is that the correlation filter on the  $(x, y)$ -slice behaves exactly like the 2D composite correlation filter that was discussed in Section 2.3. That filter accurately locates where the target is in the slice based on spatial appearance. However, in the case of trying to estimate background motion using purely scalar intensity data, the standard filter trained across a wide variety of textures can no longer distinguish between them. This means that using pure pixel intensity scalar data, it is not possible to distinguish motion patterns in this manner, as each frame (along the  $(x, y)$ -slice) looks almost the same. Only in some situations do the temporal slices  $((y, t)$  and  $(x, t))$  provide the motion information. An example of this is shown in Figure 48.

To explain this problem two applications are considered. In the first, a 3D MACH filter is trained to recognise a person bending down to pick up an object. An example of the trained 3D correlation filter shown in Figure 48 left. The second is a 3D MACH filter trained on a camera translating in an Easterly direction across a variety of textures (see Figure 48 right). Both videos are pre-processed to only contain edges. This aids in speeding up computation as the MACH filter will contain many zero entries in the 3D Volume as discussed in Section 3.3. Because the edges on each frame of the east translational motion sequence never align (due to variations in textures) the resulting 3D MACH filter has almost no structure. This results in very poor discrimination ability. However, since the motion is consistent, for any  $(x, t)$ -slice a pattern is visible as the pixels move from the right to the left of the video sequence. Alternatively, for the bend action sequence (shown in 48 left). Training on a specific object using spatial data provides a clear structure, looking at the  $(x, y)$ -slices the target is visible.

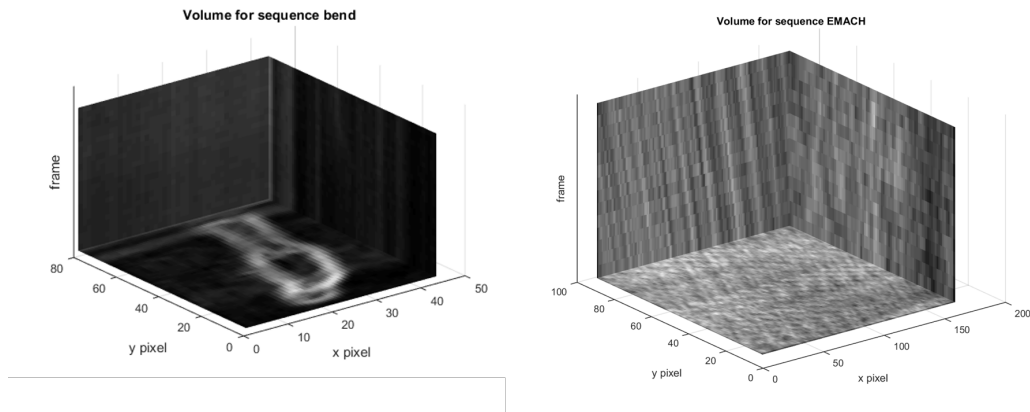


Figure 48: Slices of a 3D composite correlation filter volume. (left) Filter trained to detect human bending down motion. (right) Filter trained to detect translational motion towards the Easterly direction.

The most obvious approach to solve this problem is to pre-process the spatio-temporal volume to use motion vector data to train and test the 3D correlation filter. One approach to find the motion would be to calculate the optic flow field as discussed in Section 2.5.

However, for suitability for use onboard an MAV, simplicity and efficiency is important. The need to explicitly calculate dense optic flow adds extra computation overhead. By slicing the spatio-temporal volume along the temporal axis, this work tests if certain types of motion can be found using only pixel data directly.

For coherent translation along a set direction, there will always be a regular pattern of pixel motion formed on one of the temporal slices. For example, for motion along the North, or South direction, the pixels form a line on the  $(y, t)$ -slice. For East and West motion, the pixel motion forms a line on the  $(x, t)$ -slice. In order to recognise these motions only the patterns on the  $(y, t)$ -slice and  $(x, t)$ -slice need to be considered. The  $(x, y)$ -slice of the spatio-temporal volume can be ignored.

**Translational Motion Along Cardinal Directions** Due to how a pixel moves through the spatio-temporal domain, the North, and South motions create patterns on the  $(y, t)$ -plane. The East and West motions create patterns on the  $(x, t)$ -plane. However, in situations of motion along other directions (not parallel to a slice) such as North West, coherent pixel motion is not able to be found on a single  $(x, t)$ , or  $(y, t)$  slice. Nonetheless assuming a simplified case of a vehicle only being able to move along the 4 cardinal directions, a test was conducted to see if a composite correlation filter is able to recognise these basic motion patterns using pixel intensity values.

Rather than using the full volume to train a 3D MACH filter for each motion, only four 2D MACH filters are trained along the  $(x, t)$ -slice and  $(y, t)$ -slice:

- North and South MACH filter trained and tested on the  $(y, t)$ -plane
- East and West MACH filter trained and tested on the  $(x, t)$ -plane

Ten  $(y, t)$ , and  $(x, t)$  spatio-temporal slices were extracted for each of the four motion types. These slices were further split into training and testing datasets. The testing dataset includes labelled slices of all motion types including extra slices that were added from other motion sequences such as rotation, North West motion etc. . .

As shown in Figure 49, 2D MACH filters are able to recognise the patterns in the spatio-temporal slices with an accuracy of around 80% while keeping false alarms less than 10%. However, these test results are for a very limited set of situations. The testing motion sequences are the exact same two speeds as the training data (fast and slow sequence). Although each correlation filter is able to distinguish between the two speeds, as more speeds are introduced, extra confusion will occur. However, the most significant issue is that these correlation filters can only recognise pure translational motion. This could be extended to recognise non-linear motions such as acceleration. Rather than seeing straight lines on the slices, curves would be formed.

**Other Types of Motion** As soon as the motion occurs outside of the 4 major cardinal directions, the motion patterns are not visible on the  $(y, t)$  and  $(x, t)$  temporal slices alone. This is because the motion of a given pixel is not perpendicular to the slices through the volume, so for a given slice the motion of that pixel cannot be tracked in a single slice.

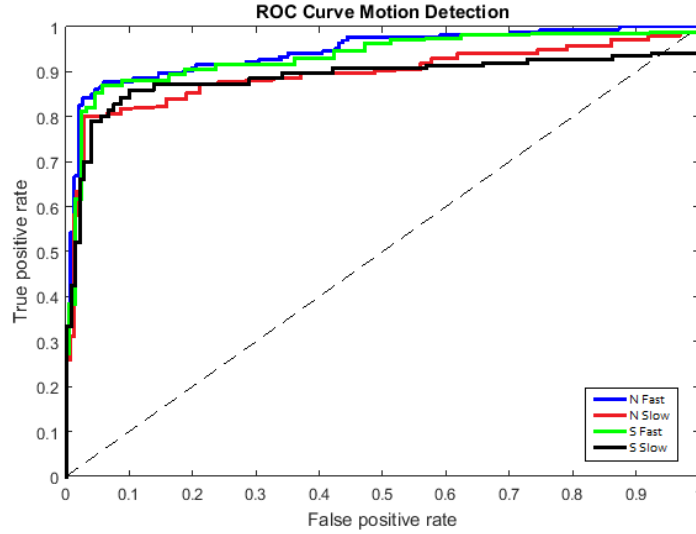


Figure 49: ROC curve for translation motion recognition on the  $y, t$ -slice.

In the case of constant rotational motion, a single pixel will move in a helical motion along the volume for each frame. At the centre of rotation, a vertical pattern occurs since the centre pixels do not move. Although the pixels do move in a regular pattern along the slices it is difficult to detect a pattern as the same pixel is not visible along every row of a temporal slice. This is the case for translational motion perpendicular to the slice. An example of rotational motion is shown in Figure 50, where the pattern varies through each slice in the volume of rotational motion.

One solution to detecting motion not parallel to the  $(y, t)$ , or  $(x, t)$  slices, is to slice the spatio-temporal volume at non-arbitrary angles to detect motions. For example in the case of North West translational motion, the motion of the pixel along a slice in the direction of motion would form a straight line. However, due to the number of slices required to recognise a wide range of motions, it is not computationally efficient. Other questions include how many angled slices are required to effectively recognise motion patterns, and what are the memory requirements to compute all of these extra slices? Also what happens when there is motion due to other objects in the scene? This makes it rather difficult to practically detect motions using just scalar data.

Using this approach of a 3D MACH filter with scalar data is best suited to locating deformable dynamic targets which can be defined by spatial appearance. For example, people performing certain actions and not general background motion. However, by extending the filter to use other higher level data (such as flow fields, or temporal energies) the recognition of motions can be improved. This is possible because there is less variation in the spatial plane in the data for a given motion sequence. This is discussed in the next section.

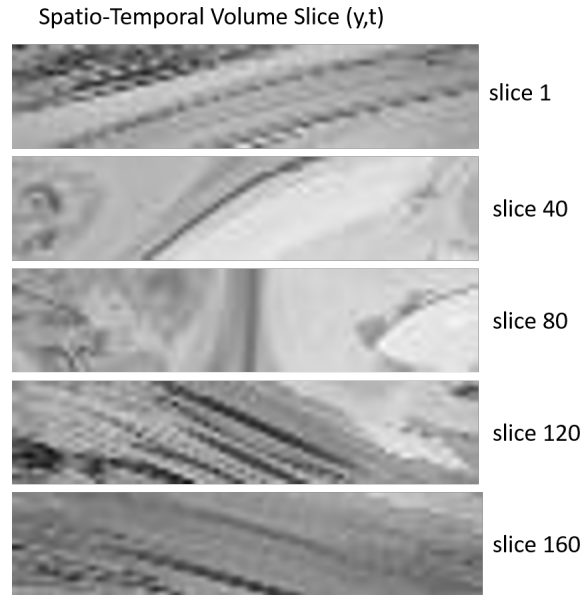


Figure 50: Spatio-temporal slices spaced evenly along a the  $(y, t)$ -plane for a purely rotational motion on a top down scene. The pattern of pixel motion is mirrored on either side of the centre of rotation.

## 4.4 Spatio-Temporal Pattern Based Motion Detection

For motion detection, using vector data such as optic flow will help to generalise the correlation filter. Matching vector data allows us to match motion vector patterns across each slice of the video volume, not just pixel intensities. For the same motion sequence in various scenes, the general optic flow pattern will remain relatively constant, variations will be due to noise, and other ambiguities, such as lighting changes.

Fitting vector data directly into the correlation filter framework is not possible as the standard Fourier transform is not generalised to use vector data, as associated with vector flow fields. Rodriguez [59] showed that MACH filters can be generalised to use optic flow vector data through the employment of Clifford algebra in three dimensional Euclidean space. This allows for a Clifford Fourier transform functions to be used on vector field data. However, this involves a considerable amount of the extra computation not suited to use onboard an MAV. This section will rather look at using spatio-temporal energy distributions along with 3D scalar composite filters applied to these energy distributions to identify motion patterns in a video sequence.

### 4.4.1 Spatio-temporal Energy Volumes

To investigate the spatio-temporal volume the motion energies need to be calculated. To do this, the training videos as described in Section 4.3.1 were used. Second order separable steerable Gaussian filters based on the work in [74]. This method has seen extensive adoption, specifically in the field of visual attention by [102], [103], [104]. These methods of

using the spatio-temporal volume are inspired by recent advances in Neuroimaging [105] and studies on the human visual system [106]. Which indicate that humans use models to focus attention based on visual cues. Motion is an important aspect, but other features such as colour and context are also important.

The spatio-temporal energies are found using equation (56) as separable 3D steerable filters derived from [74] with a size of  $6 \times 6$  pixels<sup>7</sup>:

$$E(\phi, \theta, x, y, t) = [G_{\text{even}}^{\phi, \theta}(x, y, t) \star I(x, y, t)]^2 + [G_{\text{odd}}^{\phi, \theta}(x, y, t) \star I(x, y, t)]^2 \quad (58)$$

with the following spatial Gabor directions in the  $(x, y)$ -plane:

$$\phi \in \{0^\circ, 45^\circ, 90^\circ, 135^\circ\} \quad (59)$$

and the following spatio-temporal Gabor directions in the  $(x, t)$ -plane:

$$\theta \in \{0^\circ, 45^\circ, 90^\circ, 135^\circ\}. \quad (60)$$

Angles  $\phi$  and  $\theta$  are defined consistently with (55). For a given time  $t$  (video frame number), features calculated with (59) require only intra-frame data from the input video, i.e., the  $(x, y)$ -plane data. On the other hand, features calculated with (60) are truly spatio-temporal, i.e., they require inter-frame data from the  $(x, t)$ -plane and  $(y, t)$ -plane.

This filtering results in 13 spatio-temporal features<sup>8</sup> in the volume. However, only 8 of them contribute to motion. A description of each filter is given in Table 10. Each of these volumes takes up a large amount of memory, so limiting the computation to only the critical volumes required for the application is important. The convolution theorem of the Fourier transform is exploited to improve computation depending on the size of filter and videos used (see 2.2.2).

Using these features, the energy of each filter is recorded and stored in a histogram which describes the type of motion in each region of the spatio-temporal volume. Since motion is the primary interest, only the 8 motion energy filters will be considered. For example, if we consider the case of a bar moving from left to right, the right and diagonal right motion components of the histogram exhibit the most energy as shown in Figure 51. This histogram for energy distributions shows that the motion in that region is clearly in a rightwards direction. So using these energy histograms, a spatio-temporal volume can be segmented into regions exhibiting similar types of motion. This example is for a simple video of a rightward movement. For more complex scenes, the histogram does not always provide a clear indication of the direction of motion, as indicated by incoherent motion in Table 1.

<sup>7</sup>The filter size was empirically tested to produce the best results best response to larger smooth motion patterns for the test video resolutions

<sup>8</sup>There are only 13 instead of 16, since the  $0^\circ$  in (60) is only used for the  $0^\circ$  in (59) to find the temporal energy in the volume (see Table 10).

Name	Detection
$E_1$	Temporal
$E_2$	Right to left motions
$E_3$	Vertical edges
$E_4$	Left to right motions
$E_5$	Diagonal motions: bottom right to top left
$E_6$	Diagonal edges: bottom left to top right
$E_7$	Diagonal motions: top left to bottom right
$E_8$	Up motions
$E_9$	Horizontal edges
$E_{10}$	Down motions
$E_{11}$	Diagonal motions: bottom left to top right
$E_{12}$	Diagonal edges: top left to bottom right
$E_{13}$	Diagonal motions: top right to bottom left

Table 10: Description of the spatio-temporal energy features calculated using a bank of steerable Gabor filters as defined in equation (58).

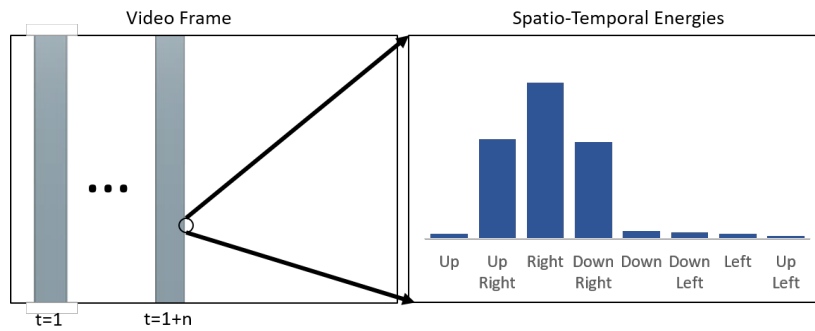


Figure 51: Histogram (right) of oriented spatio-temporal motion responses for a bar moving from left to right (left).

**Spatio-Temporal Energy Distributions for Self Motion Recognition** Once a video segment has been filtered using the spatio-temporal filters, each pixel in the spatio-temporal volume can be characterised by 8 spatio-temporal motion features for each of the 8 cardinal directions. By combining the responses from each of the 8 motion energy volumes for each  $(x, t)$ -slice in the volume, certain types of motion can be recognised. A simplistic approach to find which energy feature has the highest response to find the direction of travel. Although this could work under ideal situations it is better to combine this with the location of each motion. For example, in the case of plane rotational motion, the combination of motion directions would be close to zero as each side of the image has opposite motion and no clear direction can be found.

However, by splitting the volume into regions, more insight can be gained into the global motion present. For clockwise rotational motion, the motion on the left region would always be downwards, while in the right region the pixels would be moving upwards. An example of this is given in Figure 52. The sub regions near to the quadrant boundaries are not included, as the histograms in the border regions are often a combination of high energy from both



quadrants. This can lead to incorrectly representing the motion in that quadrant, this is specifically the case with rotational motion.

