TRANSNAV International Journal on Marine Navigation and Safety of Sea Transportation
Volume 5 Number 2 June 2011

Multirole Population of Automated Helmsmen in Neuroevolutionary Ship Handling

M. Lacki
Gdynia Maritime University, Gdynia, Poland

ABSTRACT: This paper presents the proposal of advanced intelligent system able to simulate and demonstrate learning behavior of helmsmen in ship maneuvering. Simulated helmsmen are treated as individuals in population, which through environmental sensing learn themselves to safely navigate on restricted waters. Individuals are being organized in groups specialized for particular task in ship maneuvering process. Neuroevolutionary algorithms, which develop artificial neural networks through evolutionary operations, are used in this system.

1 INTRODUCTION

One of the main tasks in Artificial Intelligence is to create the advanced systems that can effectively find satisfactory solution of given problem and improve it over time. Intelligent autonomous agents used in these systems can quickly adapt to current situation, i.e. change their behavior based on interactions with the environment (Fig. 1), become more efficient over time, and adapt to new situations as they occur.

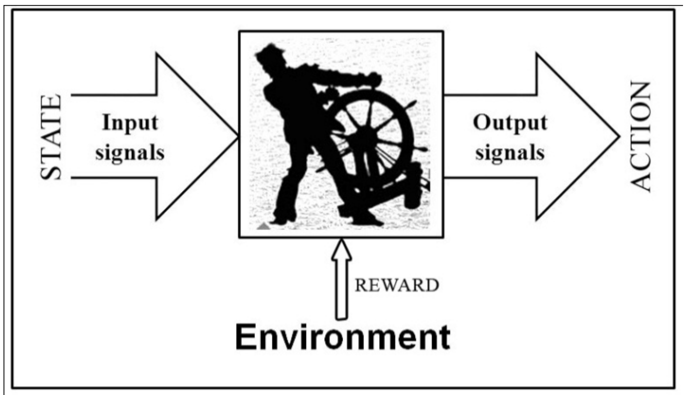

Figure 1. Interaction of helmsman with an environment.

Such abilities are very important for simulating helmsman behavior in ship maneuvering on restricted waters.

For simpler layouts learning process can be performed using classic approach, i.e. Temporal Difference Reinforcement Learning (Tesauro 1995, Kaelbling, Littman & Moore 1996) or Artificial Neural Networks with fixed structures (Braun & Weisbrod 1993). Dealing with high-dimensional spaces is a known challenge in Reinforcement Learning approach (Łącki 2007) which predicts the long-term reward for taking actions in different states (Sutton & Barto 1998).

Evolving neural networks with genetic algorithms has been highly effective in advanced tasks, particularly those with continuous hidden states (Kenneth & Miikkulainen 2005). Neuroevolution gives an advantage from evolving neural network topologies along with weights which can effectively store action values in machine learning tasks. The main idea of using evolutionary neural networks (ENN) in ship handling is based on evolving population of helmsmen.

The neural network is the helmsman's brain making him capable of choosing action regarding actual navigational situation of the vessel which is represented by input signals. These input signals are calculated and encoded from current situation of the environment.

In every time step the network calculates its output from signals received on the input layer. Output signal is then transformed to one of available actions influencing helmsman’s environment. In this case the vessel on route within the restricted waters is part of the helmsman’s environment. Main goal of the agents is to maximize their fitness values. These values are calculated from helmsmen behavior during simulation. The best-fitted individuals become parents for next generation.

2 NEUROEVOLUTION OF AUGMENTING TOPOLOGIES

Neuroevolution of Augmenting Topologies (NEAT) method is one of the Topology and Weight Evolving Artificial Neural Networks (TWEANN's) method (Kenneth & Miikkulainen 2002). In this method the whole population begins evolution with minimal networks structures and adds nodes and connections to them over generations, allowing complex problems to be solved gradually starting from simple ones.

