Sensorimotor neural systems for a predatory stealth behaviour camouflaging motion

Anderson, Andrew

QUEEN MARY
AND WESTFIELD COLLEGE
UNIVERSITY OF LONDON

Department of Computer Science

Research Report No. RR-03-06 ISSN 1470-5559 December 2003

Sensorimotor neural systems for a predatory stealth behaviour camouflaging motion

Andrew Anderson

Sensorimotor neural systems for a predatory stealth behaviour camouflaging motion

Andrew James Anderson

A thesis submitted to the University of London
in partial fulfilment of the requirements
for admission to the degree of
Doctor of Philosophy

Department of Computer Science
Queen Mary, University of London
2003

ABSTRACT

This thesis presents the design and testing of a computational model of a stealth strategy inspired by the apparent mating tactics of male hoverflies. The stealth strategy (motion camouflage) paradoxically allows a predator to approach a moving prey in such a way that it appears not to move. In the model the predators are controlled by artificial neural control systems operating using realistic levels of sensory input information.

Preliminary investigations concerned the selection of an appropriate neural network approach on which to base the design of the predator's control systems. A comparison between the performance of a Self Organising network incorporating Radial Basis functions and Multilayer Perceptrons trained using Backpropagation indicated the latter to be the more appropriate solution in terms of both accuracy and network size. Simulated control systems built from three Multilayer Perceptrons were shown to be able to accurately employ motion camouflage to approach prey in two and three dimensional environments. In these simulations the prey moved along either real hoverfly flight paths or artificially generated trajectories. The camouflaged approaches were also shown to demonstrate the control system's ability to predict future prey motion.

The performance of the control systems was further tested in a psychophysical experiment that masqueraded as a computer games competition. The basis of the competition was a 3D computer game written to allow comparison of the distances that different strategies would allow a predator to approach to before being identified. In the game the player takes the role of the prey, who must shoot approaching missiles to gain a high score. It was shown statistically that missiles employing motion camouflage were in general able to approach closer before being detected and shot than the non-motion camouflage missiles tested. Therefore it was concluded that the majority of experimental subjects were susceptible to motion camouflage. It is suggested that motion camouflage control systems may be of interest to biologists, visual psychophysicists and engineers. Possible uses for motion camouflage would be in controlling predatory agents in computer games and military manoeuvres.

DECLARATION

I declare that this thesis has been composed by myself, that it describes my own work, that it has not been accepted in any previous application for a degree, that all verbatim extracts are distinguished by quotation marks and that all sources of information have been specifically acknowledged.

Additionally, some parts of the work presented in this thesis have been published in the following articles:

Anderson, A. J. & McOwan, P. W. Humans deceived by predatory stealth strategy camouflaging motion. *Proc. R. Soc. Lond. Biology Letters*. In press.

Anderson, A. J. & McOwan, P. W. 2003. Model of a predatory stealth behaviour camouflaging motion. *Proc. R. Soc. Lond.* B, 270, 489-495.

Anderson, A. J. & McOwan, P. W. 2002. 3D simulation of a sensorimotor stealth strategy for camouflaging motion. *International Conference on Neural Information Processing, Singapore*. 1805-1810.

Anderson, A. J. & McOwan, P. W. 2002. Towards an autonomous motion camouflage control system, *International Joint Conference on Neural Networks, WCCI 2002 Hawaii*. 2006-2011.

Anderson, A. J. & McOwan, P. W. 2002. An artificial neural sensorimotor control system for a biologically inspired stealth behaviour. EPSRC PREP 2002.

Note also that following completion of this thesis Mizutani, Chahl and Srinivasan (2003) have published evidence that motion camouflage is employed by dragonflies.

Mizutani, A., Chahl, J. S. & Srinivasan, M. V. 2003. Motion camouflage in dragonflies. *Nature*, 423, 604.

ACKNOWLEDGEMENTS

I acknowledge Peter William McOwan for his continued enthusiasm as my supervisor over the duration of this thesis. My father John Graeme Anderson, for somewhere along the line he must be responsible for me doing this in the first place. On a similar theme my mother Carole Anderson. Also the various members of the Peter McOwan splinter faction of the Vision Group at Queen Mary's (Keith Anderson (Mr Nasty, not forgetting nemesis Miss Nice), Adam Sherwood (Hussar), prospectively Ross Everitt) and the remainder of the Vision Group past and present (Jeff Ng (Darkman), Ong Eng-Jon, Jack (Ting-Hsun) Chang, Yongmin Li, Jamie Sherrah, Lukasz Zalewski, Andrew Graves (Phlebas), Tony Xiang, Vejen Hlebarov, Fabrizio Smeraldi, Lourdes de Agapito Vicente, Sean Gong, Dennis Parkinson, and finally the next generation Hayley Hung, Alessio Del Bue (Pazuzu) and Justin Lim) for any attempts made to subvert the course of the average week. One should also not forget Matt Bernstein and Pablo Armelin in this respect. This work was completed under an EPSRC grant.

1 INTRODUCTION

Motion camouflage (Srinivasan & Davey, 1995) is a stealth strategy intended to allow one moving body (a shadower) to hide its motion from another moving body (the prey). The specifics of how this is accomplished are covered in detail in **section 2.1**, although at this stage a brief description is helpful. A shadower may camouflage its motion by moving along a path such that its image projected onto the prey's eye emulates that of a stationary object in the environment (as perceived by the moving prey). Motion camouflage is a problem of sensorimotor integration, the shadower must be able to transform sensory input into an appropriate motor reaction. The main aim of this thesis was to design an autonomous sensorimotor motion camouflage control system.

Biological processes have solved problems of sensorimotor integration efficiently and robustly over and over again (**section 3.1** shall see that there is evidence that male hoverflies employ motion camouflage to track females). It would be highly desirable to be able to replicate these solutions artificially. In the past few decades there has been a widespread and concentrated effort on designing neurally inspired computational algorithms. However, although a large amount of empirical data has been collected, neural systems are often poorly understood from a computational viewpoint. Consequently, although they bear some resemblance to biology, artificial neural networks are quite far removed from the true complexity of their biological counterparts. Nevertheless, as shall be seen in **section 3.3.3** artificial networks have been employed in attempt to explain the operation of sensorimotor neural systems. This thesis applies artificial neural networks to the problem of motion camouflage. This approach has the inherent advantage that it may enable a greater understanding of and allow prediction of the biological systems themselves.

The main question initially placed over the project was whether artificial technology is sufficient to provide a solution to motion camouflage. There was good evidence to say that it is. Cliff & Miller (1996) were able to co-evolve sensorimotor neural networks for simulation of pursuit and evasion in autonomous 'animats'. Furthermore they observed that their pursuers and evaders evolved traits similar to those found in real animals. For instance the pursuers evolved eyes on the front of their bodies, whilst evaders evolved eyes pointing sideways. In another biologically flavoured study Krink & Vollrath (1996) were able to optimise the web construction techniques of a population of simulated artificial spiders so that they bore statistically confirmed similarities to their biological cousins. Both of these studies were successful and yielded results of some biological importance.

The main novel contributions of this thesis are as follows:

- The provision of the first autonomous motion camouflage control system capable of tracking a prey moving in 3D space;
- Demonstration that humans are susceptible to motion camouflage approaches.

The thesis also provides a detailed analysis of motion camouflage and compares the performance of two different neural approaches when applied to motion camouflage.

The organisation of the thesis is as follows. In **chapter 2** motion camouflage is described and analysed in detail. Algorithms for motion camouflage are proposed and some of the factors constraining the technique identified. **Chapter 3** looks to biology for inspiration. The chapter identifies animal behaviour that has relevance to motion camouflage, discusses how biological sensory systems can retrieve the sorts of input information necessary for motion camouflage from the environment and examines neural sensorimotor controllers. Finally the chapter provides an overview of artificial intelligence and discusses the choice of neural algorithms made to model motion camouflage. **Chapter 4** goes on to make the first steps towards the implementation and testing of an autonomous motion camouflage control system using a self organising neural network. **Chapter 5** applies the Backpropagation algorithm to the same task as **chapter 4**. The comparative results show that Backpropagation is the more appropriate technique on which to base control system training throughout the remainder of the thesis. In **chapter 6** the simulations are modified to bring them closer to the real world.

Specifically, the inputs received by the controller are set to resemble those that could be expected to be retrieved by a real sensory system. Control systems are shown to be able to employ motion camouflage to approach prey moving along real hoverfly flight paths. **Chapter 7** successfully extends the 2D simulation of **chapter 6** to 3-Dimensions and investigates the effect of further reducing the sensory input information provided. In **chapter 8** the control systems are tested in a psychophysical experiment. A computer game that incorporates the camouflage control systems of **chapter 7** is implemented and the experiment conducted under the guise of a computer games competition. In each game the player is the prey whose aim is to spot and shoot approaching predators (some of whom approach along motion camouflaged paths). This is used to demonstrate that the experimental subjects are susceptible to motion camouflage. Finally, **chapter 9** presents a summary of the work undertaken, conclusions and suggestions for areas of future study.

2 THE PROBLEM, MOTION CAMOUFLAGE

This chapter aims to provide a thorough introduction to motion camouflage and in doing so enable an appreciation of what this thesis attempts to achieve and the nature of the problems posed. The chapter is organised in four sections. **Section 2.1** provides an overview of camouflage techniques, introduces motion camouflage and suggests possible applications for motion camouflage. **Section 2.2** proposes different algorithms for motion camouflage and their requirements. **Section 2.3** addresses the influence of shadower looming (an unavoidable consequence of the shadower's approach or retreat from the prey) on the success of motion camouflage. **Section 2.4** evaluates some of the factors constraining camouflaged approaches. Finally the main points of the chapter are discussed in **Section 2.5**. For ease of explanation this chapter shall concentrate on camouflaged approaches in 2-Dimensional environments. However as shall be seen in **chapters 7** and **8** motion camouflage is equally valid in 3D.

2.1 Camouflage and motion camouflage

Use of camouflage as both an offensive and defensive strategy is widespread throughout nature. Both animals and plants employ a wide variety of techniques intended to deceive predators and prey. Camouflage is most frequently associated with disguising appearance. Many animals have evolved appearances that enable them to blend into their background (e.g. the Huntsman spider of **Fig 2.1a**). Some would have to ensure that they choose their background carefully to remain camouflaged (e.g. the typical form of the peppered moth *Biston betularia* is camouflaged against lichen covered birch trees (McFarland 1999)), others such as crab spiders may change colour to match the substrate (Foelix 1996). Animals that are not naturally camouflaged may conceal themselves by decorating their bodies. For instance spider crabs adorn themselves with sea anemones so that they resemble the ocean floor (Owen 1980). Camouflage need not only be used to make the subject inconspicuous. Advantages may be gained by mimicking something different. This could be an uninteresting object e.g. the grasshopper *Arantia rectifolia* resembles a leaf McFarland (1999), something dangerous e.g. the hoverfly *Episyrphus* *balteatus* studied in **chapter 6** and shown **Fig 2.1b** resembles a wasp), a decoy e.g. the eye spots on butterflies' and moths' wings may direct attack to non-vital bodily parts or alternatively may act to intimidate prey (Owen 1980). Equally, mimicry may be used as a lure. The scent of the bolas of the Bolas spider imitates that of moth pheremones (note that in this case camouflage is not visual) (Haynes *et al.* 2002).

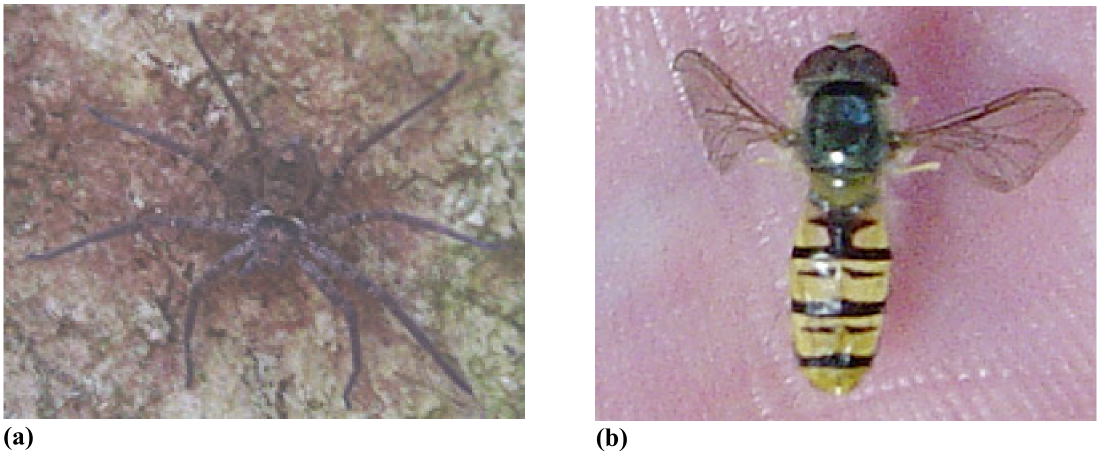


Fig. 2.1(a) A Huntsman spider is camouflaged against the tree bark upon which it sits. **(b)**. The hoverfly *Episyrphus balteatus* mimics a wasp.

Certain behaviours also may be classed as camouflage. For example beetles and possums feign death in order to discourage predators; the Eurasian Bittern sways its long neck so that it resembles the reeds against which it is camouflaged blowing in the wind (Owen 1980); and hunting cats employ stealth approaches characterised by sporadic bursts of movement, when their prey is not paying attention. The subject of this thesis is another stealth behaviour that is intriguingly designed to camouflage motion.

Motion camouflage was first suggested in 1995 by Srinivasan & Davey as a strategy that would allow one moving body (a shadower) to disguise its motion from another moving body (the prey). The basis of motion camouflage is that the shadower maintains a trajectory such that it's image projected on to the prey's retina emulates that of a distant stationary object (a fixed point). This trajectory requires the shadower to always remain directly in between the fixed point and the prey i.e. on the straight line (constraint line) connecting the position of the fixed point to the present position of the prey. For example, if the shadower wanted to catch the prey and were to start its approach positioned in front of a rock, it would ensure that it is always directly in between the rock and the current position of the prey. The optic flow of the shadower projected onto the prey's eyes would then emulate the rock. In other words, the prey would always see the shadower silhouetted against the rock, and not be alerted to the shadower's approach. An example trajectory where the shadower is approaching the prey is shown in **Fig 2.2a**. Motion camouflage could alternatively be used to retreat from the prey (**Fig 2.2b**) or to hide the shadower’s movement to a particular destination (**Fig 2.2c**). As such motion camouflage could either be used offensively or defensively. The majority of this thesis shall concentrate on motion camouflage used to approach the prey. As shall be seen in **section 3.1.1** there is evidence that motion camouflage is used by male hoverflies tracking females.

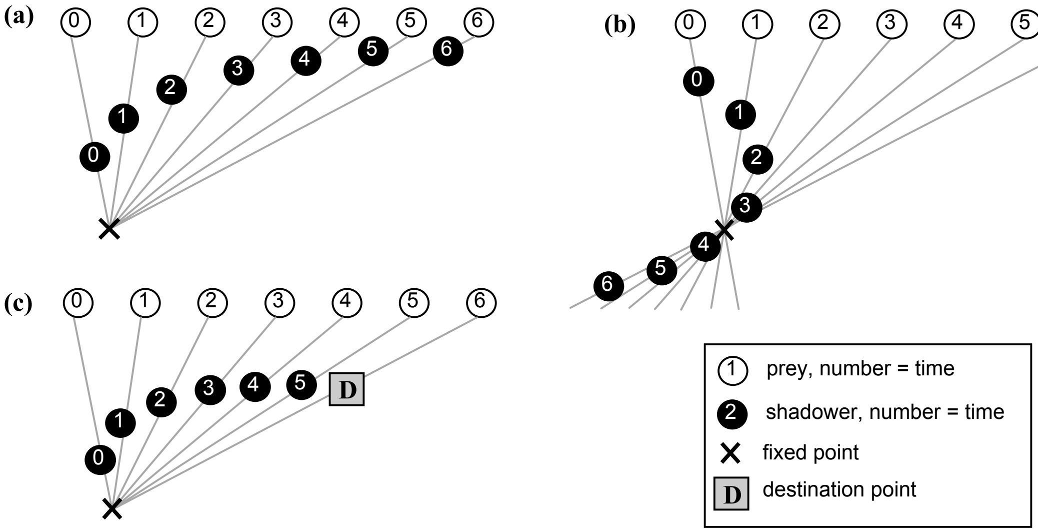


Fig 2.2a,b,c. Three possible shadower trajectories demonstrating motion camouflage. **(a)** Motion camouflage used to approach the prey, **(b)** retreat from the prey, **(c)** reach a destination point. Shadower and prey are shown at seven time instants. At each instant the shadower can be seen to adopt a position on the camouflage constraint line joining the current position of the prey to the fixed point. The fixed point could be a landmark in the environment or the initial position of the shadower.

2.1.1 Motion camouflage applications

Perhaps the most obvious real world applications for motion camouflage would be military based. For the past century the military has benefited from camouflaging troops and equipment. The mottled green and brown patterns that camouflage against vegetation are well known (see **Fig 2.3a**), their colour selected to imitate the background and their pattern, to disguise the outline of the camouflaged object. Similar to the animals discussed in the previous section, military camouflage is not confined to concealing objects against their environment. For example, it is impossible to paint ships a colour that would camouflage them adequately against the constantly changing colour of the sea and sky. Instead in WWI ships were painted with so called Dazzle camouflage designs (Williams 2001). This camouflage was characterised by large hard edged patches of contrasting colour (see **Fig 2.3b**). The intention of Dazzle camouflage was to disguise the true outline of the ship, thus confusing enemy gunners as to which side of the ship they were looking at. This (arguably) made it more difficult to estimate the ship's course and therefore more difficult to aim torpedoes. The military have also made extensive use of decoys. For example, in WWII, many false airbases were constructed to divert enemy attacks from genuine targets. Although these camouflage techniques have proved successful against humans, technological advances have rendered some, such as Dazzle camouflage ineffective. Radar and thermal imaging may be employed to determine a target's position. Modern camouflage must combat such technology. For instance, with respect to thermal imaging, contemporary military vehicles employ cooling systems and decoys may emit heat. With respect to radar avoidance, the stealth bomber is formed of many flat surfaces and sharp edges that reflect radio signals away from their source (Aronstein & Piccirillo 1997). Machinery (including the stealth bomber) may also be coated in radar absorbent material that reduces the energy of the returned radio signal.

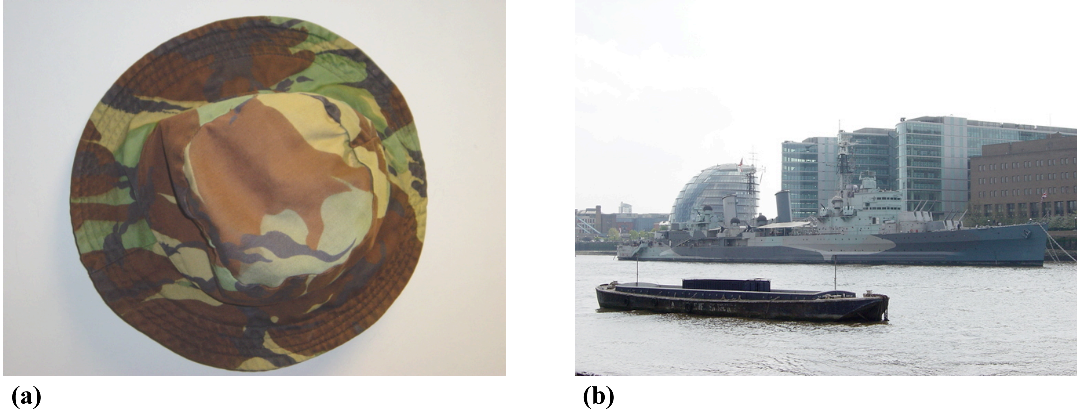

(a) **(b)**

Fig 2.3 (a) A military camouflage pattern. **(b)** The HMS Belfast (background), painted with a Dazzle camouflage design.

It is suggested that motion camouflage would be a useful strategy for aircraft or missile guidance. Camouflaged approaches would be effective against any prey that is not capable of sensing the shadower's depth directly. This implies that motion camouflage would not be successful against radar unless the shadower is capable of absorbing and/or diverting the radio signal. This situation will be addressed shortly. Considering thermal imaging, a motion camouflage missile or craft could conceivably disguise its heat from an infra red tracker by approaching against an external heat source (such as the sun). Alternatively, the missile could even create its own external dummy heat source. Such a missile could adopt a conventional approach to travel to an opportune position, at which point it would release an explosive charge. The missile would then be able to adopt a motion camouflaged approach against the heat of the explosion in the background (whilst the prey is more likely to assume that the missile itself has exploded).

Stealth aircraft aside, there seems little that could be done to properly camouflage a single approaching shadower from an accurate radar system. However, motion camouflage could still give an important advantage. If multiple missiles/attack craft were to approach simultaneously along the same camouflage constraint line (behind one another) only the foremost missile would be visible to any radar/infra red system. If the prey has no reason to expect that it is under attack from more than one missile, this could prove an ingenious method of overloading it's defences. Either way the prey would have no means of estimating the number of missiles currently approaching. Similar tactics could be employed on the ground to conceal the number of military vehicles approaching.

Finally it should be stressed that camouflage is multi-faceted in that a body may have to simultaneously employ several strategies to maintain camouflage (e.g. to conceal audio, visual and olfactory cues). In this respect, motion camouflage should not be thought of exclusively as a stand alone technique but as a technique that may be used to augment other camouflage strategies. For example, a static body may be perfectly camouflaged in appearance against its background, however may risk exposure if it moves. If this is the case, motion camouflage could be employed to minimise the chance of detection as the camouflaged body moves. Where either strategy alone would fail, the combined strategies may be successful.

The other application for motion camouflage would be for entertainment purposes. Motion camouflage could be incorporated into the artificial intelligence of 'bots' in computer games. Indeed, artificial intelligence is rapidly becoming one of the major selling points of computer games (Laird 2000). Again the likely use of the strategy would be to help conceal the approaches of predatory agents (e.g. missiles, space ships etc). An example implementation of motion camouflage in a computer game is presented in **chapter 8**.

2.2 Algorithms for Motion Camouflage

Three motion camouflage algorithms are introduced in this section. The first is the algorithm originally provided by Srinivasan & Davey (1995). As shall be seen in **section 2.2.2** for practical purposes the Srinivasan & Davey algorithm is incomplete. Two alternative algorithms are proposed in **sections 2.2.3** and **2.2.4**. Essential requirements for each are identified, though in the interests of maintaining simplicity at this stage the algorithms are discussed at a high level. Lower level constructs and implementational issues form the majority of this thesis. This section assumes that the shadower:

- possesses the relevant sensory apparatus to locate the position of prey and fixed point;
- has reaction time and flexibility of movement adequate to maintain a camouflaged trajectory;
- is at a distance from the prey such that the prey is not able to sense any difference in the shadower’s range as it moves. The problem of shadower looming is covered in **section 2.2**.

2.2.1 Positioning of the Fixed Point

Any algorithm describing motion camouflage must take into account the initial position of the shadower relative to the fixed point. Srinivasan & Davey (1995) suggest two possibilities for this position:

- The fixed point is taken to be a landmark. In this case the shadower would hope to appear to be or be located at this landmark;
- The fixed point is located at the initial position of the shadower (or indeed, less conveniently, anywhere along the line connecting the initial position of shadower and prey). Here the intention would be that the prey regards the shadower itself to be a stationary landmark.

The importance of this choice is made apparent when considering the calculations that the shadower must perform to maintain a camouflaged path. In the first case, the shadower may be able to sense (e.g. see) the positions of both prey and fixed point and would only need to ensure that it is positioned in between these. Depending on the scope of vision of the shadower this need not be such an advantage as both fixed point and prey must be viewed simultaneously (or at least the positions of both monitored frequently). In the second, the shadower has no exact visual reference with which to estimate the position of the fixed point and must co-ordinate itself using another method.

2.2.2 Srinivasan & Davey algorithm

Srinivasan & Davey (1995) show that, in theory, the shadower will remain camouflaged as long as it knows its instantaneous distance from the fixed point by:

- always viewing the prey frontally;
- always pointing radially away from the fixed point;
- making corrective yaws of angle $\Delta\theta$ and its lateral motion $\Delta\lambda$ such that the magnitudes are related by: $\Delta\lambda/\Delta\theta = \rho$ where ρ = the distance of the shadower from the fixed point (see **Fig 2.4**). Note that hoverflies are capable of rotating

independently of the direction they are flying (Collett & Land 1975), see also **section 3.1.1**).

The ratio $\Delta\lambda/\Delta\theta = \rho$ assumes that the lateral component of the shadowers movement alone would carry it along the circumference of a circle centred at the fixed point with radius ρ (for the ratio to hold the lateral component of movement alone must correspond to an arc on the circumference of this circle). However, for practical purposes, the algorithm is lacking in two respects. Firstly it implicitly makes the assumption that the shadower already knows the appropriate yaw or lateral component of movement to make. Unless the shadower can be certain of the prey's pattern of movement in advance, neither of these terms can be known instantaneously and both must be estimated. Furthermore, in order to estimate either, the shadower must have a concept of its distance from the prey (see **Fig 2.4c**). Secondly, the algorithm does not specify how the shadower's lateral movement relates to its movement forward of backward. The algorithms proposed in the following sections address these problems.

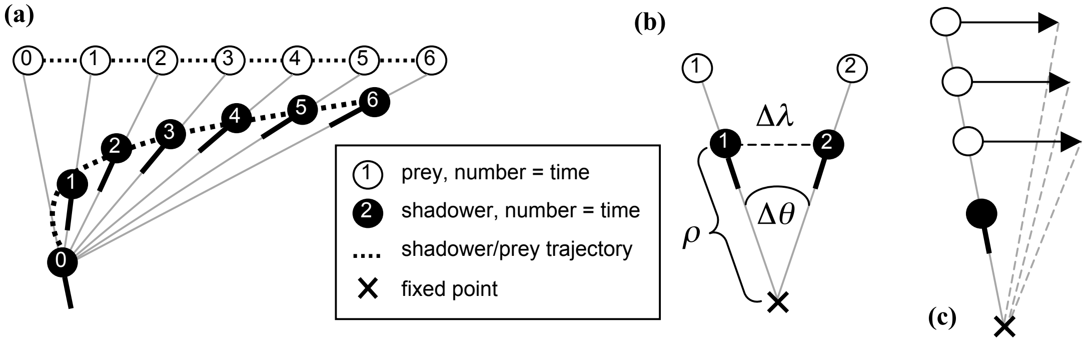


Fig 2.4. Demonstration of the motion camouflage algorithm of Srinivasan & Davey (1995).

(a) A possible shadower trajectory. The shadower can be seen to continually rotate so as always to view the prey frontally.

(b) The underlying mechanism. The shadower makes corrective yaws ($\Delta\theta$) and corrective lateral movements ($\Delta\lambda$) such that the two are related by $\Delta\lambda/\Delta\theta = \rho$. As the distance ($\rho$) increases the lateral movement ($\Delta\lambda$) must be made proportionally larger in relation to the yaw ($\Delta\theta$).

(c) to be able to estimate $\Delta\theta$ the shadower must have a concept of the distance to the prey. The same prey movement is shown with the prey at different distances from the shadower, each would require the shadower to move differently to maintain camouflage.

2.2.3 Responsive algorithm

Adopting the responsive algorithm the shadower would move so that it is correctly camouflaged according to the time instant at which it commenced its movement. However on completion of the move the prey may have moved elsewhere. Thus, although this strategy may never involve the shadower lying on the current constraint line it would enable a reduction in apparent motion. This is especially true if the shadower is able to detect very small changes in $\Delta\theta$ over very short time intervals Δt and respond by making corrective movements that are so rapid that they are not noticed by the prey. As Δt, and the responsive corrective movements tend to 0 this algorithm becomes equivalent to that of Srinivasan & Davey (i.e. the shadower spots infinitesimally small changes in the position of the prey and responds by making infinitesimally small movements to maintain camouflage).

To perform this algorithm the shadower would need to be able to estimate the position and orientation of the constraint line relative to itself. This may be calculated from the following information using trigonometry (see also **Appendix 1**):

- the angle subtended at the fixed point by shadower and prey (to estimate this the shadower must have some concept of the distance to the prey);
- the distance between shadower and fixed point.
- the set of positions the shadower can expect to reach with its next movement.

Alternatively:

- the angle subtended at the prey by shadower and fixed point (for this estimation the shadower must have a concept of the distance to the fixed point);
- and the distance between shadower and prey;
- again, the set of positions that the shadower can expect to reach with its next movement.

The rotational requirement of the Srinivasan & Davey algorithm is equivalent to the estimate of the angle subtended at the fixed point by shadower and prey. This is not to say that in practice rotation may not be helpful to the shadower (see **section 6** and **7**), simply that it is not strictly necessary for performing motion camouflage.

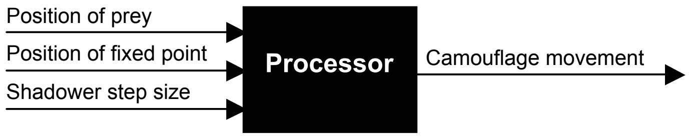


Fig. 2.5. Information and processing required by the responsive algorithm.

The responsive algorithm is relatively sparing in its processing requirements (see **Fig 2.5**). The shadower need only recognise that it is not on the present constraint line (i.e. it is not directly in between prey and fixed point) and move to rectify the situation. However, if the shadower is not able to react quickly enough, the prey will notice the lag in the shadower's response (an example responsive shadower trajectory and the corresponding visual error (i.e. retinal slip) incurred is shown in **Fig. 2.7a**). In this case, if the shadower were able to reliably predict the position of future constraint lines (i.e. indirectly predicting the prey's forthcoming movement) and move towards these instead then it should be able to improve its camouflage.

2.2.4 Predictive algorithm

The predictive algorithm requires the shadower to be able to predict future camouflaged positions based on prior prey movement and move accordingly. As such, it is an extension of the responsive algorithm adding complexity and some limitations. It also introduces the possibility of performing perfect motion camouflage. Furthermore a predictive capability could help the shadower to regulate its movement to optimise certain parameters. For example, prediction could help the shadower to optimise its approach trajectory to reach the prey in the minimum time, or with the minimum energy expenditure.

Regarding the limitations, the conditions in which the predictive algorithm would be useful are dictated by a combination of prey characteristics and environmental factors. These may be reduced to:

- foremost whether it is indeed possible to predict forthcoming prey movements;
- if so, whether it is possible to obtain information sufficient to make accurate predictions of forthcoming prey movements;
- finally whether it is possible to process this information in time to allow undetected movement to the next camouflaged position.

To predict future camouflaged positions the shadower requires some knowledge of past events. It is suggested that in addition to the requirements of the responsive algorithm the shadower is likely to have to:

- possess some sort of memory of previous positions of the constraint line/prey;
- be able to combine this memorised information with the current sensory input to predict a position on the next constraint line;
- time its movement so that it coincides with the prey (so as not to over or under step the next constraint line). For this it is likely to be helpful if the shadower possesses a memory of its own recent movement.

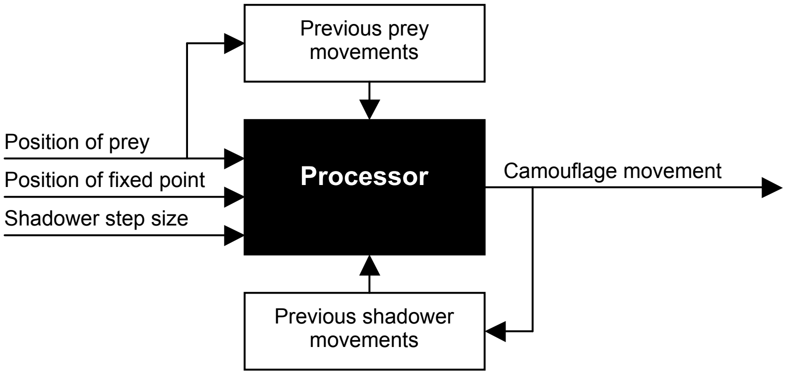


Fig. 2.6. Information and processing required by the predictive algorithm.

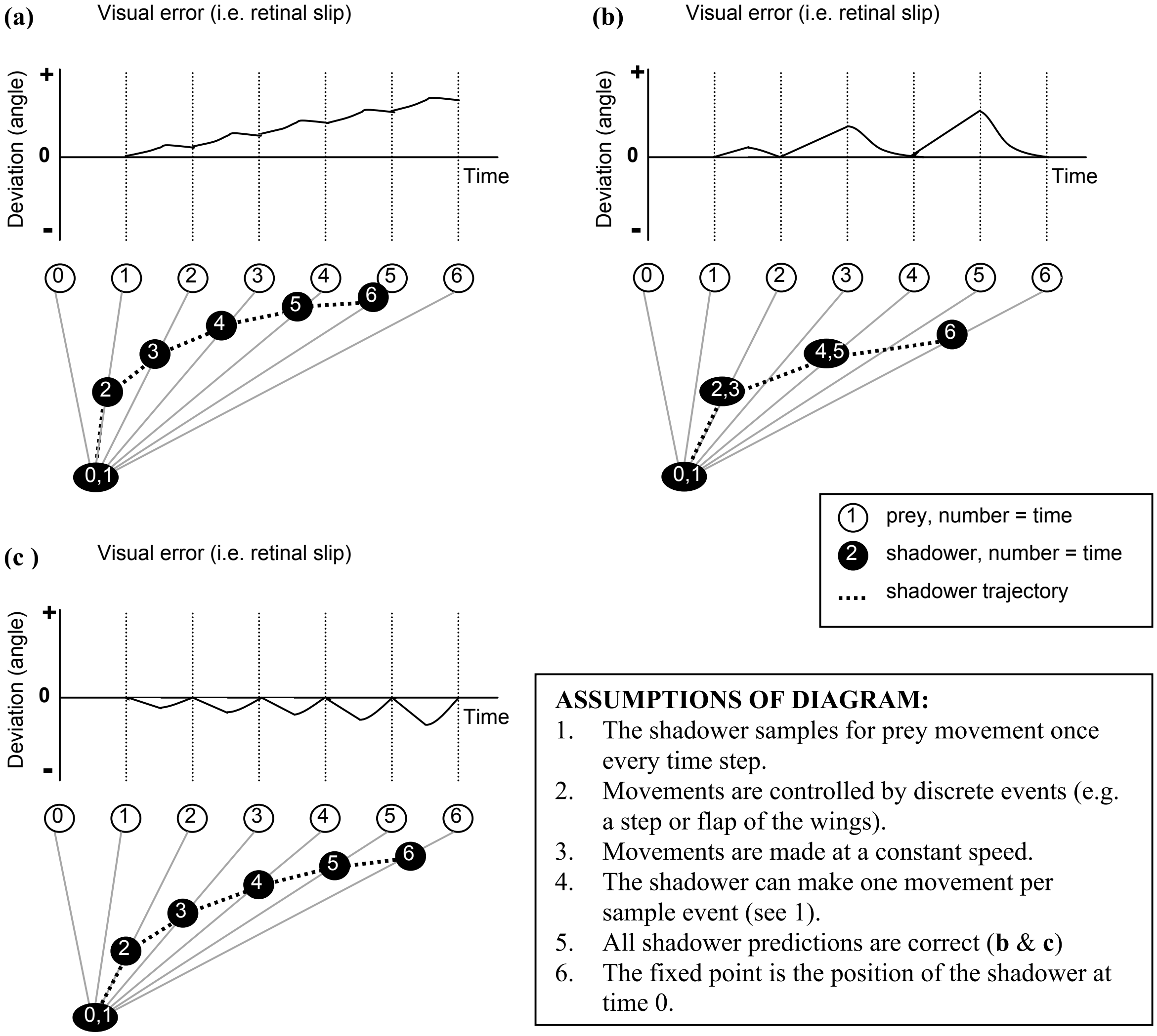
Two examples of shadowers following predictive trajectories are shown in **Fig. 2.7b** & **c**. The two respective shadowers differ in the time that it takes them to make predictions. This difference is highlighted by the increased visual error seen in **Fig. 2.7b**. However, depending upon the vigilance of the prey, either could be adequately camouflaged.

2.2.5 Discussion of algorithms

This section has introduced three different algorithms for motion camouflage. The Srinivasan & Davey algorithm and two adaptions of this made for practical purposes. These provide the grounds for the models presented throughout this thesis. This section has achieved two purposes:

Firstly, to show that there is not necessarily a definitive motion camouflage algorithm that is optimal in all situations. The responsive algorithm rarely involves the shadower lying on the present constraint line but can not be surpassed if it is not possible to predict future prey movement. Alternatively the predictive algorithm could allow perfect motion camouflage but is liable to increased error resulting from incorrect prediction. However, it is reasonable to expect that an entity performing motion camouflage would at some level have to adopt the sorts of strategies described in this section which rely upon the following: Physically, a shadower must have rapid reactions and be sufficiently dextrous to follow a camouflaged path. It must have a concept, whether explicit or implicit, of its own distance from the fixed point and the angle subtended at the fixed point by itself and the prey. To estimate this angle the shadower must possess the sensory apparatus to sample the spatial arrangement of important objects in its environment. Finally it must have the ability to manipulate this information to produce the appropriate motor responses which may require mechanisms to predict subsequent prey movements.

Secondly, the section has emphasised the apparent importance of the prey's characteristics in determining the appropriate camouflage algorithm to use. For example, returning to **Fig.2.7**, suppose that the prey is very good at detecting movement and also



ASSUMPTIONS OF DIAGRAM:

1. The shadower samples for prey movement once every time step.
2. Movements are controlled by discrete events (e.g. a step or flap of the wings).
3. Movements are made at a constant speed.
4. The shadower can make one movement per sample event (see 1).
5. All shadower predictions are correct (**b** & **c**)
6. The fixed point is the position of the shadower at time 0.

Figs 2.7a,b,c. Three hypothetical motion camouflage strategy trajectories. The line graphs represent an approximation of the visual error (measured as the angle subtended at the prey by the shadower and the fixed point). The visual error diagrams are drawn, though they have been verified with values calculated using trigonometry. The magnitude of the error increases as the shadower approaches the prey as a result of geometry (coupled with the fact that the shadower is constrained to a single movement per time step). The shadower always remains at the initial position for a single time step in order to monitor the first prey movement from which to base its subsequent movements.

(a) (***Responsive***) The shadower detects the movement of the prey and then updates its position to lie on the constraint line correct for the previous position. Whilst moving the shadower continues to monitor movement so that it may always lie on the previous constraint line. Therefore detection and movement are undertaken at the same time within a single time step. Tracking this prey trajectory the shadower is unlikely to occupy a position on the current constraint line after it has commenced motion.

(b) (***Delayed predictive***) The shadower detects movement of the prey, accurately predicts the forthcoming position of the constraint line and moves there. Monitoring and prediction take one time step, movement a second. Using this strategy the shadower occupies a position on the current constraint line once every second time step.

(c) (***Perfect predictive***) The shadower simultaneously detects movement of the prey, accurately predicts the forthcoming position of the constraint line and moves there accordingly. Whilst moving the shadower continues to monitor the movement of the prey and accurately predicts the next prey position, so that with its next movement it can meet the constraint line. Now the shadower lies at a position on the constraint line once every time step. However, due to the shadowers straight step (and constant speed) it still deviates from the constraint line in between time steps.

has a very high sample rate (i.e. it performs checks to see whether objects have moved very frequently). In this case a shadower adopting the *responsive* strategy of **Fig 2.7a** or the *perfect predictive* strategy of **Fig 2.7c** would be most likely to remain undetected. Both involve small frequent movements, so small that they could be discounted as being associated with the motion of an approaching object. In contrast the shadower of **Fig 2.7b** (*delayed predictive*) would periodically appear to make quick movements. Now, consider a prey that samples less frequently, perhaps at evenly spaced intervals of two time steps, starting at time 0. Here the strategies of **Fig 2.7b** (*delayed predictive*) or **Fig 2.7c** (*perfect predictive*) would leave the shadower perfectly camouflaged, whereas movement would be obvious in **Fig 2.7a** (*responsive*). If the sample frequency were doubled only the shadower of **Fig 2.7c** (*perfect predictive*) would remain totally concealed. Lastly there could be a prey that simultaneously samples for movement over a wide range of frequencies where the shadower of **Fig 2.7c** (*perfect predictive*) would be the most likely to be camouflaged.

2.3 Shadower looming

Up until this point, discussion has been made under the assumption that the shadower is at such a distance from the prey that it appears as a structureless dot and any looming cues resulting from the shadower's approach are insignificant. Motion camouflage need not involve a change in the distance between shadower and fixed point, but as almost all applications would involve this, shadower looming can not be ignored. Short of the shadower changing its morphology on approach or retreat there seems no obvious method of eliminating looming cues. This section therefore aims to illustrate the effects of looming and identify methods by which looming cues may be reduced. It is hoped that this shall give an impression of the constraint imposed on motion camouflage by shadower looming and consequently an idea of the situations in which motion camouflage is likely to be successful.

Section 2.1 has already recognised that the relative success of motion camouflage is likely to be prey specific. In a given situation the relationship between the:

- precision of prey vision;
- frequency that the prey is able to process visual information;

will determine the:

- speed that the shadower may approach (or retreat);
- related to this speed, the spatial range over which motion camouflage can successfully be performed without looming cues arousing prey attention.

This need not imply that motion camouflage becomes useless beyond the point at which the prey notices that the fixed point appears to have changed size. Motion camouflage may still give the shadower an advantage by minimising its apparent motion, especially if the environment is dominated by movement.

It is possible to gain an impression of the impact of shadower looming through approximating the apparent size of the shadower as an angle ϕ in two dimensions. ϕ is the angle subtended at the prey's retina by the two outmost points of the shadower given by:

$$\phi = 2 \cdot \arctan\left(0.5 \cdot w_s \cdot d_s^{-1}\right) \tag{2.1}$$

where w_s is the width of the shadower (the distance between the two outmost points) and d_s is the distance between shadower and prey. **Fig. 2.8** shows the variation in ϕ as d_s is changed and w_s held constant. As the shadower moves closer to the prey, the angle at the prey's retina can be seen to rise exponentially to π radians. As the separating distance tends to infinity the angle gradually tends toward 0. So, as already mentioned, motion camouflage is a technique likely to work better at distance. However the minimum distance at which motion camouflage is effective may be reduced if the fixed point is a landmark of greater size than the shadower.

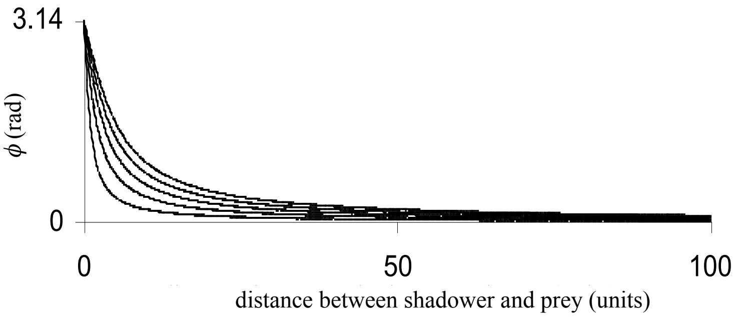


Fig. 2.8 The change in apparent size of shadower on approach. The apparent size is measured as the angle subtended at the prey's retina by the two outmost points of the shadower. Lines are plotted for 5 shadower widths of {2,4,6,8,10} units, with the lowest line representing a width of 2 and the upper line a width of 10.

