Learning Neural Networks for Visual Servoing using Evolutionary Methods

Nils T Siebel
Cognitive Systems Group
Institute of Computer Science
Christian-Albrechts-University of Kiel
24098 Kiel, Germany
nils@siebel-research.de

Yohannes Kassahun
Research Group Robotics
DFKI Lab Bremen
University of Bremen
28359 Bremen, Germany
kassahun@informatik.uni-bremen.de

Abstract

In this article we introduce a method to learn neural networks that solve a visual servoing task. Our method, called EANT, Evolutionary Acquisition of Neural Topologies, starts from a minimal network structure and gradually develops it further using evolutionary reinforcement learning. We have improved EANT by combining it with an optimisation technique called CMA-ES, Covariance Matrix Adaptation Evolution Strategy. Results from experiments with a 3 DOF visual servoing task show that the new CMA-ES based EANT develops very good networks for visual servoing. Their performance is significantly better than those developed by the original EANT and traditional visual servoing approaches.

1. Introduction and Related Work

Artificial neural networks are capable of modelling complex mappings between the inputs and outputs of a system up to an arbitrary precision. Another advantage of neural networks is that they can learn to solve a problem. However, with an increase in complexity of a given task the required complexity of the neural network also increases. Such a complex neural network is difficult to develop due to the high dimensionality of the space in which its parameters live. This so-called "curse of dimensionality" is a significant obstacle in machine learning problems [3]. This may be the reason why until today neural networks are not widely used to solve robot vision tasks. One of the most important robot vision tasks, visual servoing [14, 9], traditionally uses a simple P-type controller—an approach known from engineering—or sometimes, more elaborate techniques like trust-region methods [10].

There have also been visual servoing approaches using neural networks, or combined Neuro-Fuzzy approaches [7]. Many of these methods reduce the complexity of the problem (e.g. control the robot in as few as 2 degrees of freedom, DOFs) to avoid the problems of learning a complex neural network. Others use a partitioning of the workspace to learn a network of "local experts" that are easier to train [4, 8]. A neural network that controls a robot to move around obstacles is presented in [12]. The network's weights are optimised by a genetic algorithm, however, its structure (topology) is pre-defined and does not evolve.

We avoid all pre-designing of the solution. Instead, *our method learns both the structure (topology) and the parameters of the neural network* without being given any information about the nature of the problem. To achieve this, we have previously developed a method called EANT (Evolutionary Acquisition of Neural Topologies) [11]. In this article we present an improvement of EANT by use of an optimisation technique called CMA-ES [6] to *develop a neural network from scratch by evolutionary reinforcement learning* to solve the visual servoing problem.

The remainder of this article is organised as follows. In Section 2 we formulate the problem and describe our approach to the solution. Section 3 contains an experimental evaluation of the method; Section 4 concludes the article.

2. Learning Networks for Visual Servoing

2.1. Visual Servoing Setup

Our robot arm is equipped with a camera at the end-effector and has to be steered towards an object of unknown pose, see Figure 1. This is achieved in the visual feedback control loop depicted in Figure 2. In our system a neural network shall be used as the controller, determining where to move the robot on the basis of the object's visual appearance. Using the standard terminology by Weiss et al. [14] it is a "Static Image-based Look-and-Move" controller.

The object has 4 circular, identifiable markings. Its appearance in the image is described by the *image feature vec-*

Proceedings of the Sixth International Conference on Hybrid Intelligent Systems (HIS'06)
0-7695-2662-4/06 $20.00 © 2006 IEEE

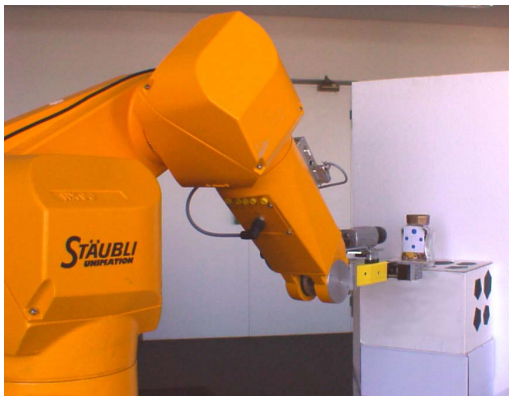

Figure 1. Robot Arm with Camera and Object

tor $y_n \in \mathbb{R}^8$ that contains the 4 pairs of image coordinates of these markings. The desired pose relative to the object is defined by the object's appearance in that pose by measuring the corresponding *desired image features* $y^\star \in \mathbb{R}^8$ ("teaching by showing"). Object and robot are then moved so that no Euclidean position of the object or robot is known to the controller. The input to the controller is the image error $\Delta y_n := y^\star - y_n$ and additionally the 2 distances in the image of the opposing markings 1-3 and 2-4, resulting in a 10-dimensional input vector. The output of the controller/network is a relative movement of the robot in the camera coordinate system: $(\Delta x, \Delta y, \Delta z)$.