The NEAT method consists of solutions to three main challenges in evolving neural network topology:

1 Begin with a minimal structure and add neurons and connections between them incrementally to discover most efficient solutions throughout evolution.
2 Cross over disparate topologies in a meaningful way by matching up genes with the same historical markings.
3 Separate each innovative individual into a different species to protect it disappearing from the population prematurely.

2.1 *Genetic Encoding*

Evolving structure requires a flexible genetic encoding. In order to allow structures to increase their complexity, their representations must be dynamic and expandable (Braun & Weisbrod 1993). Each genome in NEAT includes a number of inputs, neurons and outputs, as well as a list of connection genes, each of which refers to two nodes being connected (Fig. 2).

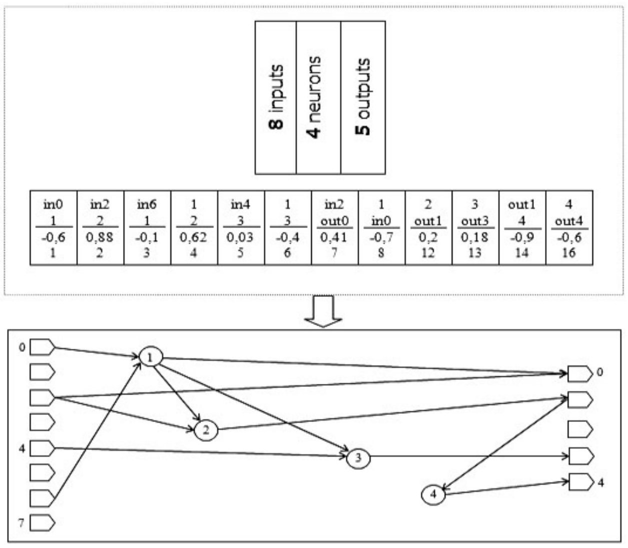

Figure 2. Genotype and phenotype of evolutionary neural network.

In this approach each connection gene specifies the output node, the input node, the weight of the connection, and an innovation number, which allows finding corresponding genes during crossover. Connection loopbacks are also allowed, as shown in figure 2.

2.2 *Mutation*

Mutation in evolutionary neural networks can change both connection weights and network structures (Fig. 3). Connection weights mutate as in any neuroevolutionary system, with each connection either perturbed or not.

Structural mutations, which form the basis of complexity, occur in two ways. Through mutation the genome can be expanded by adding genes or shrunk by removing them. In the add connection mutation, a single new connection gene is added connecting two previously unconnected nodes. In the add node mutation, the new node is placed, thus allowing to create new connections in future possible mutations.

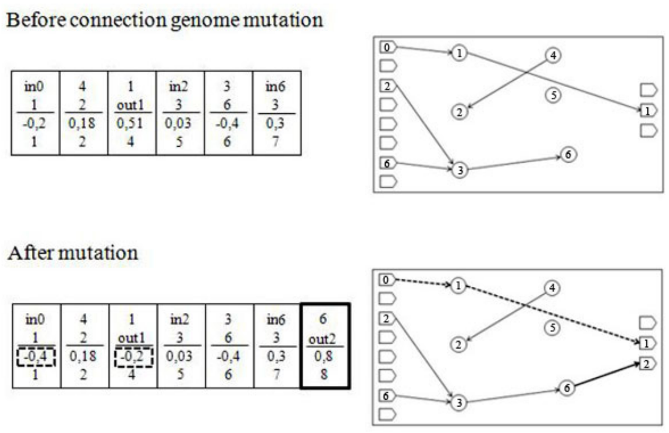

Figure 3. An example of weights and connection mutation in connections genome.

2.3 *Crossover*

Through innovation numbers, the system knows exactly what genes match up with each other. Unmatched genes are either disjoint or excess, depending on whether they occur within or outside the range of the other parent's innovation numbers.

In crossing over operation (Fig. 4), the genes with the same innovation numbers are lined up. The offspring is then formed in one of two ways: in uniform crossover, matching genes are randomly chosen for the offspring genome. In blended crossover, the connection weights of matching genes are averaged. These two types of crossover were found to be most effective in ENN in extensive testing compared to other crossover methods (Kenneth & Miikkulainen 2002).