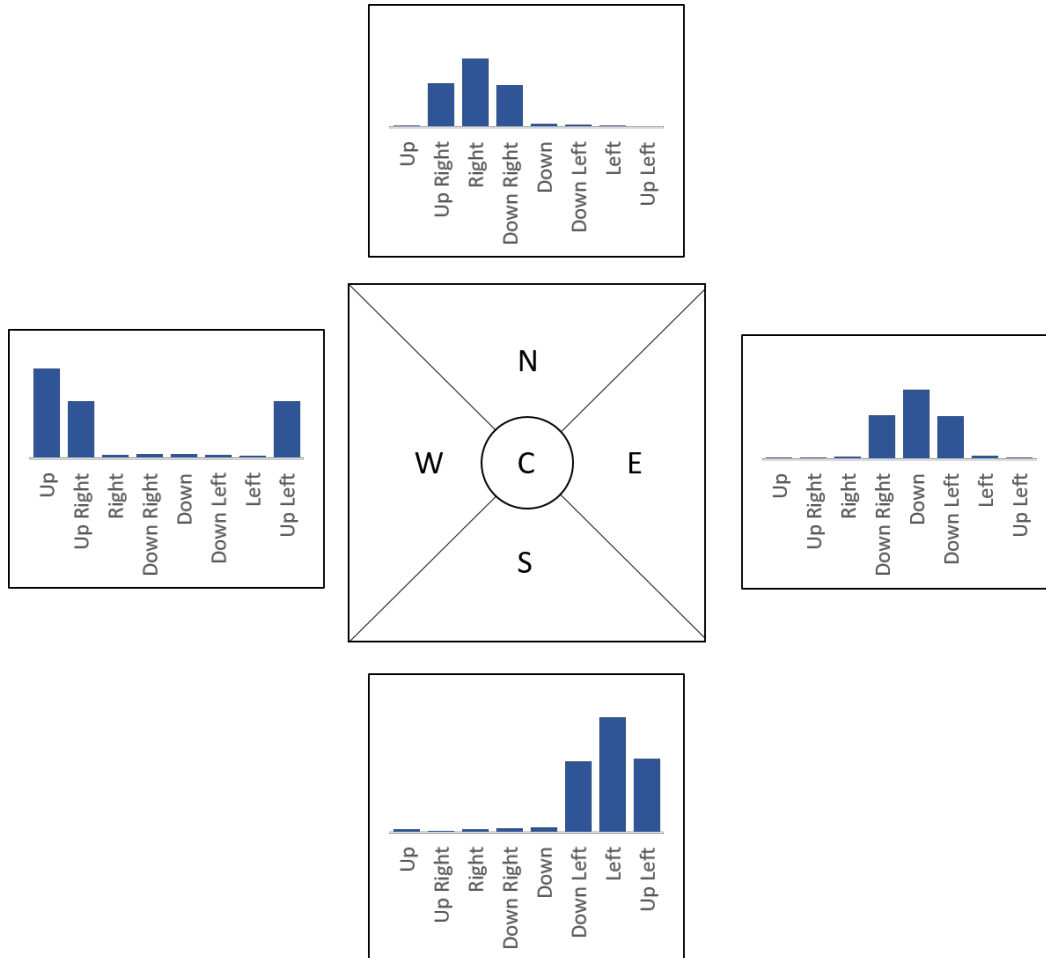


Figure 52: For constant motion, the energy histogram will show specific responses for each oriented energy filter. The histograms are representative for a clockwise rotation motion.

For translational motion, all the responses in each quadrant will be similar along the direction of motion. In the case of zooming in/out (change in altitude) all the quadrants would have opposing motions inwards or outwards. And for rotational motion, each opposite quadrant would have opposing motion with the top and bottom quadrants exhibiting strong left/right motion and the side quadrants exhibiting strong up/down motion. By obtaining the average of the histograms for each region, some understanding of self-motion of the camera can be found.

This method is able to provide some heuristics of the motion in the scene. However, it is largely qualitative information. For situations when there are large regions of smooth texture, this approach breaks down. This is a situation that will confuse even the human visual system. Also with situations such as rotational motion, near to the quadrant boundaries the energy histograms will look quite similar. Looking at the average of each quadrant

for rotational motion is not the best approach. Furthermore, there are some situations (eg. incoherent motion, or scintillation) which do not produce conforming energy distribution histograms. This is the main difficulty to overcome with this approach. For more complex motion types, the quadrant boundaries would need to be changed dynamically for each type of motion. This adds further complexity. Alternatively, this notion of certain motion types producing particular patterns on the image plane can be looked at from a pattern matching method using correlation filters can be used (see Section 2.3.4).

#### 4.4.2 Minimising Spatio-Temporal Energies

As discussed in Section 2.6, the spatio-temporal energy vectors can be calculated across the volume. To recognise global motion, only the motion components of the spatio-temporal energy vectors are used. For each of the type of motion, the direction vectors will look similar following reasons as discussed in Section 4.4.1, all-be-it at different magnitudes.

Using the spatio-temporal energy distributions alone to train the correlation filters will not be ideal for two reasons. First, the spatio-temporal energy response volumes still rely on the spatial appearance of edges. This is because the energy response is strongest along edges. This in itself is very useful for locating targets. Secondly, and most importantly, due to the number of volumes involved<sup>9</sup>, a large number of correlation filters will need to be processed for each volume, for each motion type to be detected.

To solve both these issues, low-level spatio-temporal energy volumes are used to construct three higher-level energy volumes. These three volumes characterise the essential motion information: a direction volume, a dispersion volume and a temporal energy volume.

**Spatio-temporal Direction Volume** Since the angle of each steerable filter is known, it is possible to work out the direction components of each pixel in the volume. This direction will be between  $0^\circ - 360^\circ$ . The average for each point across all volumes are obtained, to include all contributions of motion energies. However, simple average is insufficient when looking at angles. To calculate this, a circular average is applied across all of the  $n$  spatio-temporal volumes (8 volumes in this work) using the following equation (61) from [107] to find the prominent direction  $d$ :

$$d(x, y) = \arctan \frac{\sum_i^n E \sin(\theta_i)}{\sum_i^n E \cos(\theta_i)} \quad (61)$$

where  $\theta$  represents the angle of motion on the  $(x, t)$ -plane the oriented spatio-temporal feature is detecting.  $E$  is the energy for the given energy volume for the pixel.  $d$  is the direction for that pixel.

This is calculated for each frame across all the directional energy volumes to give a new single direction volume. However, there are some situations of incoherent motion where the sum of cos and sin of the energies cancel each other out, resulting in a non-prominent direction. For this, the dispersion volume is used.

<sup>9</sup>At least 8 spatio-temporal motion energy volumes are required to cover basic motion directions (see Section 4.4.1)

**Spatio-temporal Dispersion Volume** The dispersion volume is a measure of incoherent motion for each pixel. The direction volume simply adds the energy cos and sin responses together. By considering the distribution of energies through a dispersion value, it is possible to define if the given region exhibits a strong single coherent motion, or rather a combination of incoherent motion directions.

To find the dispersion ratio, the sum of energies for each orientation across the volume is found, represented by the Oriented Energy Sum, or OES. By taking into account the projected components of each energy for each orientation, the Component Energy Sum, or CES is found. By dividing these two values, the dispersion ratio is obtained. This is shown in equation (62) and illustrated in Figure 53.

$$0 < \text{Dispersion} = \frac{\text{CES}}{\text{OES}} < 1 \quad (62)$$

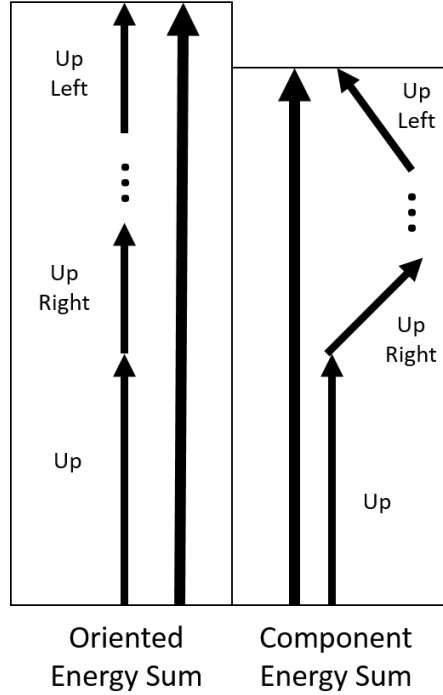


Figure 53: The dispersion of each direction is found by comparing the sum of oriented energies with the sum of component oriented energies as given in equation (62).

When this value is close to one, this means that both values are similar, and here is a clear prominent direction. When this value is closer to zero, there is no prominent direction. Therefore, this region has no clear motion structure, this similar to a situation scintillation like motion. By applying a threshold to the dispersion, only the areas with a prominent singular direction can be used. This effectively filters out regions of incoherent motion or scintillation. In this work, for the motion video sequences, the threshold was manually set such that coherent motion regions to have a dispersion value greater than 0.5.

**Temporal Energy Volume** The temporal energy volume is simply the energy from the temporal filter (see Table 10), giving a representation of which regions in the volume are moving the most. In reality, there is almost never a situation of zero temporal energy as there is always some residual energy caused by changes in illumination. By using this temporal volume a threshold can be applied to segment regions that are stationary. This threshold is usually dependent on the scene and in this work, it was manually tuned. Furthermore, this temporal energy can be used to find the speed of motion. However, this will require camera calibration and is something that future work could address (see Chapter 5).

**Using Vector Components** As previously discussed in this section, a correlation filter cannot be used with vector data directly, and the Clifford Fourier transform is too computationally expensive [59]. An effective work around is to simply use the vector directional components. To obtain the components, the direction volume is first pruned by removing any vectors that a dispersion less than 0.5. The remaining direction volume is then scaled with the temporal volume to produce a vector field volume. This volume is split into two separate scalar volumes, one for the  $x$ , and another  $y$  components of the vectors. This allows the standard 3D correlation filter to be used with each scalar component in parallel. A detection is only considered if both components return a match. An overview of this process can be seen in Figure 54.

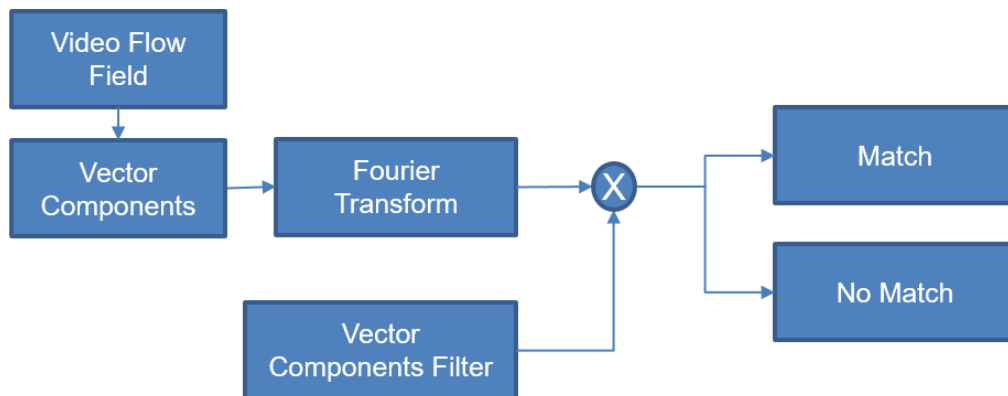


Figure 54: The steps involved in using Spatio-Temporal 3D correlation filters to recognise motion patterns.

It should also be noted that the extra computation of using a correlation filter for each vector component is still significantly faster than performing a Clifford Fourier transform on the volume.

**Comparing Spatio-Temporal Vectors to Optic Flow Vectors** As mentioned previously this approach is not limited to only spatio-temporal flow vectors; other vectors such as optic flow can also be considered. As a comparison, the same test was conducted with the flow fields calculated using the classical Horn Schunk optic flow method as discussed in Sec-

tion 2.5.2. This is shown in Figure 55, for a typical scene of rotational motion containing relatively high contrast in the textures. Both vector fields are only showing the directions.

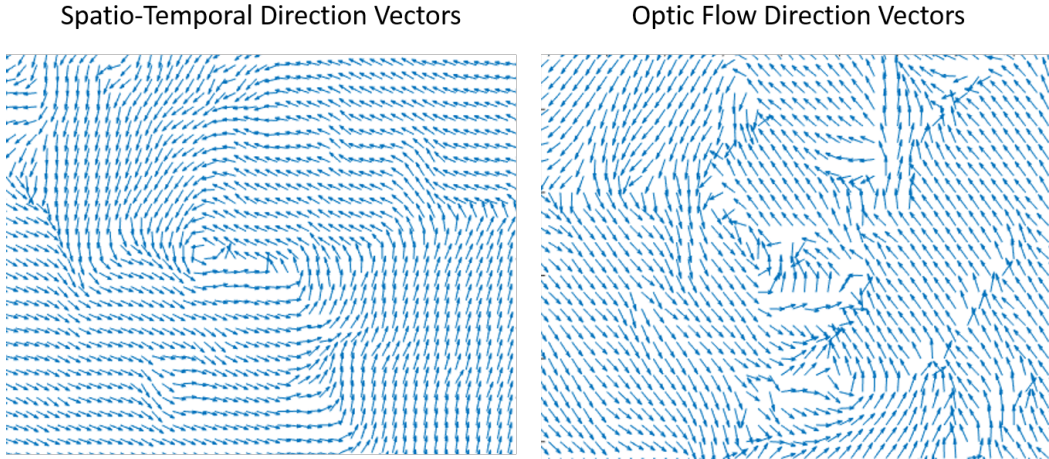


Figure 55: Comparison between motion direction vectors for in plane rotation. The spatio-temporal motion direction vectors are shown on the left, while the Horn Schunk optic flow field (Section 2.5) is shown on the right. The spatio-temporal flow vectors have been filtered to exclude any vectors with high dispersion values (see Section 4.4.2). The magnitude of both vector fields have been uniformly set to be the same for comparison purposes.

It is shown that for this basic scene of plane rotation, the directions of the optic flow vector field and spatio-temporal vector fields are similar. There are also discontinuities in both methods. For the optic flow method, these discontinuities are mainly due to the image flow constraint imposed on the optic flow derivation in equation (52). The optic flow vectors produced are perpendicular to the high contrast boundaries which do not necessarily align with the actual motion. On the other hand, using the spatio-temporal filtering method still produces some discontinuities. This is due to a function of the temporal size of the steerable filters used, causing greater sensitivity to certain motion speeds. However, the spatio-temporal dispersion allows us to filter out any uncertain regions due to incoherent motion, scintillation or flickering. This results in vectors that are consistently in a single direction that are used to improve the accuracy relative to optic flow for self-motion estimation vectors.

#### 4.4.3 Testing and Results

In order to test the ability of 3D correlation filters to be trained to recognise simple motions, a simple experiment was conducted using six categories of motion. Four for translational motions along each cardinal direction and two in plane rotations were used. The motion sequences from the motion dataset were used (see Section 4.3.1). Each training sequence was limited to just six frames. This is the same size as the spatio-temporal oriented filters. Both the testing and training video sequences had a resolution of  $256 \times 256$  pixels. For

testing, longer video sequences of three seconds were used. This corresponds to approximately 75 frames each, as not all test video sequences were exactly the same length. Each test sequence only exhibited a single motion over a variety of natural and man-made textures as recorded from the computer game simulation.

Spatio-temporal filtering was first conducted on the video sequence to obtain two component volumes per video. One volume for the  $x$  vector components, and another for the  $y$  vector components (see Section 4.4.2).

For each motion, the component MACH correlation was trained in parallel, resulting in two MACH filters per motion, an  $x$  and  $y$  component MACH filter. The  $x$  component MACH filter was correlated with the  $x$  vector components volume from the test video. Similarly, the  $y$  component MACH filter was correlated with the  $y$  component volume from the test video sequence.

Using the correlation thresholds obtained during the training phase of the MACH filters (see Section 2.3.4), each component MACH correlation filter was applied to every motion sequence. If the correlation was higher than the threshold, for both the  $x$ ,  $y$  component, the motion was considered to be detected.

Five of the motion sequences in each category included a small object that moved independently of the scene in the opposite direction to test the noise tolerance. The results are presented in Table 11 as a positive classification confusion matrix for spatio-temporal motion patterns.

Motion	N	S	E	W	CW	CCW
N	<b>0.84</b>	0.01	0.03	0.02	0.08	0.15
S	0.02	<b>0.87</b>	0.00	0.00	0.15	0.12
E	0.00	0.02	<b>0.89</b>	0.04	0.10	0.11
W	0.00	0.04	0.01	<b>0.86</b>	0.09	0.13
CW	0.10	0.06	0.09	0.5	<b>0.75</b>	0.17
CCW	0.03	0.12	0.09	0.13	0.14	<b>0.79</b>

Table 11: Confusion Matrix for motion classification. Mean accuracy is 83%.

The tests resulted in a mean accuracy of just over 80% with very few false alarms. There are possibilities for the false alarm rate to increase when the motion directions are less separated, or more complex motion patterns are included in the motion classification set. However, this test fundamentally shows that by using 3D MACH correlation filters, it is possible correctly and accurately classify simple self-motions in video sequences. Although these tests are relatively controlled, they have shown the ability of the correlation filters to recognize motion patterns through the spatio-temporal volume. Further work could look into the limits on the robustness which will be a function of both the MACH correlation filter training, but also the spatio-temporal Gabor filters used.