The neural network is developed by reinforcement learning in a simulation of the visual servoing scenario. For the assessment of the performance ("fitness") of a neural network N it is tested by evaluating it on 937 different robot poses. In each of these poses it was given the corresponding input and its output is executed by a simulated robot. The error measure $e(N)$ is then calculated as follows:

$$e(N) := \sum_{i=1}^{937} \left(\sum_{j=1}^{4} \|(y^\star)_{2j,2j+1} - (y_i)_{2j,2j+1}\|_2^2 + b(y_i) \right) \quad (1)$$

where y_i denotes the new image features after executing one robot movement starting at test pose i and $(y)_{2j,2j+1}$ shall denote the vector comprising of the $2j$th and $2j+1$th component of an image feature vector y. The inner sum of (1) thus sums up the squared deviations of the 4 marker positions in the image. $b(y)$ is an additional "badness" function that adds to the visual deviation error an additional positive measure to punish potentially dangerous situations. If the robot moves such that features are not visible in the image or the object is touched by the robot, $b(y) > 0$, otherwise $b(y) = 0$. The CCD chip in this simulation measures $\frac{8}{3}$ mm $\times$ 2 mm which means that $e(N)$ can reach values greater than 25,000 for bad networks.

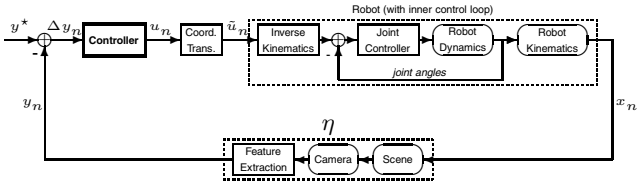

Figure 2. Visual Feedback Control Loop

2.2. Developing Neural Networks with EANT

Evolutionary Acquisition of Neural Topologies ("EANT") [11] is an evolutionary reinforcement learning system that is suitable for learning and adaptation to the environment through interaction. It combines the principles of neural networks, reinforcement learning and evolutionary methods. EANT uses a unique genetic encoding that uses a linear genome of genes (nodes). A node can be a neuron, an input to the neural network or a jumper connecting two neurons. Jumper genes can encode either forward or recurrent connections.

The linear genome encodes the topology of the neural network implicitly by the order of its elements. This enables one to evaluate the represented neural controller without decoding it. The linear genome is *complete* in that it can represent any type of neural network. It is a *compact* encoding of neural networks since the length of the linear genome is the same as the number of synaptic weights in the neural network. It is also *closed* under structural mutation and under a specially designed crossover operator.

EANT starts with initial structures that are generated using either the grow or the full method [2]. Initial structures can be chosen to be minimal. An initial network than has no hidden layers or jumper connections, only 1 neuron per output, each of them connected to all inputs. EANT gradually develops the simple initial network structure further using an evolutionary method. On a larger scale new neural structures are added to a current generation of networks. We call this "exploration" of new structures. On a smaller scale the current individuals (structures) are optimised by changing their parameters: "exploitation". Both of these optimisation loops are implemented as evolutionary processes [5]. The search stops when a neural controller with the necessary optimal structure that solves a given task is obtained.

EANT is closely related to the methods "GNARL" [1] and "NEAT" by Stanley and Miikkulainen [13]. Compared to these methods, EANT has the advantages of evaluating the network without decoding the genome that represents it, and it realises both adaptive parameter mutation (unlike NEAT) and closedness under crossover operation (unlike GNARL). More details on EANT can be found in [11].

2.3. Combining EANT with CMA-ES

Until now, EANT was only applied to learning simple control problems where is has shown a very good performance and convergence rate (that is, it learns good solutions with only few evaluations of the fitness function). Our task, visual servoing, is much more complex and therefore required some improvements to EANT on different levels.