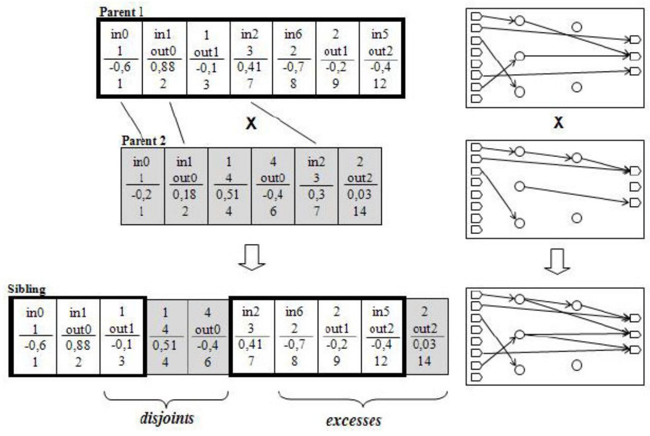

Figure 4. An example of crossover operation of equally fitted parents.

The disjoint and excess genes are inherited from the more fitted parent or from both parents, if they are equally fitted. Disabled genes (with zeroed weight) have a chance of being re-enabled during mutation, allowing networks to reuse older genes once again.

Evolutionary neural network can store history tracks of every gene in the population, allowing matching genes to be found and linked-up together even in different genome structures. Old behaviors encoded in the pre-existing network structure are not destroyed and remain qualitatively the same, while the new structure provides an opportunity to elaborate on these original behaviors.

In agent learning process, the genomes in NEAT will gradually get larger through mutation. Genomes of different sizes will sometimes result with different connections at the same positions. Any crossover operator must be able to recombine networks with differing topologies, to pass agents skills to next generations in meaningful way. Historical markings represented by innovation numbers allow NEAT to perform crossover without analyzing topologies. Genomes of different organizations and sizes stay compatible throughout evolution, and the variable-length genome problem is essentially avoided. This methodology allows NEAT to increase the complexity of the structure while different networks still remain compatible.

2.4 *Speciation*

Speciation of population can be seen as a result from the same process as adaptation (Beyer & Schwefel 2002), natural selection exerted by interaction among organisms, and between organisms and their environment (Spears 1995). Divergent adaptation of different populations would lead to speciation. Speciation of the population assures that individuals compete primarily within their own niches instead of competition within the whole population. In this way topological innovations of neural network are protected and have time to optimize their structure before they have to compete with other experienced agents in the population.

Generally, during species assigning process, as described in (Łącki 2009a), when a new agent appears in population, its genome must be assigned to one of the existing species. If this new agent is structurally too innovative comparing to any other individuals in whole population, the new species is created.

Compatibility of agent's genome *g* with particular species *s* is estimated accordingly to value of distance between two individuals. This distance is calculated with formula 1:

$$\delta = \frac{c_1 E}{N} + \frac{c_2 D}{N} + c_3 \overline{W} \qquad (1)$$

where: c1; c2; c3 - weight (importance) coefficients; E - number of excesses; D - number of disjoints; $\overline{W}$ - average weight differences of matching genes; N – the number of genes in the larger genome.

There must be estimated a compatibility threshold δ_t at the beginning of the simulation and if $\delta \leq \delta_t$ then genome *g* is placed into this species. One can avoid the problem of choosing the best value of δ by making δ_t dynamic. The algorithm can raise δ_t if there are too many species in population, and lower δ_t if there are too few.

2.5 *Fitness sharing*

Fitness sharing occurs when organisms in the same species must compete with each other for life-sustaining resources of their niche. Thus, a species cannot afford to become too big even if many of its individuals perform well.

Therefore, any one species is unlikely to take over the entire population, which is crucial for speciated evolution to maintain topological diversity. The adjusted fitness $f_i^{'}$ for individual *i* is calculated according to its distance δ from every other individual *j* in the population:

$$f_i^{'} = \frac{f_i}{\sum_{j=1}^{n} sh(\delta(i,j))} \qquad (2)$$

where: f_i – fitness value of individual *i*; *sh* - sharing function; n - number of individuals in whole population; - average weight differences of matching genes; $\delta(i,j)$ – distance between individuals *i* and *j*.