The results in table 11 show that the correlation filters are able to distinguish between the different directions. This is because the motion vectors produced by the different motion

directions each have fairly distinct vector components. For example, North motion results in the  $x$  components being very small, while the  $y$  components are large. Comparing this to an East motion will have large positive  $x$  components with small  $y$  components. A second test was conducted to look at the ability to discriminate between fast and slow motions along the same direction. Here the vectors will all have a similar direction but differ in magnitude due to the speed difference. In this test, a correct detection is counted only when both the direction and the speed are correctly identified. It is worth remembering that the motion dataset includes a fast speed and a slow speed (see Section 4.3.1), so the correlation filters are only recognising between these two classes. The results for both the North direction and the Clockwise direction are shown in Figure 56.

The results presented in Figure 56 show that filters trained to detect slow and fast motions are able to distinguish between the two, with both the slow filter and fast filter performing well. Just as with the direction test (see Table 11), filters trained for translational motions are more accurate. This is likely to be due to the fact that during a translational motion most of the motion vectors are fairly constant across the entire volume both in terms of direction and magnitude. This makes it easier to discriminate between other translational motions. However, in the case of rotational motion, although the magnitude may be similar, the directions form a circle around the centre of rotation (see Figure fig:ST-OF-Vectors), so the direction of the vectors are only constant in certain regions.

Despite the attempts to correct the spatio-temporal flow field directions, there was still additional noise in the vector fields produced. The composite correlation filters were still able to produce positive results correctly identifying most of the motion patterns. Future work can investigate this issue: some of the energy responses are influenced by both the filter and the image itself. This means that a strong energy response for one of the spatio-temporal filters could be due to motion, or clutter in the scene. A method of normalisation as discussed in future work, (see Section 5.1.2) where the effects of spatial appearance are limited, resulting in the direction vector only being influenced due to motion can be used to improve the results.

An interesting aspect of this work is the computational efficiency. The method of using vector components directly is compared to the state-of-the-art method proposed in [59] that uses vector data directly through the use of the Clifford Fourier transform to build a vectorised MACH filter. The computational efficiency comparison here is only comparing the extra cost of using vector components compared to performing a Clifford Fourier transform. Both approaches can accept any flow vectors, so they could come from an established dense optic flow method, or from the proposed spatio-temporal filtering approach in this thesis. By using vector components the processing time to correlate both component MACH filters with a  $144 \times 180 \times 200$  video sequence was 0.8 seconds. This is significantly faster than 2.32 seconds using the Clifford Fourier transform method. Both tests were computed within Matlab, using an Intel core i7-4500U with 8GB of RAM, utilising all four cores. The Clifford Fourier transform used functions included in the Clifford Multi-vector Toolbox[108].

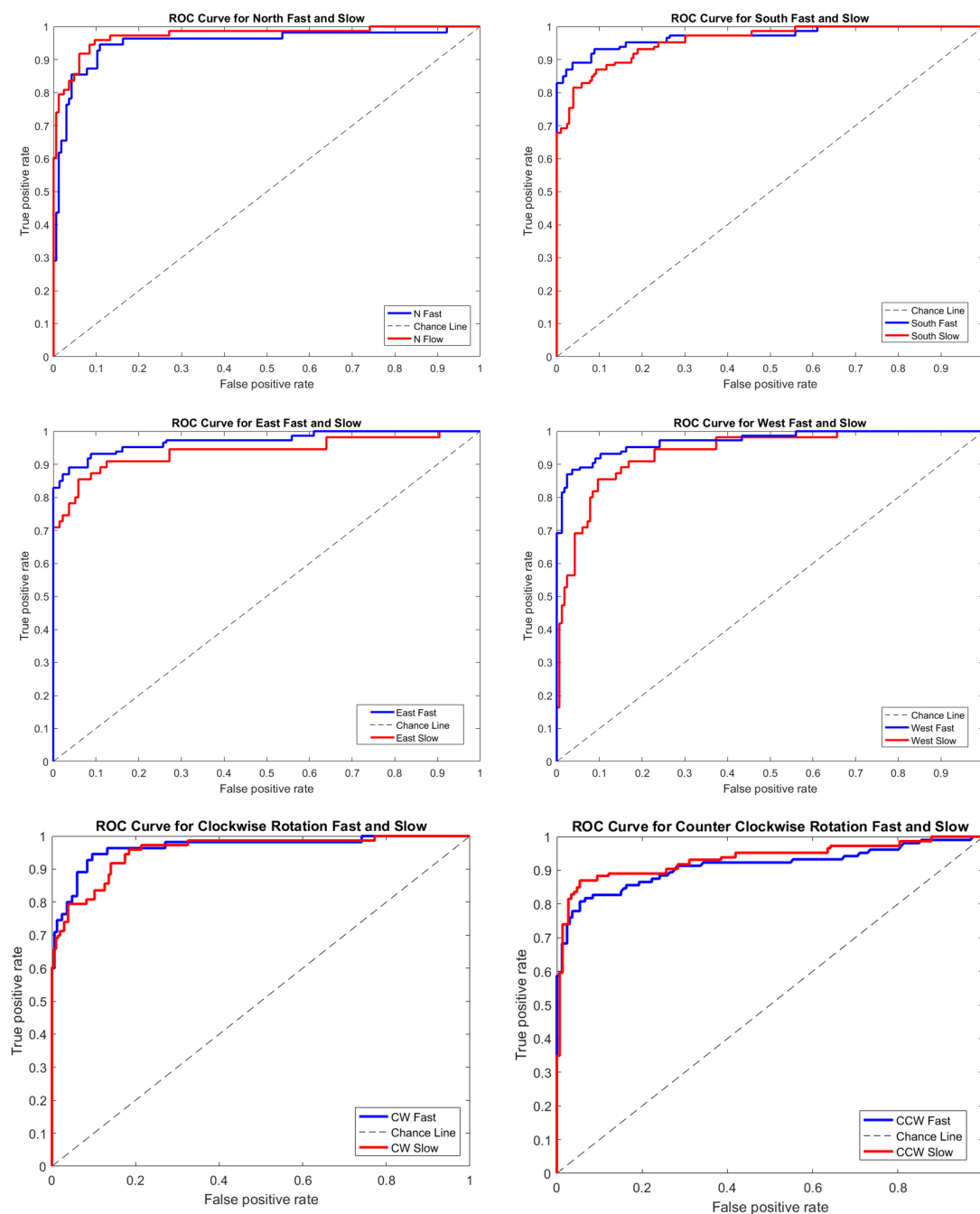


Figure 56: ROC curves showing the discrimination between fast and slow motions for each of the motion directions (North, East, South West, Clockwise, Counter Clockwise).



## 4.5 Conclusions and Interpretation

The purpose of the work in this chapter was to develop a method for using motion cues to aid in the detection of objects, and in visual self-motion estimation. The work presented in this section has only touched the surface of this intricate and complex problem but shows a possible path that future work can embark on to work towards a fully integrated solution able to run on a resource-limited MAV platform.

The first part showed that motion cues can greatly aid in the detection of symmetries in a scene. This approach utilises the same linear filtering techniques to reliably and quickly classify symmetry of objects. This approach relies on two assumptions: 1) the symmetrical properties of the object is known, and 2) the region of a potential object must first be found. The first assumption is trivial since a system is designed to find a specific object. Given that this method is designed to detect symmetry in an object, rather than finding the specific object the second assumption is acceptable. This method was shown to successfully work well alongside the object detection method proposed in chapter 3, where the algorithm can be tuned to have a higher detection rate at the cost of an increase in false alarms. By applying these symmetry cue tests to each potential region, the false alarm rates can drastically be reduced.

Using the discussion of flying insects in Section 1.2.2, a new pattern matching method has been developed using composite correlation filters for self-motion estimations. This approach relies on a directional flow field vector. This vector can be obtained using a number of optic flow methods, or spatio-temporal filtering approaches such as the method suggested in this chapter. Many optic flow methods do not provide any reliable confidence measure of the vectors [93]. Using linear spatio-temporal filtering techniques, an alternative approach was developed to construct motion flow fields. By combining several oriented energy volumes to build 3 higher level feature vectors, it is possible to describe motion at each point in the spatio-temporal volume. Using these features, the direction of motion at each point can be described. At each point, a certainty measure of the direction is also described through the dispersion value. This defines if the motion at each point is coherent in a particular direction or not. This dispersion volume was used to filter to final flow vectors for a video sequence before passing them through the correlation filter steps.

The initial testing has shown that simple motion patterns can be recognised using 3D correlation filters with reasonable accuracy. With the appropriate training dataset, more complex motions could also be detected. Furthermore, many optic flow methods only consider two adjacent frames to build the motion patterns. However, using a 3D correlation filter can combine many frames along the temporal axis to recognise longer term motions. The same filter can also recognise combined flight manoeuvres that may occur over a longer period of time.

At this stage, the computational performance has not been considered in detail. The main reason for this is that no code optimisation has been implemented. There is still future research to conduct for this approach to improve computational performance, one such starting point. Together the approach of using spatio-temporal filtering and correlation filters fit neatly into a framework of using correlation to recognise patterns for both static and dynamic patterns. These methods are well suited to parallelisation for significant performance

improvements on specialised embedded hardware.

## 5 Conclusions and Future Work

This thesis has investigated both static and dynamic perception using cues inspired by flying insects. The work conducted in each of these areas were split into two parts being static patterns and dynamic patterns.

The static patterns section of this thesis investigated how to efficiently recognise symmetries in static patterns. Motivated by honeybees, the proposed bio-inspired method uses a pattern recognition approach to quickly recognise objects with rotational symmetry. This method is not only an order of magnitude faster than the current state-of-the-art symmetry detection method but is also able to achieve better accuracy in the ROC analysis sense. This bio-inspired approach was also shown to realise a form of relevance filtering for further efficiency through the systematic choice of the training examples, with the algorithm being able to recognise symmetrical objects in 256 bit greyscale patterns while being trained with only binary patterns which contain less information than the greyscale test patterns.

The dynamic patterns section of this thesis investigated how to use dynamic patterns to aid in object detection and self-motion estimation. In this part two algorithms were proposed. The first algorithm can be considered as a bridge between static and dynamic perception, fusing together aspects of both. It was demonstrated that by incorporating motion cues from the object itself, the accuracy of the static pattern based symmetry detection method can be further improved. The second algorithm motivated by insect vision uses a pattern based approach that is able to recognise motion cues in a video sequence. This method first performs spatio-temporal filtering to estimate the motion energy in a given video sequence. This alternative approach uses linear filtering techniques and does not require direct calculation of dense optic flow that many other motion estimation methods rely on. This spatio-temporal method also has an advantage over traditional optic flow methods as it provides a measure of certainty at each point. Using this spatio-temporal energy volume, it was shown that three dimensional correlation filters can be trained to recognise motion types in the video sequence with reasonable accuracy.

### 5.1 Future Work

The suggested future work is divided into three main topics: static patterns, dynamic patterns, and integration of the two.

#### 5.1.1 Static Patterns

A future work direction aimed at improving the design efficiency of composite correlation filters could focus on further insights into relevance filtering, including a more in-depth frequency-domain analysis of the Horridge patterns (as shown in Figure 32). The Horridge patterns exhibit rotational symmetry of rectangular bars (or wedges) so further detection efficiencies could be achieved by utilising the symmetry properties of the Fourier transform [109] as shown in Figure 57.

Furthermore, the use of online training can be used to improve the tracking of objects. If an object is detected and the correlation score is above a certain threshold, that image can

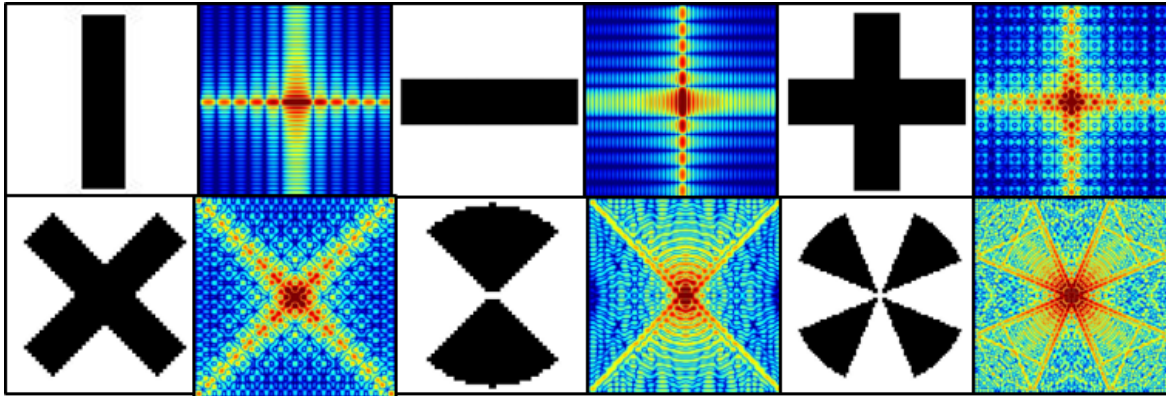


Figure 57: Some simple Horridge like patterns and their corresponding discrete Fourier transform magnitude representations

be added to the composite filter in order to make it more robust to the current scenario. This has been demonstrated to improve performance in [110].

The reasons for the main focus on flowers for testing in this thesis is because it gives a reference to the biological experiments for comparison. Future work can extend methods presented in this thesis to other objects using symmetry hierarchies as discussed in [32] for further efficiencies. For example, a flower can be generalised to a pattern of radial bars. A computer keyboard could be generalised to a rectangular shape with many smaller rectangular regions within. By recognising these lower level symmetry features they can be combined to recognise different objects. This approach can be linked to how honeybees build landmarks to recognise familiar places by combining simple features as discussed in Section 1.2.1.

### 5.1.2 Dynamic Patterns

The work described in this thesis has shown that the use of spatio-temporal filtering for motion perception is a promising avenue to follow. There are two main reasons for this, the first being that it seems to be an effective alternative to optic flow which is backed up by biologic research. The second reason for perusing this path is because it fits into a framework of correlation filters. For static perception 2D correlation filters are used  $(x, y)$ , and for dynamic perception, 3D correlation filters are used  $(x, y, t)$ .

**Extending Motion Recognition to More Complex Motion Types** Now that it has been demonstrated that one can use a pattern matching approach to recognise simple motions, future work can extend this to more complex self-motion pattern types, such as out-of-plane rotations. By using smaller filters on the  $(x, y)$ -plane, one can synthesise filters to locate key motion pattern locations such as the focus of expansion, or centre of rotation. These are important cues for obstacle avoidance and navigation. Future work can investigate using a bank of composite correlation filters to recognise the core motion types along with a certainty score, interpolation can be used to recognise variations between these motions.

**Spatio-Temporal Motion Normalisation** As discussed in Section 4.4, a dispersion measure was included to filter out regions of the volume that did not exhibit coherent motion. However, due to the spatio-temporal filters used, the energy responses are still influenced by contrast in the scene, not motion alone. Given two regions of different contrast that are moving in the same direction, the region with more contrast will yield a slightly stronger energy response even though the motion is technically the same. Future work can investigate methods to ensure the spatio-temporal energy response is purely from motion. As a starting point normalisation can be used that will take into account the contrast in the scene.

**Computational Performance** At this stage, the computational performance has not been considered as no code optimisation has been performed. Future research can be conducted to improve the performance. Together the approach of using spatio-temporal filtering and correlation filters fit neatly into a framework of using correlation to recognise patterns for both static and dynamic patterns. Which is well suited to parallelisation for significant performance improvements.

### 5.1.3 Integration

In this thesis, a common theme of applying a pattern matching approach has been utilised for the detection of symmetries, and motion. Using a framework of linear filters, both approaches can be combined into an integrated vision system. The correlation filters discussed for static perception work in the  $(x, y)$ -plane. To deal with distortions of targets, composite correlation filters are used in order to improve the detection rate. The same approach is used on the motion flow vectors but extended to 3D. Similarly, the spatio-temporal methods discussed again use linear filtering techniques. Using a linear filtering framework for both static and dynamic perception which can share some processes and information between them to allow for efficient implementation. For example, once the dynamic pattern recognition has recognised a self-motion type, this can be used to remove background motion in the scene which can be subtracted from the flow field. The remaining vectors are either caused by noise or other object motion in the scene. These regions can then be further investigated using the static pattern methods to only search those regions. Future work will study the feasibility of integrating the two approaches.