2.3.1 Reduction of looming cues through choice of large fixed point

It is suggested here that an appropriately sized, textured and placed fixed point could provide a means of reducing looming cues. The proposition is that if the fixed point is larger than the shadower and the shadower is adequately physically camouflaged against the fixed point then looming will be less apparent. This would be effective up until the point at which the apparent size of the shadower exceeds that of the fixed point. Furthermore this may help to reduce the impact of the shadower's errors. This is illustrated in **Fig. 2.9**, where the shaded area, accounting for a significant portion of the distance between fixed point and prey, represents this area of tolerance to looming and movement error.

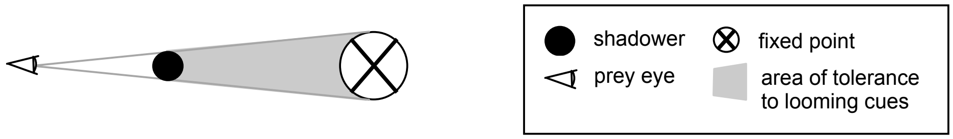


Fig. 2.9. reduction of looming cues by a large fixed point. The shadower is assumed to be camouflaged against the fixed point. It is shown at a position beyond which any movement in the direction of the prey would incur looming cues. In a perfect situation it would be camouflaged as long as it lies within the shaded area. The diagram is demonstrative of a single time instant. The shape of the shaded area of tolerance would change as the prey moves.

In two dimensions, the distance between shadower and prey d_s beyond which a large fixed point would fail to conceal looming (i.e. the angle subtended at the prey's retina by the shadower equals the angle subtended by the fixed point) is given by:

$$d_s = \frac{w_s \cdot d_{fp}}{w_{fp}} \tag{2.2}$$

where d_{fp} is the distance separating prey and fixed point, w_{fp} is the width of the fixed point (the distance between the two outmost points of the fixed point) and similarly w_s is the width of the shadower. **Figs. 2.10a&b** attempt to quantify the advantage offered by a large fixed point. The diagrams show, that in perfect conditions, as the width of the fixed becomes large, d_s asymptotes. Also the larger this width, the lesser the rate of change of d_s with respect to d_{fp}. For example, if the fixed point is four times the width of the shadower, looming cues would only be unavoidable when the shadower is located within a distance of the prey one quarter that between prey and fixed point.

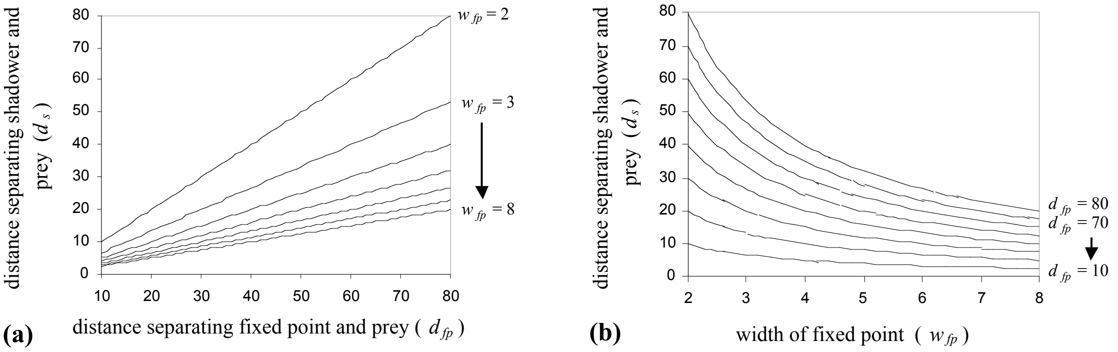


Fig. 2.10. Change in the minimum distance between shadower and prey (i.e. minimum d_s) at which a large fixed point could act to reduce looming cues. In both **a** and **b** shadower width is held constant at 2 units.

(a) Change as the width of the fixed point is held constant at 2, 3, 4, … 8 and the distance separating fixed point and prey is varied.
(b) Change as the distance separating fixed point and prey is held constant at 10, 20, 30…80 and the width of the fixed point is varied.

2.3.2 Reduction of looming cues by movement out of preys vision

The alternative method proposed here for reducing looming cues attempts to exploit prey with incomplete visual fields. It is suggested that the shadower could control its apparent size by gradually moving out of or into the preys field of vision as it approaches or retreats (see **Fig 2.11**). In doing so, it would be possible for the shadower to maintain its apparent image projected onto the prey's eye at a constant size. The successful coupling of this strategy with motion camouflage would be reliant on the fixed point being continuously on the boundary of the prey's field of view. The unfortunate implication being that the technique will only be relevant to motion camouflage if the prey stops and continues to look in the same direction, thus allowing the shadower to control looming.

The benefits of this approach can be seen to be very scenario dependent. It is limited not only by the situation in which it may be used, but also by the necessity for the prey to have a suitable field of vision and the shadower to be suitably homogenous in appearance from one extremity to the other. Nevertheless, if the prey were prone to stopping, it could provide an advantage.

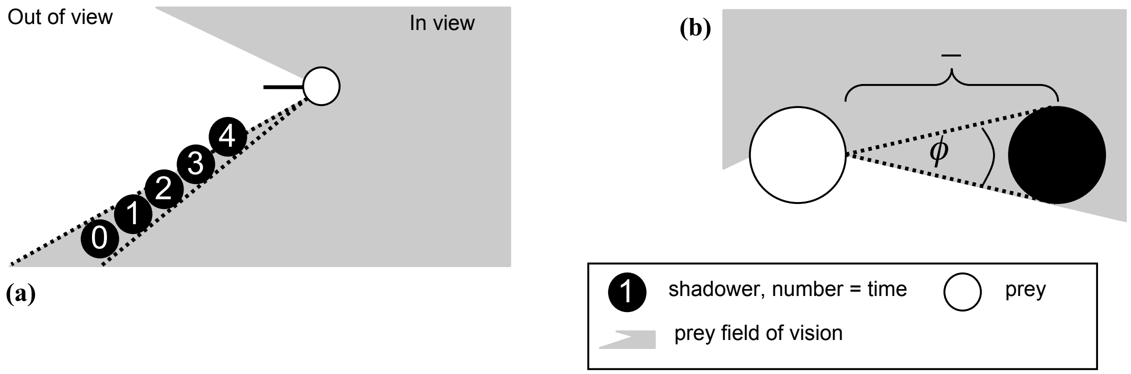


Fig 2.11 a A shadower trajectory intended to reduce looming cues. The prey is best thought of as being stationary. As the shadower approaches the prey it moves out of the preys field of vision so as to appear a constant size at all times. **(b)** To reliably follow this trajectory the shadower requires knowledge of the distance between it and the prey (_) and the angle subtended at the prey by its extremities (ϕ) (or the extremities of the fixed point) at the time when it is first touching the boundary of the visual field.

2.3.3 Summary of shadower looming

Summarising, in this section it has been seen that looming cues are an unavoidable consequence of a shadower's approach, though are only likely to be of importance when the shadower is close to the prey. It has been suggested that if the shadower is camouflaged against a large fixed point then looming cues may be reduced. However the levels of reduction and indeed the degree to which looming is disadvantageous must be seen as prey specific and a possible topic for future investigation.

2.4 Camouflage constraint

The number of different camouflaged moves a shadower may make at any one time is very much dependent on situational factors. This section seeks to identify how the relative positions of shadower, prey and fixed point constrain movement. Possible environmental influences (e.g. wind) are ignored.

It is assumed that it is possible to break down the shadower's movement into discrete events or steps. The shadower's step size will determine the set of possible positions on the forthcoming constraint line that the shadower can reach. Looked at differently this could be seen as the set of directions in which the shadower may move so as to remain camouflaged. For a given shadower location and vector describing the prey's forthcoming move the set of positions available are controlled by:

- the angle subtended at the prey by its future position and the position of the fixed point;
- the distance between prey and fixed point;
- the distance between shadower and fixed point.

The effects of these factors are illustrated in **Fig. 2.12**. The level of constraint can be seen to increase as the angle between the forthcoming prey position and the fixed point gets larger. Maximum constraint occurs when the angle subtended at two contiguous prey positions by the fixed point is equal (i.e. an isocoles triangle is formed by contiguous prey positions and fixed point). Also as the shadower moves towards the prey, geometry imposes additional constraint.

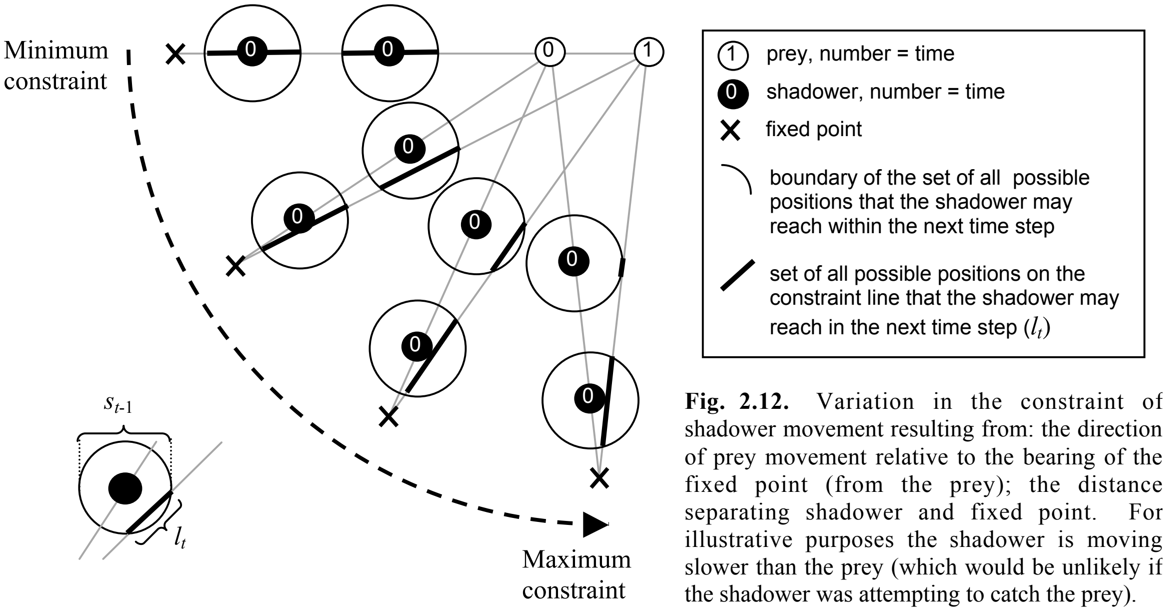

Minimum constraint
0
0
0
1
0
0
0
0
0
0
0
s_{t-1}
l_t
Maximum constraint
1 prey, number = time
0 shadower, number = time
fixed point
boundary of the set of all possible positions that the shadower may reach within the next time step
set of all possible positions on the constraint line that the shadower may reach in the next time step (l_t)

Fig. 2.12. Variation in the constraint of shadower movement resulting from: the direction of prey movement relative to the bearing of the fixed point (from the prey); the distance separating shadower and fixed point. For illustrative purposes the shadower is moving slower than the prey (which would be unlikely if the shadower was attempting to catch the prey).

In order to investigate these relationships it is necessary to provide a more formal measure of constraint (see also **Fig 2.13d**): Let l_t denote the distance encompassing the set of points that the shadower can reach on the constraint line at time t. Let s_t denote the maximum value of l_t (when the shadower is moving toward or away from the prey, this distance corresponds to twice the shadower's step size i.e. the diameter of the circles in **Fig 2.12**). So if the shadower is enabled complete freedom of movement within a circle of radius r, and a movement is assumed to last one time step, then s_t is described by:

$$s_t = 2 \cdot r_{t-1} \tag{2.3}$$

Equ (2.3) allows constraint to be measured in terms of a value C_t:

$$C_t = \frac{l_t}{s_t} \tag{2.4}$$

C_t therefore assumes values in the range 0 to 1. 0 when the shadower can not actually reach the constraint line and 1 when the shadower is already positioned on the constraint line.

Figs 2.13a-c demonstrate the change in C_t for the same prey movement (shown in **Fig 2.13d**) as a result of different shadower and fixed point positions. Replicating **Fig 2.12** at time t-1 the shadower is camouflaged. Both shadower and prey move a distance of one unit per step (i.e. $r_{t-1} = 1$). No significance is attached to the fact that the shadower is moving at the same speed as the prey, it is purely for illustrative purposes. **Fig 2.13a** reiterates the information presented in **Fig 2.12** and helps to clarify the extent of the constraint caused by the angle subtended at the prey by forthcoming prey position and the fixed point (a_{fp}). When the shadower is close to the fixed point a_{fp} has little relevance,

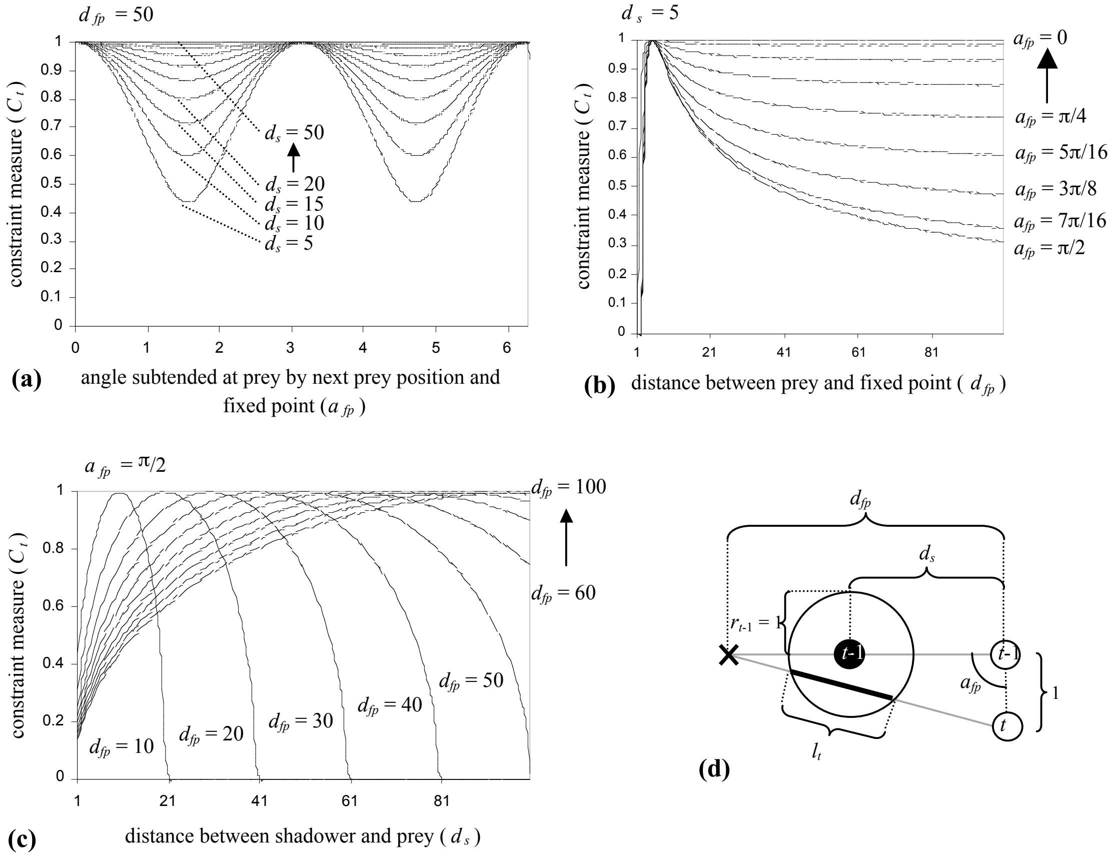


Fig. 2.13a,b,c. Change C_t corresponding to different positions of fixed point and shadower relative to the prey.

(a) Change in C_t as a_{fp} is varied between 0 and 2π radians. The distance separating shadower and prey is held constant at 50 units. Each line represents a different distance between shadower and prey.

(b) Change in C_t as the distance between shadower and fixed point is varied between 1 and 100. The distance separating shadower and prey is held constant at 5. Each line represents a different a_{fp}.

(c) Change in C_t as the distance between shadower and prey is varied between 1 and 100. a_{fp} is held constant at π radians. Each line represents a different distance between fixed point and prey.

however as the shadower approaches the prey its importance increases rapidly. In the example the difference between the maximum and minimum C_t value resulting from change in a_{fp} is approximately 0.56. This range will change depending on the distance between fixed point and prey (d_{fp}). In **Fig 2.13b** the shadower is held at a constant distance from the prey whilst the distance between prey and fixed point (d_{fp}), is varied, each line represents a different a_{fp}. This diagram indicates that once beyond a certain distance, the size of d_{fp} is inconsequential to C_t. Again the significance of a_{fp} on the value

of C_t is highlighted. The relatively small distance between shadower and prey (d_s) has been picked to emphasise the pattern of the relationship. As d_s is increased C_t can be expected to increase wherever $d_{fp} > d_s$. The sudden drop in C_t when $d_s > d_{fp}$ is explained with reference to **Fig 2.13c**. **Fig 2.13c** represents one of the more severe scenarios for motion constraint where a_{fp} is held constant at π/2 radians. The distance to the shadower is varied and each line represents a different d_{fp}. As has been seen, as the shadower moves further away from the fixed point, the level of constraint increases (C_t decreases). What is especially interesting is that this graph identifies a limitation in the use of motion camouflage for retreating from the prey. As can be seen in **Fig 2.13c**, when d_s is just over double d_{fp} the shadower can not physically move far enough to maintain camouflage (C_t = 0). Indeed, for each configuration of shadower, prey and fixed point (with the exception of the case when the prey is moving directly toward or away from the fixed point), there will be critical distance beyond which the shadower will not be able to reach the constraint line. This critical distance will vary with a_{fp} and d_{fp}. Investigation of the effects on the critical distance of varying these parameters is not undertaken here as emphasis throughout the remainder of the thesis is placed on the shadower approaching the prey.

In closing, the section has raised two important points. Firstly, that for the same prey trajectory there can be considerable variation in the constraint of camouflaged movements. Ignoring possible environmental influences the level of constraint has been shown to be dependent upon the relative positioning of shadower, fixed point and prey, their physical abilities and directions of movement. It follows that different prey trajectories will constrain motion differently. This suggests that it may be possible to design prey trajectories that maximise the constraint imposed on shadower movement. Secondly, that the shadower may be able to minimise the level of constraint on its movement if it is able to predict the bearing at which the prey is likely to appear and the direction that is likely to move in, and position itself accordingly (i.e. if the shadower can guess where its prey is going to come from, it may benefit from lying in wait in an opportunistic position). Aside from this, the level of constraint has been shown to increase the closer the shadower is to the prey, again emphasising that motion camouflage is a technique better suited to gaining an advantage at distance.

2.5 Summary

This chapter has introduced motion camouflage as a stealth strategy designed to conceal the motion of one moving body from another. Two general algorithms for motion camouflage have been proposed and were named as the responsive and predictive algorithms. The major difference between the algorithms is that the responsive algorithm, theoretically, can be performed using only the current perceptual input, whereas the predictive algorithm requires knowledge of past events. Throughout this thesis a particular area of interest will be the variation in the shadower's performance as the input information available to the shadower is changed. For instance the shadower may not have direct access to its distance from the fixed point and could have to estimate this from other cues (**chapter 3** discusses methods used by animals to retrieve such information). Advantages and disadvantages of the algorithms were suggested and it was concluded that neither is universally suitable (i.e. if the prey trajectory is erratic, prediction may be impossible. Alternatively, if the prey is particularly attentive it may notice the lag in response characteristic of the responsive algorithm).

Shadower looming was identified as a factor that may reveal the motion of the shadower, and methods to reduce the impact of this (including choice of a large landmark as fixed point) proposed. The chapter also identified that the relative positions of shadower, prey and fixed point will constrain the number of camouflaged moves available to the shadower. This implies that different prey trajectories constrain motion camouflage differently and as such it may be possible to design prey trajectories that maximise this constraint. Finally, both of the above observations suggest that motion camouflage is a technique likely to be more successful the greater the distance from the prey (i.e. looming cues are less obvious and constraint of camouflage movement is less).

3 BIOLOGICAL MECHANISMS AND ARTIFICIAL INTELLIGENCE

This chapter introduces the processes existing in biology that provide the basis of the motion camouflage models presented in this thesis. The chapter commences in **section 3.1** with an account of animal behaviours that have relevance to motion camouflage and a consideration of the sorts of algorithms employed by these animals. **Section 3.2** looks at how animals sense the spatial information necessary for motion camouflage from their environment. **Section 3.3** focuses on control systems that can transform such sensory input into appropriate motor responses. This section proposes underlying models for motion camouflage control systems, relating to the algorithms suggested in **section 2.2**, before discussing the neurophysiological circuitry of a biological sensorimotor control system. An account of the motor system is also provided. Finally **section 3.4** provides an overview of the field of artificial intelligence giving precedence to computational algorithms inspired by neurophysiology. This section also covers the selection of a neural approach to model motion camouflage.

3.1 Naturally occurring behaviour relevant to motion camouflage

Although, to the author's knowledge, there is no definitive proof that motion camouflage exists in nature, there are examples of behaviour that could be motion camouflage and examples of behaviour dependent on similar knowledge. The section concentrates upon insect behaviour. It is of some significance that the examples are taken from the insects, which, in comparison with mammals tend to have much smaller and less complex nervous systems.

3.1.1 Tracking and interception in hoverflies

The behaviour exhibited by male hoverflies (*Syritta pipiens*) tracking females recorded and described by Collett & Land (1975) is a possible example of motion camouflage. This tracking behaviour which is thought to have a sexual function is characterised by the male hoverfly:

- maintaining a constant distance from the target;
- continually adjusting its orientation to face the target;
- sometimes supplementing this turn with sideways velocity.

Srinivasan & Davey (1995) re-examined the same tracking trajectories to discover that some closely resembled those that would be expected of a shadower performing motion camouflage. However, in cases the shadower appeared to abruptly switch the position of the fixed point to either a nearby location or to a point at infinity. For this reason they were unable to conclude whether the male hoverflies were employing motion camouflage, or whether the apparent shadowing was an artefact of another behaviour. Regardless, the tracking fly can be seen to be employing an algorithm that is similar to the responsive algorithm (**section 2.2.3**). What is uncertain is whether there is any use of a fixed point. **Section 3.1.2** shows that the calculations necessary to estimate the position of a fixed point are not beyond other insects. Firstly, with respect to the predictive algorithm, is there any evidence that males can predict female movement?

Collett & Land (1978) show that male hoverflies are indeed able to make accurate predictions of female movement and use these to compute an interception course. What is interesting is that the majority of the male's interception trajectory appears to be determined by the movement of the female within the first few milliseconds of the chase. Once the male has made the prediction, it keeps to the same course irrespective of the forthcoming movements of the female (i.e. an open loop response, see **section 3.3.1**). Collett & Land (1978) suggest that the mechanism used to predict the female's path is dependent on a number of assumptions made about the image motion of the female projected on to the male's eye at the moment of first sighting. They explain that if the male assumes that a female has a constant size, it will be spotted at a constant distance. Then if it is further assumed that females travel at a constant speed, the image velocity at first sight will give the line along which the female is moving. Unless the female unexpectedly changes course the male's attack routine will be successful. As might be expected amorous male hoverflies may find themselves chasing wholly inappropriate targets resembling females at distance (including peas fired from pea shooters in the experiments of Collett & Land). However as long as the strategy works occasionally on a female, this need not be of great consequence.

3.1.2 Homing by path integration

The behaviour demonstrated by hoverflies is ample at least to show that there is the potential for existence of motion camouflage in nature. However, evidence that hoverflies utilise a fixed point is lacking. As the fixed point need not be visible (i.e. either because it is not a visible landmark (**section 2.2.1**) or because the shadower does not have all round vision) the shadower is likely to have to be capable of dead reckoning to determine its position. Many animals demonstrate the ability to precisely locate points or areas of interest that they can not directly sense. This is shown in homing behaviour.

A variety of different classification schemes for categorising homing behaviour exist based upon the orientation mechanisms, genetic information and individual experience of the animal in question (Papi 1992). Of interest here is path integration. Path integration enables an animal to navigate directly back to its starting point despite a complex outbound journey. This is achieved by the animal continuously integrating all directions taken and distances travelled into a vector describing the starting position. Such skills could be employed to determine the position of the fixed point.

Path integration may be divided into two groups, allothetic and idiothetic, according to the mechanism of orientation used. Allothetic path integrators depend upon external reference points, whereas idiothetic integrators use only internally generated signals. In practice animals tend to use a combination of both (Papi 1992).

The predominant method of navigation in bees and ants is allothetic, based on a sky light compass (although land mark cues may be helpful, they are not essential) (Wehner 1996). The Saharan desert ant *Cataglyphis* is able to return in a straight line to its nest in a featureless landscape from distances in the range of 200m. In order to do this, the ant requires angular and distance information for each part of the outward journey. Direction is retrieved from an internal celestial compass that takes into account the position of the sun, spectral and in particular polarised light patterns in the sky. It is thought that the ant compares the sky to a fixed neural template residing in its retina and uses the difference between the two to determine the direction. Distance comes from measuring the image flow on the eye over time (see also **section 3.2.2**). What is surprising is the approximate method used to compute the homeward vector. Mathematically, the simple solution would be to continuously calculate a vector summation (having reduced movements to vectors defined in a Cartesian co-ordinate system). *Cataglyphis* does not do this. Muller & Wehner (1988) show that if *Cataglyphis* is forced to follow an outbound path through narrow channels, that involves a single sharp backward angle, it will trace an incorrect return journey. The degree of error is dependent on the size of the angle. This error would not result from a vector summation technique. Muller & Wehner (1988) go on to propose a non-trigonometric model that accurately describes the behaviour. This is based upon a calculation of the arithmetic mean of the angles steered weighted by the distance covered in the relevant direction. Computation is performed continuously and does not require explicit memory of all previous legs of the journey. But, surely the error is a cause for concern for the ant? This is not the case. Behaving naturally *Cataglyphis* rarely makes the sharp backward turns that cause it to offset. Moreover, even if it does the ant tends to turn equal amounts in both directions cancelling out errors. So, under normal circumstances using this approximate system the ant can navigate perfectly well.

An example of idiothetic path integration is provided by the wandering spider *Cupiennius*. The spider can re-locate its retreat after night-time excursions lasting several hours and also find prey that it has been forced to abandon. With respect to the latter ability Seyfarth *et al.* (1982) found through experimentally eliminating the different senses that the spider could navigate back to the position where it had left its dead prey using only mechanical stress sensors located on its legs (lyriform organs). Although the method that the spider uses to process this information is unknown, it makes judgmental errors in similar circumstances to those described above for *Cataglyphis*, that are in accord with the Muller & Wehner model.

3.1.3 How fielders determine where to run to catch a ball

A final area identified to be of interest to motion camouflage is the field of research concerned with explanation of the mechanism that allows humans to determine the correct path to catch a ball. This task differs from motion camouflage in two ways. Firstly that the trajectory of a ball is likely to be less variable than that of the prey in motion camouflage (unlike the prey, once launched the ball has no control over its own trajectory). Secondly, that the path of a motion camouflage shadower is more constrained than that of a fielder who is not concerned with camouflaging motion. Nonetheless, fielding is of interest as it is an instance of a closed loop control system employed for the purpose of interception (control systems are discussed in **section 3.3.1**).

The first theory proposed for how fielders choose a path to intercept the ball was originated by Chapman (1968). The theory makes the simplifying assumption that the ball is hit directly toward the fielder. It postulates that the fielder chooses their path so as to cancel the optical acceleration of the ball. In explanation, once the ball has been hit, as long as the fielder runs so that the angle of elevation of the ball (α) is within 0 to $\pi/2$ rad they will catch the ball (McLeod, Reed & Dienes 2001). Note that if $\alpha > \pi/2$ rad the ball will have travelled over the fielder's head. If the fielder runs forwards or backwards at a rate such that the vertical optical velocity ($\tan\alpha$) is held constant, α will remain in this range (i.e. if the fielder runs to far in, the rate of change of $\tan\alpha$ will increase and the fielder will have to slow down and *vice versa*). This is equivalent to running so as to cancel optical velocity acceleration, hence the theory has been named Optical Acceleration Cancellation (OAC).

OAC does not cover the case where the ball is hit to either side of the fielder. Also it has been argued that humans are not capable of measuring acceleration to the precision that would be required for OAC to be a viable technique (McBeath, Shaffer & Kaiser 1995). Addressing these issues, McBeath, Shaffer & Kaiser (1995) proposed the Linear Optical Trajectory (LOT) model. The basis of the LOT model is that the temporal problem of OAC is transformed into a spatial problem. The LOT model conjectures that the fielder moves in such a way that the optical trajectory (i.e. the path of the ball projected on the fielder's retina as the fielder runs) of the ball is linear. Restating the problem, if β is the azimuth of the ball (i.e. the angle subtended at the fielder by the balls launch point and the vertical projection of the ball on the ground) and ψ is the angle between the linear optical trajectory and the horizontal, the fielder would choose their path so that α and β continuously increase to keep ψ constant (see **Fig 3.1**). This means that the fielder has move to control tanα/tanβ so that tanψ and therefore ψ is constant. McBeath, Shaffer & Kaiser (1995) backed up their theory with two pieces of empirical evidence. Firstly that fielders' interception paths tend to be curved (as predicted by the LOT model). Secondly, footage taken from head mounted video cameras indicated that the optic trajectory of the ball was approximately linear. However, later testing of the LOT model undertaken by McLeod, Reed & Dienes (2001) showed that when the ball was hit on trajectories outside the range of those studied by McBeath, Shaffer & Kaiser (1995), in that α and β were greater, the paths adopted by fielders would have resulted in a non-linear optic trajectory. All results were however consistent with the fielders using the OAC model.

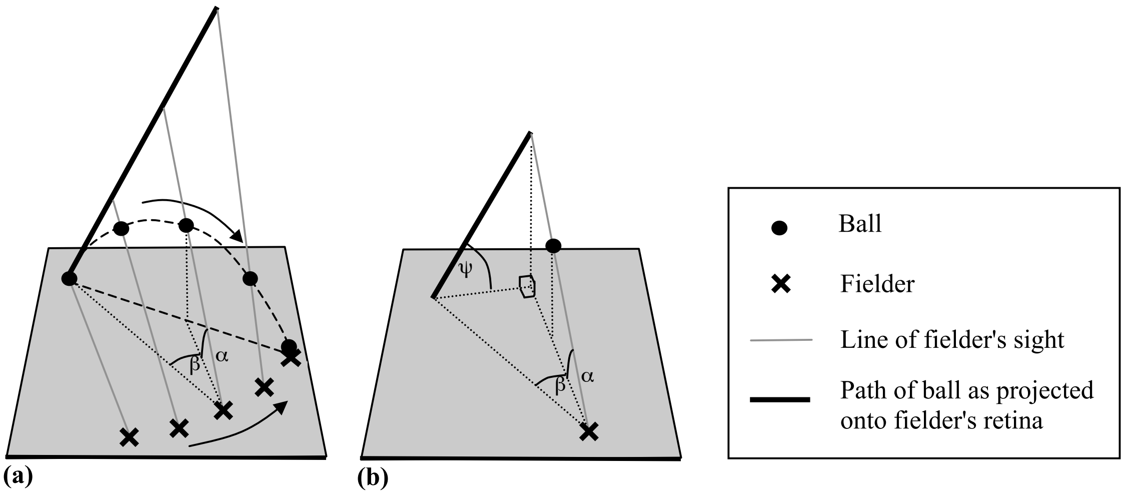


Fig. 3.1 (a) Path taken by baseball fielder to catch ball demonstrating the LOT model. **(b)** Terminology.

Marken (2001) proposes that an extension of the OAC model, Optical Velocity Control (OVC) is sufficient to explain fielder's behaviour. Marken views the fielder as controlling two separate perceptual variables (i.e. consisting of two control systems). The first is the perception of vertical optical velocity (tanα). Similar to OAC, the OVC model acts to keep the optical velocity constant. The addition of OVC is that a critical value at which the velocity should be maintained is suggested. The second system controls the lateral optical displacement (the degree to which the ball is non central in the fielder's view). Effectively, this control system acts to ensure that the fielder always faces the ball. In a computational simulation Marken (2001) demonstrates that his technique is sufficient to replicate the paths adopted by fielders in the empirical tests of McBeath, Shaffer & Kaiser (1995) and McLeod, Reed & Dienes (2001). This model does not however appear to address McBeath, Shaffer & Kaisers' (1995) initial concern that humans may not have the sufficient apparatus to detect acceleration sufficient for calculation of the approach path. As such, whether or not it will be universally accepted as a 'unified fielder theory' remains to be seen. Regardless, the explanation for the fielder's path choice is likely to lie with a control system similar to those described in the above examples. The next section looks at the sensory systems that could be used to obtain the input information necessary for motion camouflage.

3.2 Sensory systems

3.2.1 Senses that could supply motion camouflage input

In **section 2.2** each of the motion camouflage algorithms required the shadower to have knowledge of the direction and distance of objects in its environment. In the previous section, both hoverfly and ant gained this information via their visual system. However vision is not the only sense that may be employed to sample the spatial structure of the surroundings. The most notable alternatives are hearing and electromagnetic senses.

The barn owl (*Tyto alba*) is capable of catching prey in total darkness using hearing alone. It achieves this though comparing the difference in intensity, spectral quality and time of arrival of the sound reaching its two asymmetrically positioned ears (Taylor 1994). Perhaps a more famous example of the use of hearing to estimate distance is echolocation. Bats (*Chiroptera*) emit brief high frequency sound pulses. By measuring the time it takes the echo from these pulses to return they can accurately determine the distance to the object emanating the echo. Through various aural specialisations bats are able to gain an accurate understanding of their spatial surroundings. Alternatively, dogfish (*Scyliorhinus*) use local distortions in the earth's electric field to detect buried prey (McFarland 1999).

Despite these other senses this section shall concentrate on the visual system. There are several reasons for this. Firstly, as has been seen, all the external information necessary for motion camouflage can be obtained though vision. Assuming that there is adequate light, vision tends to be the most accurate source of spatial information available to an animal (Wehner 1997) especially concerning distant objects. Supporting this, few fast moving animals do not depend upon vision and most of these operate in low light (notable exceptions are found with deep sea fish and web spinning spiders). On the other hand, this is not to say that motion camouflage could not be performed using hearing alone, or that hearing could not supplement vision. Furthermore it does not exclude motion camouflage from having some use in disguising the shadower from a prey whose primary spatial sense is hearing. For instance, with echolocation, the accuracy of the estimate of target distance decreases with target range (as the signal deteriorates). So similar to vision, within a certain range the prey may not be able to sense a change in distance of the shadower.

The remainder of this section shall be divided between discussion of how visual systems are able to determine the direction and distance of objects. There are two (known) types of eyes that have the potential to provide the information necessary for motion camouflage. These are the single lensed eye characteristic of mammals, and the compound eye characteristic of insects. The single lensed eye uses a convex lens to focus light on a concave map of photoreceptive cells (the retina). A compound eye is formed from an array of *ommatidia*. Each *ommatidium* contains a lens and a light sensitive *rhabdom*. Unlike single lensed eyes, compound eyes are stationary and virtually fixed focus. Therefore compared with the single lensed eye they have poor resolution. Their advantage is that they have a much wider field of vision, so much so that many insects including hoverflies are able to simultaneously view the majority of their close surroundings. The two eye types use different methods to process information. Coverage is given of both. For further information on compound and single lensed eyes see Bruce, Green & Georgeson (1996).

3.2.2 Estimating an object's direction from vision

The spatial organisation of either the photoreceptive cells in the retina or the *ommatidia* of the compound eye is such that each cell will respond to light arriving from a small set of spatial angles. So assuming that an object can be successfully segmented from its surroundings, determining its direction is trivial. However, in many cases what is important is not the relative direction of an object from the eye, but the relative direction of the object from the front of the body. Therefore if the eyes and head are not stationary relative to the body, then visual information must be combined with other information concerning, for example, the position of the eye relative to the head and the position of the head relative to the body. In humans this information is derived from commands sent to eye muscles and information received from the neck and the inner ear (Carpenter 1996).

3.2.3 Estimating an object's distance from vision

Retrieving depth from the essentially two-dimensional images received by the eyes is significantly more complex than determining object direction. Evolution has resulted in a variety of different strategies. Firstly, binocular depth cues are considered.

When focussed on a point, the degree of convergence of the eyes gives an indication of the distance to that particular point. This convergence measure has to be obtained from the ocular muscles and is both imprecise (Carpenter 1996) and as it stands has relevance only to the point in focus. Information regarding the distance to a second object, relative to the present point of fixation is given by the muscular effort required to bring that object into focus. However, as the level of accommodation necessary to keep an object that is displaced in focus decreases rapidly with object distance, the technique can only be thought of as useful for providing distance information about nearby objects.

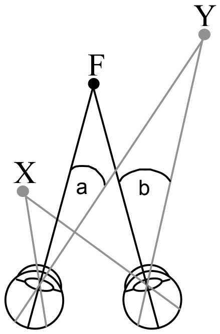


Fig. 3.2. Binocular disparity in single lensed eyes. Both eyes are focussed on point F. Points X and Y are disparate. The angular disparity of point Y is given by the difference between angles a and b. Through knowing the sign and magnitude of angles a and b, the degree of convergence of the eyes on the point of focus, and the distance between the eyes it is possible to calculate the absolute distance of point Y. Note that the degree of convergence and distance between the eyes gives the distance to F.

Although the degree of blurring may be used to ascertain depth over the whole image (Pentland 1987 referenced by Bruce, Green & Georgeson 1996), more accurate information can be extracted from binocular disparity (binocular stereopsis). Each eye has a slightly different view of the world. This means that if both eyes are focused on a particular point, the image of that point will be projected onto the same position on each retina. Conversely, the image of any other point will be projected onto different positions on each retina. Of importance in calculating distance, is the angular difference (more specifically, the distance along the circumference of the retina) between the projection of focused and unfocused points on each retina. Through knowing the magnitude and sign of both differences, it is possible to calculate the distance of the unfocused point relative to the focused (see **Fig 3.2**). In order to be able to calculate absolute distances accurate information regarding the convergence of the focused eyes and the relative positions of the eyes would be necessary. Again, binocular stereopsis has difficulty judging the difference in depth between far away objects as the angular difference between projections in each eye becomes increasingly small. Its usefulness is also dependent on the distance separating eyes and the ability of the brain to correctly match images of corresponding points in both eyes (see Bruce, Green & Georgeson (1996) for more information on the correspondence problem).

Binocular vision is not essential for measuring depth. There is conclusive evidence that it is used by members of only one insect order, the praying mantids (*Mantodea*) (Srinivasan *et al.* 1999)[*] and their predominant methods of sensing absolute depth are probably monocular. Certainly in humans it is likely that comparisons between the visual size of well known objects (e.g. the hand) and distant objects are highly instrumental in this respect. Other monocular cues include perspective, image overlap and importantly image motion.

To humans perhaps the most obvious use of image motion is motion parallax. During head movement the image of nearby stationary objects can be observed to move across the retina at a greater speed than those further away. Bees use this to their advantage to expose potential predators approaching against camouflaged backgrounds (Srinivasan *et al.* 1999). Furthermore image motion can be used to obtain accurate measures of absolute distance. Praying mantids use side to side head movements (peering) to generate retinal image motion that allows estimation of the distance of stationary targets. Kral (1988) suggests that it is speed of image motion rather than amplitude which is important to mantids. As the speed and amplitude of head motion vary during peering both may have be taken into account in the calculation. For instance target range may be derived in terms of the ratio of peering speed to image speed. Sufficed to say that the exact neural mechanism responsible has yet to be explained. An alternative use of image motion is in the estimation of distance travelled. Both bees (Srinivasan *et al.*, 1999) and *Cataglyphis* (Wehner, *et al.*, 1996) measure distance travelled by integrating optic flow over time. Srinivasan *et al.* report that through artificially varying the speed of optic flow it is possible to prompt bees to over and underestimate the distance to their destination. Finally, flies regulate their flight speed by maintaining the speed of the image motion of their environment at a constant level. This

[*] As compound eyes are immobile and fixed focus, accommodation and convergence are not possible. Regarding binocular stereopsis, even if insects are capable of the required neural processing their eyes are too close together and of such low spatial acuity for it to be of much use.

means that an insect may react to the optical illusion of high speed flight by moving backwards (Srinivasan *et al*. 1999).

3.3 Sensorimotor control systems

Having considered the sensory mechanisms that may be used to obtain the information required by motion camouflage it is now necessary to look at control systems that may be used to transform this into motor signals. Use of control systems has been traced back to 300 BC by Lewis (1992) with the Greeks attempts to measure time using water clocks. These water clocks used a float regulator to maintain the water in a tank at a constant level (i.e. a similar function to the ballcock in modern toilets). Maintenance of water at a constant depth ensured that water would flow out of the tank at a constant rate. This allowed time to be measured by the volume of water that had flowed from the tank. The introduction of a mechanical control system, Watt's flyball governor, arguably marked the onset of the industrial revolution. Watt's governor allowed automatic operation of steam engines that previously had to be regulated by hand. Watt's governor consisted of two weights attached by pivots to a rotating pole. As the speed of the engine (and hence rotation of the pole) increases, the weights are swung outwards and upwards by centrifugal force operating a valve releasing steam that acts to slow down the engine. Thus the engine is maintained at a constant speed.

This section commences with a brief introduction to the study of control systems, before continuing the discussion in light of motion camouflage. The mechanism of a biological sensorimotor control system, the vestibulo-ocular-reflex is examined and the section is concluded with an account of the motor system.

3.3.1 Control systems background

A control system will typically consist of an interconnection of components that provide a desired function (Stefani *et al*. 1994). A system is classically considered to contain a controller and a plant. The plant receives input from the controller that is intended to modify it so as to produce a desired outcome (or output). The plant may also receive external input signals that could disturb its behaviour with respect to this desired outcome. So, a successful controller must be able to operate regardless of the environmental situation.