The sharing function *sh* is set to 0 when distance δ(i,j) is above the threshold δ_t; otherwise, sh(δ(i,j)) is set to 1 (Spears 1995). Thus, sum of *sh* calculates the number of organisms in the same species as individual i. This reduction is natural since species are already clustered by compatibility using the threshold δ_t. A potentially different number of offspring is assigned to every species. This number is proportional to the calculated sum of adjusted fitness values f_i' of its members.

In the first step of species reproduction process the system eliminates the lowest performing members from the population. In the next step the entire population is replaced by the offspring of the remaining organisms in each species (Fig. 5).

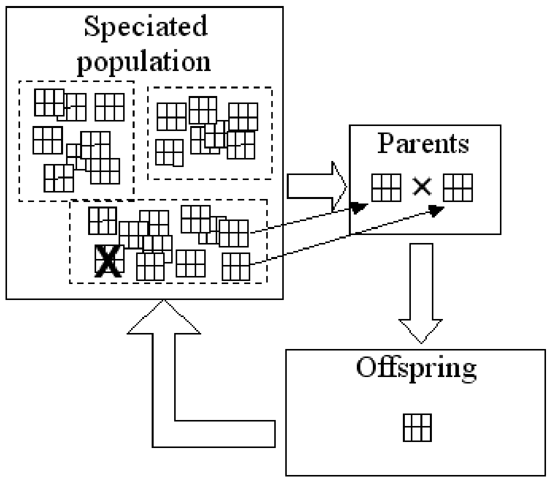

Figure 5. Evolution within one species in speciated population.

The other selection methods in speciated population are also considered in future research, i.e. island selection or elite selection of best fitted individuals of every species with particular task.

The final effect of speciating the population is that structural innovations are protected.

3 MULTIROLE SHIP HANDLING WITH EVOLUTIONARY NEURAL NETWORK

The main goal of authors work is to make a system able to simulate a set of navigational situations of ship maneuvering through a restricted coastal area. This goal may be achieved with Evolutionary Neural Networks (ENNs).

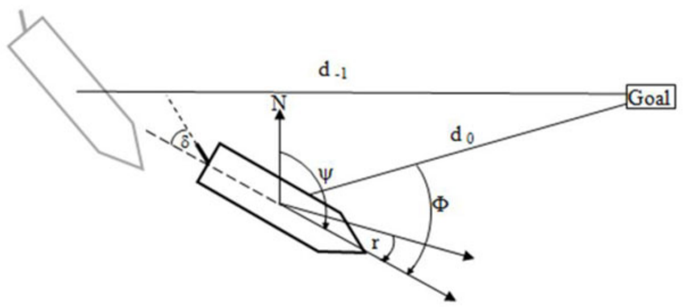

Figure 6. Sample data signals of ship handling with ENN.

Navigational situation of a moving vessel can be described in many ways. Most important is to define proper state vector from abundant range of data signals (Fig. 6) and arbitrary determine fitness function values received by the agent. Fitness calculation is of primary meaning when determining the quality of each agent. Subsequently it defines helmsman's ability to avoid obstacles while sailing toward designated goal.

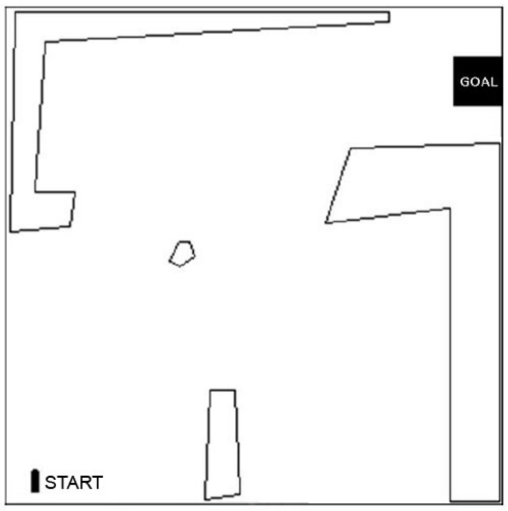

Figure 7. Model of simulated simplified coastal environment.