**Using Spherical Sensors** As shown in Appendix A, using spherical sensors can remove some of the ambiguities with flow vectors. Future work can extend the methods discussed in this thesis to utilise omnidirectional vision. Detections of certain targets around the sphere (using composite filters) can be tracked in order to gain a knowledge of the vehicle's position relative to these targets. This is a concept that is used by honeybees for navigation and is shown in Figure 11, where the detection and tracking of certain targets relative to one another around the visual panorama can be used to recognise familiar places. This, in turn, can be used for dynamic perception for self-motion tracking.

## References

- [1] A Horridge. The Compound Eye of Insects. *Scientific American*, 237:108–120, 1977.
- [2] J Zufferey. *Bio-Insect Inspired Flying Robots: Experimental Synthesis of Autonomous Indoor Flyers*. EPFL Press Distributed by CRC Press, Lausanne, 2008.
- [3] R Żbikowski. Fly like a fly. *IEEE Spectrum*, 42(11):46–51, 2005.
- [4] J Harrison. Arthropods such as this *Calliphora vomitoria* fly have compound eyes. [https://en.wikipedia.org/wiki/Eye#/media/File:Calliphora\\_vomitoria\\_Portrait.jpg](https://en.wikipedia.org/wiki/Eye#/media/File:Calliphora_vomitoria_Portrait.jpg) Last accessed Aug 2015.
- [5] M Land and D Nilsson. *Animal Eyes*. Oxford Animal Biology Series. Oxford University Press, 2nd edition, 2012.
- [6] A Elliott. Insect Vision and Static Perception. Technical report, Cranfield University, 2014.
- [7] H Krapp, B Hengstenberg, and R Hengstenberg. Dendritic Structure and Receptive-field Organization of Optic Flow Processing Interneurons in the Fly. *Journal of Neurophysiology*, 79(4):1902–1917, 1998.
- [8] G Taylor and H Krapp. Sensory Systems and Flight Stability: What do Insects Measure and Why? *Advances in Insect Physiology: Insect Mechanics and Control*, 34:232–316, 2007.
- [9] M Franz and H Krapp. Wide-field, motion-sensitive neurons and matched filters for optic flow fields. *Biological Cybernetics*, 83(3):185–97, sep 2000.
- [10] A Horridge. What the honeybee sees: a review of the recognition system of *Apis mellifera*. *Physiological Entomology*, pages 2–13, 2005.
- [11] A Horridge. What does an insect see? *The Journal of Experimental Biology*, 212(17):2721–9, sep 2009.
- [12] M Srinivasan. Honey bees as a model for vision, perception, and cognition. *Annual Review of Entomology*, 55:267–84, jan 2010.
- [13] J Chahl, M Srinivasan, and S Zhang. Landing Strategies in Honeybees and Applications to Uninhabited Airborne Vehicles. *The International Journal of Robotics Research*, 23(2):101–110, 2004.
- [14] H Esch and J Burns. Distance estimation by foraging honeybees. *The Journal of experimental biology*, 199(Pt 1):155–62, 1996.
- [15] M Srinivasan. Honeybees as a model for the study of visually guided flight, navigation, and biologically inspired robotics. *Physiological reviews*, 91(2):413–60, apr 2011.

- 
- [16] K Götz. The Optomotor Equilibrium of the Drosophila Navigation System. *Journal of Comparative Physiology*, 99:187–210, 1975.
- [17] W Reichardt. Autocorrelation, A Principle for the Evaluation of Sensory Information by the Centre Nervous System. *Sensory Communication*, 56:303–317, 1956.
- [18] K Hausen. Decoding of retinal image flow in insects. *Reviews of Oculomotor Research*, 1993.
- [19] H Krapp. Neuronal matched filters for optic flow processing in flying insects. *International Review of Neurobiology*, 44, 1999.
- [20] R Wehner. Matched Filter-neural Models of the External World. *Journal of Comparative Physiology A*, 161(4):511–531, 1987.
- [21] I Song, J Bae, and S Kim. *Advanced Theory of Signal Detection*. Springer, Berlin, Germany, 2002.
- [22] M Srinivasan, R Pinter, and D Osorio. Matched Filtering in the Visual System of the Fly: Large Monopolar Cells of the Lamina are Optimized to Detect Moving Edges and Blobs. *Proceedings of the Royal Society B: Biological Sciences*, 240(1298):279–293, jun 1990.
- [23] E Davies. *Computer and Machine Vision: Theory, Algorithms, Practicalities*. Elsevier, Amsterdam, The Netherlands, fourth edition, 2012.
- [24] G Martin. *Transformation Geometry. An Introduction to Symmetry*. Springer, New York, NY, 1982.
- [25] V Gesù, M Tabacchi, and B Zavidovique. Symmetry as an intrinsically dynamic feature. *Symmetry*, 2(2):554–581, 2010.
- [26] W Hong, A Yang, K Huang, and Y Ma. On symmetry and multiple-view geometry: Structure, pose, and calibration from a single image. *International Journal of Computer Vision*, 60(3):241–265, 2004.
- [27] Y Liu, H Hel-Or, C Kaplan, and L Gool. Computational symmetry in computer vision and computer graphics. *Foundations and Trends in Computer Graphics and Vision*, 5(1–2):1–195, 2009.
- [28] K Suresh and A Sirpotdar. Automated symmetry exploitation in engineering analysis. *Engineering with Computers*, 21(4):304–311, may 2006.
- [29] O Schmitt and M Hasse. Radial symmetries based decomposition of cell clusters in binary and gray level images. *Pattern Recognition*, 41(6):1905–1923, 2008.
- [30] Q Guo, F Guo, and J Shao. Irregular shape symmetry analysis: theory and application to quantitative galaxy classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(10):1730–43, oct 2010.

- [31] J Haunert. A symmetry detector for map generalization and urban-space analysis. *ISPRS Journal of Photogrammetry and Remote Sensing*, 74:66–77, nov 2012.
- [32] Y Wang, K Xu, J Li, H Zhang, A Shamir, L. Liu, Z. Cheng, and Y. Xiong. Symmetry Hierarchy of Man-Made Objects. *Computer Graphics Forum*, 30(2):287–296, apr 2011.
- [33] M Park, S Lee, P Chen, S Kashyap, A Butt, and Y Liu. Performance Evaluation of State-of-the-Art Discrete Symmetry Detection Algorithms. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2008, 23-28 June 2008, Anchorage, AK, USA*, pages 3745–3752. IEEE, 2008.
- [34] V Prasad and L Davis. Detecting rotational symmetries. In *10th IEEE International Conference on Computer Vision (ICCV’05), 17–21 Oct 2005, Beijing, China*, pages 954–961. IEEE, 2005.
- [35] G Loy and J Eklundh. Detecting symmetry and symmetric constellations of features. *Computer Vision ECCV 2006*, 3952:508–521, 2006.
- [36] D Lowe. Distinctive Image Features from Scale-Invariant Keypoints. *International Journal of Computer Vision*, 60(2):91–110, nov 2004.
- [37] Y Keller and Y Shkolnisky. A signal processing approach to symmetry detection. *IEEE Transactions on Image Processing*, 15(8):2198–2207, 2006.
- [38] M Chertok and Y Keller. Spectral symmetry analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(7):1227–38, jul 2010.
- [39] A Berner, M Wand, N Mitra, D Mewes, and H Seidel. Shape Analysis with Subspace Symmetries. *Computer Graphics Forum*, 30(2):277–286, apr 2011.
- [40] D Reisfeld, H Wolfson, and Y Yeshurun. Context Free Attentional Operators: the Generalized Symmetry Transform. *International Journal of Computer Vision*, 14(1995):119–130, 1995.
- [41] G Turin. An introduction to matched filters. *IEEE Transactions on Information Theory*, 6(3):311–329, jun 1960.
- [42] G Turin. An introduction to digital matched filters. *Proceedings of the IEEE*, 64(7):1092–1112, 1976.
- [43] S Kay. *Fundamentals of Statistical Signal Processing*, volume II Detecte. Prentice-Hall, Upple Saddle River, NJ, 1998.
- [44] J Woods. *Multidimensional Signal, Image, and Video Processing and Coding*. Elsevier, Amsterdam, The Netherlands, second edition, 2012.



- [45] J Fernandez, V Boddeti, A Rodriguez, and B Kumar. Zero-aliasing correlation filters for object recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37(8):1702–1715, 2015.
- [46] W Freeman and E Adelson. The design and use of steerable filters - Pattern Analysis and Machine Intelligence, IEEE Transactions on. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(9):891–906, 1991.
- [47] B Kumar and E Pochapsky. Signal-to-noise ratio considerations in modified matched spatial filters. *Journal of the Optical Society of America A*, 3(6):777–786, 1986.
- [48] A Vanderlugt. *Optical Signal Processing*. Wiley, New York, NY, 1992.
- [49] I Leonard. Face recognition based on composite correlation filters: analysis of their performances. *Face Recognition: Methods, Applications and Technology*, pages 57–80, 2012.
- [50] D Bolme, J Beveridge, B Draper, and Y Lui. Visual object tracking using adaptive correlation filters. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2010, 13-18 June 2010, San Francisco, CA, USA*, pages 2544–2550. IEEE, 2010.
- [51] A Mahalanobis, R Muise, and S Stanfill. Quadratic correlation filter design methodology for target detection and surveillance applications. *Applied optics*, 43(27):5198–5205, sep 2004.
- [52] A Rodriguez, V Boddeti, B Kumar, and A Mahalanobis. Maximum margin correlation filter: A new approach for localization and classification. *IEEE Transactions on Image Processing*, 22(2):631–643, 2013.
- [53] D Lefebvre, H Arsenault, P Garcia-Martinez, and C Ferreira. Recognition of Unsegmented Targets Invariant under Transformations of Intensity. *Applied Optics*, 41(29):6135–6142, oct 2002.
- [54] A Mahalanobis, B Kumar, and D Casasent. Minimum average correlation energy filters. *Applied optics*, 26(17):3633–40, sep 1987.
- [55] A Mahalanobis, B V K V Kumar, and D Casasent. Minimum average correlation energy filters. *Applied Optics*, 26(17):3633–3640, 1987.
- [56] P Refregier. Filter design for optical pattern recognition: Multicriteria optimization approach. *Optics Letters*, 15(15):854–856, 1990.
- [57] B Kumar, A Mahalanobis, and D Carlson. Optimal trade-off synthetic discriminant function filters for arbitrary devices. *Optical Letters*, 19(19):1556–1558, oct 1994.

- [58] O Johnson, W Edens, T Lu, and T Chao. Optimization of OT-MACH filter generation for target recognition. In D Casasent and T H Chao, editors, *Optical Pattern Recognition XX*, pages 1–9. SPIE, 2009.
- [59] M Rodriguez, J Ahmed, and M Shah. Action MACH: A spatio-temporal maximum average correlation height filter for action recognition. *26th IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, pages 1–8, 2008.
- [60] W Pratt. *Digital Image Processing*. Wiley, Hoboken, NJ, fourth edition, 2007.
- [61] K Spackman. Signal Detection Theory: Valuable Tools for Evaluating Inductive Learning. In *Proceedings of the Sixth International Workshop on Machine Learning*, pages 160–163, San Francisco, CA, USA, 1989. Morgan Kaufmann Publishers Inc.
- [62] T Fawcett. An introduction to ROC analysis. *Pattern Recognition Letters*, 27(8):861–874, jun 2006.
- [63] H Chao, Y Gu, and M Napolitano. A survey of optical flow techniques for robotics navigation applications. *Journal of Intelligent & Robotic Systems*, 73(1-4):361, 2014.
- [64] B Lucas and T Kanade. An Iterative Image Registration Technique with an Application to Stereo Vision. In *Proceedings of the 7th International Joint Conference on Artificial Intelligence - Volume 2*, volume 130 of *IJCAI’81*, pages 674–679, San Francisco, CA, USA, 1981. Morgan Kaufmann Publishers Inc.
- [65] B Horn and B Schunck. Determining optical flow. *Artificial Intelligence*, 17(1-3):185–203, aug 1981.
- [66] B Schunck. The image flow constraint equation. *Computer Vision, Graphics, and Image Processing*, 35(1):20–46, jul 1986.
- [67] E Adelson and J Bergen. Spatiotemporal energy models for the perception of motion. *Journal of the Optical Society of America A*, 2(2):284–299, 1985.
- [68] A Watson and A Ahumada. Model of human visual-motion sensing. *Journal of the Optical Society of America A*, 2(2):322–342, 1985.
- [69] R Dosil, X Fdez-Vidal, and X Pardo. Motion representation using composite energy features. *Pattern Recognition*, 41(3):1110–1123, 2008.
- [70] D Bradley and M Goyal. Velocity computation in the primate visual system. *Nature Reviews Neuroscience*, 9(9):686–695, 2008.
- [71] D Heeger. Model for the extraction of image flow. *Journal of the Optical Society of America. A, Optics and image science*, 4(8):1455–71, aug 1987.
- [72] J Daugman. Uncertainty relation for resolution in space, spatial frequency, and orientation optimized by two-dimensional visual cortical filters. *Journal of the Optical Society of America A*, 2(7):1160–1169, 1985.

- 
- [73] A Spinei, D Pellerin, D Fernandes, and J Herault. Fast Hardware Implementation of Gabor Filter Based Motion Estimation. 2(1):1–18.
- [74] K Derpanis and J Gryn. Three-dimensional nth derivative of Gaussian separable steerable filters. In *12th IEEE International Conference on Image Processing, ICIP 2005, 11–14 Sep 2005, Genova, Italy*, volume 3, pages 553–556. IEEE, 2005.
- [75] V Larrowe. Band-Pass Quadrature Filters. *IEEE Transactions on Electronic Computers*, 15(5):726–731, 1966.
- [76] W Choi, S Tse, K Wong, and K Lam. Simplified Gabor wavelets for human face recognition. *Pattern Recognition*, 41(3):1186–1199, mar 2008.
- [77] R Wildes and J Bergen. Qualitative Spatiotemporal Analysis Using an Oriented Energy Representation. In D Vernon, editor, *Computer Vision—ECCV 2000*, volume 1843 of *Lecture Notes in Computer Science*, pages 768–784. Springer, 2000.
- [78] A Belardinelli, A Carbone, and W Schneider. Classification of multiscale spatiotemporal energy features for video segmentation and dynamic objects prioritisation. *Pattern Recognition Letters*, 34(7):713–722, 2013.
- [79] R Pless. Spatio-temporal Background Models for Outdoor Surveillance. *EURASIP Journal on Applied Signal Processing*, 2005(14):2281–2291, 2005.
- [80] G Prenner, R Bateman, and P Rudall. Floral formulae updated for routine inclusion in formal taxonomic descriptions. *Taxon*, 59(1):241–250, 2010.
- [81] H Citerne, F Jabbour, S Nadot, C Damerval, and C Damerval. The evolution of floral symmetry. *Advances in Botanical Research*, 54:85–137, 2010.
- [82] A Horridge. *What Does the Honeybee See? And How Do We Know?* Australian National University E Press, first edition, 2009.
- [83] M Nilsback and A Zisserman. 17 Category Flower Dataset. <http://www.robots.ox.ac.uk/vgg/data/flowers/17/index.html>, Last accessed Feb 2015.
- [84] C Oberprieler, S Himmelreich, M Källersjö, J Vallès, L E Watson, and R Vogt. Anthemideae. In V A Funk, A Susanna, T Stuessy, and R Bayer, editors, *Systematics, Evolution, and Biogeography of Compositae*, pages 631–666, Vienna, Austria, 2009. International Association for Plant Taxonomy.
- [85] R Haralick. Statistical and structural approaches to texture. *Proceedings of the IEEE*, 67(5):786–804, 1979.
- [86] B Kumar and E Pochapsky. Signal-to-noise ratio considerations in modified matched spatial filters. *Journal of the Optical Society of America A*, 3(6):777, jun 1986.
-

- [87] A Mahalanobis, B V Vijaya Kumar, and R T Frankot. Intraclass and between-class training-image registration for correlation-filter synthesis. *Applied optics*, 39(17):2918–24, jun 2000.
- [88] C Mglee. Illustration of an image pyramid with 5 levels. [https://en.wikipedia.org/wiki/Pyramid/media/File:Image\\_pyramid.svg](https://en.wikipedia.org/wiki/Pyramid/media/File:Image_pyramid.svg), Last accessed Jun 2016.
- [89] B Kumar, A Mahalanobis, and R Juday. *Correlation Pattern Recognition*. Cambridge University Press, Cambridge, England, 2005.
- [90] D Bolme, B Draper, and J Beveridge. Average of Synthetic Exact Filters. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2009, 20-25 June 2009, Miami, FL, USA*, pages 2105–2112. IEEE, 2009.
- [91] H Bay, A Ess, T Tuytelaars, and L Vangool. Speeded-Up Robust Features (SURF). *Computer Vision and Image Understanding*, 110(3):346–359, jun 2008.
- [92] C Evans. Notes on the OpenSURF Library. Technical report, University of Bristol, 2009.
- [93] J Barron, D Fleet, S Beauchemin, and T Burkitt. Performance of optical flow techniques. In *Computer Vision and Pattern Recognition, 1992. Proceedings CVPR'92., 1992 IEEE Computer Society Conference on*, pages 236–242. IEEE, 1992.
- [94] S Frintrop, E Rome, and H I Christensen. Computational Visual Attention Systems and Their Cognitive Foundations: A Survey. *ACM Transactions on Applied Perception*, 7(1):1–39, 2010.
- [95] M Carrasco. Visual attention: the past 25 years. *Vision research*, 51(13):1484–525, jul 2011.
- [96] J Ahmed, S Abbasi, and M Shaikh. Fast spatiotemporal MACH filter for action recognition. *Machine Vision and Applications*, 24(5):909–918, 2013.
- [97] A Efros, A Berg, G Mori, and J Malik. Recognizing action at a distance. *Proceedings Ninth IEEE International Conference on Computer Vision*, (October):726–733, 2003.
- [98] D Fortun, P Bouthemy, and C Kervrann. Optical flow modeling and computation: A survey. *Computer Vision and Image Understanding*, 134:1–21, may 2015.
- [99] J Campbell, R Sukthankar, I Nourbakhsh, and A Pahwa. A robust visual odometry and precipice detection system using consumer-grade monocular vision. In *Robotics and Automation, 2005. ICRA 2005. Proceedings of the 2005 IEEE International Conference on*, pages 3421–3427. IEEE, 2005.

