Evolving Concepts using Gene Duplication

Marc Ebner
Ernst-Moritz-Arndt-Universität Greifswald
Institut für Mathematik und Informatik
Walther-Rathenau-Straße 47, 17487 Greifswald, Germany
Tel: (+49)3834/86-4646, Fax: (+49)3834/86-4640
marc.ebner@uni-greifswald.de

We have developed an on-line evolutionary vision system which is able to evolve detectors for a variety of different objects. The system processes a video stream and uses motion as a cue to extract moving objects from this video sequence. The center of gravity of the largest object, currently in the image, is taken as the teaching input, i.e. the object which should be detected by the evolved object detector. Each object detector only takes a single image as input. It uses image processing operations to transform the input image into an output image which has a maximum response as close as possible to the teaching input. Since the detectors only work on single images, they are basically appearance detectors. However, the appearance of the object may change as the object moves or changes its orientation relative to the position of the camera. Hence multiple different detectors will need to be combined to represent the concept of an object. We use gene duplication to evolve sub-detectors for the different appearances of an object. The difficulty in evolving these sub-detectors is that only one type of appearance is visible for any given image. Hence, the genetic operators could disrupt the genetic material, i.e. the sub-detectors which are currently not in use.

1 Motivation

The human visual system is a product of natural evolution. It is highly advanced. Indeed, to date, no artificial systems have been created which match the abilities of the human visual system. Obviously, it is of considerable interest to learn how the visual system works and how exactly it was created during the course of natural evolution.

Our long term goal is to get a deeper understanding of the human visual system and to reproduce at least parts of human perceptual behavior. Ebner (2009) has created an online evolutionary vision system which takes an image stream as input and evolves object detectors which locate interesting objects (as determined by the user). In an early prototype system, interesting objects had to be pointed out using the mouse pointer. The user would follow the object using the mouse as the object moves across the screen. The user created the teaching input (position of mouse pointer) by pressing the mouse button. Ebner (2010b) then went on to remove the user input from the system. Motion was found to be a valuable cue to provide the teaching input. Hence, independently moving objects were extracted from the image stream and the center of gravity of the detected motion was taken as the teaching input. Evolved object detectors worked on single images, i.e. were able to detect the desired object without the motion cue. After all, humans are also able to correctly name objects which are shown in images.

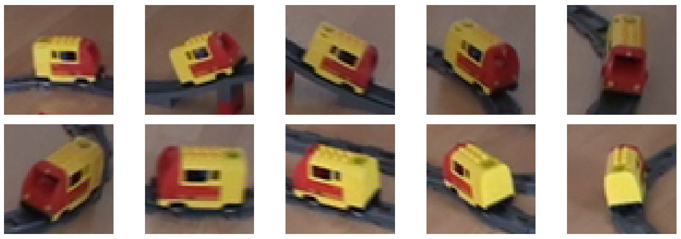
Figure 1: The spatial distribution of colors changes as on object (toy train) moves along a track.

In Ebner's work, it became apparent that detectors for objects without unique colors are more difficult to evolve than detectors which can be based on unique colors by which the object can be identified. If an object cannot be uniquely identified by a single color, one has to take the spatial relationship between colors or the shape of the object in general into account to create a successful detector for such an object. How-

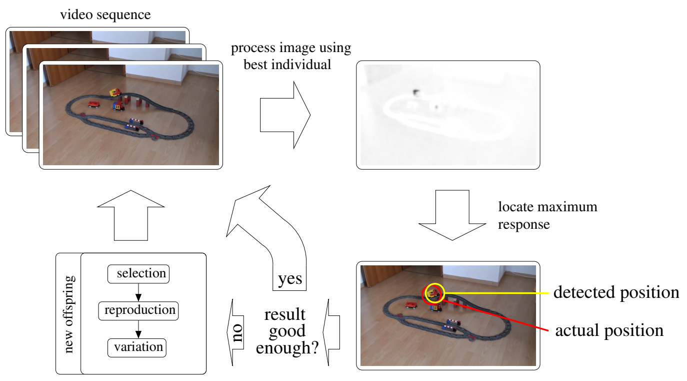

Figure 2: System overview.