Conceptually the most simple control system is the *open loop* or *ballistic* system (**Fig 3.3a**). Here the plant output has no influence upon subsequent inputs to the controller. A commonly used example that illustrates the basic concepts of control systems is the central heating system. A ballistic central heating system would pay no regard to the household temperature. Once set at a certain level it would continue to put the same effort into heating until externally instructed to do otherwise (or running out of power). As the purpose of the central heating system is to maintain a constant temperature, this is undesirable. It will not be able to compensate for diurnal or seasonal fluctuations in temperature. Thus the performance of an open loop control system can be regarded to be vulnerable to unpredicted events (noise). How could it be improved so that it can operate in a noisy environment? Keeping to a strictly ballistic system, a possible improvement could be made by giving the controller access to information concerning the environmental noise beforehand. This system would be termed a *feed forward* system (**Fig 3.3b**). The controller of the central heating system could have access to a list of previous temperature readings for different times of the year and base heating effort on expected values. However, a far better solution to the problem would be to give the controller a temperature sensor. Then, when the temperature is below the desired level the heating is switched on and *vice versa* (i.e. a thermostat). This is the basis of the *closed loop* or *feedback* system (**Fig 3.3c**). In feedback some or all of the plant outputs

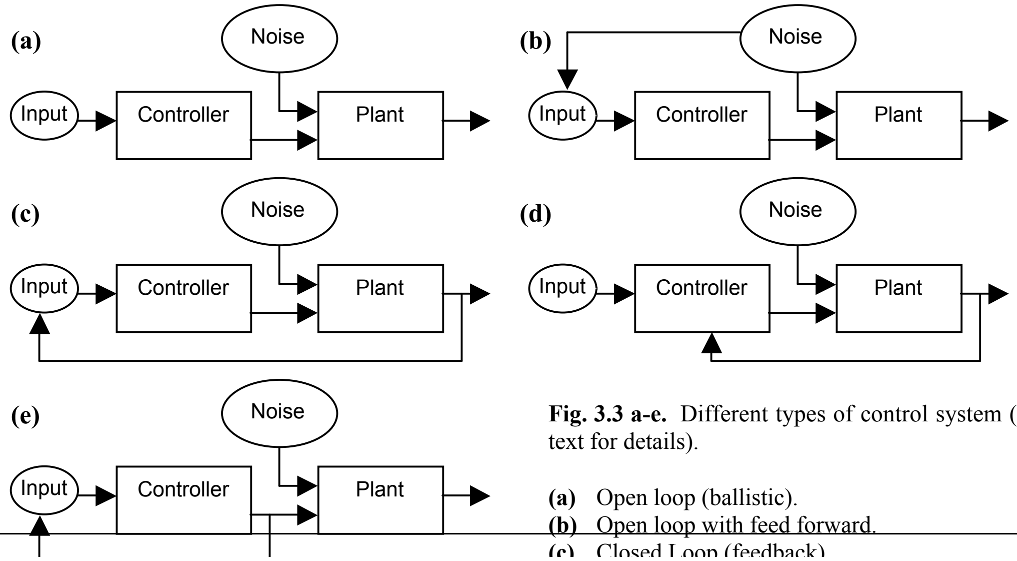


Fig. 3.3 a-e. Different types of control system (see text for details).

(a) Open loop (ballistic).
(b) Open loop with feed forward.
(c) Closed Loop (feedback)

are measured and used by the controller. The thermostat is illustrative of negative feedback. Here the actual output is subtracted from the desired output to give an error signal. The controller uses the negative of this error to form the input to the plant. Notice that the controller is not concerned with what is causing the error (unlike feedforward), just in how to rectify it. With particular relevance to neurophysiology Carpenter (1988) identifies two other types of feedback, *parametric* and *internal feedback*. Parametric feedback (**Fig 3.3d**) defines feedback used to alter parameters in the controller. Parametric feedback could be used to re-calibrate the controller of a faulty central heating system that maintained temperatures five degrees in excess of the desired value. Internal feedback (**Fig 3.3e**) on the other hand involves a prediction of the actual plant output being fed back to the controller. Returning to our central heating system, this would be of use if the temperature sensor produces temperature readings less frequently than the controller passes instructions to the plant. Through a prediction of the actual temperature, the controller may be able to pass more valuable instructions to the plant than otherwise. The accuracy of the model making the predictions could be monitored by parametric feedback.

Sensor readings are prone to noise and inaccuracy. It is usually desirable to attempt to remove such noise through filtering. For instance, filtering is successfully employed to remove noise from radio communications signals whilst retaining the important information. A particularly important development in filtering was the proposal of the Kalman filter (Kalman 1960). Although this thesis does not make use of Kalman filters, a high level overview of the underlying principles is provided due to their importance to engineering applications (for further information see Maybeck 1979). The Kalman filter may be thought of as an optimal recursive data processing algorithm (Maybeck 1979). The filter is optimal in the sense that it provides the best statistical estimate of the variable in question. Criteria for optimality could be that the estimate made is on average correct, alternatively it may be more important to ensure that the maximum error is reduced to a minimum. A Kalman filter incorporates all of the information available to obtain the most likely estimate of a system variable. Note also that the filter does not require storage of all data that it has processed at previous time steps. All information required on all previous states is provided by feedback of the estimate made at the previous time step.

At its most simple level, the filter bases its estimate on a function of the previous estimate that it made and any new information currently available. The importance of each source of information to the estimate is weighted according to the confidence in its accuracy. For instance each source of information will in practice have an associated level of noise. The Kalman filter assumes this to be Gaussian white noise (noise drawn from a Gaussian distribution that is uncorrelated through time). The confidence of an information source may be represented in terms of the standard deviation in error for that information source (note that a Gaussian distribution is defined in terms of the mean and standard deviation). This formulation allows a maximum likelihood estimate of the value of the system variable and the confidence associated with it (expressed as the standard deviation of the noise). The original confidence estimates must be measured in advance and supplied to the filter. The filter's estimate of the system variable (which is statistically the best estimate based on all information previously seen) is fed-back to form part of the input at the next time step. Needless to say, although the Kalman filter provides maximum likelihood estimates, the accuracy is dependent upon the reliability of the information (e.g. confidence values) given to the filter.

As an example the central heating system used in the previous example could conceivably employ a Kalman filter as follows. Firstly, assume that the heating system has two thermometers, one of which is known to be more accurate than the other (and that the standard deviation in the noise of the thermometer readings has previously been measured. Secondly, that the heating system controller has just been switched on (and has no previous estimate or prior knowledge of the current temperature). Upon this information alone, the most likely estimate of the temperature would lie in between the readings of the two thermometers (closer to the more accurate thermometer, with the weighting made according to the relative confidences in the readings of the two thermometers). The confidence in the estimate of the temperature would be calculated according to the confidence in each temperature reading. Having taken into account both readings the filter's temperature estimate is likely to be more accurate than either of the individual thermometers (hence the standard deviation in the noise of the filter's temperature estimate is less than that for either of the thermometers). Assume that on the basis of the temperature estimate made by the filter, the heater's control system decides that it is too cold and switches the heating on. Assume also that the rate of change of the temperature with heating is constant and (as an extreme example) that the next time the temperature is measured the most accurate thermometer has broken. The knowledge that the Kalman filter has at its disposal is its previous estimate of the temperature (with associated confidence calculated from the combined confidence in thermometer accuracies), the reading of the less accurate thermometer and the knowledge that the current heat is a linear function of the previous heat. This information would be combined in the manner previously described for the new temperature estimate.

3.3.2 Possible motion camouflage control systems

Adopting the same terminology as the previous sub-section, let the shadower be the plant of the motion camouflage control system. The desired behaviour of the shadower is that it move so as to consistently be positioned on the camouflage constraint line. Necessary external information is obtained through senses such as the visual system. The regulation of movement is the job of the controller. Conceptually there are a variety of different control system designs that could satisfy the requirements for motion camouflage.

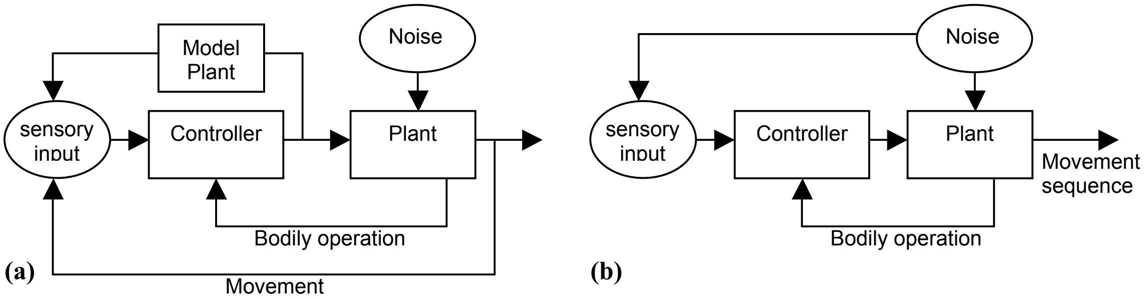


Fig. 3.4. a, b. Abstract representation of two possible motion camouflage control systems.

(a) Closed loop with parametric and internal feedback. Parametric feedback is used to report whether the body is functioning as expected. Internal feedback, to make a prediction of state when sensory input is not available. This system could be applicable to either the responsive or the predictive algorithm (assuming that sensory input is monitored frequently). **(b)** Open loop with parametric feedback and feedforward relevant to the predictive algorithm. The system bases a sequence of output movements upon a single input and sample of environmental noise. Whilst undertaking the movement sequence, sensory input is ignored.

The responsive algorithm, (that requires a rapid reaction to prey movement) is a closed loop system: the output of the plant, movement, affects the controller input (e.g. optic flow). Being thorough, the system would benefit from parametric feedback to ensure that the operation of the body is as expected (for instance, the parameters in the controller may have to compensate for a damaged wing or cold muscles). Conceivably the system could be further improved through use of internal feedback. This would be advantageous if muscular movements can be made more rapidly than sensory inputs can be processed. For example if a sensory input is received once every three movements, two out of three of the movements could be prompted by a prediction of the result of the system output. A diagram of the responsive control system is depicted in **Fig 3.4a** incorporating both parametric and internal feedback.

Depending upon the extent to which the prey trajectory may be predicted from a given amount of sensory information, the control system for the predictive algorithm could be either open or closed loop. If the prey movement is erratic requiring frequent monitoring then the control system would need to be closed loop. In this case the control system design would be similar to that of the responsive algorithm. The difference being that the predictive controller would need to be more complex, taking into account the recent history of the pursuit (see **section 2.2.4**). Alternatively, an open loop controller would be appropriate if a substantial portion of the prey trajectory can be reliably predicted from a small sample of sensory information. This is the same principle as the hoverfly of **section 3.1.1** computing the interception path according to image motion of the target at the moment of first sighting. Here a whole sequence of target movements (defining the shadower trajectory) would be based upon a single sensory input. In this case, the system could possibly improve performance through use of feedforward to sample environmental noise. For instance, returning to the hoverfly example, knowing the speed and direction of wind could enhance the accuracy of prediction. It is suggested that feedforward would be of lesser use in the closed loop systems due to the frequent sensory sampling implicit in such a system. The predictive open loop system with feedforward is shown in **Fig 3.4b.**

The optimal control system for motion camouflage will to a large extent depend upon prey motion and capabilities (see **chapter 2**) but also upon the level of importance of remaining camouflaged. If it were possible for a shadower to predict the prey's path in advance, then an open loop system would be suitable for calculating the shadower's entire approach route. However in the more likely event that it is not possible to predict substantial portions of the prey's trajectory, then the control system would need to closed. All that remains is to look at the operation of biological controllers.

3.3.3 An introduction to the workings of biological sensorimotor controllers

A biological central nervous system characteristically comprises an immense number of massively interconnected analogue processing units (the nerve cells known as neurons). Putting this into perspective the human brain is estimated to contain approximately 10^{10} neurons, with each connected to as many as 10^4 others. Calculation of complex functions is distributed over a set of neurons where processing is undertaken simultaneously. This may be contrasted to a conventional digital computer which uses a single unit to process instructions sequentially (although parallel computers may use two or more processing units, the number is many orders of magnitude less than a brain). Even though an individual artificial unit may process in excess of a million times more instructions per second than any neuron, the computational power and efficiency of a (sufficiently sized) set of neurons operating in parallel can be vastly greater. However care should be taken when interpreting such comparisons to acknowledge that the operation of an artificial processing unit and a computer is very different to that of a neuron and the brain.

The operation of a single neuron is itself quite complex. Although within and between species there are many different designs of neuron, the general properties of each are similar. In the interests of simplicity the discussion that follows shall concentrate on the flow of information in and between neurons and avoid the chemical processes involved. For further information in this area the reader is directed toward a standard neurophysiology text such as Carpenter (1996).

A standard neuron consists of a set of dendrites, a cell body (soma) and an axon. Inputs are received along the dendrites and progress to the soma. If the accumulation of input is sufficient an electric pulse (action potential) is fired along the axon. If it is not, the neuron does not output. The amplitude of the electric pulse fired is always the same, irrespective of whether the total input is marginally or hugely in excess of the firing threshold. However the amplitude does vary between neurons. Following firing there is a delay before the neuron is capable of firing again and then a period during which the firing threshold returns to its normal level. In this latter period a greater effort would be required to make the neuron fire. As such, in biological systems it tends to be the frequency of a neuron firing rather than an individual firing event that tends to be of consequence.

The output of the neuron is subsequently propagated along the axon until it reaches a synapse with the dendrite of another neuron. At the synapse the two neurons do not touch. The incoming signal is transferred from axon to dendrite by means of chemical neurotransmitters. It is received by receptors on the dendrite. Once activated the receptors can act to increase or reduce the chance of the neuron firing (i.e. the input is either excitatory or inhibitory). The position of synapse relative to the soma influences the contribution made to the state of the neuron. In general, the closer the synapse, the greater the effect. Through changing the modifications of the input signals made at the synapses it is possible to alter the behaviour of a neuron. It is precisely through such changes that learning is thought to occur. For instance, Hebbian learning postulates that the synapses between cells are strengthened when the firing of pre-synaptic cell is followed by firing of the post-synaptic cell.

Structures of neurons often specialise in performing specific tasks. These networks tend to be distributed over expansive areas of the central nervous system and frequently share neurons with other circuits. One such is the relatively simple network that provides the sensorimotor Vestibular-Ocular-Reflex (VOR).

The purpose of the VOR is to stabilise the image on the retina during head movement, by instigating compensatory eye movements. For instance, if the head is moved to the left, the present image will be stabilised by moving the eyes to the right at a similar speed and acceleration. The VOR can be nothing other than an open loop system, the time scale over which it is performed is too small for direct feedback to be returned. There is however parametric feedback. If the output eye movements are inappropriate (e.g. if direction/size altering lenses were to be worn) visual feedback is used to recalibrate the response (Churchland & Sejnowski, 1992).

The VOR involves a three-neuron-arc (Anastasio & Robinson, 1989) - three sets of neurons connected in a chain. The first layer passes signals concerning the present state of head movement from the semi-circular canals in the inner ear to an intermediate set of neurons, the vestibular nuclei. From here the signal is projected to the eye muscles that perform the necessary eye movement. During this time the signal must be transformed. This is necessary because the canal and muscle co-ordinate systems are rotated from each other and are both non-orthogonal (Anastasio & Robinson, 1989). In order to achieve the transformation the information from different canals must be combined. This is apparently achieved at the vestibular nuclei where input from the semi circular canals is distributed across the neurons (Anastasio & Robinson, 1989). The calculation is dispersed over the nuclei where different neurons undertake different parts of the computation. The vestibular nuclei are not exclusively dedicated to performance of the VOR. Some neurons are shared with the smooth pursuit (tracking a moving image with the eye) and saccadic (abrupt high speed eye movements to new areas of interest) subsystems whose signals are also distributed over the vestibular nucleus. The problem that faced neurophysiologists was to explain this organisation.

An attempt to do so was made by creating artificial neural network models of the VOR (Anastasio & Robinson, 1989, Anastasio, 1991) . These models used a learning rule (back-propagation, see also **chapters 5**, **6** and **7**) to train the network to transform simulated input from the semi-circular canals to an appropriate motor response value. This learning rule adapts the artificial synaptic weights from a random starting point toward an error minimum for the response. Although it was not a plausible argument that the biological system had used the same learning rule it was reasoned that a combination of evolution and biological learning processes could have found a similar error minimum. Investigation of the trained artificial networks indicated that they exhibited some similarities to their real counterparts. In particular it was found that some of the artificial vestibular nuclei were dedicated solely to either pursuit or VOR and some were shared. Also resembling the biological network was the simultaneous inhibition and excitation of different motor neurons by the neurons of the vestibular nucleus. These results suggest that althought the biological and artificial networks may learn in different ways, both optimisation processes have a similar outcome.

As has been implied, the operation of biological systems is both very different to human engineering and consequently can be very difficult to understand. However biological systems work and they do so well. Although it is possible to conceive some of the benefits of the neural organisation (for instance distributed processing is robust to damage. Loss of several neurons is inconsequential to the operation of the network), it is very difficult to design such systems and do so without returning to conventional engineering practices. This is especially true when large numbers of neurons are involved. Little is known about how biological systems regulate changes to synaptic weights over large networks, however artificial algorithms such as Backpropagation are available. In this section it has been seen that they are able to reproduce some of the functionality of neural systems.

3.3.4 Control structures in the motor system

Before finishing this chapter with an introduction to artificial intelligence, this section looks briefly at the motor system. With respect to motion camouflage this system would execute the instructions supplied by the control system. This introduction concentrates on the motor system of the human body as an example as this system has been studied extensively. The information covered has been largely derived from

Rosenzweig, Leiman & Breedlove (1996), for more details the reader is encouraged to consult this text.

The motor system is formed from a combination of muscles and movement controllers. Muscles are formed from long thin fibres. Muscle fibres may be categorised as either fast muscle fibres or slow muscle fibres. As the name suggests fast muscle fibres are responsible for performing rapid movements but tend to fatigue rapidly. Slow muscles are less strong but are capable of maintaining contractile forces over long periods. Most muscles are formed from a combination of fast and slow fibres. For instance the extra-ocular muscles (that move the eyes) are formed from fast fibres. Slow muscle fibres are of greater importance to maintaining posture.

The activity of muscles is controlled by motor neurons travelling down through the spinal column. The pattern of firing of motor neurons control the onset, coordination and completion of muscular operation. Motor neurons responsible for operating fast muscles tend to have a wide diameter, so that signals may be propagated rapidly. Slow muscles tend to be controlled by smaller neurons.

Motor neurons (which typically integrate many thousands of signals received at their synapses) connect to muscles at relatively large structures known as neuromuscular junctions. The diminished response from fatigued muscles following excessive use is in part due to a reduction in the effectiveness of these neuromuscular junctions. The ratio of motor neurons to muscular fibres is proportional to the precision required in control of the muscle. For example, motor neurons controlling ocular movements may synapse at as few as three fibres, whereas neurons responsible for leg movements may synapse at many hundred fibres.

Movement coordination is also highly dependent upon information fed back from the muscles, tendons and joints. For example this feedback may describe the current state of the muscle or the position of a limb. Feedback is provided by muscle spindles and Golgi tendon organs. Muscle spindles run parallel to the muscles themselves, whereas Golgi tendon organs are attached at one end to the muscle and at the other to the tendon (see **Fig. 3.5**). The spindles are largely responsible for giving feedback when the muscle is stretched and the Golgi organs responsible when it is contracted (thus detecting when the muscle and tendon are overloaded). The sensitivity of muscle spindles is controlled by motor neurons known as gamma motor neurons. This change of sensitivity is manifested by a change in length (and tension) of the spindle.

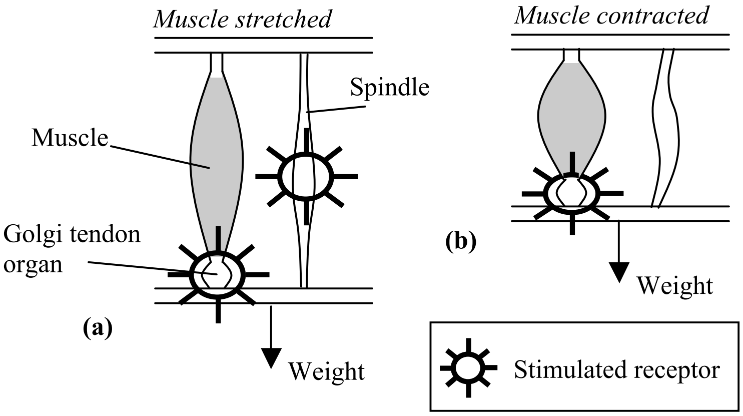


Fig. 3.5. Muscle receptors. **(a)** the muscle is stretched. Both the Golgi tendon organ and the spindle are excited. **(b)** only the Golgi tendon organs are excited. If the muscle were relaxed none of the receptors would be excited.

Motor neurons typically receive many thousands of inputs. Some of these are signals fed back from muscles, the vast majority can be traced back to the cortex and in particular the primary motor cortex. This is thought to be the area responsible for initiating voluntary movements. The primary motor cortex comprised from a topographic map of neurons with different regions responsible for controlling different areas of the body. What is especially interesting is the organisation of cells, in that adjacent areas of the body are controlled by adjacent areas of the map. The area of the motor cortex devoted to each body region is proportional to the precision of movement required for the corresponding bodily area. For example the hands and tongue occupy disproportionately large areas of the cortex.

3.4 Artificial Intelligence

Artificial Intelligence has been defined by Minsky (1968) as "the science of getting machines to do things that would require intelligence if done by men". In fact, this is only one of many definitions of artificial intelligence. Russell & Norvig (1995) list eight alternative definitions that have been proposed, that they organise into four categories according to the aim and measure of success of artificial intelligence. These are concerned with provision of systems that:

- think like humans;
- act like humans;
- think rationally;
- act rationally.

In these definitions the difference in aim is whether or not the system is intended to emulate thought processes or behaviour. The difference in success is whether or not the intelligence of the system is measured against a human or against alternative criteria. The difference between these definitions reflects the different approaches taken towards solving artificial intelligence. In this thesis the division is seen as being between the symbolic approach and the connectionist approach. Before these approaches are introduced the following definition of artificial intelligence provided by Schalkoff (1990) seems to broadly cover the categories mentioned above: "The field of study that seeks to explain and emulate intelligent behaviour in terms of computational processes."

An early theme in AI was to set up a computational model of logic. There had already been a long history of philosophic and mathematical study of logic to act as foundation. This went back to the work of Aristotle (Logic) and was added to by many eminent people in the history of science including Galileo, Descartes and Leibniz (Luger 2001, Russell and Norvig 1995). In the 19th century Boole initiated a major advance by expressing the laws of logic in mathematical form.

Boole's calculus was at the level of propositions, where a proposition is a basic statement that is either true or false. For example 'the integer 2 is prime' is a true proposition. Boole showed how more complex propositions (sentences) could be built up out of basic propositions and symbolic operators (AND, OR and NOT) and how to evaluate these complex propositions as true or false. Certain sentences, for example (P OR (NOT P)) where P is a proposition, are always true irrespective of the truth value of P itself. These are called valid sentences, or tautologies. The calculation of true/false is well known in the field of computer science as Boolean Algebra.

Subsequently Frege (1879) extended the expressiveness of logical sentences by introducing Predicates, which are propositions incorporating variables. Predicates are true or false according to the value of the variable. For example 'x is prime' and 'm is less than n' are predicates. Along with the propositional operators Frege introduced quantifiers (For all) and (There exists). These allowed sentences such as the following to be made: (There exists x) ('x is a natural number' AND 'x is even' AND 'x is prime') which in this case is true because 2 is such a natural number. This logical system is known as the First Order Predicate Calculus. It is possible to have 2nd and higher order logics by introducing variables which range over functions, and over functions of functions and so on. This is out of the scope of this introduction.

The other major early advance of mathematical logic was the introduction of proof or deduction. This identified a subset of sentences as theorems by a purely formal technique (deterministic rules for transforming one sentence into another so that if the first is a theorem so is the other). The whole process was based upon an initial selection of sentences to be axioms (which may be thought of as basic theorems). There were now two ways of assessing sentences, the formal derivation of a theorem and the calculation of whether it was valid following the approach of Boole. If it is possible to select axioms and rules for transforming sentences so that the set of theorems are identical to the set of valid sentences a logical system is said to be complete. Deduction rules can always be represented algorithmically so that from a later AI point of view it became theoretically possible to code up logical reasoning in a program.

By the time of the publication of Principia Mathematica by Whitehead and Russell (1910) the propositional calculus and first order predicate calculus were well established and it was known that the propositional calculus was complete in that the provable sentences were valid and *vice versa*. The result for the predicate calculus was not established until 1931 when Gödel demonstrated the completeness theorem. However if the Predicate Calculus is augmented by enough extra structure to express arithmetic Gödel (1931) also showed in his incompleteness theorem that there are true sentences which cannot be proved.

In principle all true arithmetic statements can not be established algorithmically (likewise for any equally complex domain). This is an example of a non computable problem as investigated by Turing (1936). Turing aimed to establish the limits of what could and could not be computed algorithmically. Turing defined an abstract device, the Turing Machine, which has a finite number of internal states and an unbounded storage medium (the tape) each cell of which can hold one of a finite number of symbols (with no loss of generality they can be 0 and 1 as with typical digital computers) and at any computation step one cell is read/written to. The machine operates according to a totally defined set of rules taking as input the internal state and the symbol read and as output a change in state, a write to the cell and a selection of a new cell. Turing provided an argument, which is generally accepted, that any algorithmic process can be emulated by a Turing machine. As such something is non computable if it can be shown that a Turing machine cannot compute it. Turing's abstract machine predated digital computers as they are recognised today. Turing himself was involved in the early development of digital computers and very early understood the potential for AI. He is famous for proposing the Turing test whereby if an observer is incapable of discriminating between a human and a computer in a typed interaction the machine is judged to have demonstrated intelligence. The advent of attempts to build artificial systems really began after World War II when digital computers had demonstrated their ability to perform the computation necessary for AI investigations. During this period the symbolic approach to artificial intelligence dominated.

3.4.1 The symbolic approach

The symbolic approach to artificial intelligence is effectively concerned with the development of software programs built on a set of logical rules grounded in Boolean algebra. These rules manipulate symbols, so as to logically search to solve a problem. The commercial product of this approach are expert systems. These are systems designed to capture knowledge of experts in the field so as to make decisions about real world problems. Human experts' understanding and practical experience are coded into a set of rules, which, together with the logical rules of inference allow the systems to deduce answers to questions. Expert systems tend to be written for specialist domains and incorporate thousands of rules. A famous example of an expert system is MYCIN (Shortliffe 1976), designed for the diagnosis of infectious blood diseases. The system asks a series of questions in response to which a patient would enter their symptoms. Based on a search guided by a set of rules, MYCIN is able to rate possible conclusions, often giving more accurate diagnoses than human experts. However, despite this success, expert systems may be criticised for being brittle, in that if knowledge of a certain situation has not been provided they are likely to give an incorrect response (Luger 2001). Also conventionally, expert systems are not able to learn from their experiences, so unless the new knowledge is supplied they will fail again in the same situation.

The symbolic approach to artificial intelligence has been seen to explicitly code knowledge as symbols to that are manipulated sequentially by logical algebra. In this respect it is attempting to recreate our perception of our own thought processes. Certainly this is the case with MYCIN, where a task that is already understood has been automated in a manner that is understood. As so much of what is commonly regarded as intelligent behaviour involves the logical manipulation of signals it is reasonable to expect that this approach will always have significance to AI. It is also deserves mention that the automating of logical systems pioneered by AI investigators has been found useful in the field of high integrity computing. Here automated proof tools may be used to see whether a safety related program fulfils its requirements (Rushby 1993).

However, artificial intelligence faces many other problems that are poorly understood and do not necessarily have any direct relationship with human thought. For example interpreting sensory information and acting on that interpretation. It is probably fair to say that many insects' ability to interpret their vision surpasses current artificial technology, even though no insect could pass the Turing test. In such a case where the solution to a problem is not understood, an alternative approach is to model the underlying mechanism responsible for solving the problem. This is the basis of the connectionist approach. Whereas the symbolic approach has its roots in introspection the connectionist approach is inspired by neurophysiology. The connectionist approach is characterised by parallel distributed processing. Calculations are distributed over many different processing units (i.e. neurons) and undertaken simultaneously. Following this approach knowledge is coded implicitly (for example as the synaptic weightings of neurons) rather than explicitly as symbols.

3.4.2 The connectionist approach

The first work widely recognised as artificial intelligence was connectionist. This was the proposal of a computational model of a neuron by McCulloch & Pitts (1943). A McCulloch & Pitts neuron is shown in **Fig 3.6a**. The neuron takes an array inputs that are weighted through multiplication (analogous to the alteration of the incoming signal that takes place at synaptic connections see **section 3.3.3**). If the sum of the weighted inputs (i.e. the inner product of the input vector and the weight vector) exceeds a certain threshold value, the neuron fires (i.e. 1 is output), otherwise it outputs 0. Although simple, this model is capable of modelling basic Boolean operations. But, to do so, the weights had to be set by hand. In 1958, Rosenblatt presented a learning rule for the Perceptron (a McCulloch & Pitts neuron). Learning is achieved by modifying the weights in the following way (the weights are initialised randomly). If the Perceptron output is too small (i.e. 0 is output when the target is 1) a value proportional to the corresponding input is added to each weight. If the output is too large, the value is subtracted This learning rule is guaranteed to teach the Perceptron any problem that it can model in a finite number of steps. The Perceptron learning rule is only applicable to networks with binary output. It was not long before Widrow (1962) proposed the Least

Mean Square algorithm, capable of training Perceptrons with linear output functions (capable of performing linear regression). This algorithm firstly involves calculation of the error of the network with each presentation of input (where the error is represented as the square of the difference between the target output and the actual output). The error is considered as a function of the weights. Each weight is changed in proportion to the negative of the slope of the error with respect to that weight (e.g. if the rate of change is positive, the weight is increased, if it is negative it is decreased). These discoveries were seen as extremely encouraging, until Minsky & Papert (1969) showed that the functions that Perceptrons (with binary or linear outputs) were capable of modelling were extremely limited, to the extent that they could not manage many fundamental classification tasks. For example the Perceptron is not capable of distinguishing between two classes that can not be separated by a straight line. Such problems are termed to be linearly inseparable (see **Fig. 3.7**), the most famous example being the XOR function. As a result interest in the connectionist approach dwindled.

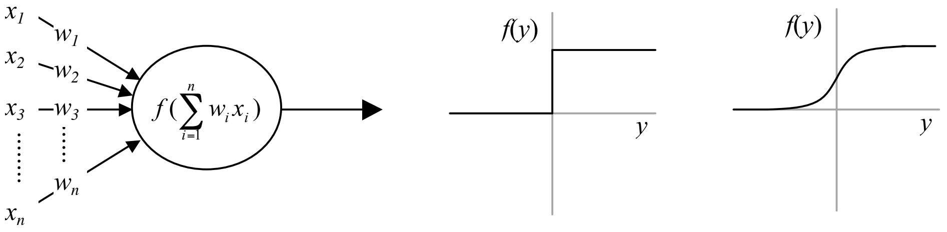


Fig. 3.6. **(a)** A McCulloch and Pitts node. Inputs are represented by x_i, weights by w_i. **(b)** Heaviside activation function (utilised by Rosenblatt's Perceptron. **(c)** Sigmoidal activation function (utilised by Multilayer Perceptron).

Nevertheless, research continued over the following decade or so. Self organising maps, inspired by the topographical arrangement of neurons in the brain were developed (Willshaw & von der Malsburg, 1976, Kohonen 1982) as were the autoassociative memories of Hopfield (1982). Self organising maps (as shall be seen in **section 3.4.6**) are unsupervised classifiers (i.e. they require no teacher) and are appropriate for tasks such as data compression. Hopfield's auto-associative memories were based on studies in physics of the energy dynamics of lattice models. Applications included image restoration (e.g. matching a corrupted image with the intact original). The publication of the Backpropagation algorithm (Rumelhart, Hinton & Williams 1986) is arguably responsible for the resurgence of interest in neural computing. This provides a method to train networks formed from multiple layers of Perceptrons (Multilayer Perceptrons or MLPs). Multilayer Perceptrons act as universal function approximators, capable of performing the non-linear classification tasks upon which the Perceptron had failed.

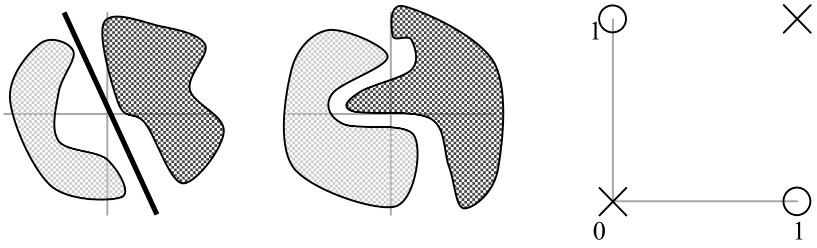
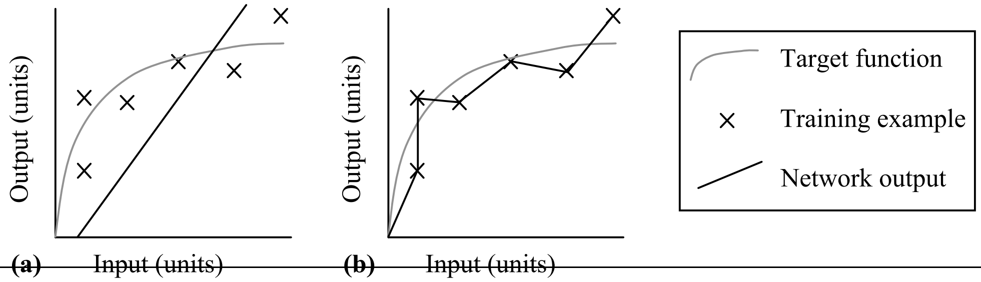


Fig. 3.7. Examples of classification tasks. **(a)** The two classes are linearly separable (the line represents a possible decision boundary). **(b)** The classes are linearly inseparable. **(c)** The XOR problem represented graphically. The two classes X and O can not be separated by one straight line.

Motion camouflage is a problem of sensorimotor integration, that effectively reduces to non-linear regression (in that the motor output is a function of the input, where both input and output are measured on a continuous scale, rather than as discrete classes). Bearing in mind that biological systems have solved such problems successfully motion camouflage lends itself to the connectionist approach. The remainder of this section provides an overview of four of the more successful neural network approaches: Backpropagation, Radial Basis Function networks, Support Vector Machines and Kohonen Self Organising Maps.

The first three approaches are categorised as supervised learning. This is to say that the networks are trained to learn a function from a training set of inputs and target responses (as with the Perceptron and Least Mean Square training algorithm). Before going on to introduce these networks it is helpful to provide a brief introduction to a generic problem posed in attempting to learn the most accurate representation of the

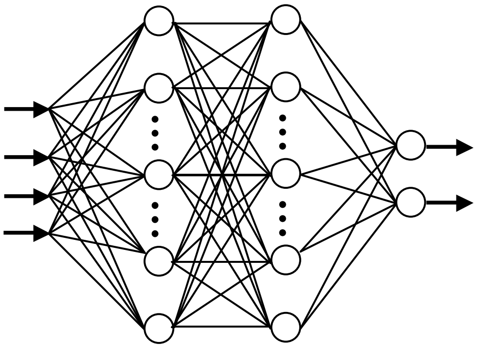


target function.

The objective of training may be seen as to minimise the expected error of the network (i.e. to make the network approximate the target function as closely as possible). As the true target function describing the training set is likely to be unknown, the accuracy of the network is dependent upon the available training data. There may be many different networks (describing different functions) that are capable of modelling the training data accurately. The difficulty is in choosing the network that best matches the actual target function. Choice of an overly complex network (e.g. a network with a large number of nodes) risks over-fitting the training data, i.e. the network may model details of the training data that are artefacts of the training sample chosen (e.g. due to noise or too few training samples). Therefore despite perfectly modelling the training data the network may have a large expected error. Conversely a network that is too simplistic may ignore important details that are representative of the truth. In general the ideal choice for the classification function lies between the two. This trade-off is also known as the bias/variance dilemma. The network error may be seen as originating from two sources, the $(\text{bias})^2$ and the variance (see Bishop (1995) for details), where the bias refers to the average error of the network from the actual target function and the variance is the extent to which the network is sensitive to the data in the training set (i.e. the variance is effectively a confidence interval that is a consequence of the inadequacy of the training data set). As shown in **Fig 3.8a** if the network is too simplistic, it will have a high error and comparatively low variance (i.e. the error is high regardless of the training data set). In **Fig 3.8b**, the error is low, however the variance is high (a different training is likely to produce a markedly different network). Specifically, the problem posed to learning machines, is how to judge when the optimal balance has been struck between bias and variance.

3.4.3 Backpropagation

Backpropagation is an algorithm that may be used to train Multilayer Perceptron networks. Multilayer perceptrons are universal function approximators which may be applied to both classification and regression problems. The technique has been established for almost 20 years and is still used widely. Example applications include: spatio-temporal signature recognition (Everitt & McOwan 2003), visual emotion classification (Anderson & McOwan 2003), speech recognition (Rimer & Martinez 2002), missile guidance (Efe, Kaynak, & Wilamowski 2002), public key cryptography (Yee & DeSilva 2002), prediction of tree growth factors (e.g. trunk circumference) from environmental factors (Jutras *et al.* 2002), predictions of stockmarket growth (Wang, Okunbor & Lin 2002) and recognition of Mongolian characters (Efe, Kaynak, & Wilamowski 2002).



→	Network input/output
—	Connection: (weight * input)
○	Hidden/output node. Sigmoidal function of sum of weighted inputs

Fig. 3.9. Abstract representation of a Multilayer Perceptron with 2 hidden layers. Note that in Backpropagation the connections between nodes are used to propagate error signals back through the network (see **section 5.1.1**).

A Multilayer perceptron (see **Fig 3.9**) is a feed-forward neural network formed from multiple layers of nodes (i.e. the input signal is propagated forward through the network layer by layer, along connections between nodes in neighbouring layers). The first layer of the network corresponds to the vector presented as input. The final or output layer of the network corresponds to the network output. In between there can be any number of hidden layers. MLPs tend to be fully connected (though it is not essential). This means that each node in a hidden layer or the output layer receives a signal from each node in the previous layer. Connections between nodes are weighted. Each node in the hidden layer and output layer acts to compute an activation function of the sum of its weighted inputs. Activation functions tend to be sigmoidal (e.g. **Fig 3.7c**) and are necessarily differentiable to enable training. It is the non-linearity of the activation functions that enables the network to perform non-linear classification tasks.

Backpropagation training is supervised and follows an iterative process. In each training iteration the network is provided with an input and the target response. The purpose of training is to find a combination of weights for which the network error is at an acceptable level. In accordance, at each training iteration the current network error is calculated (represented as a function of the difference between the actual output and the target output). Weights are adjusted in proportion to their estimated contribution to the error (i.e. the rate of change of the error with respect to that weight) and an adaption rate. The error signal is passed back along the connections of the network layer by layer, so giving Backpropagation its name. Although it has been shown that a network with a single hidden layer can model any continuous function (see Haykin (1999)) and a convergence proof for Backpropagation exists (based upon infinitesimally small weight changes and infinite training time (Rumelhart, Hinton & Williams 1986)) training may take many iterations and the network may get trapped in local error minima (e.g. when the error is unacceptably high but the rate of change of the error with respect to the weights is zero).

A particular problem associated with Backpropagation is deciding when to terminate training. For instance continuing training for too long risks overfitting the target function. A variety of methods to combat this problem have been suggested (see Haykin 1999 for further details on all approaches mentioned here). The first technique is cross validation. Here the training data is divided into two sets. The first set is used for training the network. Training is undertaken as normal, however training is periodically interrupted for the network to be tested (with fixed weights) on the second set of data (the validation set). The training process is stopped when the error resulting from the test on the validation set is at an acceptable level (or at a minimum). Hence as the network is being tested on data that has not been used to adapt the weights, it is more likely to demonstrate better generalisation. The problem of overfitting may also be addressed by refining the structure of the network. At the most simple level, if overfitting is suspected, the network complexity could be reduced (e.g. by removing weights) and the network retrained from the beginning. Alternatively, the complexity of the network may be reduced during training by removing or weakening weights. This process is known as network pruning. Criteria proposed for the removal of weights include the magnitude of the weight (the somewhat risky conjecture being that weights with high magnitude are more likely to be of importance to the network and therefore that weights with low magnitude may be removed). Alternatively the Optimal Brain Surgeon technique (Hassibi & Stork 1993) considers the second order derivative of the error surface in order to select weights to eliminate for which the increase in error incurred is minimal.

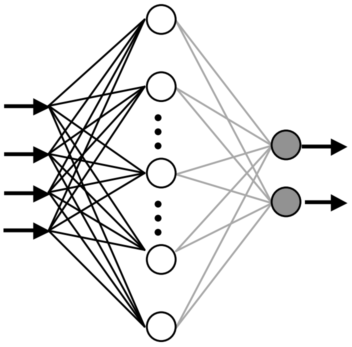
Another drawback of standard Backpropagation is that the training technique is inefficient and it may take many training iterations before the network reaches the desired accuracy (if at all). Many methods have been proposed with the aim of decreasing training time (a detailed coverage is out of the scope of the introduction, for detailed information on the methods discussed below the reader is referred to Bishop (1995)). Heuristic techniques for decreasing training time include the delta bar delta rule and Quickprop. The delta bar delta rule acts to change the adaption rate of a nodes weights throughout training (whereas with standard Backpropagation the adaption rates remain constant). The underlying principle is that if weight changes oscillate between positive and negative values on successive training iterations, the adaption rate should be decreased. If the weight change retains the same sign over successive training iterations it should be increased. Quickprop attempts to approximate the error as a function of each of the weights using a quadratic polynomial. After the coefficients of the polynomial have been estimated in consecutive training steps, the weight is moved to its minimum. In practice both of these techniques encounter a variety of difficulties and require an additional level of handcrafting to standard Backpropagation to improve training efficiency. A more principled approach is that of conjugate gradient descent. At each stage of the algorithm a line search is carried out in the direction of the change to the weight vector. The search is for three points, where the middle point has an error less than the outer points. A quadratic polynomial is fitted to these points and weights changed to lie at the minimum of the polynomial. Using conjugate gradient descent, the next search direction would not be the weight change given by Backpropagation. It would be chosen so that the component of the error gradient parallel to the previous direction remained zero (otherwise the path down the error surface is likely to zigzag). If the successive search direction satisfies this criteria it is said to be conjugate. Variants and extensions of this approach are covered in both Bishop (1995) and Haykin (1999).

3.4.4 Radial Basis Function networks

The second set of networks to be considered are RBF networks. RBF networks were pioneered in the late 1980s in the work of Powell (1987), Broomhead & Lowe (1988) and Moody & Darken (1989). Recent applications of RBF networks include: distinguishing between humans, pets and other objects for surveillance systems (Gutta & Philomin 2002), face recognition (Haddadnia, Ahmadi. & Faez 2002), protein sequence classification (Wang *et al*. 2002) and diagnosis of thyroid disease (Ozyilmaz & Yildirim 2002).

Radial basis function networks take a slightly different approach to function approximation. To model a problem, the RBF network attempts to fit a curve in multidimensional space. RBF networks are non-linear, fully connected and feed-forward. They consist of three layers of nodes, an input layer, a single hidden layer and an output layer (see also **Fig. 3.10**). Each layer of nodes serves a different purpose. The input layer represents the vector presented as input to the network. In contrast to the MLP which calculates the inner product of the input and weights, the RBF network is based on the distance between the two. Each node in the hidden layer calculates a Radial basis function (typically Gaussian) of the distance between the weights connecting that node to the input layer (i.e. the weights define the centre of the node, the function calculated at the node is also referred to as a kernel function). This calculation effectively serves to cast the problem non-linearly into a high dimensional space (the space defined by the hidden layer is typically of higher dimension than the input space). Finally, the nodes of the output layer bear more resemblance to those of a MLP by calculating a linear summation function of their inputs. Haykin (1999) traces the mathematical justification of this non-linear transformation to a high dimensional space followed by a linear transformation to Cover (1965). With respect to pattern classification problems Cover states that a problem mapped into higher dimensional space is more likely to be linearly separable than in low dimensional space.