- [100] D Floreano, R Pericet-Camara, S Viollet, F Ruffier, A Brückner, R Leitel, W Buss, M Menouni, F Expert, R Juston, M Dobrzynski, G L'Eplattenier, F Recktenwald, H Mallot, and N Franceschini. Miniature curved artificial compound eyes. *Proceedings of the National Academy of Sciences*, 110(23):9267–9272, jun 2013.
- [101] Y Song, Y Xie, V Malyarchuk, J Xiao, I Jung, K Choi, Z Liu, H Park, C Lu, R Kim, R Li, K Crozier, Y Huang, and J Rogers. Digital cameras with designs inspired by the arthropod eye. *Nature*, 497(7447):95–99, may 2013.
- [102] K Rapantzikos, N Tsapatsoulis, Y Avrithis, and S Kollias. Bottom-up spatiotemporal visual attention model for video analysis. *Signal Processing: Image Communication*, 1(2):237–248, 2007.
- [103] K Rapantzikos, N Tsapatsoulis, Y Avrithis, and S Kollias. Spatiotemporal saliency for video classification. *Signal Processing: Image Communication*, 24(7):557–571, 2009.
- [104] K Rapantzikos, Y Avrithis, and S Kollias. Spatiotemporal Features for Action Recognition and Salient Event Detection. *Cognitive Computation*, 3(1):167–184, 2011.
- [105] X Hu, K Li, J Han, X Hua, L Guo, and T Liu. Bridging the Semantic Gap via Functional Brain Imaging. *IEEE Transactions on Multimedia*, 14(2):314–325, 2012.
- [106] A Borji, D Sihite, and L Itti. Quantitative Analysis of Human-Model Agreement in Visual Saliency Modeling: A Comparative Study. *IEEE Transactions on Image Processing*, 22(1):55–69, 2013.
- [107] G Watson. *Statistics on Spheres (The University of Arkansas lecture notes in the mathematical sciences)*. Wiley-Interscience, 1983.
- [108] S Sangwine and E Hitzer. Clifford Multivector Toolbox (for MATLAB). *Advances in Applied Clifford Algebras*, apr 2016.
- [109] R Żbikowski and A Dzieliński. Fourier transform symmetry and invariance for neuro-control of NARMA models. *Nonlinear Analysis: Theory, Methods & Applications*, 30(4):1985–1993, 1997.
- [110] M Savvides and B Kumar. Efficient design of advanced correlation filters for robust distortion-tolerant face recognition. In *Advanced Video and Signal Based Surveillance, 2003. Proceedings. IEEE Conference on*, pages 45–52, jul 2003.
- [111] C Fermüller and Y Aloimonos. Qualitative Egomotion. *International Journal of Computer Vision*, 15(1):7–29, 1995.
- [112] C Fermüller. Passive Navigation as a Pattern Recognition Problem. *International Journal of Computer Vision*, 14(1):147–158, 1995.

- [113] C Fermüller and Y Aloimonos. On the geometry of visual correspondence. *International Journal of Computer Vision*, 21(3):223–247, 1997.
- [114] T Brodsky, C Fermüller, and Y Aloimonos. Directions of motion fields are hardly ever ambiguous. *International Journal of Computer Vision*, 26(1):5–24, 1998.
- [115] C Fermüller and Y Aloimonos. Ambiguity in Structure from Motion: Sphere versus Plane. *International Journal of Computer Vision*, 28(2):137–154, 1998.
- [116] J Hateren and A Schaaf. Independent component filters of natural images compared with simple cells in primary visual cortex. *Proceedings of the Royal Society B: Biological Sciences*, 265(1394):359–366, mar 1998.

## A Appendix - Planar and Spherical Optic Flow

This Appendix is an introduction to optic flow for both planar and spherical retina. This section is required as a basis for the equations used for the derivation of the Franz-Krapp Filter shown in Appendix B.

### A.1 Planar Case

This section will look at work conducted primarily by Fermüller and Aloimonos [111, 112, 113, 114, 115] who used a qualitative approach. Rather than looking at only parts of an image, a more global approach was used by looking at the patterns that the velocity fields form.

#### A.1.1 Problem Formulation (planar retina)

The mapping of scene points onto the image plane should be understood before continuing. We consider rigid body motion of the observer (camera) and describe the relationship between points in the real world (scene points) to the points the camera can see (image points) via perspective projection. This can be shown in Figure 58.

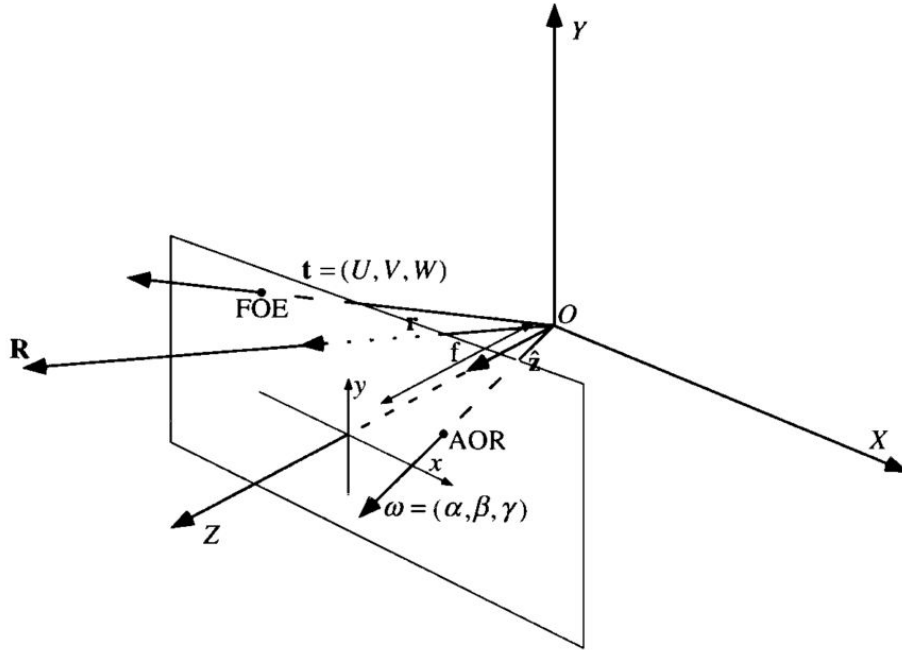


Figure 58: Image formation using perspective projection on a planar retina [113]

We consider a coordinate system  $(X, Y, Z)$  that is fixed to the observer (nodal point of the camera). The observer moves with translation  $\mathbf{t} = (U, V, W)$  and rotation  $\boldsymbol{\omega} = (\alpha, \beta, \gamma)$  in a stationary environment. Each scene point  $\mathbf{R} = (X, Y, Z)$  has the velocity component

$$\dot{\mathbf{R}} = -\mathbf{t} - \boldsymbol{\omega} \times \mathbf{R}. \quad (63)$$

The image is formed on a plane parallel to the  $XY$ -plane at distance  $f$  from the nodal point. We now consider a 3D point in the scene would be mapped onto the image plane as a 2D point,  $\mathbf{r}$  using perspective projection,

$$\mathbf{r} = \frac{f}{\mathbf{R} \cdot \mathbf{z}_0} \mathbf{R}, \quad (64)$$

where  $\mathbf{z}_0$  is a unit vector in the direction of the  $Z$ -axis and  $\mathbf{a} \cdot \mathbf{b}$  denotes the inner product of vectors  $\mathbf{a}$  and  $\mathbf{b}$ . We now need to differentiate  $\mathbf{r}$  with respect to time and substitute for  $\dot{\mathbf{R}}$  to obtain the 2D velocity of an image point.

$$\dot{\mathbf{r}} = \underbrace{-\frac{1}{Z}(\mathbf{z}_0 \times (\mathbf{t} \times \mathbf{r}))}_{\text{translation}} + \underbrace{\frac{1}{f}(\mathbf{z}_0 \times (\mathbf{r} \times (\boldsymbol{\omega} \times \mathbf{r}))}_{\text{rotation}}, \quad (65)$$

where the translational component is  $(v_{\text{tr}}\mathbf{r})/Z$  and the rotational flow component is  $v_{\text{rot}}$ . Only the translational component of equation (65) depends on the depth  $Z = \mathbf{R} \cdot \mathbf{z}_0$ , so it is only possible to recover the scaled translational component  $\mathbf{t}/Z$ . The rotational component only depends on the three rotational components  $\alpha, \beta, \gamma$ .

**Translational Vectors** If the observer moves only with translation, the 3D scene will move along parallel lines, when viewed with perspective projection on the image plane, the points will travel along a line that passes through the vanishing point. The flow at that point is zero. If the observer is moving towards the scene points, the points will emanate at the vanishing point and move outwards, this point is called the Focus of Expansion (FOE). Otherwise, if the observer is moving away from the scene points, the flow will move towards the Focus of Contraction (FOC). This can be seen in Figure 59, where the vanishing point (FOC/FOE) image coordinates  $(x, y)$  are given as:

$$x_0 = \frac{Uf}{W} \quad (66a)$$

$$y_0 = \frac{Vf}{W} \quad (66b)$$

The direction of each of the translational flow vector is determined by the location of the vanishing point.

**Rotational Vectors** In the case of purely rotational motion, every point in 3D moves along a circle in a plane that is perpendicular to the axis of rotation. The view on the image plane is the intersection of the image plane with a cone that originates at the origin and faces the Axis of Rotation (AOR). The rotational motion at this point is zero (see Figure 60).



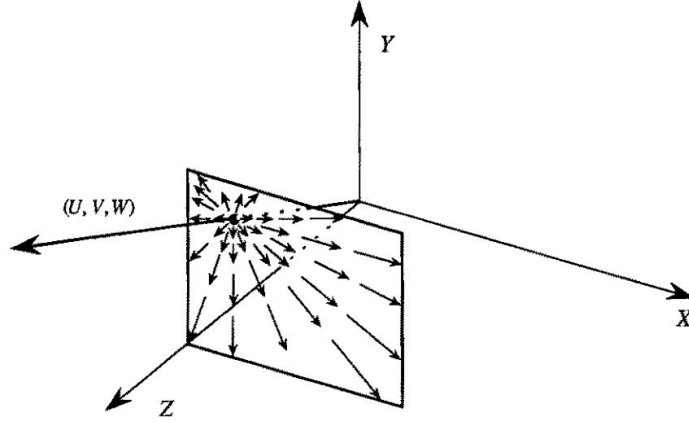


Figure 59: Translational motion viewed under perspective projection [111]

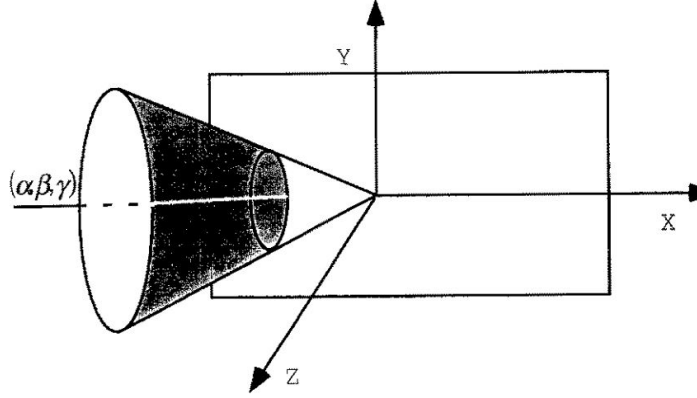


Figure 60: Intersection of the image plane with the cone of projected rotational motion [111]

The rotation axis is given by the two parameters:

$$x = \left( \frac{\alpha}{\gamma} \right) f \quad (67a)$$

$$y = \left( \frac{\beta}{\gamma} \right) f \quad (67b)$$

The axis of rotation defines the shape of the second order curves (ellipse, hyperbola, parabola, or circles) on the image plane as shown in Figure 61. These field lines are the lines along which the image points would move if the observer rotated around the axis of rotation.

We will now continue work in component notation where  $\dot{\mathbf{r}} = (\dot{r}_1, \dot{r}_2, \dot{r}_3)$ . Because we are using perspective projection,  $\dot{r}_3$  is always zero. We will denote  $\dot{r}_1$  as  $u$  and  $\dot{r}_2$  as  $v$ , so that  $\dot{\mathbf{r}} = (u, v)$ . Therefore we can re-write equation (65) in component notation where  $\mathbf{t} = (U, V, W)$  and  $\boldsymbol{\omega} = (\alpha, \beta, \gamma)$ .

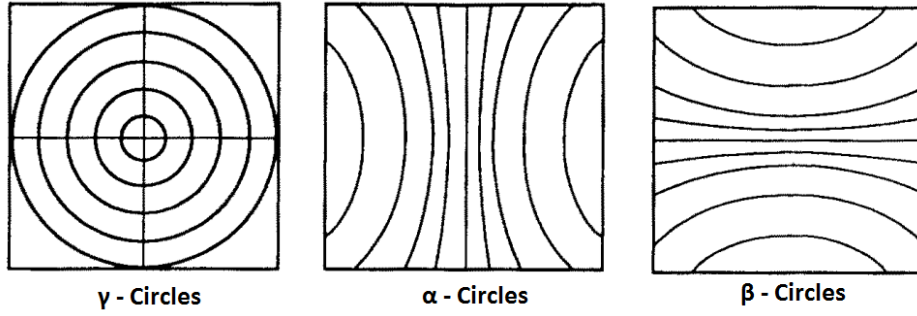


Figure 61: Second order curves on the image plane from rotation  $\gamma$ -circles form when rotating around the  $Z$ -axis,  $\alpha$ -circles form when rotating around the  $X$ -axis, and  $\beta$ -circles are formed when rotating around the  $Y$ -axis.

$$\begin{aligned} u &= u_{\text{trans}} + u_{\text{rot}} \\ &= (-x_0 + x) \frac{W}{Z} + \alpha \frac{xy}{f} - \beta \left( \frac{x^2}{f} + f \right) + \gamma y, \end{aligned} \quad (68)$$

$$\begin{aligned} v &= v_{\text{trans}} + v_{\text{rot}} \\ &= (-y_0 + y) \frac{W}{Z} + \alpha \left( \frac{y^2}{f} + f \right) - \beta \frac{xy}{f} - \gamma x. \end{aligned} \quad (69)$$

We can only compute the normal flow in an image along the gradient direction  $(n_x, n_y)$  as only the component along the brightness gradient can be recovered (see Figure 26). This normal flow vector,  $(u, v)_n$  along the gradient direction, has a flow value  $u_n$  given by:

$$\begin{aligned} u_n &= (u, v) \cdot (n_x, n_y) \\ &= u n_x + v n_y \\ &= \left( (-x_0 + x) \frac{W}{Z} + \alpha \frac{xy}{f} - \beta \left( \frac{x^2}{f} + f \right) + \gamma y \right) n_x \\ &\quad + \left( (-y_0 + y) \frac{W}{Z} + \alpha \left( \frac{y^2}{f} + f \right) - \beta \frac{xy}{f} - \gamma x \right) n_y, \end{aligned} \quad (70)$$

where  $(n_x, n_y)$  are the normal vectors  $(n_x, n_y) = (x/r, y/r)$ ,  $r = \sqrt{x^2 + y^2}$ .

As shown in equation (70), there are five unknown motion parameters  $(\alpha, \beta, \gamma, x, y)$ , and one scaled depth component  $W/Z$  at every point for  $u_n$ . This is not easy to calculate since we can not make any assumptions about the depth.