ever, the shape of the object as well as the spatial relationship of the colors changes as the orientation of the object relative to the position of the camera changes. This is illustrated in Figure 1. In theory, an overall object detector can be based on several sub-detectors each of which is tuned to a single appearance of an object on the screen. In the human visual system the detectors are independent of the object's size on the retina due to the special type of mapping (complex-logarithmic) from the retinal receptors to visual area V1 (Schwartz 1977).

With this contribution, we have made a move towards creating an artificial system, shaped by evolution, which is able to evolve detectors responding to an overall concept, i.e. detectors which consist of sub-detectors each of which tuned to a particular appearance of an object. The system is illustrated in Figure 2. The system takes a video stream as input. It maintains a population of image processing algorithms. The best individual of the population is used to detect the desired object. If the detected position is close enough to the actual position of the object then the next image is processed. Otherwise, evolutionary operators are used to create a new generation of offspring. Evolution continues through additional images as long as it is required, i.e. until the detection accuracy is good enough. We show that gene duplication is an important evolutionary operator which allows us to incrementally evolve concept detectors. Without gene duplication the problem is more difficult to solve.

The paper is structured as follows. In section 2, we provide a brief introduction into the human vision system. Section 3 puts our work into the context of research in the field of evolutionary computer vision. Section 4 describes our approach to concept learning using gene duplication. Section 5 describes the experiments that we have performed. Conclusions are given in Section 6.

2 The Human Vision System

The visual system is highly structured (Zeki 1993). Processing of visual information of course starts with the retinal receptors (Dowling 1987). Three types of receptors can be distinguished for color perception which measure the light in the red, green and blue parts of the spectrum (Dartnall, Bowmaker, & Mollon 1983). This information is then sent to the primary visual cortex, which is located in the back of the brain. The primary visual cortex, also called visual area 1 or V1 is also highly structured (Livingstone and Hubel 1984). Inside the so called ocular dominance segments, neurons respond primarily to stimuli presented to either the left or the right eye. Within these segments, neurons can be found which respond to light of certain wavelengths or to lines with different orientations. In short, the visual information is analyzed locally with respect to color and lines. Higher visual areas receive their input from V1 and further analyze it to recover shape, motion and the color of objects. It appears that the different aspects shape, motion and color of a stimulus are processed via separate areas of the brain.

So called matched filters (Schrater, Knill, and Simoncelli 2000; Simpson and Manahilov 2001) could be used to detect objects of known shape. An overall detector which responds in a generalized manner, i.e. whenever a random view of an object appears, can be created by simply merging (adding) the output of several matched filters. Such a detector will respond whenever one of the sub-detectors, i.e. matched filters responds. We investigate how such general detectors can be generated through artificial evolution.

3 Evolutionary Computer Vision

Evolutionary computer vision is an active research field since the early 1990s with the pioneering work of Lohmann (1991). An recent overview is given by Cagnoni (Cagnoni 2008). Evolutionary computer vision can be used to optimize a given algorithm for a particular task. In this case, the algorithm is either well known from the literature or developed by a human and an evolutionary algorithm is used to optimize parameters of the algorithm. However, evolutionary algorithms can also be used to evolve an entire computer vision algorithm from scratch using Genetic Programming (Koza 1992). This is of course a highly interesting approach as it would eventually allow the construction of systems which automatically create vision algorithms based on some type of fitness function. Ebner (2008) has been working towards the creation of an adaptive vision system. Creating such adaptive systems is especially important as many computer vision algorithms are very fragile. They work well in the lab but when taken to a different environment the algorithms often no longer work because the ambient light has changed, e.g. from artificial light to direct sunlight. The human visual apparatus adapts easily to such changing conditions. Some artificial systems compute so called intrinsic images (see Matsushita et al. (2004)). Even though it is possible to apply Color Constancy algorithms to arrive at a descriptor which is independent of the illuminant, current computer vision algorithms are far from being fully adaptive. Using evolutionary algorithms it could be possible to create an artificial vision system which also adapts to environmental conditions.