In the simplified simulation model there are no moving vessels in the area (Fig. 7) (Łącki 2009b).

Helmsman observes current situation which is mapped into input signals for his neural network. In general considered input signals indicating (i.e.):

- The ship is on the collision course with an obstacle,
- Ships course over ground,
- Ships angular velocity,
- Distance to collision,
- The ship is approaching destination,
- Ships angle to destination,
- The ship is heading out of the area,
- Danger has increased,
- Danger has decreased,
- Ship is heading on goal.

All the input signals are encoded binary (Łącki 2010a). Neural network output value is the rudder angle. It is crucial for effectiveness of simulation to determine the number of neural network outputs.

More outputs mean more calculations but on the other hand better accuracy and fidelity of designed environment. Additionally too many outputs increase complexity of the learning process, thus making an agent unable to quickly adapt to new situations. This accuracy vs. performance dilemmas were examined extensively in previous works (Łącki 2008-2010).

The fitness value of an individual is calculated from arbitrary set action values, i.e.:

- -1 for increase of the distance to goal in every time step,
- -10 when ship is on the collision course (with an obstacle or shallow waters),
- +10 when she's heading to goal without any obstacles on course,
- -100 when she hits an obstacle or run aground,
- +100 when ship reaches a goal,
- -100 when she departs from the area in any other way, etc.

In the simplified simulation model speed of the ship was constant. In the advanced model, in which one considers a set of possible task, there must be possibility for the agent to adjust speed of the vessel. Situation evaluation can be treated as multi-criteria problem which calculates a danger of getting stranded on shallow water, encountering a vessel with dangerous cargo, getting to close to shore, etc. It can be estimated with available optimization algorithms (Filipowicz, Łącki & Szłapczyńska 2005) with function of ship's position, course and angular velocity, information gained from other vessels (if considered in the model) and coastal operators.

3.1 *Multirole ship handling system*

Situation Determining Unit (SDU) is an intelligent module responsible for grouped helmsmen management (Fig. 8).

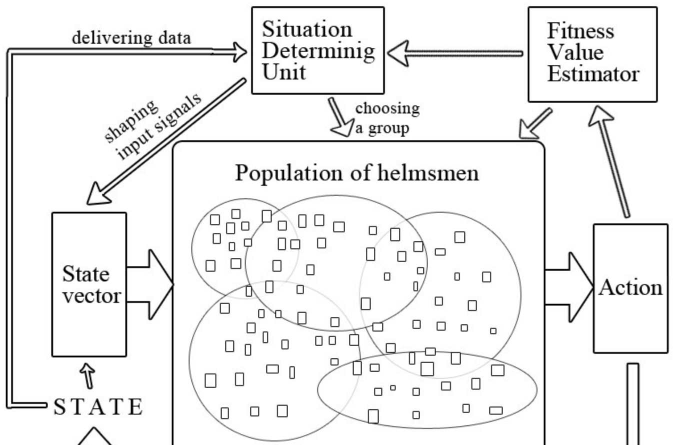

Figure 8. Complementary multirole neuroevolutionary ship handling system.

This unit constantly receives information from state of the environment, processes it, determines actual navigational situation and regarding it chooses the best fitted group of helmsmen for this task. SDU is designed with neuroevolutionary system. The best performing neural network is the one making decisions. Its performance is determined regarding fitness values reinforcing the neural network by Fitness Value Estimator. With elite selection method this neural network participates in creation of new generation of SDU's.

State vector is dynamic in that its signals depend on current navigational situation chosen by SDU. Set of available actions is determined in the same way. An example of input and output signals for general navigation task is presented on figure 9.