The motion vector consists of a rotational component which consists of three unknowns  $(\alpha, \beta, \gamma)$ , and a translational vector  $(x, y)$  which is directed everywhere towards, or from the vanishing point. We can only compute the estimates of the motion vectors projected onto the gradient direction. Therefore it is useful to re-arrange equation (70) for each of the

components:

$$\begin{aligned}
 u_n = & \frac{W}{Z}[(-x_0 + x), (-y_0 + y)][n_x, n_y] \\
 & + \alpha \left[ \frac{xy}{f}, \left( \frac{y^2}{f} + f \right) \right] [n_x, n_y] \\
 & - \beta \left[ \left( \frac{x^2}{f} + f \right), \frac{xy}{f} \right] [n_x, n_y] \\
 & + \gamma [y, -x][n_x, n_y].
 \end{aligned} \tag{71}$$

If two vectors are perpendicular, their scalar products are zero. Therefore for certain normal flow vectors, some of the components will disappear. This leads onto the idea of  $\alpha$ ,  $\beta$ ,  $\gamma$  vectors as shown in Figure 62.

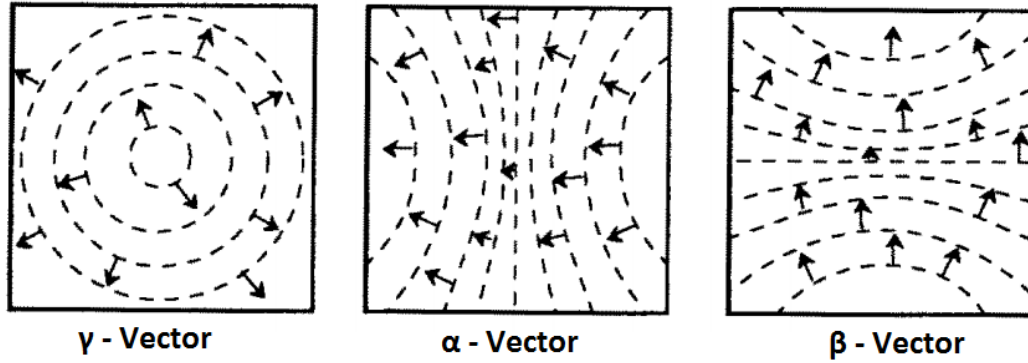


Figure 62: Positive  $\alpha$ ,  $\beta$ ,  $\gamma$  vectors.

If the normal flow vectors are perpendicular to the circle, then there is no component due to rotation for a certain axis.

If the normal flow vectors are perpendicular to the  $\gamma$ -circles, the normal flow vector does not contain a component due to rotation around the  $Z$ -axis. Similarly, if the normal vectors are perpendicular to the  $\alpha$ -circles, then there is no component due to rotation around the  $X$ -axis. There is no component around the  $Y$ -axis if the normal flow vectors are perpendicular to the  $\beta$ -circles. In Section A.1.2 we will look at these vector field lines in more detail.

### A.1.2 Copoint and Coaxis Vectors

There are two main types of normal flow vectors; the coaxial, and copoint vectors. Each of these will be discussed in this section.

**Coaxis Vectors** We will now consider an imaginary line passing through the nodal point with orientation vector  $(A, B, C)$ , where  $A^2 + B^2 + C^2 = 1$ . This line defines a family of cones originating at the origin with axis  $(A, B, C)$ . As shown in Figure 60, the

intersection of the cone with the image plane will lead to vector field lines or  $(\alpha, \beta, \gamma)$ -circles. We now look at the normal flow vectors that are perpendicular to the vector field lines. The orientation perpendicular to the vector field lines is given by the vector  $\mathbf{M} = (M_x, M_y)$  where  $M_x = -A(y^2 + f^2) + Bxy + Cxf$ , and  $M_y = Axy - B(x^2 + f^2) + Cyf$ , and the unit vector is  $\mathbf{m} = \mathbf{M}/|\mathbf{M}|$ . The normal vectors along the gradient  $(n_x, n_y)$  is equal to  $(m_x, m_y)$ . This family of vectors that correspond to the  $(A, B, C)$  axis are called the  $(A, B, C)$  coaxis vectors. Before we continue it is important to state the vector orientations.

- Positive orientation is when the  $(A, B, C)$  coaxis vector is pointing away from the axis of rotation:  $(m_x, m_y)$  or  $(n_x, n_y)$ .
- Negative orientation is when the  $(A, B, C)$  coaxis vector is pointing towards the axis of rotation:  $(-m_x, -m_y)$  or  $(-n_x, -n_y)$ .

**Translational Coaxis Components** The value  $t_n$  is the translational component of the normal flow vector,  $\mathbf{u}_n$  at point  $(x, y)$  in the direction  $(n_x, n_y)$  and is given by the translational component from equation (71).

$$t_n = \left[ \frac{W}{Z}(-x_0 + x), (-y_0 + y) \right] \cdot [n_x, n_y]. \quad (72)$$

Because we assume that the observer is approaching the scene,  $W/Z$  is positive, the sign of  $t_n$  is equal to the sign of

$$\begin{aligned} h(A, B, C, x_0, y_0; x, y) &= ((-x_0 + x), (-y_0 + y)) \cdot (n_x, n_y) \\ &= x^2(Cf + By_0) + y^2(Cf + Ax_0) \\ &\quad - xy(Ay_0 + Bx_0) - xf(Af + Cx_0) \\ &\quad - yf(Bf + Cy_0) + f^2(Ax_0 + By_0). \end{aligned} \quad (73)$$

The equation  $h = 0$  is a curve that separates the positive and negative components of the  $(A, B, C)$  coaxis vectors. This curve passes through the Focus of Expansion and the point  $(Af/C, Bf/C)$ .

When  $h(x, y) > 0$  the normal flow values are positive. When  $h(x, y) < 0$  the normal flow values are negative; similarly, when  $h(x, y) = 0$  the normal flow values are zero. This can be seen in Figure 63a. The curve  $h = 0$  for a family of coaxis vectors is uniquely defined by the FOE with coordinates  $x_0, y_0$ .

**Rotational Coaxis Components** The rotational components of the flow vectors are defined by the three rotational components  $(\alpha, \beta, \gamma)$ . The value  $r_n$  is the rotational component of the normal flow vector along the positive direction of the  $(A, B, C)$  coaxis vector. From equation (71), the rotation component  $r_n$  is:

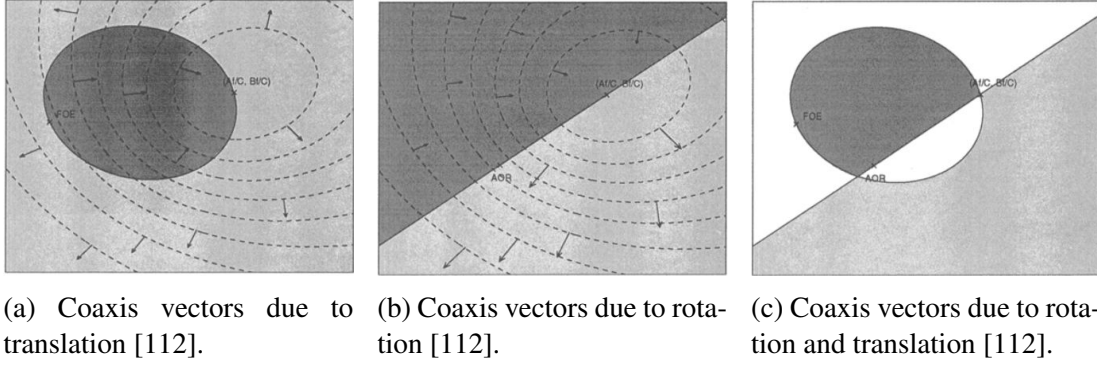


Figure 63: (a) Vectors are negative if they lie within the second order curve  $h$  defined by the FOE, and are positive at all other locations. (b) Signs of the vectors are separated by the half plane  $g$ . (c) Rigid motion defines an area of positive and negative coaxis vectors.

$$r_n = \left[ \left( \alpha \frac{xy}{f} - \beta \left( \frac{x^2}{f} + f \right) + \gamma y \right), \left( \alpha \left( \frac{y^2}{f} + f \right) - \beta \frac{xy}{f} - \gamma x \right) \right] \cdot [n_x, n_y]. \quad (74a)$$

or

$$r_n = [y(\alpha C - \gamma A) - x(\beta C - \gamma B) + \beta A f - \alpha B f] \cdot [x^2 + y^2 + f^2]. \quad (74b)$$

In a similar to the translational coaxis vector, the sign of  $r_n$  is equal to the sign of

$$g(A, B, C, \alpha, \beta, \gamma; x, y) = [y(\alpha C - \gamma A) - x(\beta C - \gamma B) + \beta A f - \alpha B f] \cdot [n_x, n_y]. \quad (75)$$

A straight line  $g(A, B, C, \alpha, \beta, \gamma)$  separates the rotational components of the  $(A, B, C)$  coaxis vectors into positive and negative signs as shown in Figure 63b. The straight line  $g$  passes through the axis of rotation  $(\alpha f/\gamma, \beta f/\gamma)$  and the point  $(A f/C, B f/C)$ .

**Combined Translational and Rotational Coaxis Constraints** To further investigate the constraints for general motion, the geometrical relations due to translation and rotation are both considered. This means that there is a line and a second order curve that separates the plane into areas containing only positive and only negative normal flow vectors. This is shown in Figure 63c. This gives an area on the image plane that contains only positive or negative normal flow vectors. These areas are known as coaxis patterns and are defined by the conic  $h$  and the straight line  $g$ . These depend on only 4 parameters, the FOE  $(x_0, y_0)$  and the AOR  $(\alpha/\gamma, \beta/\gamma)$ , but every coaxis pattern is defined by only three parameters  $(A, B, C)$ .

**Copoint Vectors** The other type of normal flow vector are the copoint vectors, these are perpendicular the lines emanating from a point  $(r, s)$ . Similar patterns to the coaxis patterns are obtained. However, in this case second order curves separate the rotational normal flow vector components, and a straight line separates the signs of the translational

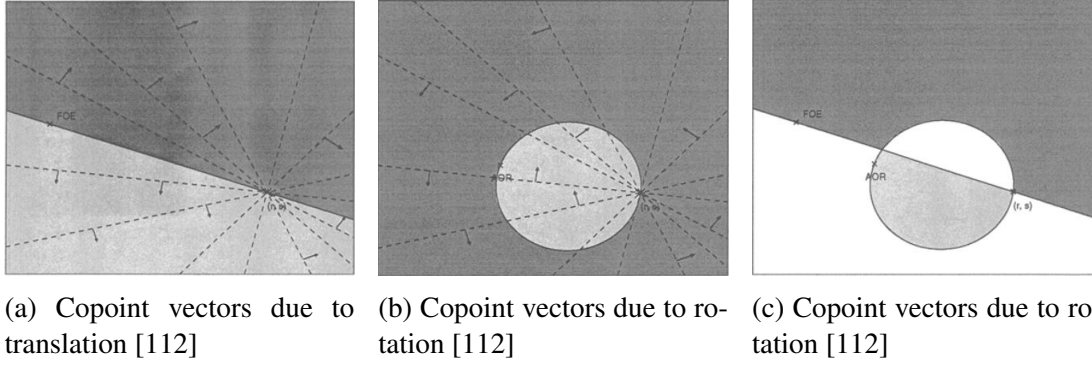


Figure 64: Copoint vectors for (a) translation, (b) rotation and (c) combination of translation and rotational motion

normal flow components. At a point  $(x, y)$  a  $(r, s)$  copoint vector of unit length in the positive direction is defined as

$$\frac{(-y + s, x - r)}{\sqrt{(x - r)^2 + (y - s)^2}}, \quad (76)$$

where  $s, r$  is the origin of the lines.

In order to define the copoint patterns, we substitute the unit vector in equation (76) for  $(n_x, n_y)$  into equation (71). Looking at the translational component defined in equation (72), we obtain the straight line  $k(r, s, x_0, y_0; x, y)$  passing through  $(r, s)$  for the translational component as shown in Figure 64a.

$$\begin{aligned} k(r, s, x_0, y_0; x, y) &= y(x_0 - r) - x(y_0 - s) - x_0s + y_0r \\ &= 0. \end{aligned} \quad (77)$$

We can then get the second order curve  $l(r, s, \alpha, \beta, \gamma; x, y)$  by looking at the rotational component in equation (74) and substituting equation (76) for  $(n_x, n_y)$ . An example is shown in Figure 64b:

$$\begin{aligned} l(r, s, \alpha, \beta, \gamma; x, y) &= -x^2(\beta s + \gamma f) - y^2(\alpha r + \gamma f) \\ &\quad + xy(\alpha s + \beta r) + xf(\alpha f + \gamma r) \\ &\quad + yf(\beta f + \gamma s) - f^2(\alpha r + \beta s) \\ &= 0. \end{aligned} \quad (78)$$

As with the coaxis patterns, we superimpose both the translation and rotational components that gives us an area of positive and negative values called the copoint patterns. This is shown in Figure 64c.

### A.1.3 Motion Recognition

There are two primary methods for recognition of motion vectors. The first recognises the motion vectors via global qualitative approach within the field of view and as discussed in [111] and [112]. There are geometrical relations between the normal flow vectors in selected regions (look at normal flow for a certain direction, and then find locations with the same sign). This allows separation between translation and rotation which separate based on the sign of the second order curve and a straight line. Only a small number of normal flow vectors are used by choosing coaxis vectors that correspond to the orthogonal coordinate axis ( $X, Y, Z$ ); this results in three  $\alpha, \beta, \gamma$  vectors as shown in Figure 62. This method can be considered as global because it uses all the information in the image and qualitative because it uses the signs of the normal flow vectors for estimating the translational and rotational axes. This method allows the optic flow to be computed in a robust way as the data is used in a global and qualitative manner and is not affected as much by discontinuity, or noise in the data.

The second alternative approach looks at the global structure of rigid motion fields [113]. It is shown that for a scene, the rigid motion vectors of certain values are constrained to lie within certain regions, forming contours. This does not mean that within these conic regions all the motion vectors will be a certain value, rather the motion vectors of a certain value will lie within the corresponding conic region for that value. The motion field possesses a global structure that is independent of the scene in view, but it does depend on the parameters of the underlying 3D motion. By looking at the geometry of the iso-motion vectors and iso-normal motion areas, each of which corresponds to the locations on the observer's retina where the motion would be a certain value. There are still some ambiguities due to the nature of a planar imaging plane. However, by using a spherical imaging plane some of these ambiguities are resolved. This is also more similar to how many insects perceive the world.

## A.2 Spherical Case

From a sequence of images, the rigid 3D motion between them can be estimated which can be used to derive the structure of the scene using the image formation equations as discussed in Section A.1.1. However, any errors in the 3D motion estimate results in a distorted version of the scene structure which can possibly result in negative depth. Negative depth means that the scene points are behind the image plane which is not correct. In terms of a planar retina, when the rotational and translational component errors are perpendicular to one another on the image plane, the optimal configuration is obtained. Spherical retina on the other hand are much better for this. This section shows that if we know the rotational error, the correct translation is the optimal one. Similarly, when we know the translational error, the optimal rotational error depends on both the direction and the value of the actual and estimated translation. This section will show that in general it is easier to estimate motion using a spherical retina and is based on research conducted in [115] and [114].

## A.2.1 Problem Formulation (spherical retina)

The 2D motion field on the image surface is the projection of the 3D motion field of scene points moving relative to the image surface. Again, we will assume rigid body motion of the observer (camera) with instantaneous translation  $\mathbf{t} = (U, V, W)$ , and instantaneous rotation  $\boldsymbol{\omega} = (\alpha, \beta, \gamma)$ . Figure 65 illustrates the image formation.