Such an adaptive vision system needs to adapt to an image stream on an image by image basis. That's why our system works with a population of algorithms which are all applied to an incoming image. The best algorithms are allowed to reproduce and create offspring which are then tested on the next incoming image. Of course this requires extensive computational resources. Computer vision algorithms usually require a lot of processing power. However, we have to apply multiple different algorithms to each input image. The time required to evaluate all of the algorithms would be extensive using single CPU processing. We have used GPU accelerated image processing to speed up the evaluation. GPU accelerated image processing is known to provide a speedup of over 40 if ten or more high level image operators are applied (Ebner 2010a).

4 Gene Duplication

Ebner has used a variant of the Cartesian Genetic Programming approach (Miller 1999) for his experiments as shown in Figure 3. Each evolved individual consists of a set of n_1 high level image processing operators and a processing matrix of low level image operators of size $n_x \times n_y$ which are applied to the input image. Each high level processing operator transforms the input image in some way to produce a modified image. Operators include edge detection, Laplacian, convolution or segmentation. In contrast to the high level operators which also take the surrounding of an image pixel into account, the operators used in the processing matrix use point operations to combine the output obtained so far on a pixel by pixel basis. The list of both types of operators is shown in Figure 4. This set of operators is fully described in (Ebner 2009). Due to the lack of space it is not possible to explain the functions of these operators in detail. The interested reader is referred to (Ebner 2009).

Ebner has called this the $n_1 - n_x \times n_y$ representation. Since there are n_y output operators on the right hand side of this representation, there are basically n_y sub-detectors in this representation. The output of these n_y sub-detectors is averaged to obtain a single output image for this detector. The output RGB pixel is interpreted as a 24 bit value and the maximum response over all image pixels denotes the position of the detected object. If multiple image pixels have the same maximum value, then we compute the center of

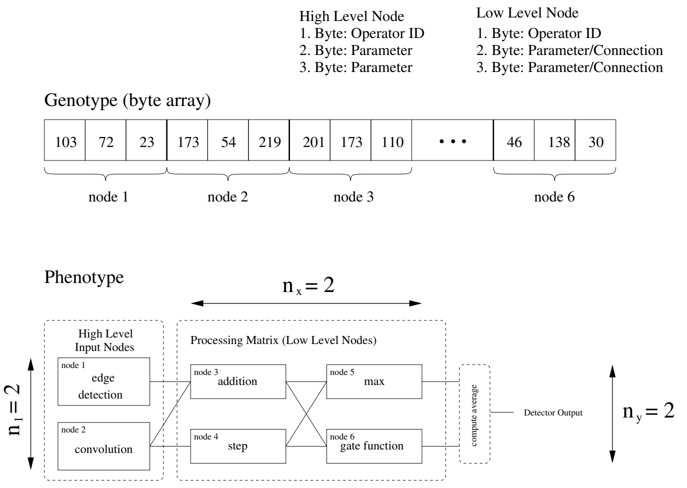

Figure 3: Each individual (byte array) is decoded into a computer vision algorithm which consists of n_1 high level operators and a processing matrix of $n_x \times n_y$ low level operators. The output of the n_y sub-detectors is averaged to compute the output of the entire detector.