In order to study the behaviour of EANT on large problems we implemented a visual servoing simulator for the setup described in Section 2.1, with 10/3 network in-/outputs. As a fitness function for the individual networks N we used the negative of the error measure $e(N)$ defined in (1). This means that fitness takes on negative values with $e(N) = 0$ being the optimal solution. Some of the studies were carried out on a GNU/Linux PC cluster. This parallelisation is helpful because of the significant amount of CPU time needed for the simulation (937 simulated robot movements and image acquisitions per evaluation of $e(N)$).

The results from some initial studies have shown that EANT develops structures of increasing complexity and performance. However, we were not satisfied with the speed at which performance improved with network complexity. We have therefore searched for a replacement for the optimisation method used in the exploitation part of EANT. For our new exploitation loop we used a global optimisation method called "CMA-ES" (Covariance Matrix Adaptation Evolution Strategy) [6] which is based on an evolutionary method like the one used in EANT's exploitation. CMA-ES includes features that improve its convergence especially with multi-modal functions in high-dimensional spaces.

3. Experimental Evaluation

3.1. Experimental Setup

In the following we will show results from experiments with the new exploitation strategy that uses CMA-ES. We will compare the results to the ones obtained with the original EANT method. With the setup as before, 10/3 in-/outputs and starting from minimal structures (3 output nodes, each connected to all 10 inputs) EANT was run again on our GNU/Linux cluster. One master process was responsible for the exploration of new structures and distributed the CMA-ES optimisation of the individuals (exploitation) to slave processes. The following parameters were used:

- up to 20 individuals in the exploration of new structures (global population size)
- mutation rate (probability for structural change) 50 %
- random addition of forward connections enabled, recurrent connections and crossover operator disabled

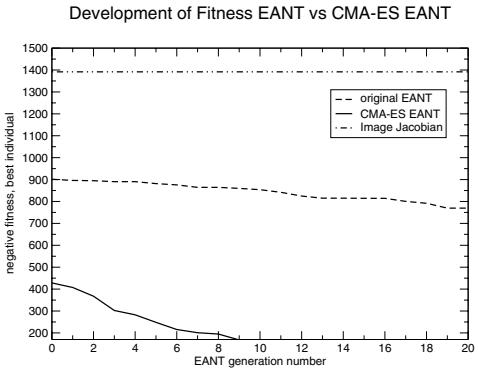

Figure 3. EANT vs CMA-ES EANT

- new hidden neurons connected to approx. 50 % of inputs ($\Rightarrow$ stochastic search for new structures)
- 3 parallel optimisations of the same individual
- up to 2 optimisation results of the same individual may be kept, so that a single structure cannot take over the whole population in less than 5 generations (very unlikely; it has not happened)

The original EANT algorithm was run with following parameters (listed are those that differ):

- new hidden neurons connected to all inputs
- exploitation: population size 7 N or 14 N, N being the network size, 15 or 30 generations; adapted as above

3.2. Results and Discussion

Figure 3 shows the development of the fitness value. Again we plot the negative fitness, this time against the EANT exploration generation since this is the determining factor for the complexity of the networks. It can be clearly seen that our version of EANT which uses CMA-ES in the exploitation of structures converges much faster to networks with a good fitness. While with the original EANT approach the fitness develops from around -900 to around -750 in 20 generations, our method reaches a value above -200 after only 9 generations. However, it should be noted that our CMA-ES exploitation uses many more function evaluations (fevals) per generation than the original one, resulting in longer running times. On the other hand, re-optimisation attempts with the original EANT exploitation that allowed for more fevals did not improve the performance significantly. It can therefore be assumed that the main performance increase stems from the techniques within CMA-ES and not from a larger allowance of fevals.

For comparison we also calculated the fitness value of the traditional Image Jacobian approach. The robot movement was calculated using the (undamped) product of the Image Jacobian's pseudoinverse with the negative image error, a standard method [9]. The resulting fitness is -1,391.72 (the fitness without moving the robot is -2,720.41). Since both the Image Jacobian and our networks calculate the necessary camera movement to minimise the image error in *one* step this is a meaningful comparison and shows that these networks can indeed be used for visual servoing. Dampening, as necessary, can be employed independent of the method that calculates the step.