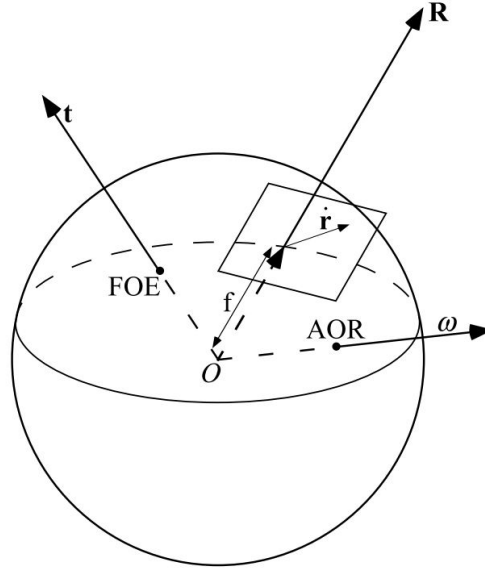


Figure 65: Image formation under perspective projection on a spherical retina [115]

Each scene point  $\mathbf{R} = (X, Y, Z)$  is measured with respect to a coordinate system  $OXYZ$  fixed to the camera. Each scene point  $\mathbf{R}$  moves relative to the camera with velocity  $\dot{\mathbf{R}}$  where

$$\dot{\mathbf{R}} = -\mathbf{t} - \boldsymbol{\omega} \times \mathbf{R}. \quad (79)$$

The negative sign for translation accounts for the perceived direction of motion, the environment translates backwards when the observer is translating forwards. Similarly for rotation, the environment is perceived to move in the opposite direction to the observers rotation. We will now consider perspective projection of the scene points onto the spherical image plane. The image is formed on a sphere of radius  $f$  (focal distance), where  $(\mathbf{r} \cdot \mathbf{r} = f^2)$ , so we can map the scene points  $\mathbf{R}$  to a point on the image plane  $\mathbf{r} = (x, y, z)$ .

$$\mathbf{r} = \frac{\mathbf{R}f}{|\mathbf{R}|}, \quad (80)$$

where  $|\mathbf{R}|$  is the norm of  $\mathbf{R}$ . If we now differentiate equation (80) with respect to time, and substitute for  $\dot{\mathbf{R}}$  into equation (79) we obtain the formula for the optic flow on the sphere.



$$\begin{aligned}
\dot{\mathbf{r}} &= \mathbf{v}_{\text{trans}}(\mathbf{r}) + \mathbf{v}_{\text{rot}}(\mathbf{r}) \\
&= \frac{1}{|\mathbf{R}|f} ((\mathbf{t} \cdot \mathbf{r})\mathbf{r} - \mathbf{t}) - \boldsymbol{\omega} \times \mathbf{r} \\
&= \underbrace{-\frac{1}{|\mathbf{R}|f} (\mathbf{r} \times (\mathbf{t} \times \mathbf{r}))}_{\text{translation}} \underbrace{-\boldsymbol{\omega} \times \mathbf{r}}_{\text{rotation}}.
\end{aligned} \tag{81}$$

Alternatively, if we look at a unit sphere of radius  $f = 1$ , the projection will become  $\mathbf{r} = \mathbf{R}/|\mathbf{R}|$  which leads to equation (81) becoming

$$\dot{\mathbf{r}} = \underbrace{-\frac{1}{|\mathbf{R}|} (\mathbf{r} \times (\mathbf{t} \times \mathbf{r}))}_{\text{translation}} \underbrace{-\boldsymbol{\omega} \times \mathbf{r}}_{\text{rotation}}. \tag{82}$$

The translational component  $\mathbf{v}_{\text{trans}}(\mathbf{r})$  depends on the depth  $Z = |\mathbf{R}|$ , the distance of  $\mathbf{R}$  to the centre of the sphere. The direction of  $\mathbf{v}_{\text{trans}}(\mathbf{r})$  is longitudinal (along great circles) pointing away from the FOE ( $\mathbf{t}$ ), or towards the FOC ( $-\mathbf{t}$ ). This can be seen in Figure 66, left. The rotational component  $\mathbf{v}_{\text{rot}}(\mathbf{r})$  does not depend on depth. Its direction is along latitudes around the AOR (clockwise ( $\boldsymbol{\omega}$ ), and counter clockwise ( $-\boldsymbol{\omega}$ )). This can be seen in Figure 66, middle.

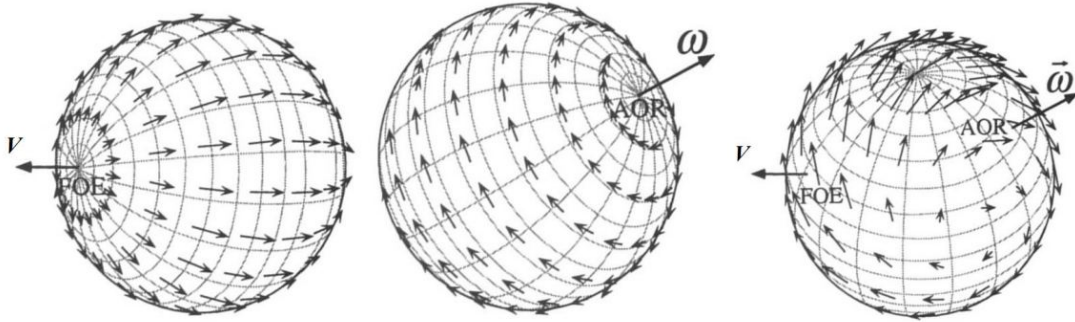


Figure 66: Optic Flow patterns on a spherical retina [114].

Left: Pure translation. Middle: Pure rotation. Right: combined translation and rotation.

As seen in Figure 66, there is an ambiguity as it is not possible to disentangle the effects of  $\mathbf{t}$  and  $|\mathbf{R}|$  (translation motion vectors will depend on the distance of scene points, where scene points close to the observer will move faster than those points further away). We are only able to derive the direction of translation.

The component of the flow  $\mathbf{u}_n$  in a specific direction  $\mathbf{n}$  (normal to the gradient) is given by

$$\mathbf{u}_n = \dot{\mathbf{r}} \cdot \mathbf{n} = \frac{\mathbf{v}_{\text{trans}}}{|\mathbf{R}|} \cdot \mathbf{n} + \mathbf{v}_{\text{rot}} \cdot \mathbf{n}. \tag{83}$$

**Distorted Space** If the exact motion parameters have been computed, the depth can be obtained from equation (83). It is more reasonable to expect some errors in the estimation of the 5 motion parameters. Because there are errors in the motion parameters, there will also be errors in the depth of the scene. The estimated parameters are represented with a hat ( $\hat{t}$ ,  $\hat{\omega}$ ,  $|\hat{R}|$ ,  $\hat{Z}$ ,  $\hat{v}_{\text{trans}}$ ,  $\hat{v}_{\text{rot}}$ ). Unmarked parameters represent the actual quantities. Errors are denoted with the subscript  $\epsilon$ , the error is defined as the difference between the actual and estimated quantity ( $\omega - \hat{\omega} = \omega_{\epsilon}$ ). Looking at equation (83), we find that an estimate of depth is

$$|\hat{R}| = \frac{\hat{v}_{\text{trans}} \cdot \mathbf{n}}{\dot{r} \cdot \mathbf{n} - \hat{v}_{\text{rot}} \cdot \mathbf{n}}. \quad (84)$$

Therefore on the image sphere, the estimated depth is

$$|\hat{R}| = |R| \cdot \left( \frac{(\mathbf{r} \times (\hat{t} \times \mathbf{r})) \cdot \mathbf{n}}{(\mathbf{r} \times (\mathbf{r} \times \mathbf{t})) \cdot \mathbf{n} + f|R|(\omega_{\epsilon} \times \mathbf{r}) \cdot \mathbf{n}} \right). \quad (85)$$

We can express the depth estimate  $|\hat{R}|$  as a multiple of the actual depth  $|R|$  with a distortion  $D$

$$D = \frac{(\mathbf{r} \times (\hat{t} \times \mathbf{r})) \cdot \mathbf{n}}{(\mathbf{r} \times (\mathbf{r} \times \mathbf{t})) \cdot \mathbf{n} + f|R|(\omega_{\epsilon} \times \mathbf{r}) \cdot \mathbf{n}}. \quad (86)$$

Equation (86) is a distortion factor for a fixed direction  $\mathbf{n}$  which is a surface in space. This distortion surface is a locus of points that are distorted in depth by the factor  $D$ . It is important to note that there will be different distortions for different directions.

### A.3 Advantages of a Spherical Retina

We will now look at coupling of motion errors on the sphere. The main focus is to look at the points in space which the 3D motion estimates give a negative depth estimate. For every direction,  $\mathbf{n}$ , the points in space with negative depth are between 0 and the  $-\infty$  distortion surface. If we look at every direction we will get a distortion surface covering a certain volume. The main interest is to minimise the negative depth volume. This is because the solution that contains the smallest negative depth volume would most likely be the correct solution for the scene.

**Rotational Error** If we already have an estimation of the rotation (from an inertial measurement unit for example), the direction of translation that minimises the negative depth volume is the correct one. This is similar to most insects that have some form of inertial sensor which gives rotational information. This rotational information might not always be correct, but it gives a basis to work from to estimate the remaining translation. This is done by subtracting the rotational motion components from the entire flow field. This will result in the translational flow.

Estimation of translational motion is simpler than the estimation of the complete 3D motion when compared to that for a planar retina.

**Translation Error** If we have estimation of translation with some error assumption, the rotational error vector  $\omega_\epsilon$  lies on different directions of the sphere. The exact value of the vector depends on the actual and estimated translation as well as the depth of the scene in view. Assuming there is an estimate of the translation (that depends on depth), the rotation estimation (that does not depend on depth) is not meaningful as it depends on the translation. This is not considered for the spherical retina case.

**Analysis on the Sphere** We will consider the case of a fixed rotational error. As shown in Figure 67, the flow vector  $\mathbf{n} = (\mathbf{s} \times \mathbf{r}) / (|\mathbf{s} \times \mathbf{r}|)$  defines a direction on the tangent plane of the sphere, where  $\mathbf{s}$  is a unit vector at each point  $\mathbf{r}$ . The rotational component  $\omega_\epsilon$  is parallel to the  $x$ -axis, so that  $\mathbf{s}$  is the set of unit vectors in the  $yz$ -plane where  $\mathbf{s} = (0, \sin \chi, \cos \chi)$  for  $\chi$  between  $[0 \dots \pi]$ . As the  $\mathbf{s}$  varies along a half a great circle (longitude), vector  $(\mathbf{s} \times \mathbf{r}) / (|\mathbf{s} \times \mathbf{r}|)$  takes on every possible orientation on the tangent plane for each point  $\mathbf{r}$ . When the set of points  $\mathbf{r}$  are on the great circle of  $\mathbf{s}$ , the direction is be zero. In Figure 67  $\mathbf{s}$  is chosen to be perpendicular to the rotational component error  $\omega_\epsilon$ . It is important to note that as  $\mathbf{s}$  varies along the great circle constantly, the value  $\mathbf{s} \times \mathbf{r}$  speeds up and slows down as it moves around the circle.

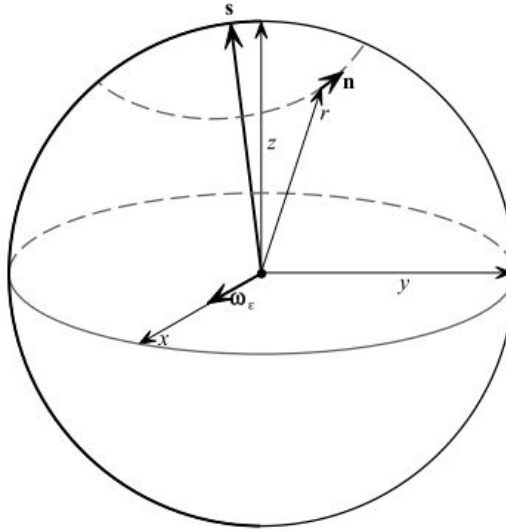


Figure 67: Sphere with tangent vector  $\mathbf{n}$  [115].

$\omega_\epsilon$  is parallel to the  $x$ -axis,  $\mathbf{s} = (0, \sin \chi, \cos \chi)$  for  $\chi$  between 0 and  $\pi$ .

Based on the estimates, we need to look at the points in space with negative depth values  $|\hat{\mathbf{R}}|$ . Looking at equation (85) for the negative depth on the sphere. Taking  $\mathbf{n} = (\mathbf{s} \times \mathbf{r}) / (|\mathbf{s} \times \mathbf{r}|)$ , and  $\mathbf{s} \cdot \omega_\epsilon = 0$  (since  $\mathbf{s} \cdot \omega_\epsilon$  are perpendicular) and setting  $f = 1$  we obtain the following inequality

$$|\hat{\mathbf{R}}| = |\mathbf{R}| \frac{(\hat{\mathbf{t}} \times \mathbf{s}) \cdot \mathbf{r}}{(\mathbf{t} \times \mathbf{s}) \cdot \mathbf{r} - |\mathbf{R}|(\omega_\epsilon \cdot \mathbf{r})(\mathbf{s} \cdot \mathbf{r})} < 0. \quad (87)$$

From the inequality in equation (87), the following constraint on  $|\mathbf{R}|$  based on the sign can be derived

$$\text{sgn}((\hat{\mathbf{t}} \times \mathbf{s}) \cdot \mathbf{r}) = -\text{sgn}((\mathbf{t} \times \mathbf{s}) \cdot \mathbf{r} - |\mathbf{R}|(\boldsymbol{\omega}_\epsilon \cdot \mathbf{r})(\mathbf{s} \cdot \mathbf{r})). \quad (88)$$

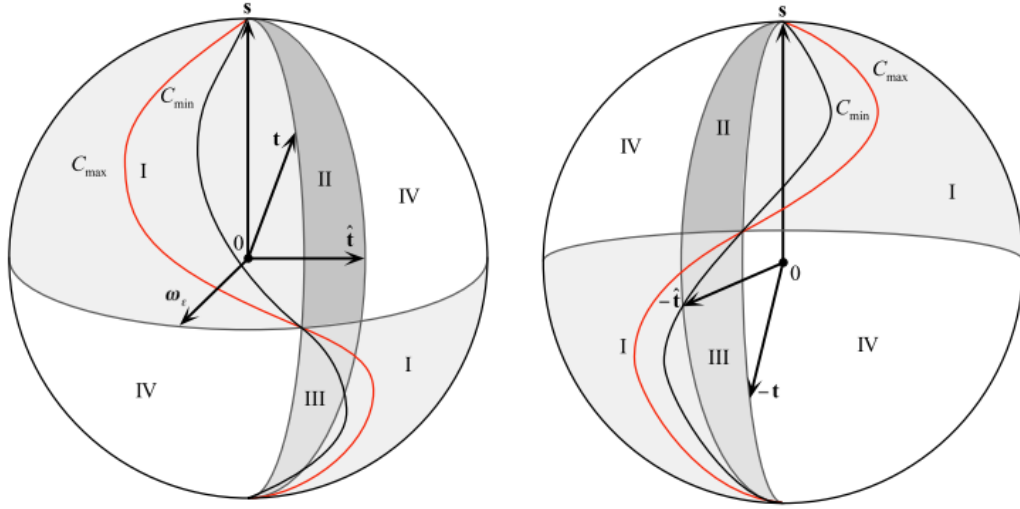
This means that at a point  $\mathbf{r}$  in the image, the constraint in equation (88) is either satisfied for all depth values of  $|\mathbf{R}|$ , or for an interval of values of  $|\mathbf{R}|$  bounded from above or below. The other case could mean that no values of  $|\mathbf{R}|$  are satisfied. Equation (87), provides a way to classify the points on the sphere in four areas (I, II, III, IV). These areas can be summarised in the table on Table 12. The locations of each area is defined by the signs of the functions  $(\hat{\mathbf{t}} \times \mathbf{s}) \cdot \mathbf{r}$ ,  $(\mathbf{t} \times \mathbf{s}) \cdot \mathbf{r}$ , and  $(\boldsymbol{\omega}_\epsilon \cdot \mathbf{r})(\mathbf{s} \cdot \mathbf{r})$ .

area	location	constraint on $ \mathbf{R} $
I	$\begin{aligned} & \text{sgn}((\mathbf{t} \times \mathbf{s}) \cdot \mathbf{r}) \\ &= \text{sgn}((\hat{\mathbf{t}} \times \mathbf{s}) \cdot \mathbf{r}) \\ &= \text{sgn}((\mathbf{r} \cdot \boldsymbol{\omega}_\epsilon)(\mathbf{r} \cdot \mathbf{s})) \end{aligned}$	$ \mathbf{R}  > \frac{(\mathbf{t} \times \mathbf{s}) \cdot \mathbf{r}}{(\mathbf{r} \cdot \boldsymbol{\omega}_\epsilon)(\mathbf{r} \cdot \mathbf{s})}$
II	$\begin{aligned} & -\text{sgn}((\mathbf{t} \times \mathbf{s}) \cdot \mathbf{r}) \\ &= \text{sgn}((\hat{\mathbf{t}} \times \mathbf{s}) \cdot \mathbf{r}) \\ &= \text{sgn}((\mathbf{r} \cdot \boldsymbol{\omega}_\epsilon)(\mathbf{r} \cdot \mathbf{s})) \end{aligned}$	all $ \mathbf{R} $
III	$\begin{aligned} & \text{sgn}((\mathbf{t} \times \mathbf{s}) \cdot \mathbf{r}) \\ &= -\text{sgn}((\hat{\mathbf{t}} \times \mathbf{s}) \cdot \mathbf{r}) \\ &= \text{sgn}((\mathbf{r} \cdot \boldsymbol{\omega}_\epsilon)(\mathbf{r} \cdot \mathbf{s})) \end{aligned}$	$ \mathbf{R}  < \frac{(\mathbf{t} \times \mathbf{s}) \cdot \mathbf{r}}{(\mathbf{r} \cdot \boldsymbol{\omega}_\epsilon)(\mathbf{r} \cdot \mathbf{s})}$
IV	$\begin{aligned} & \text{sgn}((\mathbf{t} \times \mathbf{s}) \cdot \mathbf{r}) \\ &= \text{sgn}((\hat{\mathbf{t}} \times \mathbf{s}) \cdot \mathbf{r}) \\ &= -\text{sgn}((\mathbf{r} \cdot \boldsymbol{\omega}_\epsilon)(\mathbf{r} \cdot \mathbf{s})) \end{aligned}$	none

Table 12: Using equation (87), the points on the sphere can be classified into four areas using the constraint table [115].