	Set of operators
High level input node operators (used in column 1)	`Image`, `DX`, `DY`, `Lap`, `Grad`, `ImageGray`, `ImageChrom`, `ImageLogDX`, `ImageConv1`, `ImageConv4`, `ImageConv16`, `ImageConvd`, `ImageSeg`, `0.0`, `0.5`, `1.0`
Low level node operators (used in the processing matrix)	`id`, `abs`, `dot`, `sqrt`, `norm`, `clamp(0,1)`, `step(0)`, `step(0.5)`, `smstep(0,1)`, `red`, `green`, `blue`, `avg`, `minChannel`, `maxChannel`, `equalMin`, `equalMax`, `gateR`, `gateG`, `gateB`, `gateRc`, `gateGc`, `gateBc`, `step`, `+`, `-`, `*`, `/`, `min`, `max`, `clamp0`, `clamp1`, `mix`, `step`, `lessThan`, `greaterThan`, `dot`, `cross`, `reflect`, `refract`

Figure 4: Set of image processing operators. See Ebner (2009) for a detailed description of these operators.

gravity for all these image pixels to obtain a single position inside the image. This position denotes the position inside the image at which the detector has located the object.

Unfortunately, the evolved detectors did not generalize to detect different views of the same object. We address this problem using gene duplication and gene deletion. Gene duplication and gene deletion have also been used by Koza (1995a, 1995b) in the context of tree-based genetic programming to evolve solutions to the parity problem. Haynes (1996) experimented with duplication of code segments in genetic programming which resulted in a speedup of the learning process. Hoai et al. (2005) have used gene

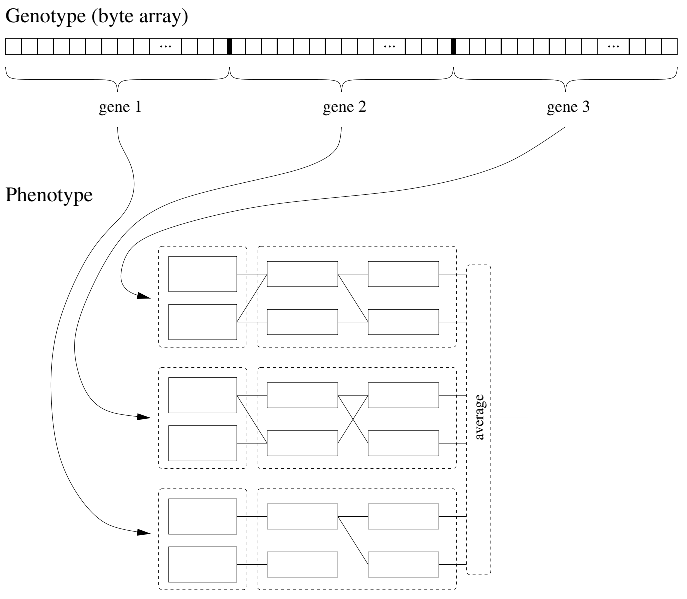

Figure 5: Each individual (byte array) consists of several genes, i.e. sub-algorithms. Each gene is decoded into a $n_1 + n_x \times n_y$ detector. The output of all detectors is averaged to obtain the overall response of the individual.

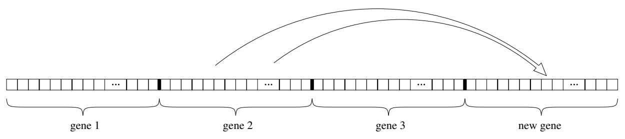

Figure 6: A randomly selected gene is duplicated and appended at the end of the individual.

deletion and duplication for tree-adjoining grammar guided genetic programming.

We add both genetic operators, gene duplication and gene deletion, to the set of operators in addition to the standard evolutionary operators crossover and mutation. In our approach, each individual may consist of several segments or genes. Each gene is basically a description of nodes as in the representation shown in Figure 3. Hence, each individual with multiple genes can be decoded into multiple image processing algorithms as shown in Figure 5. We call them sub-algorithms. The output of these sub-algorithms is averaged to obtain the overall output of the entire individual. When a gene duplication happens, then one of the genes is selected at random. This gene is most likely useful in one way or another because it has been shaped by evolution. The selected gene is duplicated and appended to the individual as shown in Figure 6. When a gene deletion occurs, then a randomly selected gene, i.e. sub-algorithm, is deleted.