Training an RBF network is undertaken in two stages. Although the training is ultimately supervised, the first stage is unsupervised. The weights connecting input to hidden layer tend to be self organised (for instance using a clustering algorithm such as k means, expectation maximisation or even training similar to Kohonen networks, see Bishop (1995) for further information). The weights then remain fixed, defining the centroid of the radial basis function for the respective node. Also the widths of the radial basis function (e.g. the standard deviation of Gaussians) may be set using a nearest neighbour technique operating on the distance between each node's centroid and the set of surrounding centroids (e.g. the root mean square distance to the P nearest centroids). Thus the hidden layer is a Gaussian mixture model. The second stage of training uses Least Mean Squares (see **section 3.4.2**). The weights connecting the hidden layer to the output are adjusted in proportion to the contribution of that node to the error of the network. As in Backpropagation the error is a function of the distance between the actual output of the network and the target output.



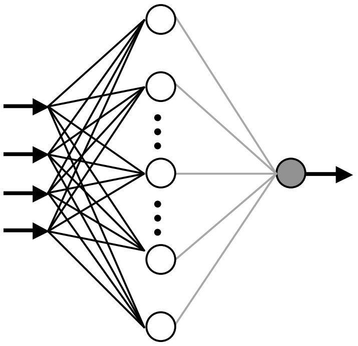
→	Network input/output
—	Connection: (weight - input)^2
○	Hidden node (RBF of sum of input connections)
—	Connection: (hidden node output * weight)
●	Output node, function of dot product of hidden layer outputs and weights

Fig. 3.10. Abstract representation of a Radial Basis Function network. Note that the sum of input connections calculated at the hidden node is a measure of the distance between the RBF centroid (determined by the nodes weights) and the current input.

Both MLPs and RBF networks are universal function approximators and as such there will always be an RBF network capable of performing the same function as an MLP and *vice versa*. However practically, the RBF network training process is likely to be less time consuming than for Backpropagation at the expense that an RBF network is likely to require more parameters for the same modelling accuracy (Haykin 1999). Determining the number of hidden nodes to accurately model a problem requires a degree of handcrafting (especially as clustering techniques are susceptible to finding sub-optimal solutions incorrect initialisation). Such problems may be remedied through use of Support Vector Machines as detailed in the next section.

3.4.5 Support Vector Machines

A third technique for designing feedforward networks that has more recently become popular is the Support Vector approach (Vapnik 1995). Recent applications of support vector machines include face detection (Ng & Gong 2002, Smeraldi & Bigun 2002, Li & Tang 2002), disease classification based on protein sequences (Sheng *et al.* 2002), phoneme classification (Ech-Sherif *et al.* 2002), prediction of airborne pollutants (Lu *et al.* 2002), recognition of abnormal usage of information systems in order to detect intruders (Mukkamala, Guadalupe & Sung 2002), arc welding quality control (Feng *et al.* 2002).



→	Network input/output
—	Connection (input * weight)
○	Hidden node (Kernel function of dot product of inputs and weights)
—	Connection (hidden node output * weight)
●	Output node, function of dot product of hidden layer outputs and weights

Fig. 3.11. Abstract representation of a Support Vector Machine. The weights on the connections between the input and each node in the hidden layer may be considered to represent the coordinates of each support vector (the number of support vectors is the number of hidden nodes).

Seen from a connectionist viewpoint, Support Vector Machines (SVMs) contain three layers of nodes (see also **Fig. 3.11**): An input layer, a single hidden layer of non-linear nodes and a linear output layer. SVMs effectively form a superclass of networks encompassing RBF networks and MLPs with a single hidden layer (with respect to these architectures SVMs provide a different means of training). With a few changes, SVM can be used to construct both pattern classification and non-linear regression networks.

SVMs are grounded on the Structured Risk Minimisation principle which directly addresses the bias variance dilemma (**section 3.4.2**). SRM is introduced as follows. This text is largely referenced from Burges (1998) and Müller *et al*. (2001).

As described previously, the objective of training is to minimise the actual error of the network (i.e. the average error when measured against the target function). The expected risk (actual error) of a network may be estimated in terms of the sum of its empirical error (the average error on the training set) and the Vapnik-Chervonenkis (VC) confidence. This estimate, which approximately quantifies the bias variance trade-off is known as the risk bound. The VC confidence itself is a function of the VC dimension (a measure of the capacity of the network discussed in the following paragraph) and the size of the training data set. The VC confidence increases monotonically with VC dimension.

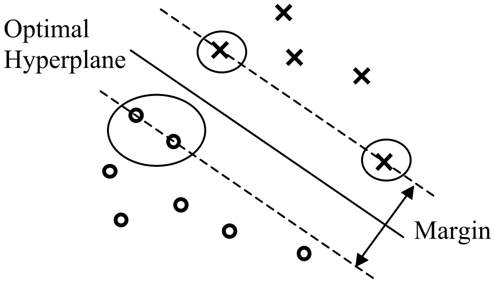


Fig. 3.12. The optimal hyperplane constructed by the support vector machine. The points circled represent the support vectors.

The capacity of a network is its ability to learn any training set without error (Burges 1998). The capacity may be measured in terms of the VC dimension. The VC dimension for a set of classification functions (e.g. a machine with all possible combinations of parameters) is the number of points that can be correctly classified, for all possible labellings of that set of points (if there are n points there are 2^n labellings). For example, a MLP with a large number of hidden nodes would have a high VC dimension and consequently would be more likely to overfit a training data set (see also **Fig 3.8b**). Therefore despite having a low empirical error, the VC confidence term would be high.

In most cases the lowest risk bound will given by a network of intermediate empirical error and VC confidence.

Given a set of networks with different capacities the risk bound, therefore, provides a principled approach to the selection of the network that is most likely to have the best generalisation performance. The risk bound is independent of the actual target function. This principle is applied to the case of obtaining the optimal classification boundary for linearly separable data as follows.

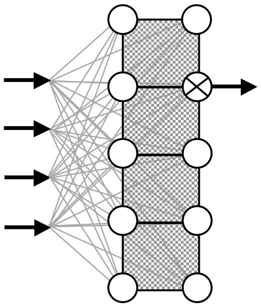
If the data is linearly separable, there may be many linear classification functions (each representing a hyperplane in feature space) that accurately separates the classes. There will only be one that can be considered to separate the data optimally. This plane is referred to as the optimal hyperplane and is most likely to give the best generalisation performance. The optimal hyperplane is positioned at maximum distance from the closest data point of each class. The sum of these two distances is referred to as the margin of separation. Points lying on the boundary of the margin of separation are support vectors. It is these critical elements of the training data alone that determine the optimal hyperplane (see **Fig. 3.12**). If all data points but for the support vectors were removed from the training set, and training repeated, the optimal hyperplane would be the same. The Support Vector Machine searches for the optimal hyperplane by maximising the margin of separation. This search is cast as a quadratic optimisation problem (that has a unique solution) based on the use of Lagrangian multipliers. The quadratic optimisation determines the weights between hidden layer and the output. Non-support vectors are given weights of 0 (i.e. the hidden layer is formed from the support vectors only and hence the number of nodes in the hidden layer are automatically determined). This maximisation process is in fact equivalent to implementing the Structured Risk Minimisation principle so as to minimise the risk bound. The hyperplane is selected so that the empirical risk of the machine is minimal. Secondly the capacity of the machine is reduced (decreasing the VC confidence) by reducing the set of hyperplanes possible (i.e. a hyperplane can not pass within a critical distance of the training data points).

The choice of a linear decision function may appear somewhat limiting. However non-linear problems may be addressed by mapping the data non-linearly into a high dimensional feature space (e.g. linearly inseparable problems may be mapped to a feature space in which they are separable). This would appear to pose two problems. Firstly finding a suitable non-linear mapping. Secondly performing calculations in the feature space may be computationally prohibitive (the feature space could have infinite dimensions). As training depends only upon the calculation of dot products, both problems can be avoided through use of kernel functions. It is possible to perform dot product calculations in this high dimensional space through use of kernel functions, without ever explicitly using or even knowing the mapping. Training follows a similar process to that described above with the mapping being simply carried out by the kernel function. The kernel functions could for instance be radial basis functions (in which case the network would emulate an RBF network), or sigmoidal (then the network would resemble a MLP).

If the SVM is to be applied to regression, a linear regression function is estimated in the feature space that lies within a specified distance of all training points. Training the network again is posed as a quadratic programming problem, in which a cost function is minimised. In training decisions must be made as to the distance from the regression estimate that training points must lie within, or to the number of training points that may lay outside the error boundary (in which case this distance is automatically determined). A problem associated with non-linear SVM is that there is no control over the number of support vectors that the algorithm may choose. With large data sets the algorithm may choose a prohibitively large number leading to inefficient execution (Haykin 1999). Secondly there are still numerical difficulties involved in the quadratic programming problem when the training set is large making computation lengthy and memory intensive (Joachims 1999).

3.4.6 Self organising maps

The final neural network approach to be considered is Kohonen's self organising feature map (SOM). The purpose of a SOM may be seen as to learn a discrete low dimensional map of a high dimensional input space. The SOM is most frequently used for data compression and pattern classification tasks. Applications for SOM's have included: document classification (Bakus, Hussin. & Kamel 2002, Freeman, Yin & Allison 2002), protein classification (Pollock, Lane & Watts 2002), Arabic character recognition (Klassen & Heywood 2002), gesture recognition (Ishikawa & Sasaki 2002) and face classification (Navarette & Ruiz-del-Solar 2002).



→	Network input/output
—	Connection: (weight - input)^2
○	Node (Sum of connections).
×	Node with least sum gives output (e.g. its 2D position on grid)

Fig. 3.13. Abstract representation of a Kohonen self organising map. The weights on the connections between the input and each node in the hidden layer may be considered to represent the co-ordinates in feature space of the nodes. The map has a 2D topology (shaded grey).

The architecture of the SOM is again fully connected and feed-forward, however the network is formed from two layers of nodes (see **Fig. 3.13**). The first layer is the input vector and the second layer is the feature map. The structure of the map is inspired by the topographical arrangement of neurons in the brain (e.g. the cortex) and is typically formed from a one or two dimensional grid of nodes, where each node is associated with a higher dimensional point in the input space. The point in input space corresponding to each node may be considered as being represented by the weights connecting the input to that node. In operation, the SOM will map the current input presented, to the node on the map that most closely resembles it (in terms of the distance weights connecting that node to the input and the actual input).

Ideally, the SOM training process maps similar inputs to similar regions on the topographic map. SOM training is unsupervised, effectively meaning that each input also acts as the target. On presentation of an input, the weights of the node that is closest to the input (in terms of a distance metric such as the Euclidian distance) are adapted to bring them closer to the current input. Also the weights of the nodes neighbouring (in terms of their position on the feature map) the closest node, will have their weights adjusted in proportion to their proximity (on the feature map). Thus particular clusters of nodes on the map will correspond to particular features in the input space. The size of the neighbourhood is gradually reduced over the training period in attempt to distinguish between the finer features in the input space. Similarly, the rate at which the weights are adapted is reduced over the training process (i.e. for exactly the same vector of weights and input, the change made to the weights would be of lesser magnitude at later stages of training). The correct setting of these rates again is likely to require a degree of handcrafting.

3.4.7 Choice of approach to model motion camouflage

This section has introduced four neural network approaches that are in widespread use. The first three (MLPs, RBF networks and SVMs) act as universal function approximators and are applicable to non-linear regression problems such as motion camouflage. Conversely, the Kohonen Self Organising Map is suitable for classification tasks rather than regression and inappropriate.

SVMs have recently become popular, with their appeal lying in their basis in statistics. Despite this, it is probably true to say that there is still to be found an application where support vector machines have been found to significantly outperform all other approaches (Hearst 1998). For example, more recently in a comparison on a regression problem (a geographical spatial analysis task) SVMs were observed to perform slightly but not significantly better than MLPs (Gilardi & Bengio 2002). Comparisons of RBF networks and MLPs on regression tasks have also generally yielded little difference. For instance in a statistical comparison, there was found to be no significant difference between the probability estimate (in a medical task) given by MLPs trained using Backpropagation and RBF networks (Li *et al.* 1997). In another comparison involving the prediction of river flow, MLPs trained with Backpropagation were shown to give more accurate predictions than RBF networks (Dawson & Wilby 1999). Therefore for the purposes of this thesis there is no reason to expect (a priori) that any one of these techniques should give a dramatically better result than any other.

The majority of this thesis concentrates on MLPs trained using Backpropagation. The main reason for this is that the work is aimed to interest biologists and psychologists as well as computer scientists. The MLP architecture and Backpropagation training technique are arguably the most widely known and best understood in the biological community. From the perspective of biological plausibility it is thought fair to say that the architecture of the MLP resembles the wiring of biological systems at least as well as the other architectures discussed. The Backpropagation algorithm also allows greater flexibility in training a MLP than the SVM approach (e.g. a SVM is constrained to a single hidden layer). Two criticisms of this choice of Backpropagation are: that Backpropagation training is time consuming. This is conceded, however the time taken to train the networks in early experimentation was not judged to be too great a hindrance to warrant use of a different technique (or trial of techniques intended to hasten the training process). Secondly that to counter the bias variance trade-off Backpropagation requires a certain amount of handcrafting (methods such as network pruning and cross validation are effectively an afterthought for Backpropagation, in comparison to the SVM that approaches them directly). However, as shall be discussed in **section 5.1.2** the majority of the simulations presented in this thesis (with the exception of the experiments using hoverfly trajectories in **chapter 6**) represent the somewhat unusual situation where there is an unlimited supply of noise free training data available (i.e. the training data is automatically generated, it should also be noted that large training sets may pose problems for the SVM, see **section 3.4.5**). As such, methods employed largely to counter problems arising from small noisy training samples are not of such great relevance to this thesis. Nevertheless, a variant of cross validation is employed in an attempt to avoid networks becoming too finely tuned to particular training samples.

As a comparison to the MLP control system, a motion camouflage controller is implemented using a hybrid network incorporating elements of the Kohonen self-organising map with radial basis functions. This architecture had previously been suggested for robot homing (Rao & Fuentes 1995) and it was thought interesting to see how its performance would compare to the MLP (MLPs have been compared to RBF and SVM before (see above) and it would not be especially surprising to get comparative or even better performance with an RBF network or SVM). A control system based on this self-organising architecture is implemented in **chapter 4**. In **chapter 5** a similar control system is implemented using Backpropagation. The performance of the two approaches is compared in **section 5.3**).

3.5 Summary

Section 3.1 presented evidence that the relatively simple neural systems of insects are capable of solving problems similar to motion camouflage and provided a high level explanation of the methods used. The path integration techniques used by animals have been instrumental in suggesting methods that a shadower could use to determine the position of the fixed point without the aid of a reference point. Notable has been the simplicity of the algorithms used for the problems discussed. The behaviour of the hoverfly intercepting a target and of the ant navigating homeward can be described by simple but not obvious algorithms (see Collett & Land (1978) and Muller & Wehner (1988) respectively). At the expense of this simplicity, there comes a certain level of approximation and unreliability. However, in both cases described this does not seem to significantly compromise the animal in its natural surroundings.

Section 3.2 provided an overview of the sensory methods used by biological organisms to recover the spatial information necessary for motion camouflage. It was seen that the visual system is capable of providing this information alone, although other senses such as hearing could potentially be used or used to supplement the performance. Retrieval of direction and depth was seen to rely on complex calculations involving integration of several cues and high level processing (e.g. assuming the size of a well known object). Often some of the necessary information is generated internally (e.g. accommodation and convergence rely on information generated by the ocular muscles). Nevertheless, although eyes may be tricked by illusions, they are capable of providing accurate spatial information. It is indeed inadequacies of prey vision that motion camouflage seeks to capitalise upon. Motion camouflage minimises image motion through the shadower emulating the optic flow of a stationary object. As such the strategy stands a high chance of success against prey using image motion to sense depth. Conversely it is unlikely to be of practical value at distances where the prey can benefit from binocular depth cues, such as stereopsis, accommodation and convergence. However, as has been seen binocular depth cues become rapidly less precise as the distance to the target increases. **Chapter 2** has already identified that motion camouflage is a technique likely to be more successful at distance.

Section 3.3 commenced with an introduction to the basic concepts of control systems from an engineers perspective, then went on to suggest possible control systems for motion camouflage before closing with an introduction to biological networks. It was identified that there is not likely to be a definitive underlying model for a motion camouflage control system. Similar to **chapter 2**, where there was no definitive motion camouflage algorithm, the optimal will be very much dependent on situation. For most applications it is anticipated that the closed loop system that is tolerant to unexpected events would be preferable. It has also been shown that biological control systems are very complex and consequently difficult to understand. At the same time, whilst their complexity is unintuitive to human designers, they are efficient, they are adaptive and most importantly they work.

Finally **Section 3.4** provided an overview of the field of artificial intelligence. The symbolic approach (based on the logical manipulation of symbols) and connectionist approach (parallel distributed processing inspired by neurophysiology) were introduced and contrasted. This thesis adopts the connectionist approach (i.e. it was not obvious how to directly derive a solution to the motion camouflage problem, therefore the most profitable approach was seen as to attempt to train a mechanism that is likely to be able to learn the answer). Four of the more widely used connectionist learning algorithms were introduced and contrasted. The section concluded by selecting the MLP and the Backpropagation training algorithm upon which to concentrate work throughout the remainder of the thesis.

4 FIRST STEPS, IMPLEMENTATION OF A RESPONSIVE CONTROLLER WITH A SELF ORGANISING MAP

As mentioned in the previous chapter, this thesis applies two different neural approaches to motion camouflage. The first of these and the subject of this chapter is a self organising network incorporating radial basis functions. The network was originally employed by Rao & Fuentes (1995) for perceptual homing in an autonomous robot and they argue network architecture bears some similarity to the structure of the mammalian cerebellum. The second approach and the subject of **chapter 5** is Backpropagation. The problem task in this chapter and the next is the implementation of the responsive (motion camouflage) algorithm when the control systems are given sensory input information known to be sufficient to calculate camouflaged movements. It was reasoned that this preliminary investigation would give a good indication of the performance of the networks when applied to more difficult problems such as predictive motion camouflage. Hence, work in later chapters builds on the results of this chapter and the next.

The chapter commences in **section 4.1** with a general description of the methods used to train and test the networks in this and later chapters. In **section 4.2** the architecture, training procedure and operation of the self-organising map is described. The remainder of the chapter follows the form of an experimental write-up and describes the specifics of training, discusses the results of testing and suggests possible improvements to the system.

4.1 General Methods

The underlying methods used to train the networks are similar throughout the thesis. They are therefore described here in a separate section. Specific parameters and configurations are identified where relevant throughout the text. The calculation of camouflaged positions from the exact locations of shadower, prey and fixed point is described in **Appendix 1**.

4.1.1 Assumptions of the simulation

Throughout the remainder of the thesis, all motion camouflage simulations shall be based upon the following assumptions:

- The controller receives input at discrete time steps;
- After receiving input the controller initiates movement at the same time step, (the move will be completed by the next time step when the new input is received);
- The duration of and distance travelled in each movement are the same (i.e. the shadower moves at constant speed, in **chapters 4** and **5** the prey is constrained to move at a constant speed, in later chapters this constraint is relaxed);
- The shadower has freedom to move in any direction at any time step;
- The shadower can rotate independently of the direction that it moves;
- The fixed point is the initial position of the shadower.

These are at least two criticisms of the first three assumptions. The first criticism questions how closely the simulated control system that operates in discrete time steps models a real biological system. Biological sensors, control systems and motor structures were introduced in **chapter 3**. From the information presented in this introduction, it can be said without hesitation that any real world camouflage control system would not make decisions and movements at equally spaced time steps. Not least because a biological system would be controlled by the frequency signals of a multitude of neurons firing (not necessarily concurrently). A real system would also have to combat a whole range of problems such as muscular fatigue (e.g. muscles taking longer to contract than they had previously) that are not accounted for in the simulation. Nevertheless, this does not mean that it is not possible to model at least some of the processes in discrete time. For example the human visual system is deceived by animations where an image is rapidly updated at discrete intervals (indeed animation underlies the psychophysical experiment based on a computer game presented in **chapter 8**). The models presented throughout this thesis should be thought of as developing an initial framework from which to examine the basics of the motion camouflage problem. Once the framework has been established it may be extended. This incremental approach may be likened to the development of the fielder algorithms described in **section 3.1.3** culminating in Marken's (2001) simulation (which made a similar assumption of discrete time to make the problem tractable).

The second criticism concerns the shadower's movement. A real world shadower is unlikely to move at a constant speed and unlikely to be able to move in any direction at any time. Again, both were pragmatic initial assumptions made so as to provide a foundation for the study of camouflage control systems (that may be built upon in future work). However, in practice, giving the shadower complete freedom of movement is not as contentious as it may at first seem. Camouflaged approaches are by definition constrained (i.e. the shadower must lie on the constraint line). As long as the shadower approximates a camouflaged approach an argument can be made that it is moving realistically (qualified by the hoverfly flight paths that resemble motion camouflage observed by Srinivasan & Davey (1995)). Moreover casual inspection of motion camouflage trajectories generated by the control systems (e.g. tracking a hoverfly prey, **Fig 6.2**) give no reason to suspect that (for the parameters used in this thesis) the shadower's approach would require any more complex movement than the prey's (e.g. in **Fig. 6.2** the shadower's turns are less tight than the prey's). Given additional and specific information on the movement dynamics of a particular shadower (artificial or biological), incorporation of these into the model would constitute a worthy future progression of the work presented here.

The main reason for the selection of the fixed point at the initial shadower position is that it removes the need for the fixed point to be an existing object (see also **section 2.2.1**). Although in this chapter and the next the shadower is able to see the fixed point, in **chapters 6** and **7**, the position is estimated using path integration (in which case it makes no difference whether or not the fixed point can be seen). Note also that the choice of fixed point to be the initial shadower position is seen as the most natural by Srinivasan & Davey (1995).

4.1.2 Responsive training algorithm

The responsive algorithm introduced in **section 2.2.3** had one requirement, that the shadower make the lateral component of its movement sufficient to stay on the constraint line correct for the previous prey position. However for the purposes of network training it is also necessary to specify:

- the motivation for camouflage (i.e. whether the shadower wishes to catch, observe or retreat from the prey);
- the appropriate response when the shadower is not able to move far enough to reach the constraint line.

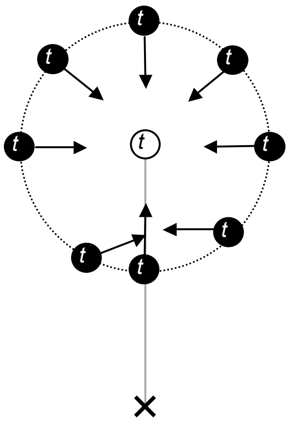


Fig. 4.1. Responsive training algorithm, shown at time = 0. Note: the shadower is also trained to rotate at time *t* to view the prey (positioned as it was at time *t*) frontally.

```
if ( distance to constraint line <= step size )
{
        move to the intersection of step and constraint line nearest to the prey
}
else
{
        move directly towards the nearest point on the constraint line
}
```

Addressing the first point, the training algorithm used throughout this thesis encourages the shadower to approach and catch the prey as quickly as possible. With respect to the second, if the shadower is incapable of actually reaching the constraint line it is trained to move as close to it as possible. This means that if the angle subtended at the prey by the fixed point and shadower is greater than or equal to $\pi/2$ radians (i.e. the prey is in between the fixed point and the shadower), then the shadower should move directly toward the prey (the nearest point on the constraint line is the prey). Otherwise it should move in a direction perpendicular to the constraint line (to bring it as close to the constraint line as possible). Finally, in keeping with the Srinivasan & Davey algorithm (**section 2.2.2**) the shadower is also trained to rotate to always view the prey (as positioned at the time of movement) frontally. The responsive training algorithm is illustrated in **Fig 4.1**.

4.1.3 Sensory inputs and motor outputs

In this chapter and the next it was decided to give the shadower access to information that it was known could allow calculation (using trigonometry) of appropriate motion camouflage responses. What was required for the simulation was a simple and reliable technique of supplying this information with some biological plausibility. **Section 3.2** saw the diversity of biological sensory systems and strategies used in nature to retrieve such information. It was also seen that if the angle of convergence of the eyes were known it would be possible to calculate the absolute distance to any point through binocular disparity.

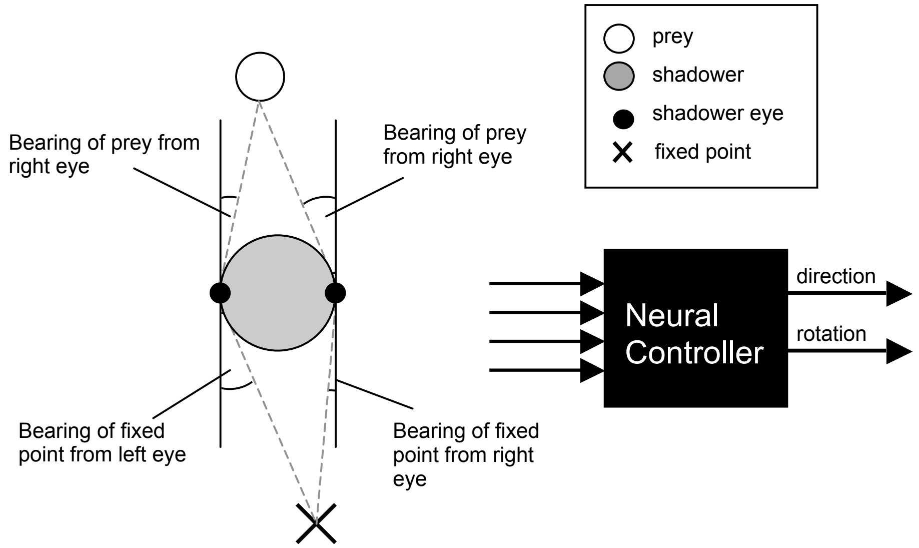

prey
shadower
shadower eye
fixed point
Bearing of prey from right eye
Bearing of prey from right eye
Bearing of fixed point from left eye
Bearing of fixed point from right eye
Neural Controller
direction
rotation

Fig. 4.2 Sensory inputs and motor outputs of responsive controller. The controller is provided with the bearing of prey and fixed point from both left and right eyes. It is expected to transform the information into an appropriate direction of movement and rotation (both relative to shadower orientation).

In a similar vein it was decided to give the shadower access to the angle of prey and fixed point relative to two eyes (positioned on opposite sides of the head). This information describes the exact position of both prey and fixed point relative to the shadower. As such it is likely to be far more detailed than the sensory input received by any biological organism. This choice could be further questioned, for if it were possible for the shadower to have such accurate spatial information then it is equally plausible that the prey could have also. The implication being that camouflage could be broken by looming cues (see **section 2.3**). However, as the primary concern was to gain an understanding of how well different neural architectures were likely to perform at motion camouflage, these points are regarded of secondary importance here. Nevertheless, they are addressed in **chapters 6** and **7**. The shadower is assumed to move a constant distance at each time step. The controller is expected to output a direction of movement and a rotation similar to those specified by the responsive training algorithm. This input and output configuration is displayed in **Fig 4.2**.

4.1.4 Training and testing procedures

This section describes the general training and testing procedures used throughout the thesis. The specific parameters used are identified throughout later text where relevant. Training and testing were undertaken by running sets of trials, during which the shadower would attempt to perform motion camouflage on a prey led along a fixed trajectory. Initial shadower positions were set at random distances (within a set range) from the initial prey position. The fixed point was always taken to be the initial shadower position.

In this chapter, in training the shadower was directed along camouflaged trajectories using the training algorithm. At each step along these trajectories the neural controller of the shadower would attempt to learn the correspondence between sensory input and motor output (i.e. supervised learning). During training in later chapters the control system was left to determine its own route and trained with the target output that it should have made after each step.

In testing the shadower was left to autonomously decide its course. For each shadower movement, recordings were made of the:

- directional and rotational deviations (error) of the controller output from the targets specified by the training algorithm (i.e. the angle difference between the actual direction moved and the target output);
- visual error, the angle subtended at the prey by the fixed point and the shadower;

- change in visual error from each shadower step to the next.

These were used to provide a measure of the overall quality of the camouflage trajectory.

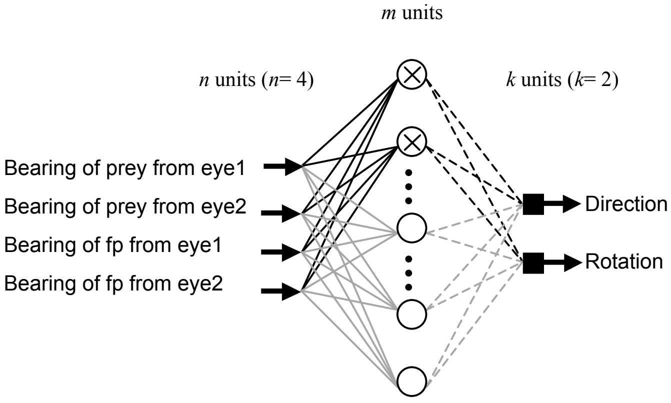
4.2 Self organising network

The self organising network used by Rao & Fuentes is one of a number of biologically inspired approaches to robot homing proposed recently. Lambrinos *et al.* (2000) developed a robot homing system based upon mechanisms used by the desert ant *Cataglyphis* (discussed in **section 3.1.2**). They developed a polarised light compass in analogue hardware that complemented proprioreceptive signals fed back from the wheels of the robot to estimate the current orientation. The distance travelled at each leg of the outward journey was estimated from the wheel encoders. The additional information supplied by the polarised light compass was statistically shown to improve the homing accuracy of the robot. Although the primary method of navigation in *Cataglyphis* is path integration, it has been shown that the ant will also make use of landmark cues to pinpoint the position of its nest if they are available (Wehner, Michel & Antonsen 1996). It has been suggested that this is achieved by the ant memorising a panoramic 'snapshot' of the layout of the landmarks surrounding its nest. Homing is achieved by comparing the visual snapshot to the current image viewed and moving to increase the similarity of the two images (Cartwright & Collett 1983). Roughly speaking, the homing animal moves in a direction such that the features in the current image that are most similar to those of the snapshot are brought closer to each other. If the feature in the current image is smaller in size the animal would move toward the feature. Lambrinos *et al.* augmented their robot with a modified snapshot algorithm. When tested in a simple landscape (where prominent landmark features could be segmented easily), the robot was shown to be able to successfully navigate back to within 20-30cm of the its starting point following short outbound trips of between 2-4m. Möller *et al.* (2000) have started making the first steps toward progressing the system to operate in more complex environments, though proper tests have yet to be undertaken. Although the homing accuracy of the robot as it stands is impressive, the robot is not directly applicable to the simulation of this chapter in that the shadower is supplied sensory input that gives the exact relative positions of the prey and fixed point (Lambrinos *et al*.s' algorithm would have relevance for a robotic application of motion camouflage, where the robot was required to use path integration to estimate the position of the fixed point). What is required for the control systems of this chapter is a method capable of learning a relationship between sensory input and motor output.

Hafner (2000) presented a neural network for robot navigation attempting to explain the formation of cognitive maps in mammals. The network is inspired by the identification of 'place' and 'head direction' neurons in rats. These cells fire as the rat moves around its environment (regardless of whether the rat can see where it is currently). The purpose of the network is to create a map of its environment as it is explored such that each output node corresponds to a particular spatial location. Similar to a conventional Kohonen map (**section 3.4.6**) the map is formed from two layers. As with the map of Rao & Fuentes described next, no assumption is made of the spatial arrangement of nodes in the output layer. The network is trained in two stages. In the first stage as the robot explores its environment the sensory inputs are self-organised. The nodes in the output layer are fully interconnected in attempt to capture the spatial relationship between places (represented by nodes) visited consecutively. This is modelled by increasing the strength of the connection between nodes that have been adapted consecutively. The direction that the robot has travelled between consecutive adaptions is also recorded. The second stage of training attempts to add metric information to the map (so that the nodes relate to positions in the environment). Positions are estimated using a force model previously used in graph drawing. In this physics based model all nodes are given repulsive charges, and nodes with strong connection weights are connected with springs. The repulsive and attractive forces are based upon the distance between nodes and the angle between connected nodes (i.e. the distance is small for nodes corresponding to similar sensory input vectors). The spatial organisation of the nodes follows an iterative technique. At each iteration a randomly selected node is moved according to the forces acting upon it. The process is repeated until the sum of the forces between nodes reaches a minimum (global or local). In a simulated simple environment the network was shown to be able to generate a map corresponding to the layout. However problems with the technique were evident in that nodes corresponding to distant spatial positions that were separated by an obstacle became connected. This was the result of the algorithms inability to disambiguate between nodes representing similar sensory vectors. As with conventional Kohonen maps this network is likely to require handcrafting to select the appropriate number of nodes and learning rates for the task. It faces additional difficulties, if the force model reaches a local minimum.

The neural network of Rao & Fuentes (1995) borrows from the Kohonen approach and the RBF network approach (**section 3.4.6** and **3.4.4** respectively). It was originally applied to a robot guidance task. The robot was trained to home via specific paths to a location based upon its current perceptions. An aspect of particular concern with this approach, as with Hafner's network, is that it is liable to encounter difficulties in the instance of replicate sensory inputs (i.e. the robot receives a similar input in two different places). This is likely to have more serious consequences with Rao & Fuentes's network in that it does not take into account the previous node that fired (i.e. previous place visited). This is not a cause for concern for the motion camouflage problem task of this chapter where there is a single solution for each combination of inputs.

Rao & Fuentes's technique firstly involves the creation of a sparse look-up table of mappings of sensory input to motor output, where sparse is used to indicate that the look-up table only contains a small subset of the entire set of possible input/output mappings. In the case of the homing application, this look up table was created as the robot was tele-operated around its environment associating current sensory inputs to the motor output. The look up table may be thought of as a network consisting of three layers of units, with the first layer representing the current sensory input and the third layer the motor output. In between these is a hidden layer that maps the input to the response. The network is fully connected such that the weight vector connecting the input to each hidden unit is of the same dimensionality to that of the input vector. Likewise the weight vector connecting each hidden unit to the output is of similar dimensionality to the output vector. **Fig. 4.3** displays the structure of the network.



→	Network input/output
—	Connection: (weight - input)^2
○	Node (Sum of connections).
×	Nodes with least sum contribute to output
- - - -	Connection: (weight * output)
■	Average

Fig. 4.3. Structure of Rao & Fuentess' self organising network. In this diagram two nodes contribute to the ouput of the network. Connections responsible for the output of the network are highlighted in black.

In training, the weights between input and hidden layer are adapted in a similar way to a conventional Kohonen map (e.g. the weight that is closest to the current input is brought closer to the current input). However the topographical arrangement of output nodes characteristic of conventional Kohonen maps (see **section 3.4.6**) is not present in the architecture examined here. The subset of nodes closest to the input are adapted in proportion to their Euclidian distance from the training input presented (i.e a weight that is close to the input will be adapted less than one that is further away). The corresponding set are adapted in a similar fashion to bring them closer to the target output (though for the output layer each of the weights is adapted by the same amount).

To enable the networks to generalise to novel situations, Rao & Fuentes advocate an averaging scheme based on radial basis functions (in this case Gaussian). The output of the network is calculated based upon a weighted average of the response of the output nodes. As with RBF networks, therefore, the problem is modelled locally (i.e. only a subset of the hidden nodes contribute to the output). The network attempts to interpolate between the response of a set of outputs that it has seen in similar situations. Each node's influence upon the network output is proportional to its distance from the current sensory input. Therefore a node whose input weights are very close to the input will have a high influence on the output. It was the generalisation (potentially) afforded by the interpolation of the RBF functions that made this technique appear attractive for application to motion camouflage. For the network to be regarded as an efficient solution to motion camouflage, it would be necessary for the population averaging to yield an accurate response from a hidden layer with few nodes. The following two sections formally describe the calculations involved in the training and operation of the network.

4.2.1 Training

This section defines the training method presented by Rao & Fuentes (1995). The terminology follows that of Rao & Fuentes. The exact parameters used in the simulation are described in **section 4.3**.

Letting:

$\vec{s}$ = the current sensory input vector consisting of *n* units.

$\vec{w}_i$ $(1 \le i \le m)$ = the vector of weights between hidden unit *i* and the sensory input layer.

$\vec{m}_i$ $(1 \le i \le m)$ = the vector of weights between hidden unit *i* and the motor output layer.

Then repeat over each trial:

1. Calculate the (euclidian) distance $d_i = \|\vec{w}_i - \vec{s}\|$ between every $\vec{w}_i$ and current input $\vec{s}$.
2. Select the hidden units $N(i*)$ that represent the $f(t)$ smallest distances between hidden unit and sensory input. Where *t* represents the number of training iterations and $f(t)$ decreases as *t* increases. In fact, for this application preliminary tests suggested that it was most profitable to adapt a single unit at each iteration.
3. For each hidden unit in the set $N(i*)$ modify the weight vector using a Gaussian gain function by:

$$\vec{w}_i := \vec{w}_i + \gamma(d_i, t) \cdot (\vec{s} - \vec{w}_i) \qquad \text{where } i \in N(i*). \tag{4.1}$$

$$\gamma(d_i, t) = e^{-d_i^2/\sigma(t)} \qquad \text{where the Gaussian spread term } \sigma(t) \text{ decreases as } t \text{ increases} \tag{4.2}$$

4. For each hidden unit in the set $N(i^*)$ modify the vector of weights connected to motor output using the current motor input vector $\vec{m}$ (supplied by responsive training algorithm) by:

$$\vec{m}_i := \vec{m}_i + \beta(t) \cdot (\vec{m} - \vec{m}_i) \quad \text{where } i \in N(i^*). \tag{4.3}$$

$$\beta(t)(0 < \beta(t) < 1) \qquad \text{where } \beta(t) \text{ decreases as } t \text{ increases.} \tag{4.4}$$

4.2.2 Operation

1. (as in training) Calculate the (euclidian) distance $d_i = \|\vec{w}_i - \vec{s}\|$ between every $\vec{w}_i$ and current input $\vec{s}$.
2. Select the hidden units $N(i^*)$ that represent the *f* smallest distances between hidden unit and sensory input where *f* is a constant.
3. Calculate the motor output. The output of each motor unit ($o_j, (1 \leq j \leq k)$) is calculated on the basis of a Gaussian weighted average of the weights m_{ij} between motor output unit *j* and hidden unit *i* for $i \in N(i^*)$:

$$o_j = \sum_{i \in N(i^*)} e^{-d_i^2/\sigma} \cdot m_{ij} \tag{4.5}$$

As an experimental control included to allow investigation of the relative benefit of using the Gaussian weighted averaging, the following linear average (i.e. mean) of the weights m_{ij} is also adopted:

$$o_j = \sum_{i \in N(i^*)} \frac{m_{ij}}{f} \tag{4.6}$$

4.3 Methods

Three sensorimotor self-organising maps were trained using the algorithm described in **section 4.2.1** in the manner described in **section 4.1.4** (multiple networks were trained in order to investigate whether the training technique repeatedly gave networks of similar quality). The parameters used for training and configuration of the map are displayed in **Tables 4.1a&b**, respectively. The parameters were selected following a set of preliminary pilot experiments. In explanation of the tables, each trial lasted 100 steps and the shadower and prey moved a constant distance per step. The initial position of shadower relative to the initial position of the prey was determined randomly within the bearing and distance ranges shown. At each step of the trial, the prey's direction was changed by a constant angle (determined randomly prior to the trial). This meant that the trajectory of the prey always resembled an arc (the radius of which is randomly determined). Training parameters for the self-organising map are self explanatory (**Table 4.1b**). Weights connecting sensory layer to hidden layer and motor layer to hidden layer were initialised with values calculated from running the responsive training algorithm on the trials just described. (i.e. the hidden layer comprises 2000 units, meaning that 20 trials (of 100 steps) were needed to generate the initial 2000 input output mappings).

In testing the performance of the maps three possibly interrelated areas were identified as being of particular interest.

- whether the three different maps (trained in a similar manner) performed differently;
- whether there was a difference between the Gaussian and linear output averages (taking into account the number of units used in averaging);
- whether there was a difference in performance relating to different prey trajectories.

In order to examine these factors a factorial experiment was designed. Three prey trajectories were selected for experimentation in each test undertaken. The first, (*trajectory a*) involved the prey travelling in a straight line. During the second, (*trajectory b*), the prey turned 0.01 radians at each step and the final (*trajectory c*) the prey turned 0.02 radians at each step. As such the trajectories represent the minimum, mid and maximum sharpness of turning that could be expected in the training set. For each of these trajectories 50 trials were run. The initial positions of the shadower were selected randomly for each trial within the ranges allowed by training (see **Table 4.1a**). However the same set of 50 random starting positions were used for each prey trajectory. A single test therefore consisted of 3 prey trajectories, each with 50 trials, leaving a total of 150 trials per test. Tests were undertaken on shadowers using every combination of trained map (3 maps), with the Gaussian averaging method (using {2,3,4} units for output, see also **Table 4.1c**) and the linear averaging method (using {1,2,3,4,5} units for output, see also **Table 4.1.d**). This gives a total of (5*3)+(3*3) = 24 tests.

Tables 4.1a,b Parameters defining the **(a)** training procedure and **(b)** self organising map training configuration. **(c)** and **(d)** identify parameters used in averaging for motor output. See **section 4.2.2** for terminology

(a) Training procedure	
Number of steps in trial	100
Shadower step size (units)	0.5
Prey step size (units)	0.4
Prey direction change constant per step?	Yes
Prey direction change (per step) range (rad)	$-0.02 \le x \le 0.02$
Initial distance range of shadower from prey (units)	$50 \le x \le 90$
Initial bearing of shadower from prey (rad)	$0 \le x \le 2\pi$

Selected randomly before trial

(b) SOM training	
No units in hidden layer	2000
$\sigma(t)$	0.0005
$f(t)$	1
$\beta(t)$	0.2

(c) Gaussian averaging	
f	{2,3,4}
σ	0.01

(d) Linear averaging	
f	{1,2,3,4,5}

4.4 Results

How well did the neural maps perform? **Figs 4.4a,b,c** display shadower trajectories obtained using one of the better configurations (using the linear average of three units for output). Casual inspection suggests that the performance for all three prey trajectories and from each starting position is competent. This is upheld by **Figs 4.5 a & b** which display the mean visual error per trial (plotted against the initial bearing of the shadower from the prey and the start distance of the shadower from the prey respectively. There are a few notable examples where the mean visual error is particularly high. It is thought that these are a direct result of the shadower over shooting the prey (e.g. the right most shadower in **Fig 4.4c**) which could easily result in maximum error (i.e. π radians). Supporting this, the large errors tend to occur when the start position of the shadower is at a bearing from the prey similar to the direction that the prey will move (i.e. shadower and prey move toward one another). Also there is a tendency for high error when the start distance is small. Finally, **Fig 4.5c**, a plot of the mean visual error against the mean change in visual error from one step to the next, indicates that there was never an instance of high mean error that was not paired with high deviation change. High values of deviation change would result from the shadower over stepping the prey.