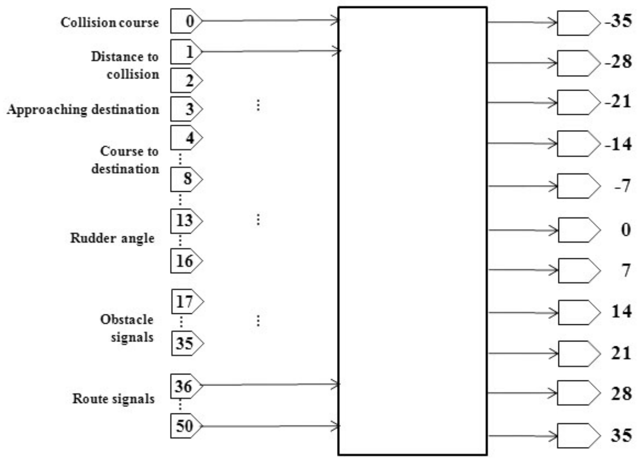

Figure 9. Input and output signals in general navigation task.

List of considered possible tasks:

- General navigation – during this task SDU observes possible collision warnings. All the vessels are treated as points. Action: coarse rudder angle.
- Collision avoidance – The vessels involved in collision situation are treated as 2D objects. Action: precise rudder angle, propeller thrust control,
- Turning – Action: rudder deflection, propeller thrust and bow/stern thrusters' control,
- Mooring – Action: propeller thrust and bow/stern thrusters' control, rudder deflection.

Two different multirole models of restricted waters environment can be taken into consideration in the advanced system:

- Single vessel multirole population simulation,
- Multi-task multi-agent simulation.

In the first model the main goal is to train population of helmsmen to safely handle a particular model of a ship in specified single multirole task (i.e. safe passage trough congested water channel, approach to dock and mooring). In this case agents compete simultaneously with each other in one population (may be grouped into species) to handle a ship as best as possible. Any other vessels in the area are treated as stationary or moving objects on reasonably predictable routes (Fig 10).

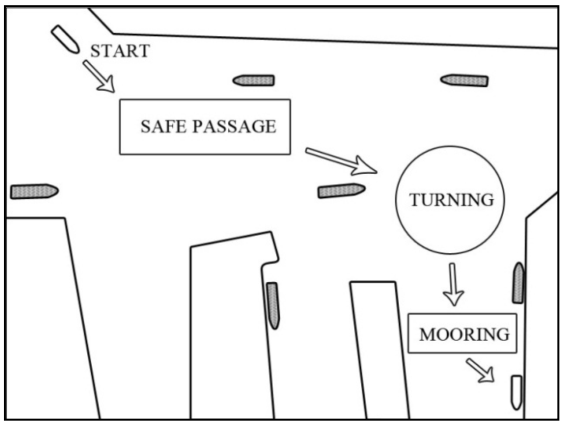

Figure 10. Example of an environment with simultaneous multirole population of helmsman.

The second model consist a small number of different vessels in the same restricted waters with helmsmen as the best trained agents for particular ships (Fig. 11). In this case every helmsman has different goal and different environmental situation, depending on actions taken by other helmsmen on the other vessels.

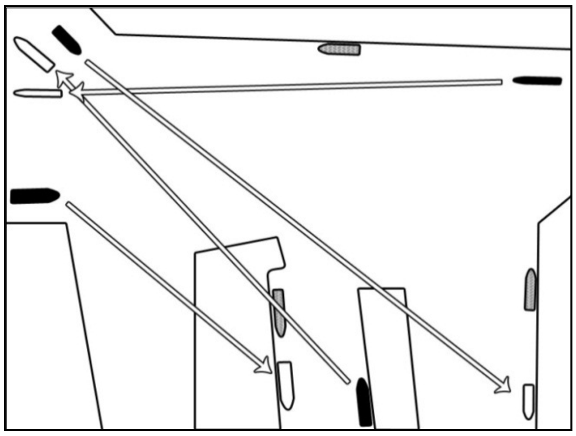

Figure 11. Example of multi-task multi-agent environment. There are four agents allocated on four vessels. Starting points of the vessels are indicated with black outlines while goals for them are marked with white ones.