The output of an individual with a single gene which has just been duplicated is almost identical to the original individual. However, mutation is now free to change the contents of one the sub-algorithms without compromising on the function of the original gene. Hence, it is now possible to evolve concept detectors. Each concept detector consists of several sub-algorithms whose output is averaged. As long as one particular view is present in the current image, then one of the sub-algorithms will respond.

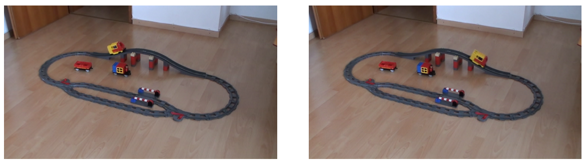

Figure 7: Two images from the toy train image sequence. The sequence was taken with a stationary camera.

5 Experiments

We have used the toy train video sequence that has already been used by Ebner (2010b, 2010a) as shown in Figure 7. It is not possible to detect the toy train through color alone. The color of the toy train is mostly yellow and red. However, other objects in the video sequence feature the same colors, e.g. the wagon in the center. The spatial arrangement of the different colors must also be taken into account to successfully detect the train on its track. A successful algorithm could detect the object showing a large yellow area on the top left and a large red area on the bottom right. Of course this spatial arrangement of colors changes as the train takes turns on the track. During previous runs of this experiment, it was not possible to find an overall detector which would detect the train as it completes a full circle on the track. Therefore, we increased the length of the sequence to ten times its size by playing it forward, backward, forward and so on. However, the problem still remains difficult.

Fitness is computed as the Euclidean distance between the position of the moving object (automatically detected as described in (Ebner 2010b)) and the object position as detected by the individualfitness of zero as it would correctly locate the moving object in the image sequence. Initially, evolution is used to find an individual which is able to correctly locate the moving object in the image sequence. Once the moving object is correctly located, i.e. with an accuracy of 10 pixels or less for five consecutive images, then evolution is turned off. The best individual found so far is then used to detect the moving object for successive images. Of course it may happen that the appearance of the object changes and that the currently used individual is no longer able to correctly detect the moving object. If the accuracy deteriorates beyond 25 pixels difference between the actual and the detected position, then evolution is turned on again. This process continues throughout the entire image sequence. Evolution is turned on whenever re-learning is needed.

Our experiments are carried out using a population of 5 parent individuals which generate 10 offspring for each image whenever evolution has been turned on. In addition to these offspring, 10 offspring are generated completely at random. This ensures an influx of new genetic material at all times. We use four different genetic operators: crossover, mutation, gene duplication and gene deletion. These operators are applied with probabilities p_c, p_m, p_{dup} and p_{del} respectively. We use 2-point crossover with randomly selected crossover points. The mutation operator either increments or decrements a randomly chosen parameter by one or it mutates all of the bits with a probability of $2/l$ per bit where l is the length of the genotype in bits. In other words, on average, two bits are mutated. The reason for the factor of two is that the genetic representation is redundant and hence a slightly larger mutation rate than $1/l$ is used.

For each new incoming image, fitness is computed for all parents as well as all offspring. Parent and offspring are then sorted according to fitness. Whenever two or more individuals achieve the same fitness they are considered to be identical even though their genetic material may be different. Hence, only one of these individuals is kept. The remaining duplicates are discarded. The best 5 individuals are selected to become parents of the next generation. Usually, approximative solutions are found within 24 generations. The evolved detectors will then be further refined by evolution.

We have carried out three experiments:

a) without gene duplication using a $2-2\times 2$ representation with probabilities $p_c = 0.1$, $p_m = 0.9$, $p_{\text{dup}} = 0$ and $p_{\text{del}} = 0$

b) with gene duplication, using a $2-2\times 2$ representation for the first generation with probabilities $p_c = 0.1$, $p_m = 0.89$, $p_{\text{dup}} = 0.005$ and $p_{\text{del}} = 0.005$

c) without gene duplication using a $10-2\times 10$ representation with probabilities $p_c = 0.1$, $p_m = 0.9$, $p_{\text{dup}} = 0$ and $p_{\text{del}} = 0$ as a control experiment.