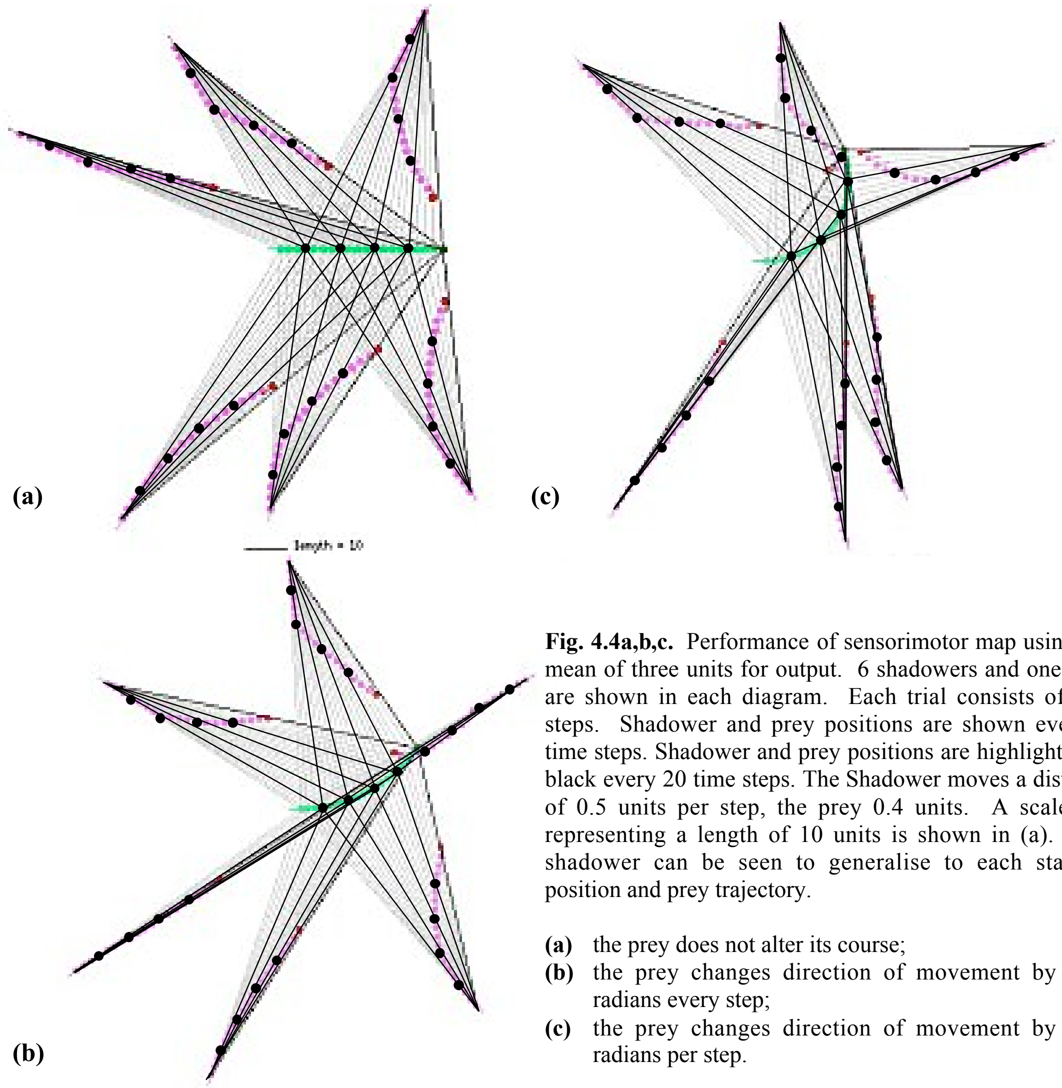


Fig. 4.4a,b,c. Performance of sensorimotor map using the mean of three units for output. 6 shadowers and one prey are shown in each diagram. Each trial consists of 100 steps. Shadower and prey positions are shown every 5 time steps. Shadower and prey positions are highlighted in black every 20 time steps. The Shadower moves a distance of 0.5 units per step, the prey 0.4 units. A scale bar representing a length of 10 units is shown in (a). The shadower can be seen to generalise to each starting position and prey trajectory.

(a) the prey does not alter its course;
(b) the prey changes direction of movement by 0.01 radians every step;
(c) the prey changes direction of movement by 0.02 radians per step.

What were the differences in performance between the different network configurations? **Figs 4.6a & b** show the mean direction and rotation error per test (i.e. the mean respective difference between actual output and responsive algorithm output). The graphs indicate that whereas the number of units contributing to the linear average had little influence upon performance, they affected Gaussian averaging dramatically. This was confirmed statistically using two-way Analysis of Variance (ANOVA, see Underwood (1997) for more details) with the different prey trajectories cross classified against the number of units averaged. **Tables 4.2a & b** show the ANOVA tables for the directional and rotational error resulting from the linear average. No significant difference was found between performance on different prey trajectories with different numbers of units. Alternatively **Tables 4.3a & b** represent similar ANOVA tables regarding the Gaussian average. These show that whilst again there was no significant

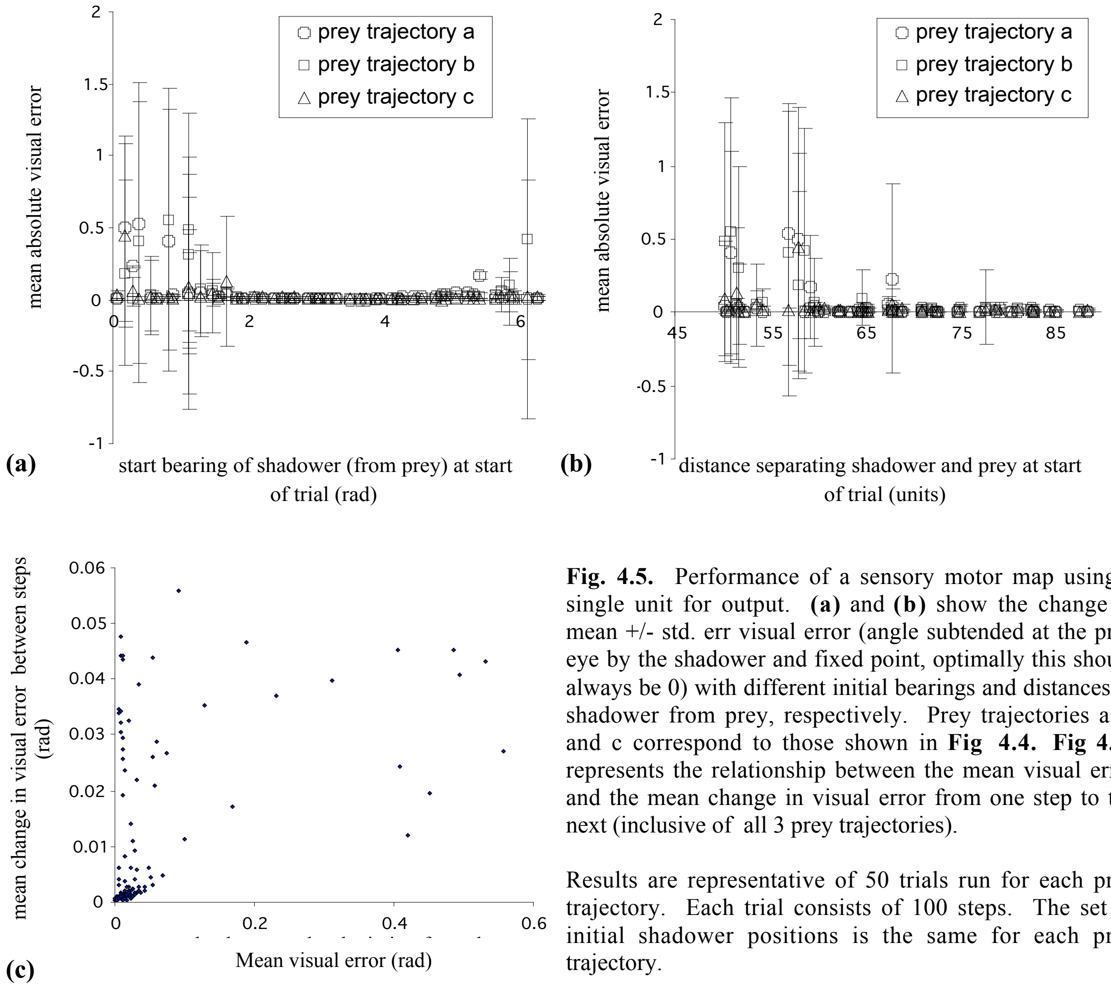


Fig. 4.5. Performance of a sensory motor map using a single unit for output. **(a)** and **(b)** show the change in mean +/- std. err visual error (angle subtended at the prey eye by the shadower and fixed point, optimally this should always be 0) with different initial bearings and distances of shadower from prey, respectively. Prey trajectories a, b and c correspond to those shown in **Fig 4.4. Fig 4.5c** represents the relationship between the mean visual error and the mean change in visual error from one step to the next (inclusive of all 3 prey trajectories).

Results are representative of 50 trials run for each prey trajectory. Each trial consists of 100 steps. The set of initial shadower positions is the same for each prey trajectory.

difference between camouflage performance on different prey trajectories, there was a highly significant difference in both rotation and direction deviations between differing number of units averaged. These results have several implications. The fact that there is no significant difference between the number of units contributing to the linear average is suggestive that a similar result could be obtained using a lesser number of units in the hidden layer. With respect to the Gaussian averaging, the difference in performance resulting from the number of units averaged indicates that either parameters (especially Gaussian spread values) were not set optimally, or that the technique is not appropriate for this application.

The remaining points of interest are whether there was a difference in performance between the different maps (initialised and trained in a similar manner) and whether there was a difference in performance between the averaging types. **Figs 4.7 a & b** display the mean directional and rotational error per test using the linear and Gaussian average on two units (results illustrated in **Fig 4.6** had suggested that the Gaussian average was most successful using two units). **Fig 4.7a** implies a difference between the directional error of the different maps using either method of averaging. This is not apparent with the rotational results shown in **Fig 4.7b**. The two graphs are also suggestive of a difference in performance between averaging types. Mean values for the linear average are always lower and standard errors for the Gaussian average tend to be greater (especially concerning the rotational data). These differences in spread meant that the data did not meet the homogeneity of variance assumption of analysis of variance (see also **Tables 4.4a & b**). Regardless the tests were undertaken and results considered in light of the inadequacy of the data. **Tables 4.4a & b** show the results of two-way nested ANOVAs conducted on directional and rotational errors. Fixed main effects are the different averaging methods and the different prey trajectories which are cross-classified, with different maps (labelled as weights in the ANOVA table) nested within averaging method. The analysis indicated once again that there was no significant difference between performances over different prey trajectories. Conversely it showed a highly significant difference between the different maps with respect to the directional deviation and also a highly significant difference between the two averaging methods in relation to the rotational deviation. Although the accuracy of these results should be treated tentatively, the significance levels are very high. The implication of the difference between maps is that training has not resulted in convergence upon an optimal solution. It can be expected that the situation could be improved through selection of more appropriate training parameters. On top of this there is no evidence that the Gaussian weighted averaging technique provides any advantage over linear averaging when applied to motion camouflage. In fact, the drop in performance of Gaussian averaging with the increase in number of units averaged (**Figs 4.6a&b**) and the significant difference between Gaussian and linear average in rotational performance implies that the Gaussian technique is the less suitable.

Table. 4.2. Two-way ANOVAs investigating the difference in directional **(a)** and rotational **(b)** error between the different prey trajectories using the mean of {1,2..5} units for output. Both main effects are fixed. Both mean errors were calculated from a total of trials per prey trajectory, 100 steps per trial. Results indicate in both cases that there was no significant difference between the different prey trajectories or between the number of units used for output (in the range). Residuals for the rotational data were positively skewed. Otherwise Levene's test showed that variances were suitably homogenous in both cases.

(a)

Source	*df*	*MS*	*F*	*P*
No. units averaged	4	0.1810	2.35	> 0.05
Prey trajectory	2	0.1493	1.94	> 0.10
P. traj * no. units avgd	8	0.0072	0.09	> 0.99
Residual	735	0.0770		
Total	749			

(b)

Source	*df*	*MS*	*F*	*P*
No. units averaged	4	0.0057	0.22	> 0.9
Prey trajectory	2	0.0479	1.83	> 0.15
P. traj * no. units avgd	8	0.0030	0.11	> 0.99
Residual	735	0.0262		
Total	749			

Table. 4.3. Two-way ANOVAs investigating the difference in directional **(a)** and rotational **(b)** error between different prey trajectories using the Gaussian weighted average of {2,3,4} units for output. Both main effects are fixed. Both mean errors were calculated from a total of 50 trials per prey trajectory, 100 steps per trial. Results indicate in both cases that there was no significant difference between the different prey trajectories. However for both direction and rotational error there is a highly significant difference between the number of units used for output (in the range). Directional and rotational data were log-transformed to resolve heteroscedasticity (confirmed using Levene's test). Residuals for the rotational data were positively skewed, however this can be expected not to have unduly affected the results (Underwood, 1997).

(a)

Source	*df*	*MS*	*F*	*P*
No. units averaged	2	1.2364	70.83	< 0.00001
Prey trajectory	2	0.0240	1.37	> 0.25
P. traj * no. units avgd	4	0.0086	0.49	> 0.7
Residual	441	0.0175		
Total	449			

(b)

Source	*df*	*MS*	*F*	*P*
No. units averaged	2	18.997	50.46	<0.00001
Prey trajectory	2	0.1049	0.28	> 0.75
P. traj * no. units avgd	4	0.0208	0.06	> 0.99
Residual	441	0.3765		
Total	449			

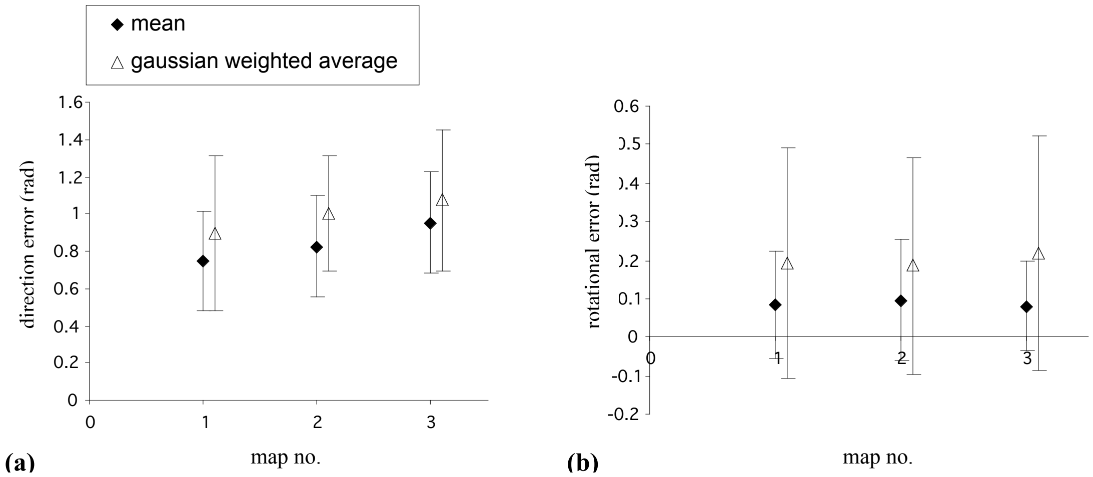


Fig. 4.7(a) Difference in the mean +/- std err angle directional error between the three maps trained in a similar manner. Results are shown for both linear and Gaussian averaging methods.

(b) Difference in the mean +/- std err rotational error between the three maps trained in a similar manner. Results are shown for both linear and Gaussian averaging methods.

Mean values represent the mean error over 150 trials: 50 trials for each of the three prey trajectories (a, b and c see **Fig 4.4**). 100 steps per trial. Results suggest that there is no significant difference in performance between different prey trajectories using any of the maps with either averaging method. (see **Tables 4.2 & 4.3**).

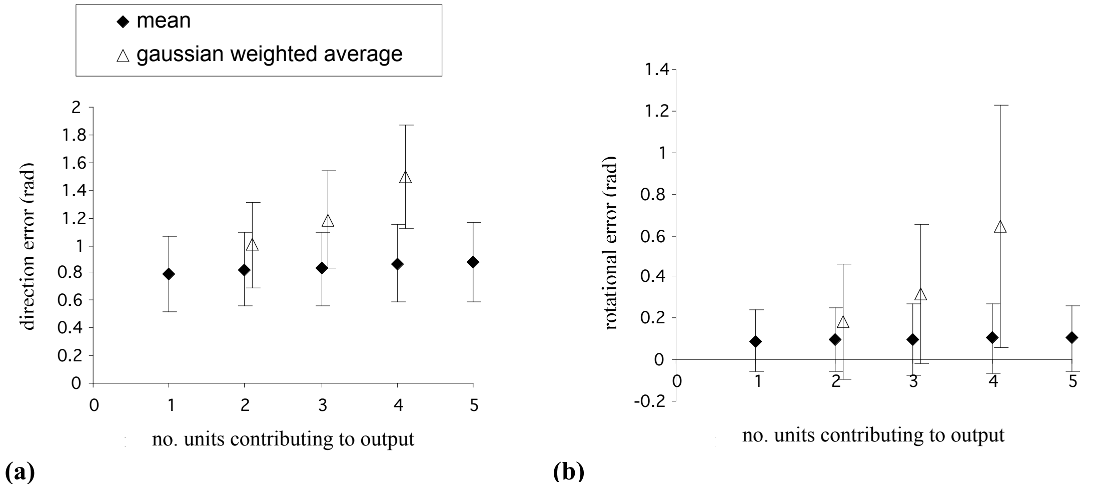


Fig. 4.6. (a) Difference in the mean +/- std err direction error between the number of output units contributing to the average. Results shown for both linear and Gaussian averaging methods. **(b)** Difference in the mean (+/-) std err rotational error between the number of output units contributing to the average. Results shown for both linear and Gaussian averaging methods. Mean values represent the mean error over 150 trials: 50 trials for each of the three prey trajectories (a, b and c see **Fig 4.4**). 100 steps per trial. Results suggest that there is no significant difference in performance between different prey trajectories using either averaging type with any number of units (within range shown) contributing to output (see **Table 4.2, 4.3**).

Table. 4.4. Two-way nested ANOVAs investigating the difference in directional **(a)** and rotational **(b)** error between linear and Gaussian averaging over 3 different prey trajectories, using 3 maps trained in a similar manner. Output averages use two units. Averaging method and prey trajectory are treated as fixed main effects. Different maps (labelled in the table as Weights (Av. Method)) are nested within averaging method. The mean error was calculated from a total of 50 trials per prey trajectory, 100 steps per trial. Results indicate in both cases that there was no significant difference between the different prey trajectories. Alternatively for both direction and rotation there is a highly significant difference between maps (within averaging types). There is also a highly significant difference in rotational error between linear and Gaussian averaging methods.

However these results should be treated with caution. Levene's test indicated heteroscedasticity in both directional and rotational data that could not be removed by transformation. This is likely to have led to the increased probability of incorrect rejection of the null hypotheses (so insignificant differences are still valid) (Underwood, 1997) . Rotational data showed heavy positive skewness.

(a)

Source	*df*	*MS*	*F*	*P*
Averaging method	1	5.043	3.53	> 0.1
Weights (Av. method)	4	1.427	13.50	<0.0001
Prey trajectory	2	0.082	0.77	> 0.45
P. traj * Av. method	2	0.004	0.04	> 0.95
P. traj * Weights(Av. method)	8	0.009	0.08	> 0.99
Residual	882	0.106		
Total	899			

(b)

Source	*df*	*MS*	*F*	*P*
Averaging method	1	2.830	52.67	<0.0001
Weights (Av. method)	4	0.030	0.56	> 0.65
Prey trajectory	2	0.075	1.39	> 0.25
P. traj * Av. method	2	0.011	0.21	> 0.8
P. traj *Weights(Av. method)	8	0.004	0.07	> 0.99
Residual	882	0.054		
Total	899			

4.5 Discussion

This section has seen that it is possible to implement a system that will competently perform motion camouflage in the conditions described using a self-organising map. This result is not altogether surprising, since the map is effectively operating as a look-up table (given a large enough table and sufficient input-output pairs it would be possible to model the problem to any degree of accuracy). Note that when a single node is used to generate output with a linear average, this approach is equivalent to the state-space look-up table discussed by Churchland & Sejnowski pp333-337 (1992). However the purpose of the implementation was not to laboriously create a large look up table. The intention of the averaging was to reduce the number of hidden units whilst maintaining accuracy. The results show that there was no significant benefit afforded through averaging. This is possibly indicative that the number of hidden units in the hidden layer could be reduced. What was of particular concern was the failure of the Gaussian weighted averaging. It can be confidently stated that this is not due to inadequacy in the network design. On closer inspection, the averaging scheme lacks normalisation and is quite capable of giving averages that are outside the range of the data averaged. Clearly this is unacceptable for any interpolation method. Notably the technique is later discarded by Rao & Fuentes (1996) who move to a normalised weighted Gaussian averaging scheme, where the multiplying term in eq 4.2 & 4.5 is replaced by:

$$\gamma(d_j,t) = \frac{e^{-d_j^2/\sigma_j(t)}}{\sum_{k=1}^{m} e^{-d_k^2/\sigma_k(t)}} \tag{4.7}$$

Preliminary pilot experiments have suggested that this method offers the potential to be superior to either the linear average or the previous weighted averaging scheme. It is however heavily dependent upon the selection of appropriate parameters, which theoretically could be found using genetic algorithms (at the expense of time).

It is likely that the self-organising map architecture can be significantly improved for the responsive algorithm. What is the potential for application to the predictive algorithm? The predictive controller must integrate the recent chase history with current sensory input to provide the motor output (unless it is capable of accurately predicting the prey's entire path in advance). The most obvious method for achieving this would be to feed previous sensory input information back into the system. In order to avoid an explosion in the number of units required by this self organising network it would be necessary to reduce the previous sensory data to a lower dimensionality. Rao & Fuentes (1996) advocate an approach where the current output is based upon previous motor outputs as well as the current sensory input. Therefore, the current sensory input indirectly contributes to future motor outputs. As such depending upon the ratio of input units to output units (in this case, there are less output units) this would reduce the data to lower dimensionality and may relieve the need to feed back data as input to the system. Further work on self-organising maps is discontinued in this thesis in light of the superior performance of back-propagation discussed in the next chapter.

5 IMPLEMENTATION OF THE RESPONSIVE ALGORITHM USING BACKPROPAGATION

This chapter applies the Backpropagation training algorithm to the motion camouflage task investigated in **chapter 4**. Backpropagation is described in **section 5.1**. The performance of Multilayer Perceptron camouflage control systems trained using Backpropagation is examined throughout the rest of the chapter. The accuracy of the MLPs of this chapter is compared with the self organising networks of the previous chapter in **Section 5.3**.

5.1 Backpropagation

Backpropagation is one of the most widely studied and used neural network training algorithms. The underlying concept has been proposed independently in various forms (see Rojas 1996), however Rumelhart, Hinton & Williams (1986) are those most credited for bringing Backpropagation to the artificial intelligence community. The algorithm can theoretically be used to train a Multilayer Perceptron network to approximate almost any function to any level of accuracy. A Multilayer Perceptron is formed from multiple layers of nodes. The first layer represents the inputs to the network. The final layer gives the network output. The layers in between are referred to as hidden layers.

Excluding the input layer, in a standard (*fully connected*) network each node receives input information from all nodes in the preceding layer. The node, bearing similarities to a biological neuron, acts to compute an output that is a function of the thresholded input received. Alike a biological system the input signals are modified by weighting factors (analogous to synapses) before they reach the node. It is through adapting these weights that the network is able to learn. The network is trained by supervised learning. At the start of training weights are randomised. The network is given an input and the desired output response. The weights on all connections in the network are adapted in a manner intended to make the actual output closer to this target. Classically this process is repeated with different example inputs and targets until the actual network output becomes sufficiently accurate, or the network converges and weight changes are zero.

Weight changes are determined through differentiation. The error of the network (i.e. difference between actual output and target output) is considered as a function of the weights. As it is not possible to directly calculate the set of weight values for which the error is at its minimum, a search technique is used: the values of each weight are changed in proportion to their likely influence upon the overall error. In other words if the magnitude of the rate of change of the error with respect to a particular weight is high, then the weight change will be relatively large. If it is low, the change will be relatively small. The weight change process is frequently likened to rolling a ball across the error surface (the surface given by the error as a function of the weights). The position of the ball is given by the current combination of weights. By rolling the ball down the steepest slopes it is hoped that it will eventually descend to the point of least altitude (and error). A more formal description is provided in the next section.

5.1.1 A formal description

In the following text, consecutive layers in the network are referred to using uppercase bold script in alphabetical order. If the final layer is **K**, the preceding layer is **J**. The number of nodes in a layer is indicated by uppercase italics corresponding to the layer letter (layer **J** has *J* nodes). Lowercase subscript is used to associate variables with layers of corresponding uppercase script. In accordance, o_j is used to refer to the output of a node in layer **J** and w_{ij} refers to the weighting of the output of node *i* in layer **I** that contributes to the input of node *j* in layer **J**. A possible network structure illustrating the connections between layers is shown in **Fig.5.1**. Notice that the first layer of nodes is represented as arrows (the inputs), stressing the difference between it and the other layers.

Typically the function to be trained is complex with many different input-output pairs required to allow the error to be minimised. The text below explains the calculation of all weight changes for a single training iteration. In practice many training iterations would be required, with training pairs commonly selected at random from the training set.

Initial weights are selected randomly. The network is trained until it converges upon a minimum (where the error slope is zero and consequently weight changes become zero) or a desired accuracy level is reached.

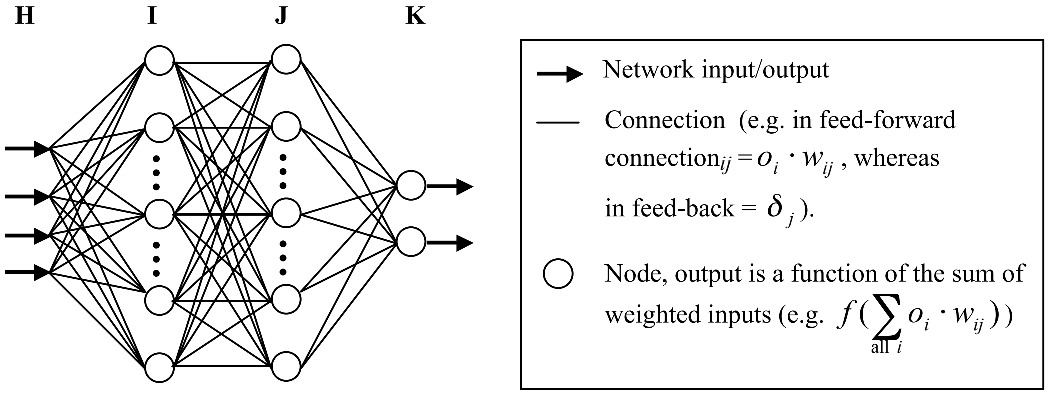

H
I
J
K
Network input/output
Connection (e.g. in feed-forward connection$_{ij} = o_i \cdot w_{ij}$, whereas in feed-back = δ_j).
Node, output is a function of the sum of weighted inputs (e.g. $f(\sum_{\text{all } i} o_i \cdot w_{ij})$)

Fig. 5.1. Abstract representation of the structure of a Multilayer Perceptron with 2 hidden layers. Layer **H** is the input layer, **I** and **J** are hidden layers and **K** is the output layer. Theoretically the network could contain any number of layers.

Each training iteration consists of two stages:

- a *feed-forward* stage, where the input is received and propagated forward through the network layer by layer to give the output;
- a *back-propagation* stage, where the output is compared to the target, and a signal relating to the error passed back through the network layer by layer. Weights are changed according to this signal.

In the feed-forward stage the output of any node at time *n* (e.g. $o_j(n)$) is given by the sum of the weighted outputs from the previous layer and a weighted bias *B* (where the bias weighting is denoted as *W*):

$$o_j(n) = f\left(\left(\sum_{i=1}^{I} o_i(n) \cdot w_{ij}(n)\right) - W_j(n) \cdot B_j\right) \qquad (5.1)$$

B is a constant usually set at 1 or –1. Classically the bias is the same for all nodes in the network. For ease of later explanation, the bias will be considered as a weighted constant output from a node in the previous layer (note that the superscript on the summation has been changed to accommodate this), so equ(5.1) becomes:

$$o_j(n) = f(\sum_{i=1}^{I+1} o_i(n) \cdot w_{ij}(n)) \tag{5.2}$$

where:

$$f(x) = \frac{1}{1+e^{-x}} \tag{5.3}$$

which constrains the output to the range $0 < f(x) < 1$. The purpose of the bias is to speed training by offsetting the weighted sum of inputs to a position where the rate of change of equ(5.3) is higher.

Considering the back-propagation stage, if **K** is the output layer and $t_k(n)$ the target output for the *kt*h node in this layer, the error of the network $E(n)$ may be considered to be:

$$E(n) = \frac{1}{2} \cdot \left(\sum_{k=1}^{K+1} (t_k(n) - o_k(n))^2 \right) \tag{5.4}$$

(i.e. half of the squared Euclidian distance between target and output).

All weights are updated in proportion to the negative of the rate of change of $E(n)$ with respect to the weight. Partially differentiating $E(n)$ yields two general formulae for weight changes:

- a formula for weights of the output layer, where the target is known explicitly;
- a formula for weights of the hidden layers, where the target is unknown.

At time $(n+1)$, the new value of any weight in the *output layer*, $w_{jk}(n+1)$ is given by:

$$w_{jk}(n+1) = w_{jk}(n) - \eta \cdot \frac{\partial E(n)}{\partial w_{jk}(n)} \tag{5.5}$$

where η is a learning rate parameter in the range $0 < \eta < 1$. If $\delta_k(n)$ is defined to be:

$$\delta_k(n) = (t_k(n) - o_k(n)) \cdot (o_k(n) \cdot (1 - o_k(n))) \tag{5.6}$$

then equ(5.5)may be rewritten as:

$$w_{jk}(n+1) = w_{jk}(n) + \eta \cdot \delta_k(n) \cdot o_j(n) \tag{5.7}$$

If w_{ij} is the weight of any node from any *hidden layer* **J**, then at time $(n+1)$ its new value is given by:

$$w_{ij}(n+1) = w_{ij}(n) - \eta \cdot \frac{\partial E(n)}{\partial w_{ij}(n)} \tag{5.8}$$

Defining $\delta_j(n)$ to be:

$$\delta_j(n) = \left(\sum_{k=1}^{K+1} \left(\delta_k(n) \cdot w_{jk}(n) \right) \right) \cdot (o_j(n) \cdot (1 - o_j(n))) \tag{5.9}$$

allows the new weight to be rewritten in a similar format to equ(5.7):

$$w_{ij}(n+1) = w_{ij}(n) + \eta \delta_j(n) \cdot o_i(n) \tag{5.10}$$

To calculate each δ in the hidden layers (equ (5.9)) it is necessary to have knowledge of each δ from the layer above. The acquisition of this information may be likened to the δ terms being propagated back along the connections of the network (see **Fig 5.1**), thus giving Backpropagation its name.

The error of the network for a single training pair was given in equ (5.4). However the purpose of training is to minimise the error over the whole set of possible input-output pairs. As equ (5.4) does not consider this whole set it means that the weight change may not follow the direction of negative gradient. This effect may be alleviated by batch training. Here weight updates would be suppressed at each training iteration and the cumulative weight change applied periodically according to the following equation (Rojas 1996), where there are N patterns in the training set:

$$w_{ij}(N+1) = w_{ij}(N) + \eta \sum_{n=1}^{N} \delta_j(n) \cdot o_i(n) \tag{5.11}$$

In practice batch training may not offer any real advantage. Without batch training the search direction is likely to follow the direction of negative gradient on average (as long as training pairs are applied randomly). Also the inherent noise added may be helpful in escaping local minima (Rojas 1996). Finally, a useful modification of the algorithm intended to reduce training time is the addition of momentum to the weight changes. Here a term is added to the weight change that is proportional to the previous weight change:

$$w_{ij}(n+1) = w_{ij}(n) + (\eta \cdot \delta_j(n) \cdot o_i(n)) + \alpha(\eta \cdot \delta_j(n-1) \cdot o_i(n-1)) \qquad (5.12)$$

where α is in the range $0 < \alpha < 1$. It follows that all previous weight changes contribute to the current weight change (in proportion to how recently they occurred) thus providing a sort of inertia. This can act to accelerate the initial convergence of the network and reduce the chance of getting trapped in local error minima.

5.1.2 Preliminary investigation of different training techniques

Primarily an investigation was undertaken in order to give an impression of whether it would be more profitable to use batch training or on-line training to train the MLPs. Whereas early attempts to train motion camouflage using batch training yielded little success, it was found possible to train the networks to a respectable degree of accuracy on-line. In order to gain confidence that this was not simply the result of an inadequate choice of parameters a more formal comparison was undertaken. Five networks were trained on-line and five networks were batch trained on similar data sets. The training parameters of all networks were otherwise the same (adaption rate = 0.0001, momentum = 0.1). The parameters were set following a series of preliminary trials intended to maximise the accuracy of the batch trained networks. The parameters were not optimal for the networks trained on-line. All networks had a single hidden layer consisting of ten nodes. The training set consisted of 5000 input and target training pairs. In each pair, the shadower, prey and fixed point were arranged randomly according to the following specification (see also **Fig 5.2a**). The fixed point was translated to a random position between 50 to 90 distance units from the prey. The shadower was translated to a position at a randomly selected distance from the prey that was less than the distance of the fixed point. The bearing of the shadower from the prey was randomly chosen to be within 0.01 radians of the bearing of the fixed point from the prey. Similarly, the shadower was orientated to face a randomly selected angle no more than 0.01 radians to the left or right of the prey.

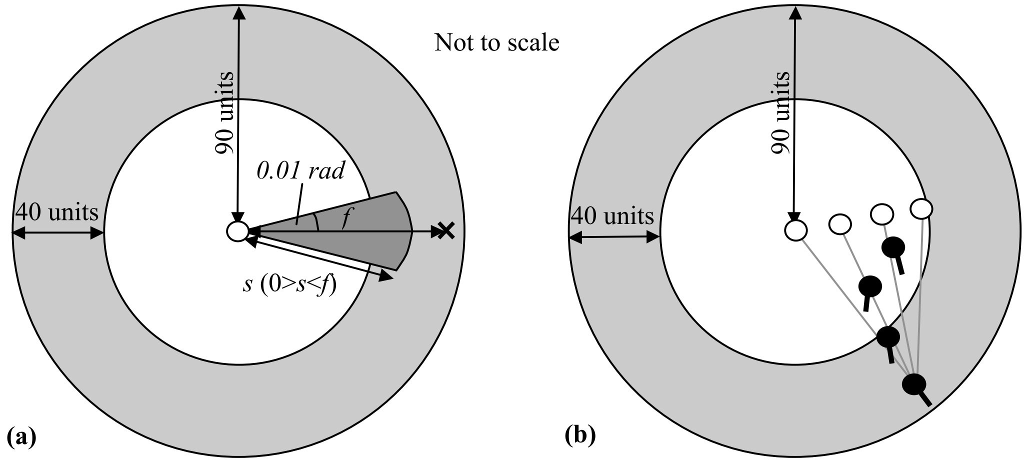


Fig 5.2. Diagramatic representations of the procedures used to select shadower, prey and fixed point positions for use in network training. **(a)** Selection procedure used to generate the training set for the preliminary experiment described in **section 5.1.2**. The light grey shaded area represents the set of possible fixed point locations relative to the prey (the centre of the circle). In this illustration the fixed point lies at a distance *f* units from the prey at point X. Given this location of fixed point, the set of possible shadower locations is represented by the segment shaded dark grey (the shadower position is chosen randomly). The segment is defined in terms of its radius *s* (where 0>*s*<*f*), and the angle subtended by the outer arc (0.02 radians). This angle is bisected by the line connecting the fixed point to the prey. The shadower would then be set to point in a randomly chosen direction no more than 0.01 radians to the left or right of the prey.

(b) Selection procedure as adopted in **section 5.3**. Variants of this procedure are applied throughout the remainder of the thesis. The fixed point, which is also the initial position of the shadower, is randomly chosen to lie in the area shaded light grey. In training, the shadower attempts a camouflaged approach on the prey (the prey is moved along a predetermined trajectory). After each move made by the shadower, the control system is trained with the target motor output (i.e. the correct response for the previous time step). Each shadower approach lasts a fixed number of time steps (in the case of this chapter 100). After this, the procedure is repeated with a different prey trajectory and fixed point position.

The networks were all trained for exactly 10^6 training iterations. In online training, training pairs were selected at random from the training set. **Fig. 5.3** shows the mean of the direction error of the networks when tested on the training set and also on the test set used in **chapter 4**. As can be seen from **Fig 5.3**, all networks trained on-line were more accurate when tested on the training data set than the most accurate network trained using batch training. Every network (irrespective of the training method used) was less accurate when tested on the test set. However, the networks trained on line gave the two most accurate performances on the test set. As such these results give good evidence to suggest that for this specific problem batch training is less suitable. It is thought likely that the failure of batch training may be the consequence of a large data set and redundant data (i.e. the training set contains many similar training pairs) as suggested by Haykin (1999).

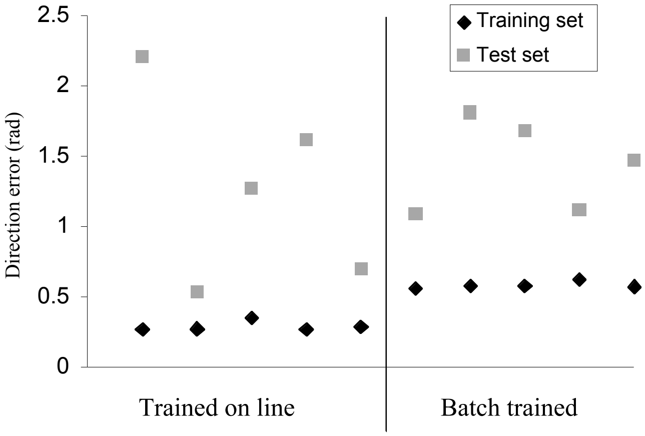


Fig 5.3 The mean direction error of the ten networks trained on-line and off-line. Black data points represent the results of tests on the training set, the grey data point above represent the results of tests on the test set for the same network.

On the basis of these results on-line training is adopted throughout the remainder of this thesis. Given that in the simulation an unlimited number of (noiseless) training data points could be generated, it was decided not to use a specific training set (as in the example above). Instead the networks were trained whilst they attempted to approach the prey using motion camouflage (see also **Fig. 5.2b**). Similar to the procedure described in **section 4.1.4**, training was undertaken by running sets of trials. Each trial lasted a fixed number of time steps. During each trial the shadower would attempt a camouflaged approach on the prey. The trajectory of the prey was predetermined prior to the trial. The initial positions of shadower and prey were randomly selected. At each training iteration the network would estimate the appropriate camouflaged move and the shadower would move accordingly. The network would then be trained with the target motor output that it ideally should have given. This process was repeated until the end of the trial. From the perspective of Backpropagation training this approach is intuitively attractive for two reasons:

- Each training example is directly relevant as the control system is responsible for getting itself into that situation. In the example above where a specific training set was used, due to possible inadequacies in the technique used to position the fixed point, shadower and prey, some of the training examples may have represented situations that the control system is never likely to be in (it is therefore less important that the control system knows how to respond in these situations). More importantly, the training set may not have accounted for situations that the control system is likely to encounter;

- The training data is adaptive in that it changes with the accuracy of the control system (this relates to the point above).

In attempt to avoid the control system becoming too finely tuned to the most recent training data (the relatively poor generalisation to the test set indicated by **Fig. 5.3** suggests that the networks trained in the experiment had overfitted the training set) a variant of cross validation is applied (see **section 3.4.3**). Periodically training is halted, the weights are held fixed and the system is tested on an alternative (validation) data set. If the network's performance is better than previous validation results, the weights of the network are recorded. Training is undertaken for a fixed number of iterations. Upon completion of training, the network that gave the most accurate results in the validation tests is selected to represent the final network state (employing this method for motion camouflage networks were never found to converge on minima whether local or global). Empirical tests suggested that this approach was more profitable than using a fixed training set. Throughout the thesis, it was always found to yield a network of respectable accuracy. Further details of the training procedure are described **section 5.2**.

5.2 Methods

Early experimentation showed that a motion camouflage controller could be implemented (operating with the same inputs and outputs to **section 4.1.2**) using a Multilayer perceptron network with a single hidden layer consisting of 10 hidden nodes. It was speculated that performance could be improved through use of two networks (receiving the same input, see **Fig 5.4**), one trained to output the motion camouflage direction, the other the rotation. Throughout the remainder of the chapter these networks will be referred to as *dual* networks. Also it was observed that the accuracy of the networks was noticeably better when the networks were tested whilst still being trained (i.e. after each time step during the test the controller was trained with the move it should have made as would be the normal procedure in training). This was thought a worthy topic of further investigation (note that throughout the remainder of the chapter networks operating conventionally are referred to as operating with fixed weights and networks undergoing constant training are referred to as operating under continuous training). So, the main questions posed by the chapter were:

- whether the Multilayer-Perceptrons performed differently to the self-organising maps;
- whether there was a difference in performance between the different Multilayer-Perceptron structures;
- whether there was a difference in performance between operation with fixed weights or during training.

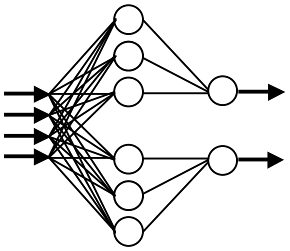


Fig 5.4. Abstract representation of the structure of the dual Multilayer Perceptrons. Both networks receive the same input but are otherwise independent. They are trained simultaneously. The key is the same as for **Fig 5.1**. Although the hidden layers in the diagram have only three nodes each, the experiments were based on networks with 10 nodes in each hidden layer.

In attempt to answer these questions 60 Multilayer Perceptrons were trained to perform the responsive algorithm. 20 of the networks had 10 nodes in the hidden layer, 20 of the networks had 20 nodes in the hidden layer and the final 20 were dual networks (see **Fig 5.4**) with 10 nodes in each hidden layer. Note that the overall number of connections in the dual networks is less than that of the normal networks with 20 nodes in the hidden layer (100 connections as opposed to 120). The networks were trained in a similar manner to **section 4** on the responsive algorithm (**section 4.1.2**) with the inputs and outputs described in **section 4.1.3** in a similar fashion to **section 4.1.4**. The only difference in the training procedure was that in training the shadower was left to autonomously decide its own movement, rather than being led along the desired trajectory (see also **Fig. 5.2b**). This strategy had been observed to hasten the Backpropagation training process and improve the final accuracy. The parameters describing the training trials are redisplayed in **Table 5.1b**. For further explanation see **section 4.3**. The network training parameters are displayed in **Table 5.1a**. The networks were trained for a fixed number of training iterations and the best performing network configuration during the training period selected as the final state to be tested. More specifically each network underwent $2*10^8$ training iterations. Every 10^4 iterations (i.e. after 10 prey trajectories of 100 steps had been trained from 10 different starting positions) the network was tested on 10 randomly selected prey trajectories. 10 trials were run against each prey trajectory from 10 random starting positions. All testing parameters were within the same ranges allowed in training (**Table 5.1b**). Network weights were fixed throughout the tests. If the mean euclidian distance between the actual output vector (direction and rotation) and the desired output vector over all 10^4 steps was less than the previous least mean, the network weight values were recorded. After training each network, the weight record that had given the least mean error during the tests in the training phase was selected for testing. None of the networks were found to converge on local or global minima.

Tables 5.1a,b Parameters defining the **(a)** Backpropagation training parameters. and **(b)** training procedure.