For a direction  $\mathbf{n}$  defined by a certain unit vector  $\mathbf{s}$  we can obtain a volume of negative depth values that consist of the volumes above areas I, II, and III. This is shown in Figure 68 for both hemispheres of a sphere. It can be seen that areas II, and III cover the same area. The size of these regions is same as the area between the two great circles between the estimated and actual values of  $\mathbf{t}$ . These two circles are defined by  $(\hat{\mathbf{t}} \times \mathbf{s}) \cdot \mathbf{r}$  and  $(\mathbf{t} \times \mathbf{s}) \cdot \mathbf{r}$ . Area I covers the remaining area of the hemisphere subtracted by the area between  $(\hat{\mathbf{t}} \times \mathbf{s}) \cdot \mathbf{r}$  and  $(\mathbf{t} \times \mathbf{s}) \cdot \mathbf{r}$ .

If the scene in view is unbounded ( $|\mathbf{R}| \in [0, \pi]$ ), the range of depth values above any point  $\mathbf{r}$  in areas I and II that will result in negative depth estimates. We can consider the lower bound  $|\mathbf{R}_{\min}| \neq 0$  and the upper bound  $|\mathbf{R}_{\max}| \neq \infty$  to obtain additional curves  $C_{\min}$  and  $C_{\max}$  where

Figure 68: Classification of image points according to constraints on  $|R|$  [115]

$$C_{\min} = (\mathbf{t} \times \mathbf{s}) \cdot \mathbf{r} - |\mathbf{R}_{\min}|(\boldsymbol{\omega}_\epsilon \cdot \mathbf{r})(\mathbf{s} \cdot \mathbf{r}) = 0, \quad (89a)$$

$$C_{\max} = (\mathbf{t} \times \mathbf{s}) \cdot \mathbf{r} - |\mathbf{R}_{\max}|(\boldsymbol{\omega}_\epsilon \cdot \mathbf{r})(\mathbf{s} \cdot \mathbf{r}) = 0. \quad (89b)$$

These curves further bound the areas of negative depth values. It can also be seen that the two curves  $C_{\min} = 0$  and  $C_{\max} = 0$  intersect at the same point as  $(\mathbf{t} \times \mathbf{s}) \cdot \mathbf{r} = 0$  and  $(\boldsymbol{\omega}_\epsilon \cdot \mathbf{r})(\mathbf{s} \cdot \mathbf{r}) = 0$ . Further details about these bounds on the depth volume is discussed below for areas I and III.

**Area I** We do not obtain any negative depth estimates for the points  $\mathbf{r}$  between the curves  $(\boldsymbol{\omega}_\epsilon \cdot \mathbf{r})(\mathbf{s} \cdot \mathbf{r}) = 0$  and  $C_{\max} = 0$ .

The volume for points  $\mathbf{r}$  between  $C_{\max} = 0$  and  $C_{\min} = 0$  are bounded from below by equation (90), and from above by  $|\mathbf{R}_{\max}|$ :

$$|R| = \frac{(\mathbf{t} \times \mathbf{s}) \cdot \mathbf{r}}{(\boldsymbol{\omega}_\epsilon \cdot \mathbf{r})(\mathbf{s} \cdot \mathbf{r})}. \quad (90)$$

The volume for points  $\mathbf{r}$  between  $C_{\min} = 0$  and  $(\mathbf{t} \times \mathbf{s}) \cdot \mathbf{r} = 0$  are bounded by  $|\mathbf{R}_{\min}|$  and  $|\mathbf{R}_{\max}|$ .

**Area III** We do not obtain any negative depth estimates for the points  $\mathbf{r}$  between the curves  $(\mathbf{t} \times \mathbf{s}) \cdot \mathbf{r} = 0$  and  $C_{\min} = 0$ .

Between  $C_{\max} = 0$  and  $C_{\min} = 0$  the volume is bounded from above by equation (91), and from below by  $|\mathbf{R}_{\min}|$ .

$$|R| = \frac{(\mathbf{t} \times \mathbf{s}) \cdot \mathbf{r}}{(\boldsymbol{\omega}_\epsilon \cdot \mathbf{r})(\mathbf{s} \cdot \mathbf{r})} \quad (91)$$

The volume for points  $\mathbf{r}$  between  $C_{\max} = 0$  and  $(\boldsymbol{\omega}_\epsilon \cdot \mathbf{r})(\mathbf{s} \cdot \mathbf{r}) = 0$ , are bounded between  $|\mathbf{R}_{\min}|$  and  $|\mathbf{R}_{\max}|$ .

For any unit vector  $\mathbf{s}$ , the corresponding negative depth volume becomes smallest if  $\hat{\mathbf{t}}$  is on the same great circle of  $\mathbf{t}$  and  $\mathbf{s}$ , so  $(\mathbf{t} \times \mathbf{s}) \cdot \hat{\mathbf{t}} = 0$ . Basically the negative depth volume is smallest when the estimated value  $\hat{\mathbf{t}}$  is close to  $\mathbf{t}$ .

Considering an unbounded scene where  $C_{\min} = (\mathbf{t} \times \mathbf{s}) \cdot \mathbf{r} = 0$  and  $C_{\max} = (\boldsymbol{\omega}_\epsilon \cdot \mathbf{r})(\mathbf{s} \cdot \mathbf{r}) = 0$ . The estimate  $\hat{\mathbf{t}}$  does not lie on the great circle defined by  $\mathbf{t}$  and  $\mathbf{s}$ , so  $(\mathbf{t} \times \mathbf{s}) \cdot \hat{\mathbf{t}} \neq 0$ . In order to minimise the negative depth volume  $\hat{\mathbf{t}}$  is changed so that it satisfies  $(\mathbf{t} \times \mathbf{s}) \cdot \hat{\mathbf{t}} = 0$ . This change of  $\hat{\mathbf{t}}$  causes the area of type II becomes an area of type IV, and the area of type III becomes an area of type I. This results in the negative depth volume only consisting of values above the area of type I.

In this Appendix, it was argued why spherical sensors have advantages over the planar retina. Furthermore, solving optic flow with spherical sensors is made much easier if the system makes use of an inertial sensor to provide an estimate of rotation. Spherical sensors do not have as many problems with ambiguities and are better for estimating self-motion, which is probably why all flying insects have spherical vision systems.

## B Appendix - Franz-Krapp Filter Derivation

This section is a technical summary of the Franz-Krapp filter, full details can be found in [9] and the derivation for a classical matched filter can be found in [48]. This Appendix requires an understanding of spherical optic flow (see Appendix A).

The Franz-Krapp filter models the directional response characteristic (locally) of the LTFC is as a projection of the image flow vector  $\mathbf{p}_i$  onto a unit vector,  $\mathbf{u}_i$ . So the linear combination over the receptor field is shown in Figure 15 and given as:

$$e = \sum_{i=1}^k w_i f(\mathbf{u}_i \cdot \mathbf{v}(\mathbf{r}_i) + n_i), \quad (92)$$

where  $e$  is the scalar output of the modelled LTFC,  $w_i$  are scalar weights (to be determined),  $f$  is a piecewise linear (or linear with saturation) scalar function modelling EMD response,  $\mathbf{u}_i$  is the unit vector of the prescribed pattern at a point  $\mathbf{r}_i$  on the sphere  $S^2$ ,  $\mathbf{v}(\mathbf{r}_i)$  is the actual optic flow vector on  $S^2$  at  $\mathbf{r}_i$  given by equation (82)<sup>10</sup>, and  $n_i$  is the scalar noise. It is assumed that there are  $k$  measurement points (the ommatidia) on  $S^2$ . Also  $\mathbf{a} \cdot \mathbf{b}$  is the inner product of  $\mathbf{a}$  and  $\mathbf{b}$ .

The unit vector  $\mathbf{u}_i$  of the given pattern at a point  $\mathbf{r}_i$  on the sphere which is defined as:

$$\mathbf{u}_i = \begin{cases} -\frac{\mathbf{a} \times \mathbf{r}_i}{\sin \phi_i} & \text{for prescribed pure rotation around axis } \mathbf{a}, \\ -\frac{\mathbf{r}_i \times \mathbf{a} \times \mathbf{r}_i}{\sin \phi_i} & \text{for prescribed pure translation along axis } \mathbf{a}, \end{cases} \quad (93)$$

where equation (82) was used again, because the optic flow pattern consists of both translational and rotational components. (see Figure 66).  $\phi_i$  is the angle between the filter axis,  $\mathbf{a}$  and the unit vector at point,  $\mathbf{r}_i$ .

For simplicity, the EMD response model  $f$  in equation (92) was assumed to be linear in the subsequent derivation, but it is important to mention Franz and Krapp also considered the saturation of  $f$ . With this assumption of the linearity of the EMD response  $f$ , the Franz-Krapp filter is now:

$$e = \sum_{i=1}^k w_i (\mathbf{u}_i \cdot \mathbf{v}(\mathbf{r}_i) + n_i), \quad (94)$$

which is fully linear.

### B.1 Output variance

There are distance variability effects which act only in the translational flow field, because angular translation depends on distance. Because of this, for  $k$  points on  $S^2$ , equation (82) can be rewritten as:

<sup>10</sup>For an overview of spherical optic flow see Appendix A.2

$$\dot{\mathbf{r}}_i = -\frac{1}{|\mathbf{R}_i|}(\mathbf{V} - (\mathbf{V} \cdot \mathbf{r}_i)\mathbf{r}_i) - \boldsymbol{\omega} \times \mathbf{r}_i, \quad i = 1, \dots, k, \quad (95)$$

and using equations (93) and (94), the translational part of the flow projection at a point  $\mathbf{r}_i \in S^2$  is given by

$$-\frac{1}{|\mathbf{R}_i|}(\mathbf{V} - (\mathbf{V} \cdot \mathbf{r}_i)\mathbf{r}_i) \cdot \mathbf{u}_i = -\frac{1}{|\mathbf{R}_i|}V_i, \quad (96)$$

with  $V_i = \mathbf{V} \cdot \mathbf{u}_i$ .

For confidence we denote,

$$\mu_i \stackrel{\text{def}}{=} \frac{1}{|\mathbf{R}_i|}, \quad (97)$$

it can be observed that the same self-motion in a different environment would result in a local flow projection that is different due to the local noise signal  $n_i$  and different unit vectors  $\mu_i + \Delta\mu_i$  at each point  $\mathbf{r}_i$ . The flow projection would change by:

$$\Delta\mu_i V_i + n_i. \quad (98)$$

For a linear model as in equation (94), would lead to the output variance given by:

$$\Delta e^2 = \left\langle \left( \sum_{i=1}^k w_i (\Delta\mu_i V_i + n_i) \right)^2 \right\rangle, \quad (99)$$

where  $\langle \dots \rangle$  denotes the expectation over all trials.

It can be assumed that  $\Delta\mu_i$  and  $n_i$  are statistically independent, so this expression can be simplified:

$$\Delta e^2 = \sum_{i=1}^k w_i^2 (\Delta V_i^2 + \Delta n_i^2), \quad (100)$$

where  $\Delta V_i^2$  is the variance of the translational flow projection and is given by:

$$\Delta V_i^2 = \Delta\mu_i^2 \langle V_i^2 \rangle. \quad (101)$$

## B.2 Minimisation of output variance

The optimal weights should minimise the variance  $\Delta e^2$ , which given by equation (100). It is also required that without noise, the average filters output magnitude should be equal to the magnitude of the rotational (or translational) vector  $\boldsymbol{\omega}$  (or  $\mathbf{V}$ ). This determines the optical flow component that is of interest. In the case of the rotational component  $\boldsymbol{\omega}$ , the filter's output in the absence of noise  $n_i$  is:



$$\sum_{i=1}^k w_i (\mathbf{u}_i \cdot (-\boldsymbol{\omega} \times \mathbf{r}_i)) = |\boldsymbol{\omega}| \sum_{i=1}^k w_i \sin \phi_i, \quad (102)$$

where equations (93) and (95) were used with assumptions:  $\mathbf{a} = \boldsymbol{\omega}$  (filter axis matched the rotational vector) and  $\mathbf{v}(\mathbf{r}_i) = -\boldsymbol{\omega} \times \mathbf{r}_i$ , so that the additional requirement becomes:

$$|\boldsymbol{\omega}| \sum_{i=1}^k w_i \sin \phi_i = |\boldsymbol{\omega}|, \quad (103)$$

or, equivalently:

$$\sum_{i=1}^k w_i \sin \phi_i = 1. \quad (104)$$

So by minimising equation (100), the optimal weights,  $w_i^R$ ,  $i = 1, \dots, k$  for the rotational component detection can be found. But this is subject to the equality constraint of equation (104), which can be solved using Lagrange multipliers. The fact that there is a single constraint given by equation (104) means there is only one Lagrange multiplier  $\lambda$ , so that the Lagrangian becomes:

$$F(w_1, \dots, w_k, \lambda) = \sum_{i=1}^k w_i^2 (\Delta V_i^2 + \Delta n_i^2) - \lambda \left( \sum_{i=1}^k w_i \sin \phi_i - 1 \right), \quad (105)$$

and the optimal solution for rotational component detection is:

$$w_i^R = N_R \frac{\sin \phi_i}{\Delta V_i^2 + \Delta n_i^2}, \quad (106)$$

where  $N_R$  is a suitable normalisation factor that satisfies  $\sum_{i=1}^k w_i \sin \phi_i = 1$  so that:

$$N_R = \left( \sum_{i=1}^k \frac{\sin^2 \phi_i}{\Delta V_i^2 + \Delta n_i^2} \right)^{-1}. \quad (107)$$

In an similar procedure, the constraint for a translational filter is:

$$\sum_{i=1}^k w_i \langle \mu_i \rangle \sin \phi_i = 1, \quad (108)$$

which leads to the optimal weights  $w_i^T$

$$w_i^T = N_T \frac{\langle \mu_i \rangle \sin \phi_i}{\Delta V_i^2 + \Delta n_i^2} \quad (109)$$

with  $N_T$  is a suitable normalisation factor that satisfies  $\sum_{i=1}^k w_i \langle \mu_i \rangle \sin \phi_i = 1$ .

As shown in equation (106) and equation (109), the optimal solution for both rotation and translation detection is weighted by the local variance of the noise  $\Delta n_i^2$  and the local variance of the translational flow  $\Delta V_i^2$ . Flow projections that contain high noise with large translational variation components receive lower weight because they strongly contribute to the variance of the filter output. In order to calculate the optimal weights, a prior knowledge the self-motion statistics, EMD noise  $\Delta n_i^2$ , and the distance statistics of the habitat are required. Because these parameters cannot be realistically known for the fly, they can be substituted with guesstimates or treated as free parameters that are fitted to the data. Franz and Krapp [9] adopted an approach based on a simplified world model, see Figure 16, which yields a low-dimensional parameterisation of the weights. The model is comprised of assumed distance statistics and flight statistics, but (although not implausible) it would change with the environment and flight profile. Future work would look at improving upon this model given research conducted in natural image statistics as discussed by work primarily by van Hateren [116].

It is worthwhile to consider in what sense the Franz and Krapp derivation [9] results in a matched filter. The input to the Franz-Krapp filter is a vector field on the unit sphere  $S^2$  which means that at every ‘pixel’ of  $S^2$  three data points are present, i.e. for a given image plane  $\mathbf{r} \in S^2$  there is an optic flow vector  $\mathbf{v}(\mathbf{r})$ . In the language of image processing, this is a multi-dimensional image, because for each ‘pixel’  $\mathbf{r}$  there are three independent values (components of  $\mathbf{v}(\mathbf{r})$ ), a situation not unlike processing of colour images (e.g. RGB). Franz and Krapp decided to convert the problem into a setting of scalar-valued pixels by taking the inner product of the input vector field with the pattern vector field. On the other hand, one could design three parallel filters (one for each component of  $\mathbf{v}(\mathbf{r})$ ) each of which would use the fundamental theorem of matched filtering theory [41], [48], that matched filter performs cross-correlation of the input with the pattern or, equivalently, that the Fourier transform of the filter’s impulse response is the complex conjugate of the pattern. Franz and Krapp’s use of scalar-valued pixels (by projecting the input vector field into the pattern) required a new derivation of the filter and forfeited all information except the relative orientation of the input and pattern vector fields. The main reason for using the scalar-values pixels in the Franz-Krapp filter is because it made for easier detection detection Compared to the vector approach (three parallel two-dimensional filters). Because with the vector approach there would be three outputs of which a single decision would have to be made.