Figure 8 shows how the number of genes changes during the course of evolution for experiment b). Performing 10 runs for each of these experiments took almost two days on a Linux system (Intel Core 2 CPU running at 2.13GHz) equipped with a GeForce 9600GT/PCI/SEE2.

Table 1 summarizes the results. The first two columns show the number of generations that evolution had to be turned on. The last two columns show the number of restarts which were necessary. The data is shown for the entire run (all 15800 images), as well as for the last iteration of the image sequence, i.e. only for the last 1580 images. The number of restarts, i.e. the number of times evolution had to be turned on again for re-learning, is taken as the relevant indicator.

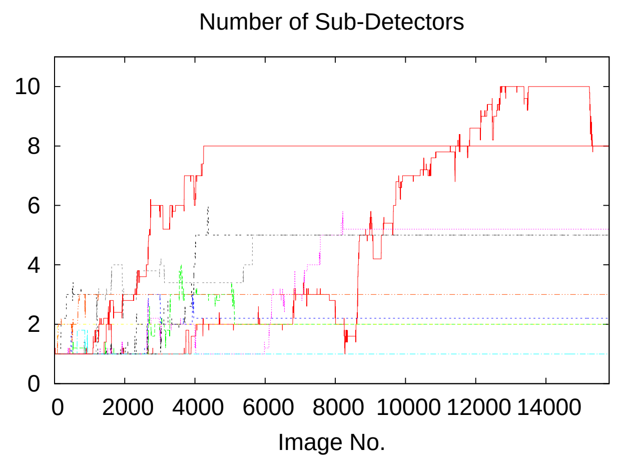

Figure 8: Evolution of the number of sub-detectors (shown independently for all 10 runs of experiment b).

A successful individual would not need any restarts because it is able to correctly detect the moving object at all times. The differences are not statistically significant using a t-Test. However, the problem got easier on average by using gene duplication. The total number of generations that evolution was turned on, could be decreased from 1407.1 to 1014.8 generations on average. The number of restarts could be decreased from 73.4 to 50.8 on average.

6 Conclusion

We have shown that gene duplication is a tool to incrementally evolve object detectors. If the representation used to evolve an object detector is too small, then the evolved detector may not be able to generalize to other views. If the representation is too complex, then evolution may fail to find a good solution. However with gene duplication, evolution can start off using a small representation. Once an approximative solution has been found, it will eventually be duplicated. Evolution is then free to modify one of the genes while still maintaining the functionality of the original gene. This makes its possible to incrementally evolve good solutions.

References

Cagnoni, S. 2008. Evolutionary computer vision: a taxonomic tutorial. In *8th Int. Conference on Hybrid Intelligent Systems*, Los Alamitos, CA, pp. 1–6. IEEE Computer Society.

Dartnall, H. J. A., Bowmaker, J. K. & Mollon, J. D. 1983. Human visual pigments: microspectrophotometric results from the eyes of seven persons. *Proc. R. Soc. Lond. B 220*, 115–130.

Dowling, J. E. 1987. *The retina: an approachable part of the brain*. Cambridge, MA: The Belk-

Table 1: Experimental results for experiments a), b), and c). The standard deviation is shown in brackets. Best results are printed in bold face.

	generations (all images)		generations (last iteration)		restarts (all images)		restarts (last iteration)	
a) no gene duplication	1407.1	(833.6)	32.6	(97.8)	73.4	(42.0)	1.7	(5.1)
b) with gene duplication	**1014.8**	(915.8)	**11.6**	(34.8)	**50.8**	(40.8)	**0.4**	(1.2)
c) control experiment	1316.4	(1098.1)	21.8	(65.4)	55.9	(47.1)	0.7	(2.1)

nap Press of Harvard University Press.