(a) Backpropagation training	
η	0.1
α	0.5
Initial weight range	$-1 \leq w \leq 1$

(b) Training procedure		
Number of steps in trial	100	
Shadower step size (units)	0.5	
Prey step size (units)	0.4	
Prey direction change constant per step?	Yes	
Prey direction change (per step) range (rad)	$-0.02 \leq c \leq 0.02$	Selected randomly before trial
Initial distance range of shadower from prey (units)	$50 \leq d \leq 90$	
Initial bearing of shadower from prey (rad)	$0 \leq b \leq 2\pi$	

Following training each network was tested. All networks were tested on 100 randomly selected prey trajectories. 100 trials were run against each prey trajectory from 100 random starting positions. The set of possible trial parameter values were the same as for training. Each network was tested with fixed weights and also under continuous training. The starting state of the networks tested during training was the same as when operating with fixed weights. The extra training time allowed during testing is not thought to have confounded the results in any way (tests were run with fixed weights on 3 of the networks after this extra training and none was observed to perform better than it had previously). To allow a graphical comparison with the self organising networks of **chapter 4** tests were also run on prey trajectories a, b and c (as shown in **Fig 4.4**).

5.3 Results (including comparison with self organising map)

The tests demonstrated a clear difference in performance between the self organising networks of the previous chapter and the Multilayer Perceptron of this chapter. Motion camouflage trajectories generated from a Backpropagation trained controller are displayed in **Fig 5.5a-c**. These results were obtained from a network with 10 nodes in the

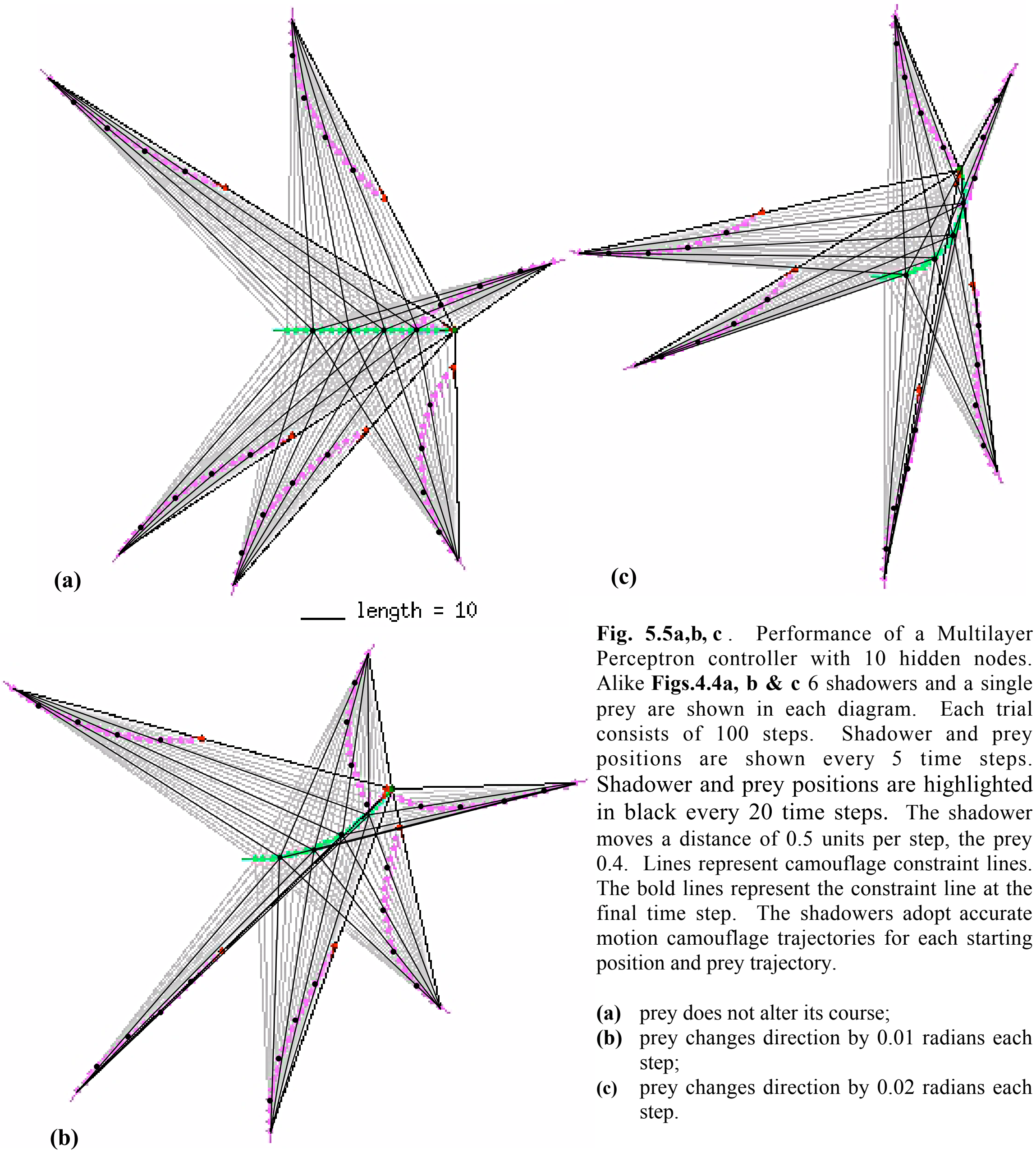


Fig. 5.5a,b, c . Performance of a Multilayer Perceptron controller with 10 hidden nodes. Alike **Figs.4.4a, b & c** 6 shadowers and a single prey are shown in each diagram. Each trial consists of 100 steps. Shadower and prey positions are shown every 5 time steps. Shadower and prey positions are highlighted in black every 20 time steps. The shadower moves a distance of 0.5 units per step, the prey 0.4. Lines represent camouflage constraint lines. The bold lines represent the constraint line at the final time step. The shadowers adopt accurate motion camouflage trajectories for each starting position and prey trajectory.

(a) prey does not alter its course;
(b) prey changes direction by 0.01 radians each step;
(c) prey changes direction by 0.02 radians each step.

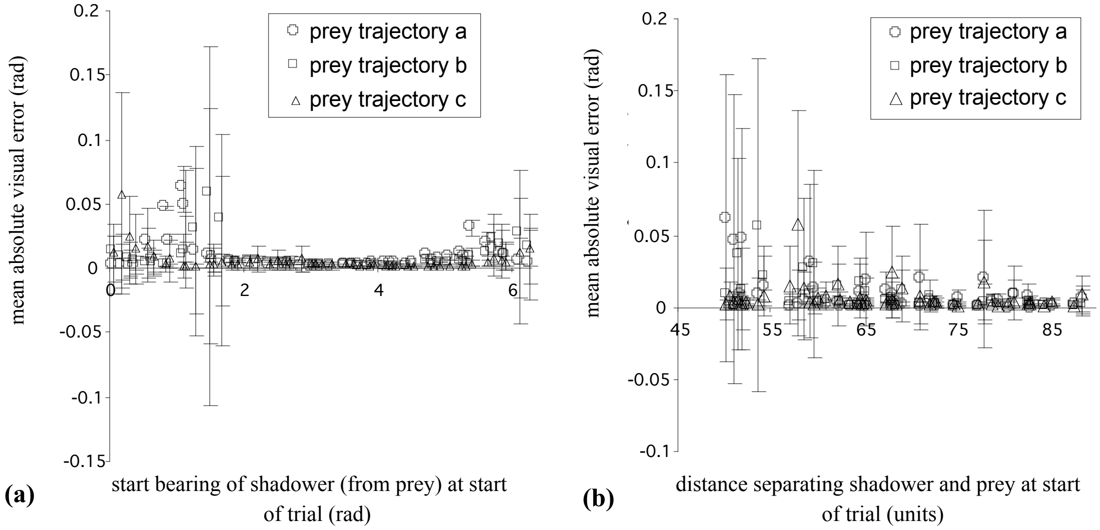


Fig. 5.6. Performance of a Multilayer Perceptron controller with 10 hidden nodes. The graphs mirror **Figs 4.5a, b**. **(a)** and **(b)** and show the change in mean +/- std. err visual error according to the initial bearing and initial distance of shadower from prey respectively. Prey trajectories a, b and c correspond to those shown in **Fig. 5.4.** Results are representative of 50 trials run for each prey trajectory. The set of initial shadower positions was randomly chosen, but the same for each prey trajectory. Visual error values were not recorded after the shadower had reached a distance within two units from the prey (when, due to geometry, small movement errors are likely to give large visual errors that will have a disproportional effect on the average).

hidden layer. On casual inspection the shadower trajectories appear to be almost faultless. In fact, as shown later in this section this network structure is the least accurate of the three investigated in the chapter. However similar diagrams for networks with 20 and (2*10) nodes in the hidden layer are virtually indistinguishable. These diagrams alone suggest that the MLPs trained with Backpropagation have given more accurate motion camouflage approaches than the self organising networks of the previous chapter (see **Fig. 4.4**). The MLP shadowers always maintain a course close to the constraint line (unlike the shadower in the top left of **Fig. 4.4a**). Also the MLP shadower trajectories appear smoother and there are no instances of the shadower overshooting the prey (as with the rightmost shadower of **Fig 4.4b**). This difference is quantified by **Figs 5.6a&b** which display the mean visual error per trial plotted against the initial bearing and the initial distance of the shadower from the prey respectively. In comparison with the corresponding graphs for the self-organising networks (**Fig 4.5a&b**) the error range can be seen to be far lesser. Ultimately use of statistical tests is unnecessary to show the superiority of the Multilayer Perceptrons over the networks of the previous chapter for motion camouflage. The worst performing network of the 60 trained in this chapter had a mean directional error (~0.26 rad) less than half that of the best sensorimotor map (~0.7

rad). The greatest mean rotational error of the MLP (~0.06rad) was also smaller than the least SOM error (~0.08rad).

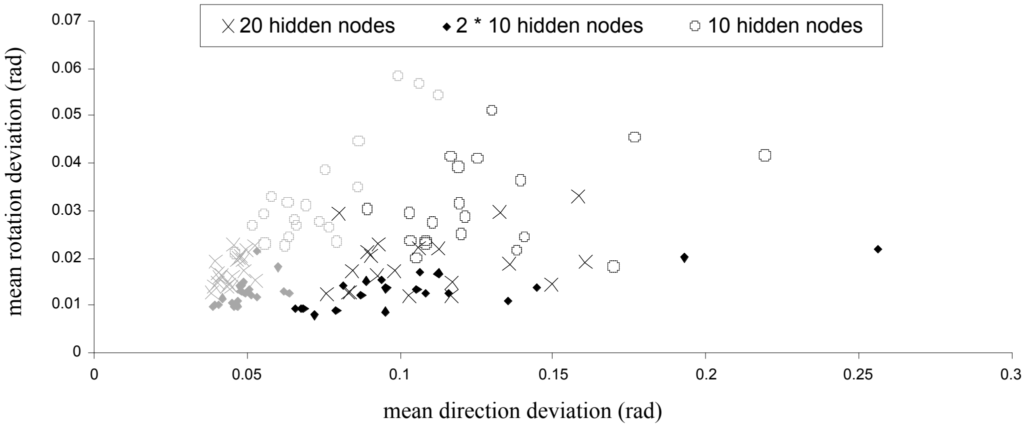
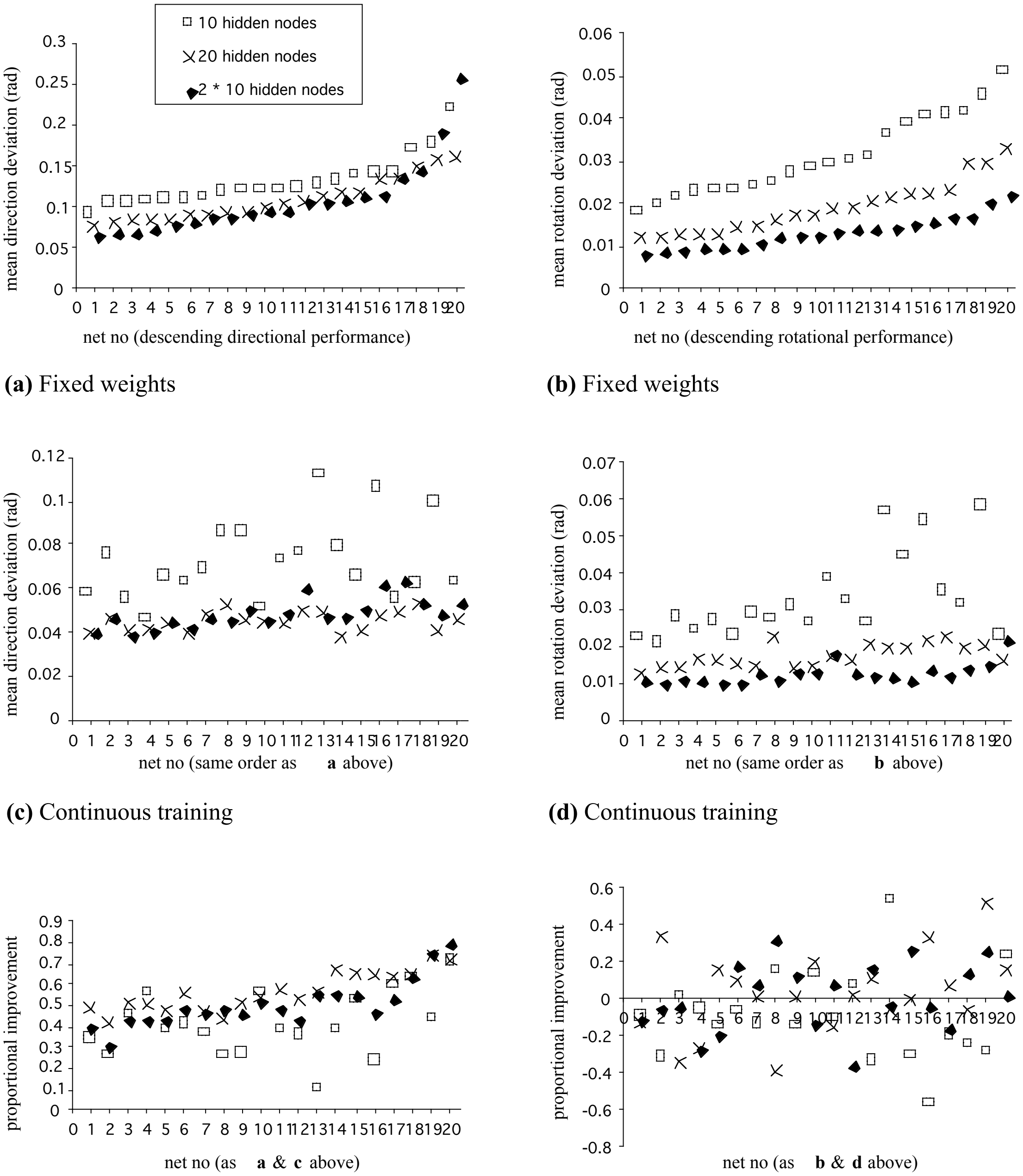


Fig 5.7. Performance of the different network structures in the two operating conditions. Mean directional error is plotted against mean rotational error. Results are shown for all 20 networks of each network structure (overall 60 nets) running either with fixed weights (black) or under continuous training (grey). Therefore in total 120 points are plotted. Results are representative of 100 trials run against each of 100 prey trajectories (as described in **section 5.3**). Each trial consists of 100 steps. At each step the prey altered its course by a constant angle chosen randomly at the start of the trial (in the range – 0.02 rad ≤ b ≥ 0.02 rad).

The next points of interest were the possibly inter-related differences in performance between the different Multilayer Perceptron structures working with fixed weights or undergoing continuous training. Mean results for each of the 60 different networks operating under both conditions are displayed in **Fig 5.7**. Note that in graph legends dual networks are indicated as having 2*10 hidden nodes. This graph indicates the following:

Operating with fixed weights (black points):

- Overall the four best performing networks were dual networks.
- There is a stronger relationship between directional and rotational performance with the dual networks (i.e. a network providing a good directional performance is likely also to give a good rotational performance). Correlations between directional and rotational means are 0.78, 0.29 and 0.33 for networks with (2*10), 20 and 10 hidden nodes respectively.



(a) Fixed weights

(b) Fixed weights

(c) Continuous training

(d) Continuous training

(e) Proportional improvement in directional mean resulting from continuous training

(f) Proportional improvement in rotational mean resulting from continuous training

Figs 5.8 a-f. Difference in mean performance between operation with fixed weights or under continuous training. **a** & **b** display the mean directional and rotational error of the different network structures running with fixed weights (i.e. the points plotted in black in **Fig 5.5**). The networks are numbered in descending order according to their mean directional (**a**) or rotational (**b**) performance (i.e. 1 is best, and net 1 in **a** is not necessarily the same as net 1 in **b**). **c** & **d** display the mean directional and rotational errors of the same networks running under continuous training (i.e. the points plotted in grey in **Fig 5.5**). Net numbers in **c** correspond to **a** and in **d** correspond to **b**. **e** & **f** show the proportional reduction in mean error (labelled as proportional improvement) in direction and rotation respectively. (If $\overline{a}$ is the mean deviation of a net operating with fixed weights and $\overline{c}$ is the mean deviation of a network operating under continuous training, the proportional improvement is given by $(\overline{a}-\overline{c})/\overline{a}$).

Operating under continuous training (grey points):

- The overall best directional performance with fixed weights is surpassed by many of the 10 hidden nodes networks undergoing continuous training.
- The least mean rotation error is slightly greater than the least with fixed weights.
- The range of mean directional errors is greatly reduced. This is most prominent with the networks having (2*10) and 20 nodes in the hidden layer.

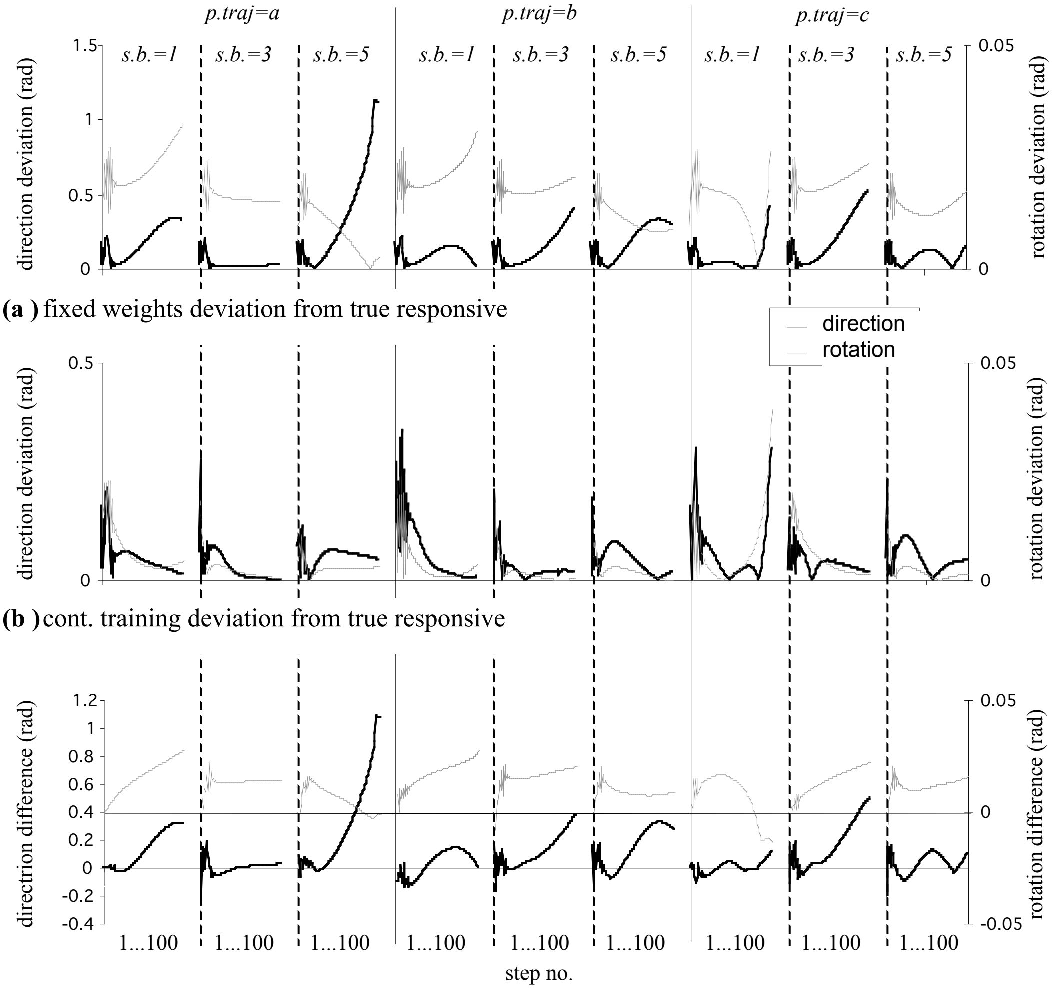
In general:

- Continuous training markedly improves the directional performance of all networks.
- Networks with ten nodes in the hidden layer tended to perform worse with respect to both directional and rotational error.

For clarity the data displayed in **Fig 5.7** is re-plotted in **Figs 5.8a-d**. These graphs help to illustrate:

- A significant difference in directional error means between the networks with 10 hidden nodes and the two other structures (confirmed statistically in **Table 5.2a**). Significance holds for both operation with fixed weights or under continuous training.
- Significant differences in rotational error means between each of the network structures (confirmed statistically in **Table 5.2b**). Significance holds for both operation with fixed weights or under continuous training.
- A significant difference in the directional error means between each network operating with fixed weights and under continuous training (confirmed statistically in **Table 5.2a**).
- No significant difference in the rotational error means between each network operating with fixed weights and under continuous training (confirmed statistically in **Table 5.2b**).

- No clear relationship between networks that perform well with fixed weights and under continuous training.



(a) fixed weights deviation from true responsive

(b) cont. training deviation from true responsive

(c) difference (fixed - c.t.)

Fig. 5.9 a – c. Difference in performance over the duration of individual trials resulting from continuous training. A dual network was used to generate the results.

Results show 3 trials run over the 3 different prey trajectories as shown and labelled in **Fig 5.3.** The bearing of the shadower from prey at the start of the trial (*s.b.*) was either 1, 3 or 5 radians. The distance separating shadower and prey at the start of the trial was always 85 units (As for all tests in this chapter shadower step size = 0.5 units, prey step size = 0.4 units). Therefore **a, b & c** each show a total of 9 trials. Note that the directional and rotational data are scaled on different axes and that the directional error scale differs between **a**, **b** and **c**.

(a) Directional and rotational error with fixed weights.
(b) Directional and rotational error whilst training.
(c) Difference between **a** and **c** at each time step.

Figs 5.8e-f display the reduction in the mean error afforded by continuous training proportional to the mean error encountered with fixed weights for each network (for directional and rotational data respectively). Whilst continuous training reduced the mean directional error mean for every network, it could either improve or degrade rotational performance. There was found to be a statistically significant difference in the error reduction between the networks with 10 hidden nodes and both of the two alternative structures (**Table 5.3a**). No such significant differences were found with the rotational data (**Table 5.3b**).

Having identified that continuous training reduced the mean directional error in the tests, it became necessary to investigate how this reduction is manifested over the duration of individual trials. **Fig. 5.9a-c** illustrate the difference in accuracy between operation with fixed weights and under continuous training over the duration of nine trials for a double network. The graphs are illustrative of 3 trials run from different starting positions for each of prey trajectories a, b and c (as displayed in **Figs. 5.5**). The graphs show that when operating with fixed weights (**Fig. 5.9a**) the directional error tended to be high during the first few steps of the trial, before abruptly dropping and then increasing over the later stages. The rotational error tended to be worse at the start of the trial, either increasing or decreasing over the later stages. Differences in the pattern of error were evident between trials. When the network was continuously trained (**Fig. 5.9b**), the patterns in directional and rotational error exhibited greater similarity to each other. Both directional and rotational errors were highest at the start of the trial, tending to diminish towards the end. It would appear to be over these later stages that continuous training offers directional improvement (**Fig. 5.9c**). At the start of the trial continuous training often seems to slightly degrade directional performance. In this example rotational performance was almost always improved by continuous training, however attention is drawn back to **Fig. 5.8f** which shows that this is not always the case.

Table 5.2. Scheirer-Ray-Hare (non-parametric equivalent of two-way ANOVA) tests investigating the difference in directional (**a**) and rotational (**b**) error between the three different network structures operating with both fixed weights and under continuous training. See also **Figs 5.7 a – d**. The mean directional or rotational error was calculated from a total of 10,000 trials per network (as described in **section 5.3**). Each trial had 100 steps (20 networks for each network structure. Therefore overall there were 6 samples consisting of 20 replicates.

(a) results indicate a highly significant difference in directional error between the different network structures and also between operation with fixed weights and under continuous training. It was found using Mann-Whitney tests that there was no significant difference between networks with (2*10) and 20 hidden nodes. Fixed weights P = 0.478, continuous training P = 0.086. However both were significantly different from the network with 10 hidden nodes. Fixed weights (2*10) vs 10 hidden nodes P = 0.005, 10 vs 20 hidden nodes P = 0.017. With continuous training both P < 0.001).

(b) results indicate a highly significant difference in rotational error between the different network structures. There is no significant difference between operation with fixed weights or under continuous training. It was found using Mann-Whitney tests that all three network structures performed significantly differently to one another (with both fixed weights and continuous training). All P-values < 0.002.

(a)

Source	*df*	*MS*	*H*	*P*
F or c.t.	1	92852	76.7372	< 0.00001
Net struct	2	8433.77	13.9401	< 0.001
Interaction	2	1329.36	2.19729	0.33
Residual	114	277.296		
Total	119			

(b)

Source	*df*	*MS*	*H*	*P*
F or c.t.	1	128.13	0.1059	0.75
Net struct	2	49315	81.513	< 0.00001
Interaction	2	170.33	0.2815	0.85
Residual	114	393.78		
Total	119			

Table 5.3. One-way ANOVAs investigating for a difference in proportional mean directional (**a**) and (**b**) rotational improvement offered by continuous training for each of the three network structures. See also **Fig 5.7e,f** for a plot of the data and definition of proportional improvement. Results generated from the same trials as **Table 5.2a,b**. Overall there were three samples consisting of 20 replicates. Levene's test showed that the variances of the data were suitably homogenous in both cases.

(a) results indicate a highly significant difference in the proportional mean directional improvement for different network structures. Post hoc LSD tests showed that there was no significant difference between networks with (2*10) hidden nodes and 20 hidden nodes (P = 0.153). Both were significantly different from networks with 10 hidden nodes. (2*10) hidden nodes vs 10 hidden nodes P = 0.019. 20 hidden nodes vs 10 hidden nodes P < 0.001.

(b) there was no significant difference in proportional mean rotational improvement between any of the network structures.

(a)

Source	*df*	*MS*	*F*	*P*
Net struct	2	0.113	7.634	0.001
Residual	57	0.01481		
Total	59			

(b)

Source	*df*	*MS*	*F*	*P*
Net struct	2	0.0821	1.712	0.19
Residual	57	0.048		
Total	59			

5.4 Discussion

The results obtained using Backpropagation were more accurate whilst requiring far fewer computational units than the self-organising networks of **chapter 4**. Therefore it can be stated with confidence that Backpropagation is more suitable for application to motion camouflage. Furthermore the progression of the Backpropagation networks to the more challenging problem of making predictions (addressed in **chapter 6**) is also likely to be less computationally demanding than for the self organising networks. In explanation, to make predictions the control system is likely to require knowledge of the recent chase history (see **section 2.2.4**). The most simple method for supplying this information would be to present previous sensory information as input. As the self-organising network has a far larger hidden layer, more additional connections would be necessary, suggesting that the computation cost would be higher than for the MLPs.

The dual network architectures were found on average, to give the least rotation error means. Also in tests with fixed weights the dual networks produced the four least direction error means (coupled with rotation error means less than any of those produced from alternative network structures). This suggests that there is a greater chance of obtaining high motion camouflage accuracy with the dual networks than with the alternative structures (using the training method and parameters described in **section 5.2**). Also the test results suggested that a high direction accuracy was more likely to be paired with high rotation accuracy with the dual networks controllers.

By continuously training the networks it was possible to dramatically reduce the movement direction error but not the rotation error. This poses two questions:

- Why did continuous training reduce the direction error?
- Why did continuous training not have the same effect on rotation?

With the regular prey trajectories used, each set of inputs were similar to the previous set. Likewise each set of consecutive target outputs were similar to the previous outputs. (**Fig. 5.9a** shows the direction outputs and **Fig. 5.9b** the difference between consecutive direction outputs over the motion camouflage trajectories shown in **Fig. 5.9e**). It is therefore thought that by always adapting to the immediate situation (i.e. the previous step) the networks could have improved their performance. This is consistent with the general pattern of reduced directional error towards the end of the trials and the relatively high error at the start of trials as seen in **Fig. 5.9c**. The poor performance at the beginning of trials (excluding the first) could be explained by the network having adapted to the different conditions at the end of the previous trial. One possible reason that continuous training gave this advantage is that the fixed weight networks contained too few nodes to accurately model the problem. Supporting this the performance of the 20 hidden node networks was markedly better than that of the 10 hidden nodes networks. Alternatively, if in fact there were sufficient nodes, it is implied that the training procedure and parameters were not conducive to finding the global error minimum.

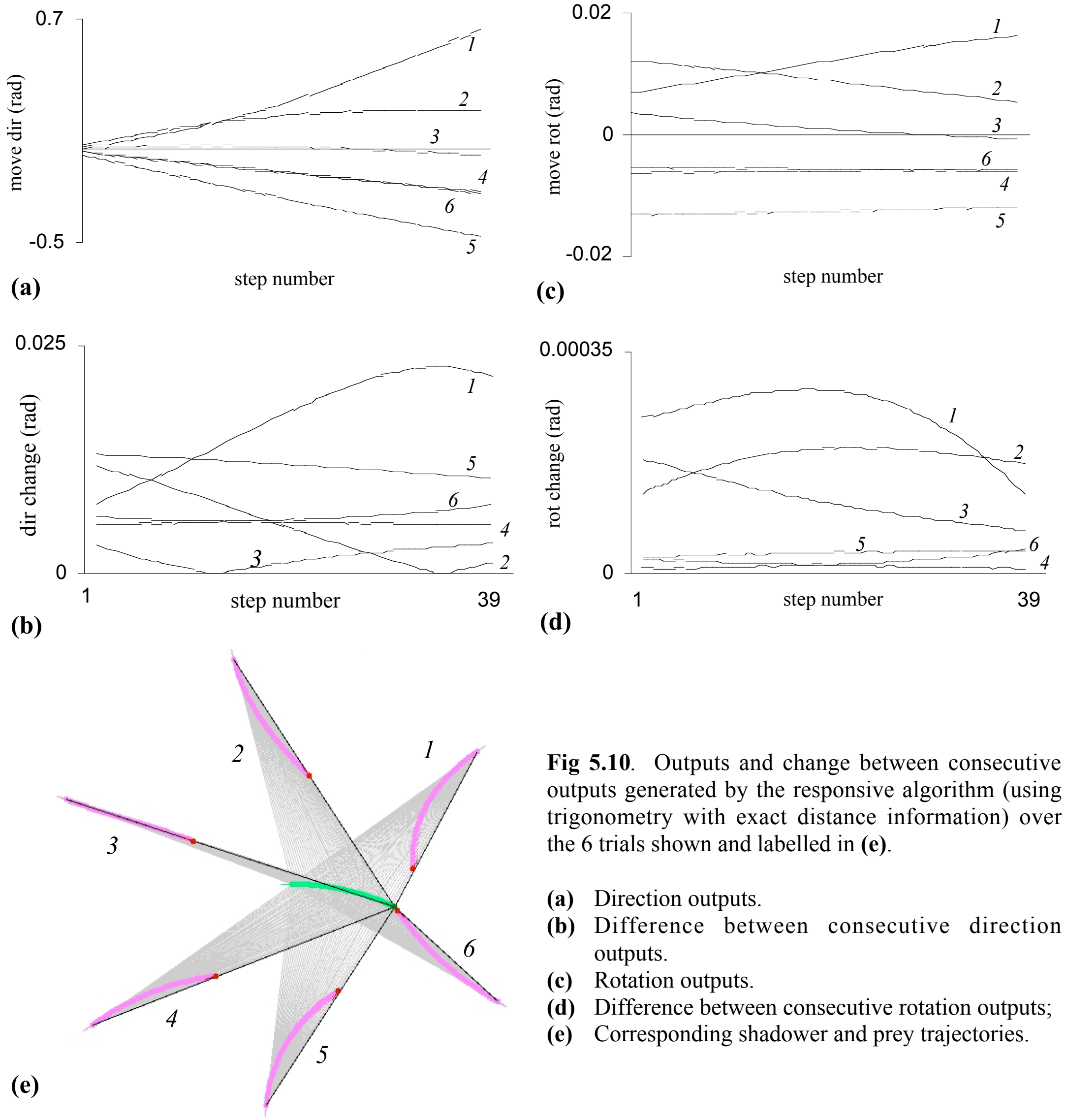


Fig 5.10. Outputs and change between consecutive outputs generated by the responsive algorithm (using trigonometry with exact distance information) over the 6 trials shown and labelled in **(e)**.

- **(a)** Direction outputs.
- **(b)** Difference between consecutive direction outputs.
- **(c)** Rotation outputs.
- **(d)** Difference between consecutive rotation outputs;
- **(e)** Corresponding shadower and prey trajectories.

If this explanation for improved direction performance is correct, then why was rotational performance not improved? The rotation outputs and differences between consecutive rotation outputs corresponding to the trajectories of **Fig. 5.10e** are shown in **Fig. 5.10c** and **d** respectively. Inspection of the output differences of **Fig. 5.10c** gives no obvious reason to expect that rotation should behave any differently to direction. However, it can be seen from **Fig 5.10c** that the rotation output over the duration of a trial does not necessarily follow a similar pattern to the direction output. This implies that the rotation and direction error surfaces are different (as would be expected). Therefore it is suggested that this difference may involve the solutions to local direction problems being closer to each other (or at least easier to reach from each other using on-line training) than is the case for rotation. Alternatively it is possible that the failure in rotational improvement could result from choice of unsuitable training parameters. However, casual experimentation involving alterations of learning rate (η) and momentum (α) terms failed to yield a rotation performance better than the best achieved with fixed weights.

Regardless, to be able to benefit from continuous training a controller would have to have the ability to teach itself autonomously. Although this requirement is somewhat contradictory to a supervised learning technique, it is thought that it may be possible to train an external network to act as teacher. After each movement this teaching network would be required to recognise what would have been the appropriate move and use this to train the motion camouflage network. In this case measures of self error (the input to the teaching network) could be the degree to which the prey and fixed point are non central in the shadowers front and behind view (ideally if the prey were directly ahead and the fixed point directly behind each of the shadower's four visual input angles would be equal). However, in later chapters where the shadowers vision of the fixed point is removed the only error measure would be the direction of the prey and the shadower would have to assume that it is facing directly away from the fixed point. It is unclear whether this 'unsupervised-supervised' approach would offer any real advantage over supplying the motion camouflage network with the additional computational power required for the teaching network. Due to the improvements in performance observed from increasing the network size (see **chapter 6**) continuous training is not further investigated in the remainder of this thesis.

6 TOWARDS AN AUTONOMOUS MOTION CAMOUFLAGE CONTROL SYSTEM

In the previous chapters the sensory input received by the control system explicitly gave it its exact position relative to the prey and the fixed point. In a real situation the shadower would be unlikely to have access to such accurate information. It is also unlikely that the shadower would be able to see both prey and fixed point simultaneously (especially if the fixed point does not physically exist, see **chapter 2**). As the shadower must concentrate on viewing the prey, it has no alternative but to use dead reckoning to estimate the position of the fixed point. Also, the binocular vision previously afforded the shadower gave it more accurate depth information than would be expected in a real world situation (as discussed in **section 3.2.2** binocular vision tends to be inaccurate for judging the depth of distant objects). Beyond this, if the shadower could have such accurate depth vision, then so could the prey. The prey would consequently be less likely to be fooled by camouflaged approaches as it would be able to spot looming.

In this chapter the amount of spatial information provided by vision is reduced to a level resembling that which an insect or robot could retrieve. Additionally, the control systems are trained to perform the predictive algorithm, tracking prey moving along real hoverfly flight paths as well as the regular trajectories of previous chapters. Again the chapter is organised in the form of an experimental write-up.

6.1 Methods

This section lists the control system inputs, outlines the design of the control systems, describes the prey trajectories used in training and testing and finally details the experimental procedure where it differs from that of previous chapters.

6.1.1 Control system inputs and outputs

As previously discussed, for motion camouflage to be successful, the prey must not be able to sense any difference in shadower range as the shadower approaches. This implies that the shadower must be at such a distance that the prey can not make use of binocular depth cues (e.g. binocular stereopsis, convergence, accommodation see **section 3.2.2**) to judge the distance of the shadower. Equally the shadower would be unlikely to be able to use these techniques to estimate prey distance. Therefore in this chapter the shadower's sensory input is reduced to:

- the direction of the prey;
- the image motion of the prey (i.e. the angle subtended at the shadower's retina by the present and previous prey position).

To enable prediction the shadower is also afforded a short memory of recent prey image motion and in order to interpret this a memory of its own recent movement. The shadower is expected to estimate the distance to the fixed point based on the memory of its movement (**section 3.1.2** saw that this task is not beyond the wandering spider *Cupiennius* which is capable of dead reckoning based entirely upon signals from mechanical stress sensors located on its legs).

Similar to **chapters 4** and **5** the controller outputs the direction to move and the rotation about which to turn. As before the shadower is trained to rotate so as always to view the prey frontally and keep the fixed point directly behind. As the shadower no longer receives any external input giving the direction of the fixed point, the accuracy of the rotation becomes more important. If the shadower rotates accurately it can assume that the fixed point is directly behind it, leaving only the distance to be estimated. The shadower step size is held constant and modelled implicitly by the system.

6.1.2 Control system architecture

In light of the results of **chapter 5** where it was found advantageous to split the control system into separate networks corresponding to the outputs, it was decided to form the control system from three Multilayer Perceptrons (**Fig. 6.1**). The first network is trained to output the distance to the fixed point. The second to output the direction in which to move and the third the rotation about which to turn.

Before making a movement, the controller is provided with the direction to and image motion of the prey. This information, combined with:

- a memory of the prey image motion over the previous two time steps;
- a memory of the previous three movements made by the shadower;
- and the previous estimate of fixed point distance

is given to the distance network. This network estimates the distance to the fixed point. The new distance estimate is passed to the direction and rotation networks (and fed back to form part of the next input to the distance network). Otherwise their input is identical to that of the distance network. The direction and rotation of movement are then decided and output. These are also stored in memory and subsequently fed back as input to all networks.

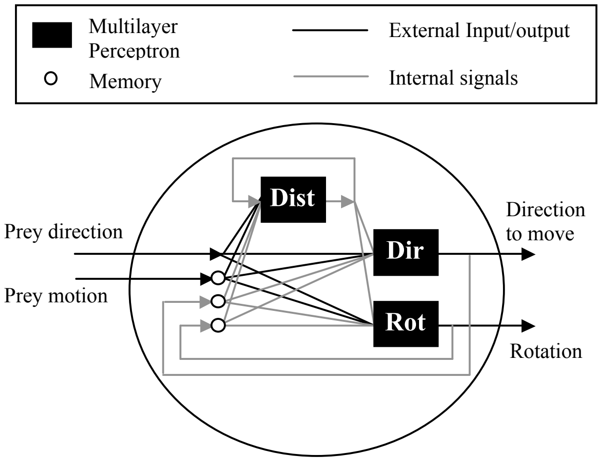


Fig. 6.1. Control system architecture. See text for explanation.

Chapter 5 has seen that the larger 20 node networks were more accurate than the smaller nets and preliminary empirical tests suggested that it would be beneficial to increase the number of layers in the networks. In accordance, in this chapter, each network was comprised of two hidden layers, the first consisting of 20 nodes, the second 10. The networks were trained with an adaption rate of 0.1 and a momentum term of 0.5.

6.1.3 Prey trajectories

Two types of prey trajectory were used to train and test the networks. The first (as in the previous chapters) were automatically generated regular arcs. Here the prey moved a constant distance and changed direction by a constant angle within the range –0.4 to 0.4 radians at each time step. These trajectories were therefore totally predictable. In training the direction change of the prey was selected randomly at the start of each trial. In testing the direction changes were selected to span the range uniformly (i.e. if 5 trajectories were selected for testing the corresponding set of direction changes would be {-0.04, -0.02, 0, 0.02, 0.04}). To enable the shadower to get closer to the prey with each step, the distance moved by the prey at each time step was set to 9/10 that of the shadower.

The second set of trajectories were the digitised flight paths of real hoverflies. 3 hoverflies of species *Episyrphus balteatus* (see **Fig 1.1**) were captured and used to generate the experimental trajectories. The hoverflies were individually placed in a box (1m * 1m * 30cm (height)) covered with perspex and filmed from above. The camera was located approximately 70 cm above the box. The camera recorded 24 frames per second. The footage was digitised and the positions of the hoverflies over their flights manually segmented and transformed in to sequences of 2D co-ordinates. In total the footage yielded 91 sequences, each consisting of 40 frames.

The step sizes of shadower and prey were increased to 5 and 4.5 units respectively (from values of 0.5 and 0.4 units in previous chapters). The reason for this was to exaggerate the difference in visual error between a shadower following a responsive approach and a shadower following a totally camouflaged trajectory. In explanation, it was assumed that the best camouflaged approach a shadower incapable of prediction could adopt would be a responsive approach. As such a shadower capable of prediction would be expected to have a visual error less than that of a responsive shadower. However the smaller the shadower's step size, the closer a responsive approach to perfect camouflage and the less the difference between the two approaches in visual error (and the less the benefit from making predictions). Therefore increasing the step size not only served to increase the importance of prediction to maintaining camouflage but also made it easier to distinguish whether or not the shadowers were making accurate predictions.

6.1.4 Experimental procedure

Beyond successfully training a controller, the following areas were identified as being of particular interest:

- Whether it is possible to improve motion camouflage through prediction.
- Whether the control systems are able to perform on the alternative trajectory type to those used in training.

Overall, 6 controllers were trained with the predictive algorithm (in the manner described in **section 4.1.4**, parameters used in training and testing are displayed in **Table. 6.1**). Three were trained on the hoverfly training data and the remaining three on automatically generated regular trajectories.

Table. 6.1. Training and testing parameters

Number of steps moved per trial	39
Shadower step size (units)	5
Prey step size (units)	4.5
Initial distance of shadower from prey (units)	$200 \leq d \leq 400$
Initial bearing of shadower from prey (rad)	$0 \leq b \leq 2\pi$

Similar to **section 5.2** each control system was trained for a fixed number of iterations. The training algorithm resembled that of **section 4.1.2** except that the shadower was trained to move to the future constraint line (the position of which was calculated *aposteriori*). During training, tests were performed and a record kept of the weight configuration of the best performing state of the controller. Specifically, trials were run from 10 randomly selected starting positions on ten different prey trajectories (in the training set). Accuracy was measured by the mean directional error. On completion of training, the best performing state of the controller found during training was selected as the final state (it had been found that the state of the control system after the last training iteration was not necessarily the best).