4. Conclusions and Future Work

Our aim was to develop neural networks automatically that can be used as a controller in a visual servoing scenario. In order to achieve this with a minimum of predetermined modelling an evolutionary method called *EANT*, *Evolutionary Acquisition of Neural Topologies*, was used. EANT uses evolutionary search methods on two levels: in an outer optimisation loop, *exploration*, new networks are developed by gradually adding new structures to an initially minimal network. In an inner optimisation loop, *exploitation*, the parameters of current networks are optimised. EANT was used with a complete simulation of a visual servoing scenario to learn neural networks by reinforcement learning.

Based on initial experiments we decided to improve EANT's exploitation part. This loop was replaced with a new strategy that uses CMA-ES as its optimisation algorithm. Experiments with this new method have shown much improved results over the original EANT method for developing a visual servoing controller. The performance is also significantly better than the traditional visual servoing approach.

Our experimental results indicate that the new EANT method with CMA-ES is promising and bears further investigation and development. Our future work will be to investigate how the CMA-ES optimisation can be combined with more elaborate stop criteria to avoid spending a lot of CPU time on individuals that do not seem to converge to a promising solution. We also plan to investigate more closely the impact of optimisation parameters, e.g. CMA-ES's population size, on the convergence speed.

Acknowledgements

This work was in part supported by the the European Community, grant COSPAL (IST-2003-004176), which is gratefully acknowledged[1]. The authors also wish to thank Nikolaus Hansen, the developer of CMA-ES, for his kind support which helped us to quickly start applying his method.

[1]However, this paper does not necessarily represent the opinion of the European Community, and the European Community is not responsible for any use which may be made of its contents.

References

[1] P. J. Angeline, G. M. Saunders, and J. B. Pollack. An evolutionary algorithm that constructs recurrent neural networks. *IEEE Transactions on Neural Networks*, 5:54–65, 1994.

[2] W. Banzhaf, P. Nordin, R. E. Keller, and F. D. Francone. *Genetic Programming: An Introduction on the Automatic Evolution of Computer Programs and Its Applications.* Morgan Kaufmann, San Francisco, USA, 1998.

[3] R. E. Bellman. *Adaptive Control Processes*. Princeton University Press, Princeton, USA, 1961.

[4] W. Blase, J. Pauli, and J. Bruske. Vision-based manipulator navigation using mixtures of RBF neural networks. In *International Conference on Neural Network and Brain*, pages 531–534, Bejing, China, April 1998.

[5] Á. E. Eiben and J. E. Smith. *Introduction to Evolutionary Computing*. Springer Verlag, Berlin, Germany, 2003.

[6] N. Hansen and A. Ostermeier. Completely derandomized self-adaptation in evolution strategies. *Evolutionary Computation*, 9(2):159–195, 2001.

[7] K. Hashimoto, editor. *Visual Servoing: Real-Time Control of Robot Manipulators Based on Visual Sensory Feedback*, volume 7 of *Series in Robotics and Automated Systems*. World Scientific Publishing Co., Singapore, 1994.

[8] G. Hermann, P. Wira, and J.-P. Urban. Neural networks organizations to learn complex robotic functions. In *Proceedings of the 11th European Symposium on Artificial Neural Networks (ESANN 2003)*, pages 33–38, Bruges, Belgium, April 2005.

[9] S. Hutchinson, G. Hager, and P. Corke. A tutorial on visual servo control. Technical report, Yale University, New Haven, USA, May 1996.

[10] M. Jägersand. Visual servoing using trust region methods and estimation of the full coupled visual-motor Jacobian. In *Proceedings of the IASTED Applications of Control and Robotics, Orlando, USA*, pages 105–108, January 1996.

[11] Y. Kassahun and G. Sommer. Efficient reinforcement learning through evolutionary acquisition of neural topologies. In *Proceedings of the 13th European Symposium on Artificial Neural Networks (ESANN 2005)*, pages 259–266, Bruges, Belgium, April 2005.

[12] D. E. Moriarty and R. Miikkulainen. Evolving obstacle avoidance behavior in a robot arm. In *Proceedings of the Fourth International Conference on Simulation of Adaptive Behavior*, Cape Cod, USA, 1996.

[13] K. O. Stanley. *Efficient Evolution of Neural Networks through Complexification*. PhD thesis, Department of Computer Sciences, The University of Texas at Austin, Austin, USA, August 2004.

[14] L. E. Weiss, A. C. Sanderson, and C. P. Neuman. Dynamic sensor-based control of robots with visual feedback. *IEEE Journal of Robotics and Automation*, 3(5):404–417, October 1987.