Ebner, M. 2008. An adaptive on-line evolutionary visual system. In E. Hart, B. Paechter & J. Willies (Eds.), *Workshop on Pervasive Adaptation, Venice, Italy*, pp. 84–89. IEEE.

Ebner, M. 2009. A real-time evolutionary object recognition system. In L. Vanneschi, S. Gustafson, A. Moraglio, I. De Falco & M. Ebner (Eds.), *Genetic Programming: Proc. of the 12th Europ. Conf., Tübingen, Germany*, Berlin, pp. 268–279. Springer.

Ebner, M. 2010a. Evolving object detectors with a GPU accelerated vision system. In *Proc. of the 9th Int. Conf. on Evolvable Systems – From Biology to Hardware, York, UK*, Berlin, pp. 109–120. Springer.

Ebner, M. 2010b. Towards automated learning of object detectors. In *Applications of Evolutionary Computation, Proceedings, Istanbul, Turkey*, Berlin, pp. 231–240. Springer.

Haynes, T. 1996. Duplication of coding segments in genetic programming. In *Proc. of the 13th National Conf. on Artificial Intelligenc. Volume 1*, pp. 344–349. AAAI Press/MIT Press.

Hoai, N. X., McKay, R. I. B., Essam, D. & Hao, H. T. 2005. Genetic transposition in tree-adjoining grammar guided genetic programming: The duplication operator. In M. Keijzer, A. Tettamanzi, P. Collet, J. I. van Hemert & M. Tomassini (Eds.), *Proc. of the 8th European Conference on Genetic Programming*, Berlin, pp. 108–119. Springer-Verlag.

Koza, J. R. 1992. *Genetic Programming. On the Programming of Computers by Means of Natural Selection*. Cambridge, MA: The MIT Press.

Koza, J. R. 1995a. Evolving the architecture of a multi-part program in genetic programming using architecture-altering operations. In A. V. Sebald & L. J. Fogel (Eds.), *Proc. of the 4th Annual Conf. on Evolutionary Programming*, Cambridge, MA, pp. 695–717. The MIT Press.

Koza, J. R. 1995b. Gene duplication to enable genetic programming to concurrently evolve both the architecture and work-performing steps of a computer program. In *Proc. of the 14th International Joint Conference on Artificial Intelligence*, San Francisco, CA, pp. 734–740. Morgan Kaufmann.

Livingstone, M. S. & Hubel, D. H. 1984. Anatomy and physiology of a color system in the primate visual cortex. *The Journal of Neuroscience 4*(1), 309–356.

Lohmann, R. 1991. Selforganization by evolution strategy in visual systems. In J. D. Becker, I. Isele & F. W. Mündemann (Eds.), *Parallelism, Learning, Evolution*, pp. 500–508. Springer.

Matsushita, Y., Nishino, K., Ikeuchi, K. & Sakauchi, M. 2004. Illumination normalization with time-dependent intrinsic images for video surveillance. *IEEE Transactions on Pattern Analysis and Machine Intelligence 26*(10), 1336–1347.

Miller, J. F. 1999. An empirical study of the efficiency of learning boolean functions using a cartesian genetic programming approach. In W. Banzhaf, J. Daida, A. E. Eiben, M. H. Garzon, V. Honavar, M. Jakiela & R. E. Smith (Eds.), *Proc. of the Genetic and Evolutionary Computation Conference*, San Francisco, California, pp. 1135–1142. Morgan Kaufmann.

Schrater, P. R., Knill, D. C. & Simoncelli, E. P. 2000. Mechanisms of visual motion detection. *Nature Neuroscience 3*(1), 64–68.

Schwartz, E. L. 1977. Spatial mapping in the primate sensory projection: Analytic structure and relevance to perception. *Biological Cybernetics 25*, 181–194.

Simpson, W. A. & Manahilov, V. 2001. Matched filtering in motion detection and discrimination. *Proc. R. Soc. Lond. B 268*, 1–7.

Zeki, S. 1993. *A Vision of the Brain*. Oxford: Blackwell Science.