Following training, each control system was more extensively tested. 100 trials were run from randomly selected starting positions on each of 16 prey trajectories. For both trajectory types, the controllers with the least mean directional error were selected for further testing. These two also gave the least rotational and visual error means (respective to test trajectory type). These were then tested on the alternative trajectory type to those trained (i.e. the controller trained on hoverfly flight paths was tested on regular trajectories and *vice versa*). Performance was compared to that obtained with the responsive algorithm (calculated using trigonometry from exact distances).

6.2 Results

Motion camouflage trajectories generated from the controllers are displayed in **Fig. 6.2**. The performance appears to be very accurate, with the shadower able to successfully approach from different starting positions. Camouflage is almost perfect on the regular trajectories, and although the shadower has more difficulty with the erratic movement of

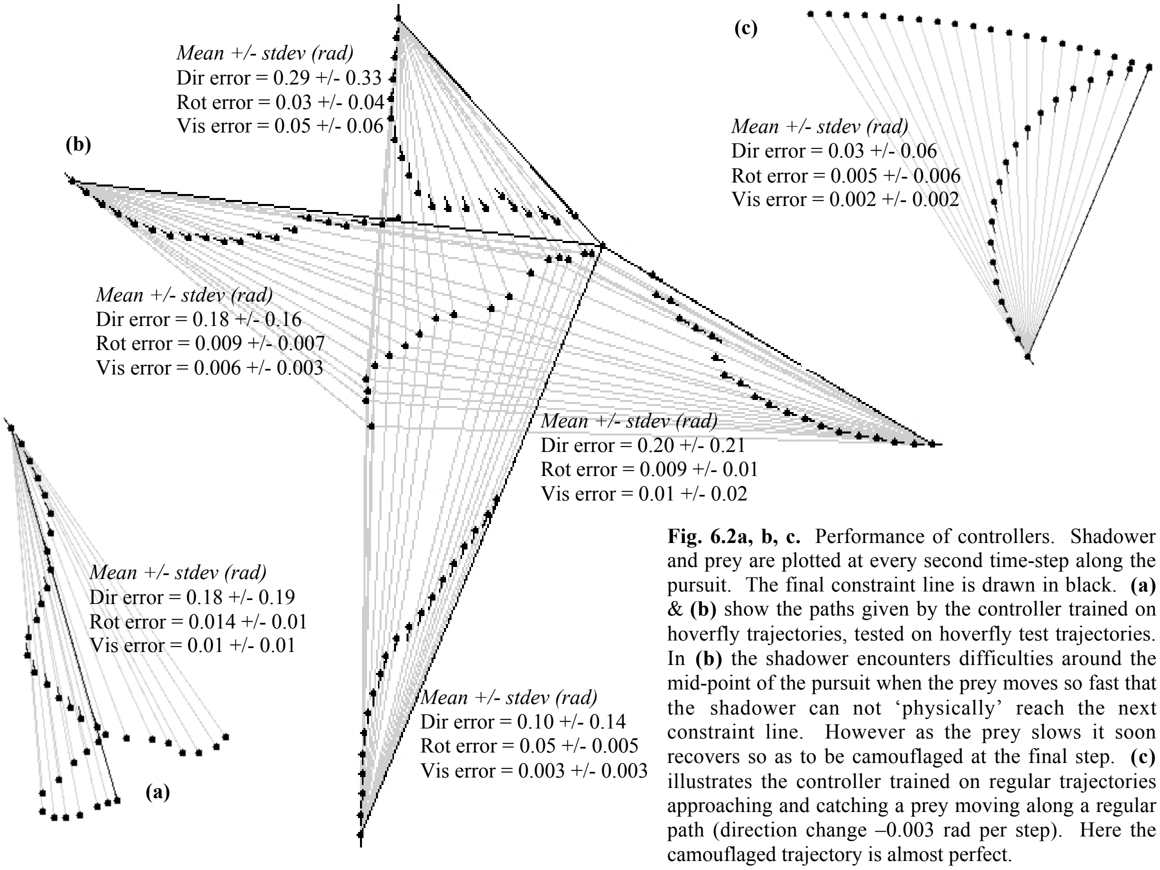


Fig. 6.2a, b, c. Performance of controllers. Shadower and prey are plotted at every second time-step along the pursuit. The final constraint line is drawn in black. **(a)** & **(b)** show the paths given by the controller trained on hoverfly trajectories, tested on hoverfly test trajectories. In **(b)** the shadower encounters difficulties around the mid-point of the pursuit when the prey moves so fast that the shadower can not 'physically' reach the next constraint line. However as the prey slows it soon recovers so as to be camouflaged at the final step. **(c)** illustrates the controller trained on regular trajectories approaching and catching a prey moving along a regular path (direction change –0.003 rad per step). Here the camouflaged trajectory is almost perfect.

the hoverfly, it is able to keep to the constraint line remarkably well (especially considering that the hoverfly occasionally moves so fast that it is impossible for the shadower to actually reach the constraint line e.g. **Fig. 6.2b**).

Fig. 6.3 compares the **(a)** visual, **(b)** directional and **(c)** rotational errors of each controller on each test. The mean errors are reduced from those recorded in **section 5**, suggesting that the increase in network size has improved performance. As might be expected, for each test trajectory type the least errors were given by the controller trained on similar data (i.e. the controller trained on hoverfly trajectories performed better than anything else on the hoverfly test data). Notably, the errors were also less than those encountered from use of the responsive algorithm, indicating that prediction had improved camouflage in all respects. All differences mentioned in this section are supported statistically (**Table. 6.2**).

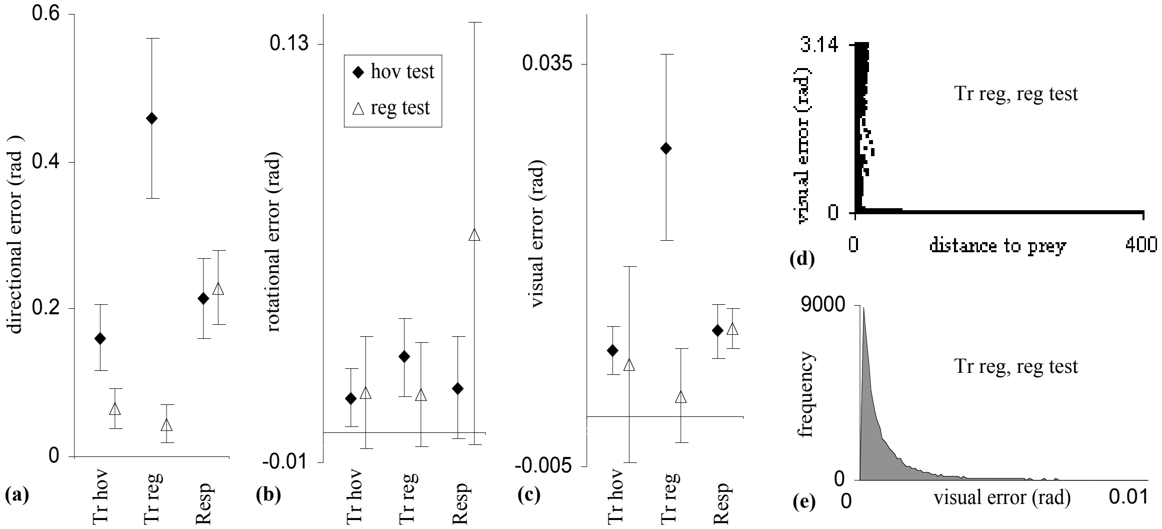


Fig. 6.3a-c. Comparison of the mean +/ stdev **(a)** directional, **(b)** rotational and **(c)** visual errors of the different controllers. Standard deviations are scaled by **(a)** 0.2, **(b)** 0.2, **(c)** 0.1 for the purposes of presentation. Each result is representative of 100 trials of 39 steps run against each of 16 prey trajectories. The high rotational errors of the responsive algorithm were incurred when the shadower was very close to the prey (and rotated to view the previous prey position). This happened more frequently with the regular trajectories when shadower and prey were more likely to move directly toward one another and meet. In **(c)** the visual error was not recorded after the shadower was ≤ 10 units distance from the prey. Beyond this distance, any slight directional error could incur a very large visual error, exerting a disproportionate influence on the mean and standard deviation. **(d)** shows the relationship between the distance separating shadower and prey and visual error for the controller trained on regular trajectories, tested on regular trajectories. As can be seen, there are only large errors when the shadower is close to the prey. **(e)** shows the frequency distribution of visual errors for **(d)**. The vast majority of errors are below 0.005 rad. 3427 (of a total of 62400) entries are above 0.01 rad and not shown on the plot.

The other area of interest was how well the control systems performed on the alternative data type to that on which they had been trained. It was found that the hoverfly trained control system performed well on the regular test set. The performance was better in all respects than the responsive algorithm, indicating that the controller was capable of making accurate predictions. The opposite was not true. The control system trained on regular data performed poorly on the hoverfly trajectories and far less accurately than the responsive algorithm. The success of the hoverfly trained controller suggests that many of the predictable characteristics of the regular trajectories were also present in the hoverfly trajectories. It was thought that the poor performance of the regular control system on hoverfly data could be indicative of a difference in dependency on inputs between controller types. For instance, in the regular trajectories the prey never changed speed or abruptly changed direction. Subsequent camouflaged moves were therefore always very similar. It is possible that the shadower could have become too reliant on its previous moves to determine its next to reliably track the hoverfly.

Table.6.2 a, b, c. Scheirer-Ray-Hare (non-parametric equivalent of two-way ANOVA) tests investigating the difference in **(a)** directional, **(b)** rotational and **(c)** visual errors (recorded only when the shadower was at a distance ≥ 10 units from the prey. See **Fig 6.3** for explanation) between the different controllers (and responsive algorithm). Replicates were the mean error per trial (16 prey trajectories with 100 trials per prey trajectory). In each case highly significant differences were found between controllers and between test trajectory types. There was also a highly significant interaction, a consequence of the poor performance on hoverfly test data of the controller trained with regular trajectories, contrasting to the improved performance of the hoverfly trained controller on regular trajectories.

It was found through exhaustive use of Mann Whitney-tests (and z -tests in the cases of testing for a difference between each test trajectory type in directional and visual errors incurred by the responsive algorithm, where variances met test assumptions) that there were highly significant differences (all $P \leq 0.0001$) between each possible combination of main effects but one: There was no significant difference between test trajectory types in the visual error of the responsive algorithm ($P = 0.75$).

(a) Directional error

	df			
Contr	2	16.91	1134.42	< 0.0001
Test traj	1	64.78	2175.93	< 0.0001
Interaction	2	39.62	2657.98	< 0.0001
Residual	9594	0.011		
Total	9599			

(b) Rotational error

	df			
Contr	2	0.7	122.28	< 0.0001
Test traj	1	0.44	76.86	< 0.0001
Interaction	2	0.91	158.96	< 0.0001
Residual	9594	0.005		
Total	9599			

(c) Visual error

	df			
Contr	2	0.08	58.78	< 0.0001
Test traj	1	0.04	14.69	< 0.0001
Interaction	2	0.21	154.29	< 0.0001
Residual	9594	0.003		
Total	9599			

6.3 Discussion

In this chapter it has been seen that accurate camouflaged approaches can be calculated with simple inputs and that performance can be improved through prediction.

What is especially interesting is that the control system has been able to gain an adequate concept of prey depth from its inputs (see also **Fig 2.4c**). There are a variety of clues that the control system has available for this estimation:

- It knows that it will be starting its approach from within a certain distance range of the prey (similarly as discussed in **section 3.1.1** it has been suggested that male hoverflies computing interception courses make the assumption that they will spot the female at a given distance (Collett & Land 1978).
- The further it is from the fixed point the closer it is likely to be to the prey.
- The difference between the previous rotation and prey image motion may provide distance information in a manner similar to that in which peering enables animals such as praying mantids to estimate target distances (see **section 3.2.2)**. Praying mantids are able to accurately estimate the depth of stationary targets from the motion of the target's image encountered as they move their heads from side to side (Kral 1998). Ideally as the shadower always faces its prey there will be no image motion. But the corrective yaw necessary to keep the shadower facing the prey (as the shadower itself moves) will be identical to the image motion should the shadower not have rotated (any prey image motion is therefore indicative of error). However, in contrast to the praying mantis, whose target is normally stationary, both shadower and prey are moving, consequently making depth estimation more approximate. Image motion alone can only reveal whether the prey is moving to the left or right. In practice, at large distances knowing this may be sufficient. The lateral component of shadower movement will always be relatively small and as long as the shadower moves in roughly the right direction, the prey is unlikely to notice. As the shadower approaches, its movement must be increasingly accurate to maintain camouflage. Then, to be useful, image motion must be combined with additional knowledge.
- Lastly, the shadower's mistakes may even be helpful in allowing it to infer its distance from the prey. The control system's only measure of its error is the extent to which the prey is non-central in its viewpoint (which is useful if the shadower is indeed facing away from the fixed point). Geometry dictates that the magnitude of

this error measure will on average be greater for a similar mistake in movement the closer the shadower is to the prey. Therefore in general, the greater the error, the closer the shadower to the prey.

That the control systems tended to specialise in tracking similar prey movement patterns to those on which they had been trained was not especially surprising. It is noteworthy that the control systems trained on the hoverfly trajectories should be able to generalise so well to tracking the artificial trajectories. This suggests that many of the predictable characteristics of the artificial trajectories were present (on occasion) in the hoverfly flight paths. The implications of this specialisation are:

- in nature, if motion camouflage shadowers do exist, they are likely to be specialist in certain prey;
- artificially speaking, motion camouflage control systems are likely to have to be tailored to particular prey.

Conversely, returning to a point raised by Srinivasan & Davey (1995) of what counter strategies a prey could use to combat the shadowers camouflage, the prey also may need to be specialist in the avoidance of its potential predators.

7 3D CONTROL SYSTEMS WITH MINIMAL SENSORY INPUT

Chapter 6 demonstrated that it was possible to implement a predictive motion camouflage control system operating using realistic levels of input in a 2D simulation. This chapter extends the simulation to 3-Dimensions. As a secondary area of interest control systems are trained and tested with and without knowledge of prey image motion in order to assess its value to correct calculation of outputs. The format of this chapter reflects recent chapters and takes the form of an experimental report.

7.1 Methods

7.1.1 3D Control system design

The design of the control systems follows that of **section 6.1.2**. Each controller comprises three Multilayer Perceptrons. As before the first of these networks is responsible for estimating the distance to the fixed point. The second, the direction to move and the third, the rotation. The control system receives the following sensory input (as described in **section 7.4** half of the control systems do not receive image motion, the architecture of the non-image motion system is shown in **Fig 7.1**).

- The direction of the prey, represented as two angles in a spherical coordinate system relative to the shadower as origin (see **Fig. 7.2**);
- The image motion of the prey over the previous 3 time steps, represented as the difference in the azimuth and difference in elevation of the prey between consecutive time steps.

And the following internally generated signals:

- Feedback of the previous direction and rotation output, each represented using two angles as for the prey direction.

- An estimate of the distance from the fixed point. At each time step the distance is estimated prior to the movement, and the new distance estimate fed directly to the direction and rotation networks, as well as back to the distance network, for the next distance estimate (the initial input to the distance network is 0).

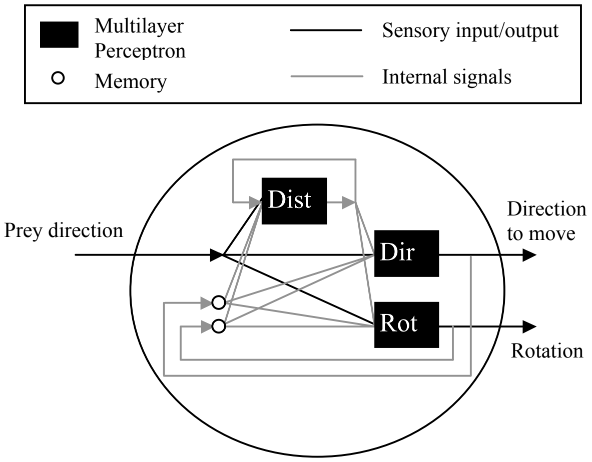


Fig. 7.1. Non-image motion control system architecture.

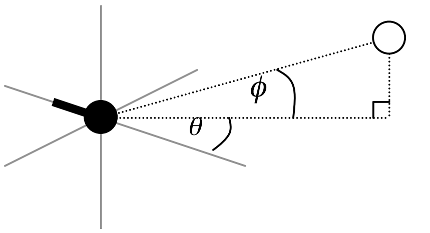


Fig. 7.2. Representation of prey direction, using azimuth θ and elevation ϕ. Direction of movement and rotations are similarly represented using azimuth and elevation.

As in **chapter 6**, each of the networks is formed from 2 hidden layers, this time the first hidden layer consisting of 40 nodes, the second 25. Each node uses the logistic activation function. Training was undertaken on line using the standard Backpropagation algorithm, with a momentum term of 0.1 and a learning rate of 0.05. All parameters were selected empirically. There was not observed to be any noticeable improvement in performance or training time, through use of alternative activation functions (e.g. hyperbolic tangent, arc tangent, natural log).

7.1.2 Prey trajectories

Control systems were trained and tested on two types of prey trajectory, both artificially generated:

Regular trajectories (resembling the 2D regular trajectories), where the prey changed direction by a constant angle at each step. The change in the azimuth and elevation of movement direction were independent and randomly determined before the trial. Step size was held constant at 4.5 units as in **chapter 6**.

Stochastic trajectories, generated as a three dimensional substitute for the hoverfly trajectories (which could only be filmed in 2D). The movement direction change at each step was controlled by a fraction of the previous direction change plus a random component. Step size was dependant upon the current direction change. Equations (7.1) and (7.2) below formally define the direction change and step, where $\Delta\theta_t$ and $\Delta\phi_t$ are the change in azimuth and elevation respectively (spherical coordinate system with prey as origin) at time *t* and r_t is the step size.

$$\Delta\theta_t = \frac{\Delta\theta_{t-1}}{p} + m \cdot z_t\,,\ \Delta\phi_t = \frac{\Delta\phi_{t-1}}{p} + m \cdot z_t \tag{7.1}$$

z_t is a random deviate (measured in radians) picked from a normal distribution with mean 0 and variance 1. z_t is selected independently for $\Delta\theta_t$ and .$\Delta\phi_t$ (i.e. the random component is not the same for azimuth and elevation). *p* and *m* are terms included to modify the influence of the previous direction, and randomly chosen component on the new direction. In the simulation $p = 2$ and $m = 1/5$.

$$r_t = 2.5 + 5 \cdot \left(\frac{m}{m + |\Delta\phi_t| + |\Delta\theta_t|} \right) \tag{7.2}$$

Thus r_t is limited to lie within the range 2.5 to 7.5 units, with a mean of approximately 4.5 units (in keeping with the other prey trajectories in this chapter and **chapter 6**). All prey trajectories were limited to 40 steps and in both cases the step size of the shadower was held constant at 5 units (just greater than that of the prey).

7.1.3 Experimental procedure

The experimental procedure was designed to investigate the following areas:

- Whether provision of image motion as input offers any advantages;
- Whether the 3D control systems were able to make successful predictions;
- Whether the 3D control systems were able to generalise to the alternative prey trajectory type to those used in training.

In total, 12 control systems were trained. 6 were trained on regular prey trajectories and 6 on stochastic trajectories. In order to investigate the value of prey image motion to the calculation, half of the control systems in each group were trained with image motion as input and half without. From each group a single control system was selected for further testing. The training process and selection procedure are similar to those of **section 6.1.4** and are described below. Note that in the 3D tests, direction error was measured in terms of azimuth and elevation components.

The controllers were trained according to the procedure described in **section 4.1.4**. The control systems were trained for a fixed number of iterations ($5*10^8$). Throughout training their performance was tested, and a record kept of the best performing control system over the training period. The tests consisted of 100 trials, 10 trials run against each of 10 different prey trajectories. Testing parameters (e.g. initial distances and bearings of shadower from prey) were selected randomly before each test. Performance was measured using the mean azimuth movement direction error. If the current state of the control system gave a lesser error than any of the previous, a second test was undertaken. If the error was still smaller, the record of the previous best controller was replaced with the current. On completion of training this record was selected to be the final state of the control system.

In order to determine which control system from each group to retain, more rigorous testing was undertaken. Each control system was tested on 10,000 trials, 100 trials run against 100 prey trajectories (of similar type to those used in training). From each group, the control system yielding the least error was then tested on the alternative prey trajectory type to that on which it had been trained. To allow testing for evidence of prediction, corresponding results for both prey trajectory types were generated using the responsive algorithm (calculated from exact positions using trigonometry).

7.2 Results

Camouflaged trajectories generated by the 3D-control systems are shown in **Fig. 7.3**. In **Fig 7.3a, b** and **c** the shadowers are tracking prey moving according to the stochastic algorithm, in **7.3d** the prey moves regularly. As can be seen, the shadowers approach accurately in each diagram regardless of prey movement or starting positions. The average errors resulting from the tests described in **section 7.1.3** are displayed in **Fig. 7.4**. The results of statistical analyses performed on the error data are displayed in **Table.**

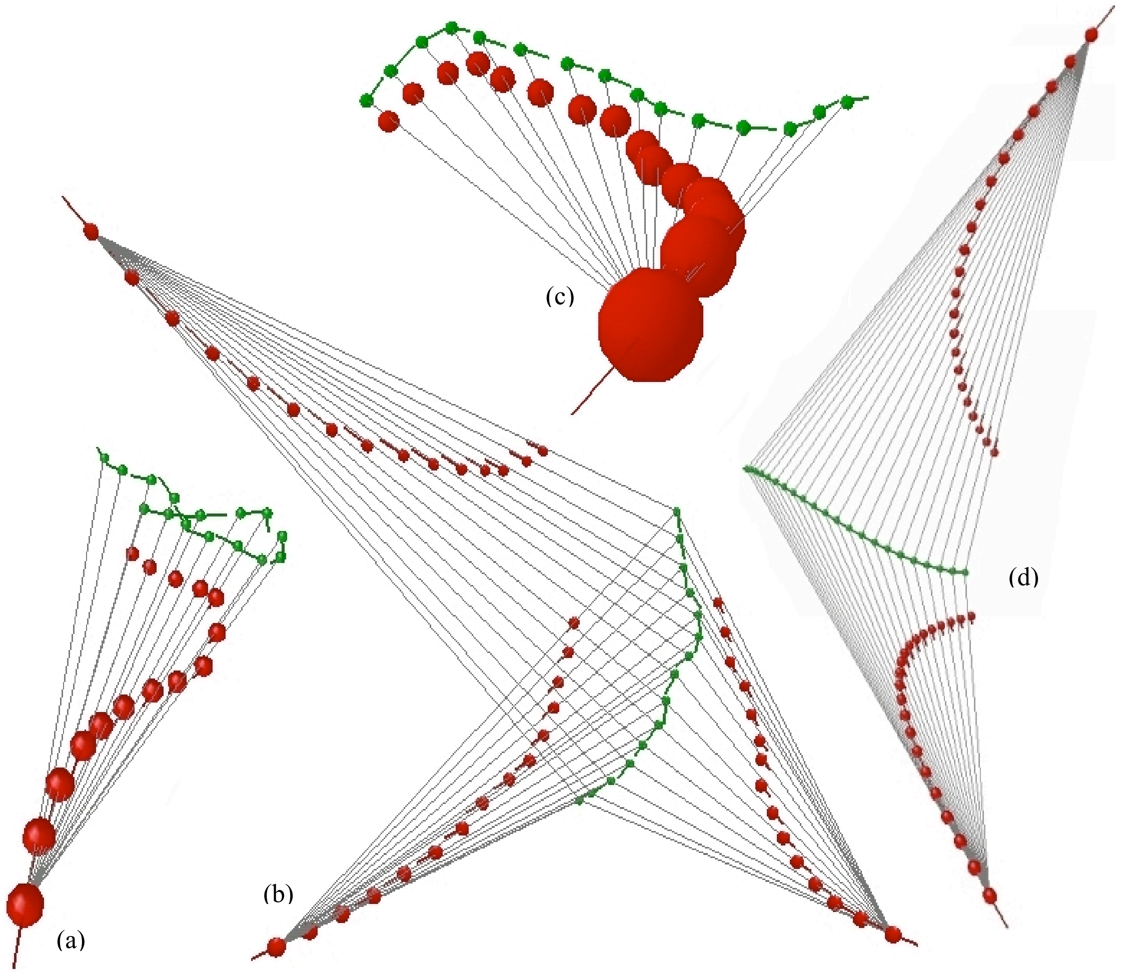


Fig 7.3. Example camouflaged trajectories generated by the 3D shadower control systems shown from different viewpoints and rendered in 3D perspective: **(a)**, **(b)** and **(c)** show shadowers tracking prey moving stochastically, **(d)** shows two shadowers simultaneously approaching a prey moving in a regular pattern.

7.1.

Table 7.1a displays the results of an analysis of the system trained on stochastic trajectories. It identifies a highly significant difference in visual error between tests on regular and stochastic prey trajectories. This difference can be seen in **Fig. 7.4**, where the error is always lesser on tests of the same prey trajectory type to those trained. There was also observed to be a significant difference in azimuth and elevation error (but not visual error) between systems operating with and without image motion. The significant azimuth interaction results from the comparatively low error of the image motion system tested on stochastic trajectories contrasting with the slightly higher comparative error on regular prey trajectories

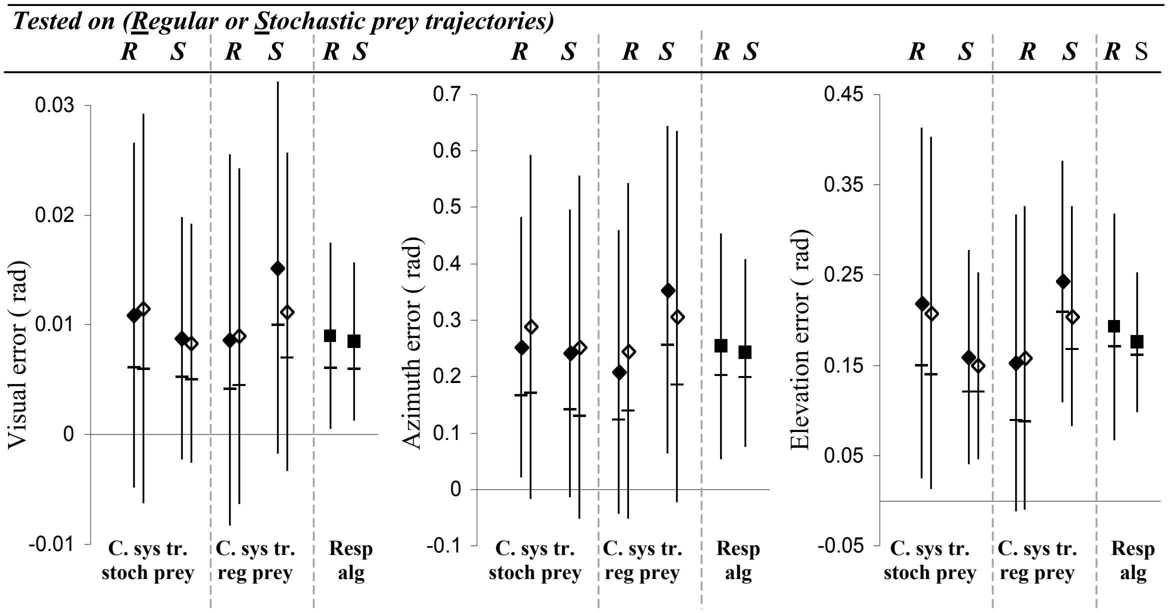


Fig. 7.4. Comparison of mean +/- std error visual, azimuth and elevation error per trial for each of the control systems and the responsive algorithm (calculated from exact locations using trigonometry) when tested against both stochastic and regular prey trajectories. Similar to [2] visual error was not recorded after the shadower was within 15 units of the prey. Within this distance any slight error in choice of direction could have an extreme effect on the average visual error, thus reducing its usefulness as a comparative error measure for the different control systems.

All tests performed on the system trained on regular prey trajectories (**Table 7.1b**) indicated highly significant differences. Conversely, this time it was the image motion system that yielded comparatively greater errors when tested on the trained trajectory type (regular) and far lesser on the alternative (stochastic).

The interaction in **Table 7.1c** and **d** again highlights the superior performance of the controllers when tested on the trained prey trajectory type. The insignificant difference in visual error between training trajectory types in **Table 7.1d** suggest that the image motion networks were better able to generalise to the alternative prey trajectory type (stochastic) to that trained. As such, no evidence was found to say that knowledge of image motion increases the maximum attainable accuracy of the control systems, only that it may enhance the system's ability to generalise to new prey movement patterns (when trained on regular movement).

Table 7.1 Scheirer-Ray-Hare tests (non-parametric equivalent of two-

		Azimuth error	Elevation error
	▨	▨	▨
		▨	▨
		▨	
	▨	▨	▨
	▨	▨	▨
	▨	▨	▨
	▨	▨	▨
	▨	▨	▨
	▨	▨	▨
	▨	▨	▨
		▨	▒
	▨	▨	▨
	▨	▨	▨
	▨	▨	▨
	▨	▨	▨

Finally, **Table 7.1e** compares the error of the two non-image motion systems to that of the responsive algorithm on tests against both trajectory types (the control systems were tested on the same prey trajectory types to those trained). Again the test indicated that all differences were highly significant. In accordance **Fig. 7.4** shows that the average errors from the relevant controllers were lower than those of the responsive algorithm. The significant interaction is a result of the comparatively lower error of the responsive algorithm on the stochastic prey trajectory test to the comparatively higher error given by the control systems. These results show that the performance of the control systems was on average closer to perfect camouflage than the responsive algorithm. However the corresponding higher standard error (see **Fig. 7.4**) suggests a more erratic performance. (i.e. when mistakes were made, they were greater than those of the responsive algorithm). In summary:

- There is no clear evidence that provision of image motion improves performance;
- There is evidence that the control systems (especially the regular control system) were capable of predicting future camouflaged positions;
- Again each control systems outperforms the others when tested on the prey trajectory type that it had been trained with. Control systems trained on stochastic prey were better able to generalise to regular prey trajectories than vice versa.

7.3 Discussion

This chapter has shown that it is possible for a shadower to calculate motion camouflaged approaches to track prey moving in 3D space given only basic input. The greater difficulty encountered with the task is reflected by the increased error in comparison to the 2D simulations. However the control systems were still able to outperform the non-predictive approaches given by the responsive algorithm (calculated using trigonometry).

As was expected from the results of **chapter 6**, the shadowers were found to specialise in tracking prey moving in the manner on which they had been trained. Again the shadowers trained on the predictable regular trajectories were less capable of generalising to the alternative prey movement type. This adds support to the suggestion made in **chapter 6** that camouflage control systems are likely to have to be tailored to approach specific prey.

It has been shown that the shadower has been able to perform effectively, using a single basic sensory input (the direction of the prey). This suggests that the performance of the network is sufficiently accurate for prey image motion not to be a necessity in self error measurement (as mentioned in the third bullet point of **section 6.3**) and shows that image motion is not essential for the training process. The simplicity of input found to be necessary for camouflage in the simulation is an encouraging result for engineers who may have an interest in designing real world motion camouflage control systems. From the point of view of biologists, the simulation has shown that the behaviour can be learnt by a system that works in a manner reminiscent of a biological nervous system. This result provides further evidence that the computation of camouflage trajectories is not beyond insects such as hoverflies. As such this increases confidence of conclusively identifying the existence of the behaviour in nature. What remains is to demonstrate that motion camouflage has a practical application. The next chapter tests whether humans are susceptible to camouflaged approaches calculated by the control systems.

8 MOTION CAMOUFLAGE APPLIED: MISSILE DEFENCE

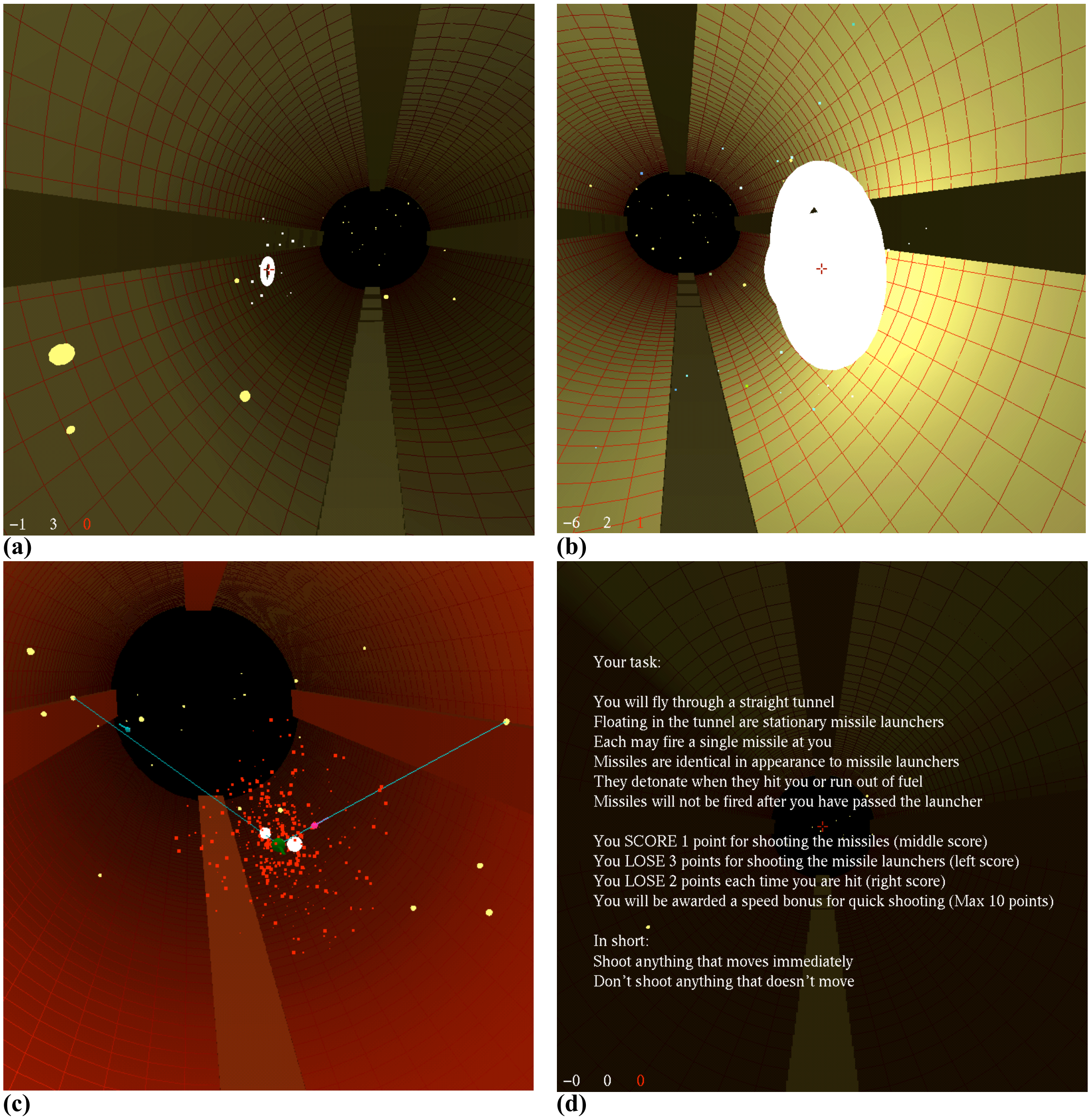
So far this thesis has shown that it is possible to implement an accurate predictive camouflage control system that can track prey moving in a 3-Dimensional simulation. What remains is to demonstrate that the networks have a practical application. To this end this final chapter presents the implementation and results of a psychophysical experiment that masqueraded as a first person perspective computer game competition. The game (Missile Defence) is used to show that humans are susceptible to the camouflaged approaches of the control systems.

8.1 Methods

Specifically, the aim of this chapter was to show that motion camouflage approaches would allow a predator to travel closer to their prey before being identified than other plausible approach strategies. The challenge was seen as to provide a simple, understandable and in order to attract subject participation, entertaining task that would allow comparison of the different approach strategies. A 3D computer game incorporating motion camouflage seemed a perfect solution which not only fulfils these criteria but also demonstrates a potential application for motion camouflage.

8.1.1 Missile Defence, game design

In the game, the player takes the role of the prey who moves in a straight line down the centre of a tunnel. The player is given the freedom to look and shoot in any direction (controlled by moving the mouse and clicking the mouse button) as is conventional in contemporary first person perspective computer games such as Quake 3 (www.quake.com.br). In the centre of the screen are crosshairs acting as sights for the player. Floating in the tunnel are stationary missile launchers which may or may not fire a missile at the prey (i.e. the player). Missiles and missile launchers are identical in appearance. Both are represented as bright yellow spheres. The player’s purpose in the game is to gain a high score by shooting the approaching missiles as fast as possible (so as to avoid being hit). One point is awarded for each missile shot and a speed bonus is awarded for fast shooting (the calculation of which is discussed later in the section). The player loses two points each time they are hit by a missile and loses three points for shooting a missile launcher. The missiles have a limited lifespan and self destruct a fixed time after launch (if they have not hit the prey). Whether the player has successfully hit a missile, shot a missile launcher or been hit is indicated by one of three explosion types as shown in **Fig 8.1**.



Your task:

You will fly through a straight tunnel
Floating in the tunnel are stationary missile launchers
Each may fire a single missile at you
Missiles are identical in appearance to missile launchers
They detonate when they hit you or run out of fuel
Missiles will not be fired after you have passed the launcher

You SCORE 1 point for shooting the missiles (middle score)
You LOSE 3 points for shooting the missile launchers (left score)
You LOSE 2 points each time you are hit (right score)
You will be awarded a speed bonus for quick shooting (Max 10 points)

In short:
Shoot anything that moves immediately
Don’t shoot anything that doesn’t move

Fig 8.1. Missile Defence screen shots. **(a)** Explosion indicating when a missile has been shot. The explosion lasts 5 frames. The view is from the player’s perspective. **(b)** Explosion indicating when a missile launcher has been shot. The explosion lasts 10 frames. The view is from the player’s perspective. **(c)** Two examples of the explosion indicating when the player has been hit by a missile or when a missile self-destructs. The explosion lasts 5 frames. The view is from third person perspective (behind the prey). The player (prey) is shown in green in the centre of the image. The missiles are exposed. The light blue sphere with a tail in the top left of the image is a Straight missile. The pink missile to the right of the prey is a Trig-MC missile. See text for a description of missile types. The blue lines are camouflage constraint lines. **(d)** Instruction screen. The background is shown at normal luminance (without explosion lighting).

From the experimental point of view, the missile launchers are the fixed points and the missiles represent predators employing different strategies to approach the prey (including motion camouflage). The speed bonus and negative scores were included to encourage the player to fire as soon as they think they see a missile but not to recklessly shoot at anything (i.e. attempting to shoot missiles before they have been fired). The speed bonus was calculated as the ratio of the mean life-span of the missiles to the maximum possible life-span multiplied by 10 points. The different types of explosion give covert feedback to the subjects about their success. The tunnel was included to enhance the player's perception of movement in the virtual environment so as to give realistic cues of positions through relative motion. The contrast between the bright yellow missiles and dark background was high to ensure that missiles and launchers were clearly visible. Use of more complex patterns of prey movement such as the curved or stochastic trajectories of previous chapters was dismissed to avoid confusing the player (in preliminary trials of the game, players had expressed that they were confused as to their course of movement when moved along more complex paths).

There are four types of missiles, categorised according to their approach strategy:

- *NN-MC* (Neural Network Motion Camouflage), NN-MC missiles are controlled by the regular camouflage control system trained in **chapter 7**.
- *Trig-MC*, Trig-MC missiles employ perfect predictive camouflaged approaches calculated from exact distance information and knowledge of future prey positions.
- *Homing*, at each time step Homing missiles move in the direction of the forthcoming prey position (see **Fig 8.2**).
- *Straight*, Straight missiles move in a straight line so as to intercept the prey as quickly as possible or to get as close to the prey before they reach their maximum range and explode (see **Fig 8.2**).

Notice that the NN-MC missiles are the only missiles that do not have access to knowledge of future prey positions. Also that it was necessary to use a predetermined prey trajectory to allow calculation of the appropriate paths for the Trig-MC, Homing and Straight missiles.

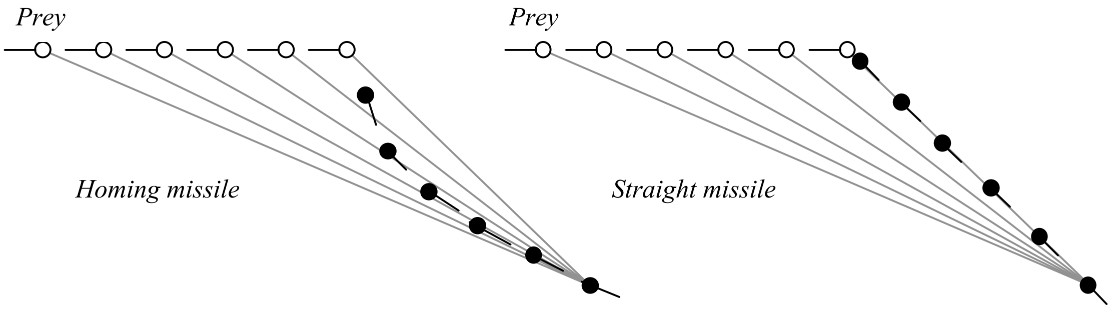


Fig. 8.2. Example approach paths of Homing and Straight missiles (shown in 2D). Grey lines are camouflage constraint lines.

In each game there are 80 missile launchers, 40 of which fire a missile. During a game 10 missiles of each of the four types are fired (in a random order). The positions of the missile launchers, missile types and launch times are determined randomly from the following specification before the start of each game. Missile launchers are placed so that their position at launch time is within the prey's field of view as shown in **Fig 8.3** (launch times are chosen at random throughout the duration of a game). The radius of missiles and launchers is 2 units and the distance of any launcher at launch time is between 200 and 400 units from the prey reflecting the parameters of **chapters 6** and **7**. Similarly the prey moves a distance of 4.5 units per step, whilst missiles move 5 units. During each game the prey moves 1200 steps and each missile moves a maximum of 40 steps before self destructing. In order to smooth the animation 10 extra frames were inserted between each pair of consecutive steps (positions of prey and missiles were calculated by linearly interpolating between steps). The game was set at a speed estimated slow enough to give the player a fair chance of shooting the missiles, but fast enough so as not to be boring. Each game takes approximately 5 minutes to complete.

The effectiveness of the four approach strategies was measured in terms of the distance from the prey at which the missiles exploded (either through being shot, hitting the prey or self destructing after reaching the end of their life).

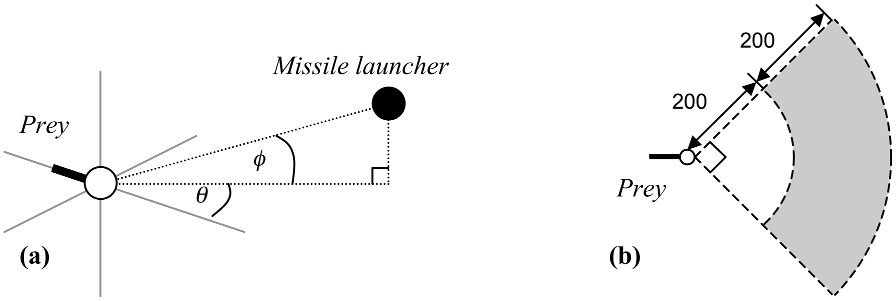


Fig. 8.3. Selection of missile launcher locations. **(a)** The azimuth θ and elevation ϕ of a launcher's position at firing relative to the prey were selected randomly from the respective ranges $-\pi/4 > \theta < \pi/4$ and $-\pi/8 > \phi < \pi/8$ (if both angles are 0 the launcher is directly ahead of the prey). The distance between prey and launcher at firing was randomly selected from the range 200 to 400 distance units. **(b)** shows a plan view of the 3D area (shaded grey) in which a missile launcher could be positioned relative to the prey.

8.1.2 Experimental procedure

The experiment was conducted in the guise of a one week long games competition. Players were invited to compete for a prize for the best Missile Defence score. The 30 experimental subjects were all members of staff or students at the Department of Computer Science at Queen Mary's University who competed voluntarily (age range 18-40 years). Competitors were allowed multiple attempts to beat the high score, though to ensure equal representation of each competitor, only the results from the competitor's best scoring game were selected for analysis (a mean of 4 games were played per player). Subjects were unaware of the purpose of the experiment.

During each game the player sat at a desk with a computer and mouse set up in conventional arrangement. The screen was approximately 50 cm from the player and the game was played within a square 19cm*19cm window. The subjects all had normal, or corrected to normal vision. At the onset of each game the player was shown a screen listing the following instructions (see also **Fig. 8.1d**):

You will fly through a straight tunnel.

Floating in the tunnel are stationary missile launchers.
Each may fire a single missile at you.
Missiles are identical in appearance to missile launchers.
They detonate when they hit you or run out of fuel.
Missiles will not be fired after you have passed the launcher.
You SCORE 1 point for shooting the missiles.
You LOSE 3 points for shooting the missile launchers.
You LOSE 2 points each time you are hit.
You will be awarded a speed bonus for quick shooting (Max 10 points).

In short shoot anything that moves immediately.
Don’t shoot anything that doesn’t move.

During the instructions an automatic demonstration was given showing missiles and missile launchers being shot (see also **Figs 8.1a&b**). The players were then left to complete the task.

8.2 Results

The results show that in general the motion camouflaged missiles approached closer than the other missile types before being detected and shot. **Fig 8.4a** displays frequency distributions of all missile explosion distances (300 of each missile type, 10 per subject per missile type). The distance distributions corresponding to both motion camouflage missile types (NN-MC and Trig-MC) are very similar. These missiles were predominantly shot within 100 units distance from the prey and rarely shot beyond 150 units. In contrast the range over which homing and straight missiles were shot was far greater, with homing missiles occasionally shot at distances in excess of 200 units and straight missiles often shot between 200 and 300 units. **Fig 8.4b** displays the mean +/- standard error of the mean distance per player at which missiles exploded. The mean distance at which NN-MC missiles exploded was 0.63 times that at which the homing missiles exploded and 0.48 times that of the straight missiles. The data shown in **Fig**

8.4b was log transformed to reduce heteroscedasticity and analysed using a One-Way ANOVA (df 119). The analysis indicated a highly statistically significant difference between missile types. *Post hoc* LSD tests indicated that there was no significant difference between Trig-MC and NN-MC approaches ($p = 0.43$). That there was a highly significant difference between any combination of motion camouflaged approach and non-motion camouflaged approach ($p < 0.0001$). Finally that there was a significant difference between the Homing and Straight approach strategies ($p = 0.02$). The lowest mean explosion distance per player was given by a motion camouflaged approach in all but four cases (see **Appendix 2**). In two of these the mean distance was similar for all missile types (the straight missiles were most successful against one player). This suggests that these two players were basing their decision to shoot solely on the missiles' looming. In the third case, the NN-MC missiles were shot comparatively early whereas the mean explosion distance for Trig-MC and Homing approaches was identical. Straight missiles were shot at the furthest distance. The fourth player shot the motion camouflage missiles at a mean distance approximately 18 units further away than the homing missiles and again straight missiles were shot at the greatest distance.

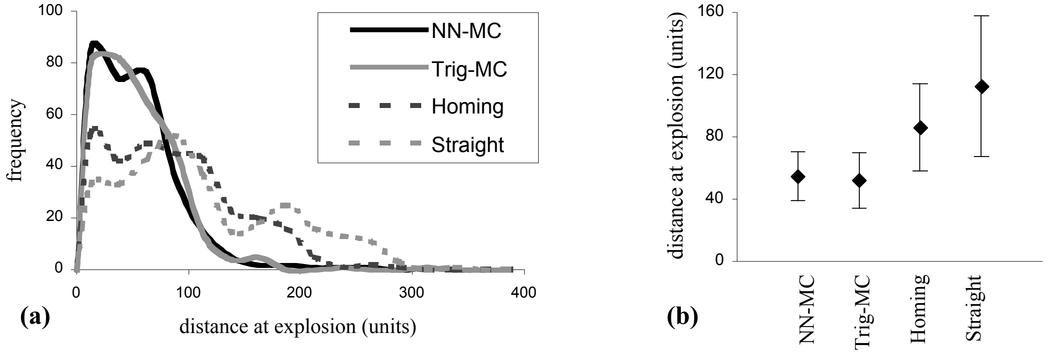


Fig. 8.4 (a). Comparison of the frequency distributions of raw explosion distances for each missile type **(b)** Comparison of the mean +/- std. error explosion distance for each missile type.

8.3 Discussion

This chapter has shown motion camouflage to be effective at allowing the missiles to travel closer to their target before identification than the other approach strategies.

Therefore it may be stated with confidence that the majority of the experimental subjects were susceptible to motion camouflaged approaches, demonstrating for the first time that humans are susceptible to motion camouflage. The chapter has also further validated the control system architecture of **chapter 7** by showing there to be no recognisable difference between the camouflaged approaches made by the neural control systems and the ideal predictive camouflaged approaches (in terms of the distance that the shadowers were able to approach to before being shot).

It was seen that the Homing missiles were able, on average, to approach closer to the prey than the Straight missiles. It is thought that this was due to the greater similarity between homing and motion camouflaged approaches than between straight approaches and camouflaged approaches. For example in **Fig 8.2** it can be seen that the homing approach resembles a camouflaged approach more closely than the straight approach. This is an interesting result in itself, in that in the instance that a motion camouflaged approach can not be calculated, homing may prove a valuable stealth alternative.

The experiment presented in this chapter was carefully designed to be a fair test of human susceptibility to motion camouflage. However there are many factors that are likely to influence the effectiveness of motion camouflage relative to the other approach strategies that were not investigated (the study of which is beyond the scope of the thesis). The remainder of this section identifies some of the factors judged to be important and discusses their implications in light of real world situations.

In the experiment the missiles were represented as plain spheres, identical in appearance to the stationary missile launchers (this ensured that the only way that the two could be differentiated was their pattern of motion). As such the results of the experiment may be considered to be representative of real world situations where the shadower is largely structureless or is at such a distance that it appears to the prey as structureless. The obvious question posed is whether shadowers with complex structures and/or moving appendages (e.g. wings or limbs) would be spotted significantly earlier than those of this chapter. In the experiment the shadowers were exposed by their looming (which seemingly became apparent at a distance of around 100 units, see **Fig 8.4a**). Therefore further investigation would be necessary to identify the relationship between the distance at which the prey notices the shadower's appearance and body part motion and the distance at which the shadower is exposed by looming.

When designing the game, the missiles' approach speed was observed to be a factor influencing the ease with which the missiles could be detected. The author perceived non-motion camouflaged approaches to be comparatively more obvious when the game was run at higher speeds than in the experiment. However at high speed it was difficult to shoot the missiles. Also the player's reaction time was proportionally greater relative to the life-span of the missiles so acting to conceal the difference between missile types (i.e. potentially confounding measurements of perception with a subject's motor ability). Relating to this it should also be recognised that in emulating a stationary point the motion camouflage missiles were the easiest to aim at. The game was not set at a slower speed (where non-camouflaged missiles) as it became less exciting (i.e. less likely to attract volunteers).

The complexity of the background is thought likely to influence the comparative success of the different approach strategies, though the relationship is difficult to predict. A complex background could serve to disguise the movement of the non-motion camouflaged predators or alternatively provide reference points that help to expose their relative movement. Similarly the level of background motion is a factor that could conceivably bias results either way. If the prey is familiar with the background, or has some understanding of the objects in the background, the shadower is likely to benefit from the use of an existing landmark as fixed point. For instance in a barren landscape with a single tree, a non-camouflaged predator commencing its approach from the tree would soon be evident as a second object in the environment. In the same situation a motion camouflage predator may benefit if the prey has an understanding that the tree is a static object. In this case, the prey may not expect any motion from the tree and could discount the shadower's looming as visual noise. It is also suggested that a shadower could use a strategic choice of fixed point to its advantage. For instance as suggested in

section 2.2.1, if the prey were able to find a large fixed point against which it is camouflaged in appearance, looming cues would be likely to be hidden longer. Alternatively the shadower could even employ motion camouflage to approach against the sun in a similar manner to fighter pilots. Lastly, in the experiment the contrast between the missiles and the background was high. In low contrast conditions, when the prey has difficulty distinguishing the shadower from the background looming is likely to be less evident. However with no contrast motion camouflage would be redundant.

In concluding, this chapter has demonstrated the effectiveness of motion camouflage as a stealth approach strategy and that the approaches of the control systems are accurate enough to deceive humans in a practical task. This section has identified factors thought likely to influence the advantage over the other approach strategies given by motion camouflage. It is suggested that insight into the effects of many of these factors could be gained through the use of simulations similar to that presented in this chapter. Finally, through showing that the experimental subjects were susceptible to motion camouflage, this chapter has also demonstrated a potential application of motion camouflage as an approach strategy for predatory agents in computer games.

9 SUMMARY, CONCLUSIONS AND FUTURE DIRECTIONS

In **chapter 2** an overview of different camouflage strategies was provided and motion camouflage was introduced as a stealth behaviour that would allow one moving body (a shadower) to conceal its motion from another moving body (the prey). The basis of the strategy is that the shadower should travel along a path such that its image projected onto the retina of the prey emulates that of a distant stationary object in the environment (the fixed point). It was seen that motion camouflage could be employed as either an offensive or defensive strategy (e.g. to approach or retreat from the prey, though this thesis concentrates on the use of motion camouflage as a predatory approach strategy). It was suggested that motion camouflage has potential military applications (e.g. as an automatic control strategy for missiles). Alternatively motion camouflage could be employed as a stealth tactic for 'bot' AI in computer games. Two general algorithms for motion camouflage were proposed (the responsive and predictive algorithms) and the advantages and disadvantages of each were identified. Both algorithms require the shadower to estimate the position of the prey and fixed point. Investigation of the format and amount of input information necessary to adequately estimate these positions was identified as a subject of particular interest for the remainder of the thesis. Shadower looming (the increase in the size of the shadower's image perceived by the prey as the shadower approaches) was recognised as a factor likely to influence the success of camouflage and methods to minimise looming were suggested. Finally it was shown that the location of the fixed point relative to the prey's movement direction would influence the level of constraint on the shadower's path (e.g. the shadower would not have to move laterally if the prey is heading straight towards the fixed point).

Chapter 3 looked to biological literature for inspiration in designing a motion camouflage controller. Firstly several sensorimotor behaviours were examined. It was seen that male hoverflies move in a manner consistent with motion camouflage to track females. Also that the homing ability of various animals demonstrates that they can accurately estimate their current position relative to their location at the start of their outbound journey without need of an explicit visual reference. This ability was identified as being useful to a shadower for estimating the position of the fixed point. Different methods for extracting the information likely to be helpful for calculating camouflaged movements from the senses were described. Following this sensorimotor integration was considered. A background to control systems theory was given and possible motion camouflage control systems were proposed. In introducing the operation of biological sensorimotor controllers a brief account of the vestibulo-ocular-reflex was provided. This was used to highlight the complexity of neural solutions to sensorimotor problems. Finally an overview of the field of artificial intelligence was provided. This introduced and contrasted the symbolic and connectionist approaches to artificial intelligence. The connectionist approach was seen as being most relevant to the design of a camouflage control system (in that it was not obvious how to derive a symbolic solution to the motion camouflage problem, therefore it was thought most profitable to train a connectionist network that is likely to be able to learn the answer). Four connectionist network models were discussed and the choice of networks on which to base studies in the forthcoming chapters was explained.

Chapter 4 covered the first steps made towards designing an autonomous motion camouflage control system. A controller for the responsive algorithm was implemented using a self-organising network. The controller was provided with external input information that was known to be sufficient to allow calculation of camouflaged movements (accurate binocular vision of both prey and fixed point). The input was however more explicit and more complete than would be expected in a real world scenario. The control system's motor output was represented in terms of two values, the direction to move and the rotation. The controller performed camouflaged approaches competently, generalising between different prey trajectories and approaching accurately from different start positions. However this performance was at the cost of a large number of artificial neurons and training was highly reliant on the appropriate setting of parameters. Methods thought likely to improve upon the system’s performance and efficiency were suggested. However study of this self organising technique was discontinued in light of the superior accuracy obtained by training Multilayer Perceptrons using Backpropagation.

Chapter 5 described the implementation of controllers for the responsive algorithm built from Multilayer Perceptrons trained using Backpropagation. All networks trained were found to give a superior performance to the most accurate self-organising network of **chapter 4**. On the basis of this success Backpropagation was used to train networks throughout the remainder of the thesis. During preliminary investigations of the application of Backpropagation, it was observed that the networks tended to be more accurate when they were tested during training (i.e. when the weights were adapted after each time step of the test) as opposed to after training (when the weights were held constant, as is conventional). Further investigation showed that this continuous training technique yielded an improved camouflage performance with respect to the motor output controlling the movement direction but not the motor output controlling rotation. However for the networks to continually operate under training conditions they would require an external teacher. Aside from the difficulties in providing this, it was not clear whether any practical advantage would be offered over concentrating the extra computational resources required for this external teacher on the original control system itself. This topic was therefore not investigated further. The test results also suggested that it was profitable to train two separate networks (corresponding to each of the two motor outputs) rather than train a single large network to estimate both outputs.

In **chapter 6**, MLP control systems given (arguably) realistic levels of sensory input information (the direction and image motion of the prey) were trained to perform the predictive algorithm. Each control system was provided with feedback of its recent outputs rather than any external reference with which to estimate the position of the fixed point. The shadowers' camouflaged approaches showed evidence of an ability to predict the prey's motion, both when the prey moved along regular flight paths and when the prey moved along flight paths filmed of real hoverflies. It was seen that control systems trained on hoverfly trajectories were capable of predicting the motion of prey following regular trajectories, whereas controllers trained on regular trajectories could not predict hoverfly flight paths. Suggestions were made as to how the control system may have extracted the information necessary to calculate camouflaged approaches from its inputs (**section 6.3**).

Chapter 7 extended the simulation of **chapter 6** to 3-Dimensions and investigated the effects of removing an input (prey image motion) that had previously been given to the control systems of **chapter 6**. The 3D control systems were trained and tested on two types of artificial prey trajectory: regular trajectories (where prey motion could always be reliably predicted) and 'stochastic trajectories' generated by an algorithm involving a random component (the stochastic trajectories were generated as a 3D substitute for the hoverfly trajectories that could only be filmed in 2D). Again the control systems were shown to be able to accurately adopt camouflaged approaches demonstrating their ability to predict prey motion. No evidence was found to suggest that knowledge of prey image motion offered any clear advantage.

Finally **chapter 8** incorporated the control systems of **chapter 7** in a computer game that formed the basis of a psychophysical experiment. In the game the player took the role of the prey whose purpose was to identify and shoot missiles fired (at the player) from stationary missile launchers. The game was used to show that missiles following motion camouflaged approaches were on average able to approach closer to the player before being detected and shot than other plausible approach strategies. Also that there was not a significant difference between the average distance at which missiles controlled by the control systems and missiles following perfect predictive camouflaged approaches were identified and shot. As such this further validated the control system architecture through use of a real task and provided the first evidence that humans are susceptible to motion camouflage.

In conclusion:

- This thesis has provided the first artificial motion camouflage control systems known to the authors.

- It has been shown, through simulation, that camouflaged movements may be computed from basic input information: if the shadower is aware of its own recent movement and the direction of the prey (assuming that the shadower continually rotates to face away from its initial position) it is possible to calculate a camouflaged movement. That the system has been implemented in a manner reminiscent of biological nervous systems adds weight to the conjecture that the calculation of motion camouflaged approaches is not beyond the computational power of insects (and also not beyond the processing capabilities of autonomous robots).
- Evidence has been found to show for the first time that humans are susceptible to motion camouflage approaches (including those made by the control systems). This result may be of interest to computer games developers and military tacticians.

Five possible areas for future investigation are suggested:

- Implementation of a motion camouflage control system for an autonomous robot. This thesis has shown in theory that the information necessary for calculation of camouflaged approaches could be retrieved by contemporary robotic sensors. Added difficulties would be introduced as the system inevitably would have to cope with noisy sensor inputs and differences between intended and actual motor outputs.
- Use of the control systems as predatory stealth agents in computer games for entertainment. The potential for this has already been demonstrated in the game used in the experiments of **chapter 8**.
- Investigation of the effect on the success of motion camouflage brought by change in factors such as: how well the shadower's appearance is camouflaged against the background (the shadower's appearance could for example be camouflaged using military or artificially generated patterns); the background complexity (e.g. the difference between a desert and a jungle, this relates to the preceding point); the relative movement speed of shadower and prey and finally

the prey's trajectories (see also next bullet point). These factors could be tested using psychophysical experiments based on simulations of a similar nature to the game Missile Defence in **chapter 8**.

- Investigation of possible prey counter strategies to combat camouflaged approaches (as originally suggested by Srinivasan & Davey (1995)). One approach to this problem would be for the prey to be controlled by neural networks that are trained or artificially evolved (or both) using the level of the shadower's error as a cost function. If both shadower and prey were to be trained/evolved simultaneously it is thought likely that interesting co-evolutionary effects would emerge. Such effects were observed in pursuit-evasion studies of Cliff & Miller (1996), where pursuers and evaders tended to be adapted to each other's current counter strategies.
- Simulation of camouflaged approaches to particular destinations rather than towards the prey (as suggested by Srinivasan & Davey (1995), and discussed in **chapter 2**). This task would be more difficult than simply approaching the prey as the shadower would additionally have to estimate the relative position of its intended destination. In this case it is suggested that the image motion of the destination, perceived by the shadower could be of importance to estimating the distance to the destination (bearing in mind that the shadower would continue to counteract prey image motion by rotating to face the prey, see also **section 6.3**).
- Use of motion camouflage to conceal multiple shadowers approaching the same prey from the same fixed point (as described in **section 2.1.1**). If a set of shadowers were to approach the prey along the same constraint line (behind one another) only the foremost shadower would be visible to the prey (note that this technique would be effective against radar and thermal imaging). This task would not require the shadowers to have any additional computational abilities to those required for normal motion camouflage. Each shadower would consider the shadower in front to be the prey (except for the first shadower who would base its movement on the real prey). However it is suggested that it would be advantageous for coordination if the shadowers were able to communicate their intended movements to each other.

REFERENCES

Anastasio, T. J. & Robinson, D. A. 1989. Distributed parallel processing in the vestibulo-oculomotor system. *Neural Computation* 1, 230-240.

Anastasio, T. J. 1991. Neural network models of velocity storage in the horizontal vestibulo-ocular reflex. *Biological Cybernetics* 64, 187-196.

Anderson, K. W. J. & McOwan, P.W. 2003. Real-time Emotion Recognition using Biologically Inspired Models. To appear at *4th International Conference on Audio and Video Based Biometric Person Authentication*.

Aronstein, D. C. & Piccirillo, A. C. 1997. Have blue and the F-117A: Evolution of the stealth fighter. American Institute of aeronautics and stronautics.

Bakus. J., Hussin. M., & Kamel. M. 2002. A SOM-based document Clustering Using Phrases. *International Joint Conference on Neural Networks*.

Batsaikhan, O. & Bsorkhuu, S. 2002. Mongolian character recognition using Multilayer Perceptron (MLP). *International Conference on Neural Information Processing.*

Bishop, C. M. 1995. Neural networks for pattern recognition. Oxford University Press.

Boole, G. 1847. The mathematical; analysis of logic: Being an essay towards a calculus of Deductive reasoning. Cambridge: Macmillan, Barclay & Macmillan.

Broomhead, D. S. & Lowe, D. 1988. Multivariable functional interpolation and adaptive networks. *Complex Systems*, 5, 603-643.

Bruce, V., Green, P. R. & Georgeson, M. A. 1996. Visual perception, physiology, psychology and ecology 3rd Ed. Psychology Press.

Burges, C. J. C. 1998. A tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge Discovery.* 2, 121-167.

Carpenter, R. H. S. 1988. Movements of the eyes 2nd Ed. Pion, London.

Carpenter, R. H. S. 1996. Neurophysiology 3rd Ed. Arnold.

Cartwright, B. A. & Collett. 1987. Landmark learning in bees. *Journal of Comparative Physiology* A. 151, 521-543.

Chapman, S. 1968. Catching a baseball. *American Journal of Physics*. 36, 868-870.

Churchland, P. S. & Sejnowski, T. J. 1992. The computational brain. MIT Press.

Cliff, D. & Miller, G. F. 1996. Co-evolution of Pursuit and Evasion II: Simulation Methods and Results. In Maes, P. Mataric, M. Meyer, J. A. Pollack, J. & Wilson, S. W. (Eds). From Animals To Animats 4: Proceedings of the Fourth International Conference on Simulation of Adaptive Behaviour (SAB 96). MIT Press Bradford Books.

Collett, T. S. & Land, M. F. 1975. Visual control of flight behaviour in the hoverfly, Syritta pipiens. *L. J.comp. Physiol.* 99, 1-66.

Collett, T. S. & Land, M. F. 1978. How hoverflies compute interception courses. *J. Comp. Physiol.* 125, 191-204.

Cover, T. M. Geometrical and statistical properties of systems of linear inequalities with applications in pattern recognition*.* *IEEE Transactions on Electronic Computers. EC-14*, 326-334.

Dawson, C.W. & Wilby, R.L. 1999. A comparison of artificial neural networks used for river forecasting. *Hydrology and Earth System Sciences* 3 529 - 540

Ech-Sherif, A., Kohili, M., Benyettou, A. & Benyettou, M. 2002. Lagrangian support vector machines for phoneme classification. *International Conference on Neural Information Processing.*

Efe, M. O., Kaynak, O. & Wilamowski, B., M. 2002. Precise Interception of a manoeuvring target by a missile. *International Joint Conference on Neural Networks*.

Everitt, R. & McOwan, P. W. 2003. Java-Based Internet Biometric Authentication System. *IEEE Transactions on Pattern Analysis and Machine Intelligence*. In press.

Feng, Y., Song, L., Li, D. & Zong. 2002. Application of support vector machines to quality monitoring in robotized arc welding. *International Joint Conference on Neural Networks*.

Foelix, R. F. 1996. Biology of spiders 2nd Ed. Oxford University Press.

Freeman. R., Yin. H., & Allison, N. 2002. Self Organising Maps for tree view based hierarchical document clustering. *International Joint Conference on Neural Networks*.

Frege, G. 1879. Begriffsschrift, eine der Arithmetischen Nachgebildete Formelsprache des Reinen Denkens. Verlag von Louis Nebert, Halle. English translation by van Heijenoort 1967 as Begriffsschrift, A formula language, modeled upon that of arithmetic, for pure thought. In van Heijenoort, J. (Ed). 1967. From Frege to Gödel: A Source Book in Mathematical Logic, 1879-1931. Harvard University Press, Cambridge, Mass.

Gilardi, N. & Bengio, S. 2002. Local machine learning models for spatial data analysis. *Journal of Geographic Information and Decision Analysis* 4, 11-28.

Gödel, K. 1931. Uber formal unentscheidbare Satze der Principia Mathematica und verwandter Systeme. *Monatshefte fur Mathematik und Physik*, 38, 173-198. (English translation: Meltzer, B. 1962. On formally undecidable propositions of 'Principia mathematica' and related systems. Oliver & Boyd.)

Gutta, S. & Philomin., V. 2002. Classification of objects in residential monitoring systems. *International Conference on Neural Information Processing.*

Haddadnia, J., Ahmadi, M. & Faez, K. 2002. Hybrid learning RBF neural network for human face recognition pseudo Zernike moment invariant. *International Joint Conference on Neural Networks*.

Hafner, V. V. 2000. Cognitive maps for navigation in open environments. Proceedings of the 6th International Conference on Intelligent Autonomous Systems 801-808.

Hassibi, B & Stork, D. G. 1993. Second order derivatives for network pruning: optimal brain surgeon. 164-171 In (Eds) Stephen Jose Hanson, S. J., Cowan, J. D. & Giles, L. *Advances in Neural Information Processing Systems*. Morgan Kaufmann, San Mateo, CA.

Haykin, S. 1999. Neural networks a comprehensive foundation 2nd Ed. Prentice Hall.

Haynes 1, C. Gemeno, C., Yeargan, K. V., Millar, J. G. & Johnson, K. M. 2002. Aggressive mimic of moth pheremones by a bolas spider. *Chemoecology*, 12, 99–105.

Hearst, M. A. 1998. Support vector machines. *IEEE Intelligent systems* 13 (4) 18-28.

Hopfield, J. J. 1982. Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the National Academy of Sciences*, USA. 79, 2554-2558.

Ishikawa. M. & Sasaki. N. 2002. Gesture recognition based on SOM using multiple sensors. *International Conference on Neural Information Processing.*

Joachims, T. 1999. Making large scale SVM learning practical. In Burges, C. J. C., Schölkopf, B., & Smola, A. J. (Eds). Advances in Kernel Methods - Support Vector Learning. MIT Press.

Jutras, P., Prasher., S. O., Yang C-C & Hamel, C. 2002. Urban tree growth modelling with artificial neural network. *International Joint Conference on Neural Networks*.

Kalman, R. E. 1960. A new approach to linear filtering and prediction problems. *Transactions of the ASME, Journal of Basic Engineering*, 82, 35-45.

Klassen., T. & Heywood. M. 2002. Towards the on-line recognition of Arabic characters. *International Joint Conference on Neural Networks*.

Kohonen, T. 1982. Self-organised formulation of topographically correct feature maps. *Biological Cybernetics*, 22, 159-168.

Kral, K. 1998. Side-to-side head movements to obtain motion depth cues: A short review of research on the praying mantis. *Behavioural Processes* 43, 71-77.

Krink, T. & Vollrath, F. 1997. Analysing spider web-building behaviour with rule based simulations and genetic algorithms. *J. Theor. Biol*. 185, 321-331.

Lambrinos, D., Möller, R., Labhart, T., Pfiefer, R & Wehner, R. 2000. A mobile robot employing insect strategies for navigation. *Robotics and Autonomous Systems*. 30, 39-64.

Laird, J. E. 2000. Human-level AI's killer application: interactive computer games. AAAI.

Lewis, F. L. 1992. Applied optimal control and estimation. Prentice-Hall.

Li, Y.C., Liu, L., Yang, T.F., & Chiu W.T. 1997. Comparing the Performance of Mathematical Models for Surgical Decisions on Head Injury Patients. *Journal of the American Medical Informatics Association, Symposium Supplement*, 870.

Li, Z. & Tang S. 2002. Face recognition using improved pairwise coupling support vector machines. *International Conference on Neural Information Processing.*

Lu, W., Wang, W., Leung, A., Lo, S. Yueng, R. Zongben, X. & Fan, H. 2002. Airborne pollutant parameter forecasting using support vector machines. *International Joint Conference on Neural Networks*

Luger, G. F. 2001. Artificial intelligence: Structures and strategies for complex problem solving. 4th Ed. Addison Wesley.

Marken, R. S. 2001. Controlled variables: psychology as the center fielder views it. *American Journal of Psychology*. 114, 259-282.

Maybeck, P. S. 1979. Stochastic methods estimation and control, Vol. 1. Academic Press Inc.

Mc Beath, M. K., Shaffer, D. M. & Kaiser, M. K. 1995. How baseball outfielders determine where to run to catch fly balls. *Science*, 268, 569-573.

McCulloch, W. S. & Pitts, W. 1943. A logical calculus of ideas immanent in nervous activity. *Bulletin of Mathematical Biophysics*, 5, 115-37.

McFarland, D. 1999. Animal Behaviour 3rd Ed. Longman.

Minsky, M & Papert, S. 1969. Perceptrons an introduction to computational geometry. The MIT Press.

Minsky, M. (Ed) 1968. Semantic information processing. MIT Press.

Möller, R., Lambrinos, D., Roggendorft, T., Pfeifer, R. & Wehner, R. 2000. Insect strategies of visual homing in mobile robots. *Biorobotics*.

Moody, J. & Darken, C. J. 1989. Fast learning in networks of locally tuned processing units. *Neural Computation*, 1, 281-294.

Mukkamala, S., Guadalupe, J. & Sung, A. 2002. Intrusion detection using neural networks and support vector machines. *International Joint Conference on Neural Networks*.

Müller, K-R., Mika, S., Rätsch, G., Tsuda, K. & Schlköpf, B. 2001. An introduction to kernel-based learning algorithms. *IEEE Transactions on Neural Networks*, 12, 181-202.

Müller, M. & Wehner, R. 1988. Path integration in desert ants, Cataglyphis fortis. *Proc. Natal. Acad. Sci* U.S.A, 85, 5287-90.

Navarette. P. & Ruiz-del-Solar. J. 2002. Interactive face retrieval using Self Organising Maps. *International Joint Conference on Neural Networks*.

Ng, J & Gong, S. 2002. Composite support vector machines for the detection of faces across views and pose estimation. *Image and Vision Computing*, 20, 359-368.

Owen, D. 1980. Camouflage and mimicry. Oxford University Press.

Ozyilmaz, L. & Yildirim, T. 2002. Diagnosis of thyroid disease using artificial neural network methods. *International Conference on Neural Information Processing.*

Ozyilmaz, L. & Yildirim, T. 2002. Diagnosis of thyroid disease using artificial neural network methods. *International Conference on Neural Information Processing.*

Papi, F. (Ed). 1992. Animal homing. New York: Chapman & Hall.

Pollock. R., Lane. T., & Watts. M. 2002. A Kohonen Self Organising Map for the functional classification of proteins based on one-dimensional sequence information. *International Joint Conference on Neural Networks*.

Powell, M. J. D. 1987. Radial basis functions for multivariable interpolation: a review. 143-167. In Mason, J. C. & Cox, M. G. (Eds). Algorithms for approximation. Oxford: Clarendon Press.

Rao, R. P. N & Fuentes, O. 1995. Perceptual homing by an autonomous mobile robot using sparse self-organising sensory-motor maps. *Proc. World Congress On Neural Networks*.

Rao, R. P. N & Fuentes, O. 1996. Learning navigational behaviours using a predictive sparse distributed memory. In Maes, P. Mataric, M. Meyer, J. A. Pollack, J. & Wilson, S. W. (Eds). From Animals To Animats 4: Proceedings of the Fourth International Conference on Simulation of Adaptive Behaviour (SAB 96). MIT Press Bradford Books.

Rimer, E. M & Martinez, T. R. 2002. Improving speech recognition learning through lazy training. *International Joint Conference on Neural Networks*.

Rojas, R. 1996. Neural networks, a systematic introduction. Springer Verlag, Berlin, New York.

Rosenblatt, F. 1958. The perceptron: a probabilistic model for information storage and organisation in the brain. *Psychological Review*. 65, 386-408.

Rosenzweig, M., Leiman, A. & Breedlove, S. 1996. Biological psychology. Sunderland MA.

Rumelhart, D. E., Hinton, G. E., & Williams, R. J. 1986. Learning internal representations by error propagation. In Rumelhart, D. E. & McLelland, J. L. (Eds) Parallel distributed processing, 1 318-62. Cambridge, MA: MIT Press.

Rushby, J. 1993. Formal methods and the certification of critical systems. SRI International report SRI-CSL-93-7.

Russell, S & Norvig, P. 1995. Artificial intelligence, a modern approach. Prentice Hall.

Schalkoff, R. J. 1990. Artificial intelligence: An Engineering Approach. McGraw-Hill.

Seyfarth, E. A. Hergenroder, R. Ebbes, H. & Barth, F. G. 1982. Idiothetic orientation of a wandering spider – compensation for detours and estimates of goal distance. *Behavioural Ecology and Sociobiology* 11: (2) 139-148.

Sheng. L., Qing, S., Wenjie., H, Aize, C. 2002. Diseases classification using support vector machines (SVM). *International Conference on Neural Information Processing.*

Shortliffe, E. H. 1976. MYCIN: Computer based medical consultations. Elsevier Press, New York.

Smeraldi, F. & Bigun, J. 2002. Retinal vision applied to facial features detection and face authentication. *Pattern Recognition Letters*. 23. 463-475.

Srinivasan, M. V. & Davey, M. 1995. Strategies for active camouflage of motion. *Proc. R. Soc. Lond*. B 259, 19-25.

Srinivasan, M.V., Poteser, P. & Kral, K. 1999. Motion detection in insect orientation and navigation. *Vision Research* 39, 2749-2766.

Stefani, R. T., Savant, C. J., Shahian, B. & Hostetter, G. H. 1994. Design of feedback control systems. Saunders College Publishing.

Taylor, I. 1994. Barn owls, predator-prey relationships and conservation. Cambridge University Press.

Underwood, A. J. 1997. Experiments in Ecology. Their logical design and interpretation using analysis of variance. Cambridge University Press.

Vapnik, V. 1995. The nature of statistical learning theory. Springer NY.

Wang, D., Lee, K. N., Dillon, T. S. & Hoogenraad, N. J. 2002. Protein sequence classification using radial basis function (RBF) neural networks. *International Conference on Neural Information Processing.*

Wang, W., Okunbor, D. & Lin, F. C. 2002. Future trend of the Shanghai stockmarket. *International Conference on Neural Information Processing.*

Wehner, R. 1992. In Papi, F (Ed). 1992. Animal homing. New York : Chapman & Hall.

Wehner, R. 1997. Sensory systems and behaviour in. Krebs, J. R. & Davies, N. B. (Eds). Behavioural ecology, an evolutionary approach 4th Ed. Blackwell Science.

Wehner, R. Michel, B. & Antonsen, P. 1996. Visual navigation in insects: coupling of egocentric and geocentric information. *J Exp Biol* 199, 129-140.

Whitehead, A., N. & Russell, B. 1910. Principia mathematica. Cambridge University Press, Cambridge.

Widrow, B. 1962. Generalisation and information storage in networks of adaline neurons, in Jorvitz, M. C., Jacobi, T., Goldstein, G. (Eds) . Self organising systems. Washington, D.C. Spartan Books, 435-461.

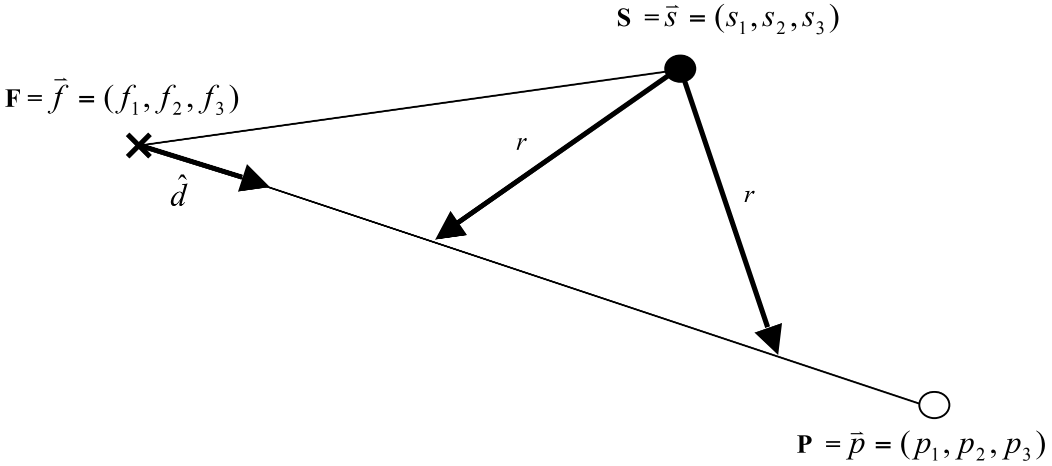
Williams, D. 2001. Naval camouflage a complete visual reference. United States Naval Institute.

Willshaw. D. J. & von der Malsberg, C. 1976. How patterned neural connections can be set up by self organisation. *Proc Royal Soc*. B. 194, 431-445.

Yee, L. P. & DeSilva, L. C. 2002. Application of Multilayer Perceptron networks in public key cryptography. *International Joint Conference on Neural Networks*.

APPENDIX 1: CALCULATION OF CAMOUFLAGED MOVEMENTS

This appendix describes how to calculate camouflaged movements when given the exact locations of shadower, prey and fixed point. In the following description, the shadower lies at point **S**, the future position of the prey at point **P** and the fixed point at point **F** (see diagram below). The shadower will move a distance of *r* units in the next time step. The camouflage constraint line is therefore the line **FP**. In the following description $\vec{a}.\vec{b}$ is used to indicate the scalar (dot product) of vectors $\vec{a}$ and $\vec{b}$ and $\vec{a} \times \vec{b}$ is used to indicate the vector product (cross product).



$\mathbf{S} = \vec{s} = (s_1, s_2, s_3)$
$\mathbf{F} = \vec{f} = (f_1, f_2, f_3)$
r
r
$\hat{d}$
$\mathbf{P} = \vec{p} = (p_1, p_2, p_3)$

Given this, if the shadower wishes to approach the prey, it must travel to the intersection point closest to **P** of the line **FP** and the surface of the sphere with centre $\vec{s}$ and radius *r*.

The unit vector in direction **FP** is:

$$\hat{d} = \frac{\vec{p} - \vec{f}}{\left|\vec{p} - \vec{f}\right|}$$

Therefore the equation of camouflage constraint line **FP** may be expressed as:

$$\vec{x} = \vec{f} + l\hat{d}$$

To find the camouflaged position it is necessary to find the appropriate value of *l*.

$$r = \left|\vec{x} - \vec{s}\right|$$

$$r^2 = (\vec{x} - \vec{s}).(\vec{x} - \vec{s})$$

$$\therefore r^2 = (\vec{f} + l\hat{d} - \vec{s}).(\vec{f} + l\hat{d} - \vec{s})$$

$$\therefore r^2 = \vec{f}.\vec{f} + l\vec{f}.\hat{d} - \vec{f}.\vec{s} + l\vec{f}.\hat{d} + l^2\hat{d}.\hat{d} - l\vec{s}.\hat{d} - \vec{f}.\vec{s} - l\vec{s}.\hat{d} + \vec{s}.\vec{s}$$

$$\therefore l^2 + 2l(\vec{f}.\hat{d} - \vec{s}.\hat{d}) + (\vec{f}.\vec{f} - 2\vec{f}.\vec{s} + \vec{s}.\vec{s} - r^2) = 0$$

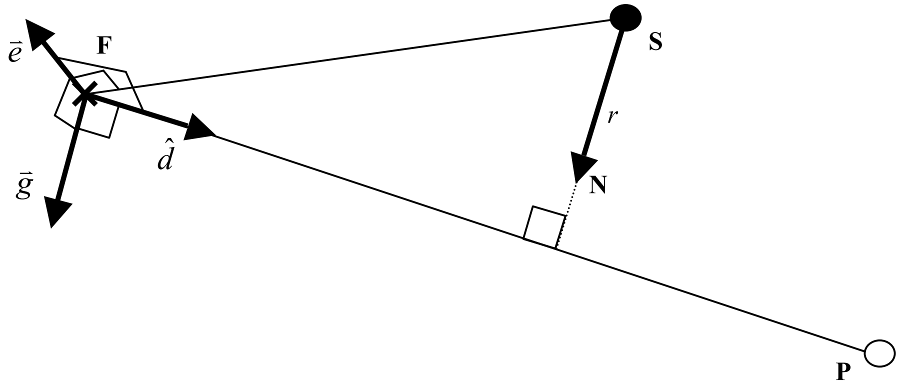
which is a quadratic equation in *l*.

If $P = 2(\vec{f}.\hat{d} - \vec{s}.\hat{d})$ and $Q = \vec{f}.\vec{f} - 2\vec{f}.\vec{s} + \vec{s}.\vec{s} - r^2$ and $P^2 - 4Q \geq 0$ then

$$l = -P \pm \frac{\sqrt{P^2 - 4Q}}{2}$$

Then the camouflaged point can be found by inserting *l* into the equation for $\vec{x}$ (i.e. the equation for the constraint line).

But if $P^2 - 4Q < 0$ it is not possible for the shadower to reach the constraint line. In this case it is necessary to find the point **N** closest to the constraint line **FP** (see diagram below).



First find the direction $\vec{e}$ perpendicular to **FP** in the plane **FSP**:

$$\vec{e} = \hat{d} \times (\vec{s} - \vec{f})$$

Then find the direction $\vec{g}$ perpendicular to both $\vec{e}$ and **FP**:

$$\vec{g} = (\vec{p} - \vec{f}) \times \vec{e}$$

Finally find the line **SN:**

$$\vec{x} = \vec{s} + l\hat{g} \ ,$$

where $\hat{g} = \frac{\vec{g}}{|g|}$ and point **N** is a distance *r* from **S**, so:

$$\mathbf{N} = \vec{s} + r\vec{g}$$

APPENDIX 2: MISSILE DEFENCE MEAN RESULTS

Mean explosion distance per Missile Defence player for each missile type. Each row represents a different player. The last four results are those discussed in **section 8.2** where motion camouflage missiles were not the most successful missiles in that they did not on average approach closest to the prey.

	NN-MC	*Trig-MC*	*Homing*	*Straight*
	61.22714	28.25051	91.62026	72.14626
	38.16898	52.83723	104.1525	175.8024
	52.48377	32.80767	68.53529	54.01627
	44.68245	25.85055	46.74302	92.26877
	43.75534	37.72294	63.30008	82.85799
	54.02676	47.82116	113.3156	193.1908
	55.45364	38.19605	98.60089	98.45228
	34.20582	45.82734	111.2292	78.36102
	87.08997	89.92067	140.1615	167.4511
	69.15591	63.07029	110.9253	164.015
	67.94329	86.28791	105.1528	168.9456
	30.45446	41.42283	101.2756	86.7762
	51.60777	48.73421	79.09258	180.3726
	58.27308	66.72043	95.468	156.1156
	70.07994	80.88847	79.33345	175.1887
	60.1408	56.89022	94.58003	86.31877
	35.30092	31.98484	53.30488	52.32194
	58.25572	53.06086	118.7129	109.7683
	68.20377	63.26802	91.00293	106.9648
	28.9242	26.30161	71.2027	117.9562
	42.74572	38.07779	52.11973	88.7012
	39.98546	43.97199	101.8809	125.7271
	43.45176	62.2387	103.3664	81.32336
	57.46436	81.23375	129.7738	160.9046
	89.17827	65.68062	94.26495	66.25488
	51.28434	30.85753	81.70093	74.50488
1	55.269	66.3583	44.01796	63.47268
2	45.7716	49.00914	42.416	40.36272
3	80.62451	45.34233	45.08126	149.6668
4	54.69307	53.38253	36.81891	102.1604