4 REMARKS

Neuroevolution approach to intelligent agents training tasks can effectively improve learning process of simulated helmsman behavior in ship handling (Łącki 2008). Neural networks based on NEAT increase complexity of considered model of ship maneuvering in restricted waters.

Implementation of multirole division of helmsmen population in neuroevolutionary system allows simulating complex agents' behavior in the environments with much larger state space than it was possible in a classic state machine learning algorithms (Łącki 2007). In this system it is also very important to change parameters of genetic operations dynamically as well as the input signals vector and set of available actions.

REFERENCES

Beyer, H.-G. & Schwefel, P. H. 2002. *Evolution strategies – A comprehensive introduction*. Natural Computing, 1(1):3–52.

Braun, H. & Weisbrod, J. 1993. *Evolving feedforward neural networks*. Proceedings of ANNGA93, International Conference on Artificial Neural Networks and Genetic Algorithms. Berlin: Springer.

Chu T. C., Lin Y. C. 2003. *A Fuzzy TOPSIS Method for Robot Selection*, the International Journal of Advanced Manufacturing Technology: 284-290,

Filipowicz, W., Łącki, M. & Szłapczyńska, J. 2005, *Multicriteria decision support for vessels routing,* Proceedings of ESREL'05 Conference.

Kaelbling, L. P., Littman & Moore. 1996. *Reinforcement Learning: A Survey.*

Kenneth, O.S., & Miikkulainen, R. 2002. *Efficient Evolution of Neural Network Topologies*, Proceedings of the 2002 Congress on Evolutionary Computation, Piscataway.

Kenneth, O.S. & Miikkulainen R. 2005. *Real-Time Neuroevolution in the NERO Video Game*, Proceedings of the IEEE 2005 Symposium on Computational Intelligence and Games, Piscataway.

Łącki, M. 2007 *Machine Learning Algorithms in Decision Making Support in Ship Handling*, Proceedings of TST Conference, Katowice-Ustroń, WKŁ.

Łącki, M. 2008, *Neuroevolutionary approach towards ship handling*, Proceedings of TST Conference, Katowice-Ustroń, WKŁ.

Łącki, M. 2009a, *Speciation of population in neuroevolutionary ship handling*, Marine Navigation and Safety of Sea Transportation, CRC Press/Balkema, Taylor & Francis Group, Boca Raton – London - New York - Leiden, p. 541-545.

Łącki, M. 2009b, *Ewolucyjne sieci NEAT w sterowaniu statkiem*, Inżynieria Wiedzy i Systemy Ekspertowe, Akademicka Oficyna Wydawnicza EXIT, Warszawa, p. 535-544.

Łącki, M. 2010a, *Wyznaczanie punktów trasy w neuroewolucyjnym sterowaniu statkiem*. Logistyka, No 6.

Łącki, M. 2010b, *Model środowiska wieloagentowego w neuroewolucyjnym sterowaniu statkiem*. Zeszyty Naukowe Akademii Morskiej w Gdyni, No 67, p. 31-37.

Spears, W. 1995. *Speciation using tag bits*. Handbook of Evolutionary Computation. IOP Publishing Ltd. and Oxford University Press.

Stanley, K. O. & Miikkulainen, R. 2002. *Efficient reinforcement learning through evolving neural network topologies*. Proceedings of the Genetic and Evolutionary Computation. Conference (GECCO-2002). San Francisco, CA: Morgan Kaufmann.

Stanley, K. O. & Miikkulainen, R. 2005. *Real-Time Neuroevolution in the NERO Video Game*, Proceedings of the IEEE 2005 Symposium on Computational Intelligence and Games, Piscataway.

Sutton, R. 1996. *Generalization in Reinforcement Learning: Successful Examples Using Sparse Coarse Coding*. Touretzky, D., Mozer, M., & Hasselmo, M. (Eds.), Neural Information Processing Systems 8.

Sutton, R. & Barto, A. 1998. *Reinforcement Learning: An Introduction.*

Tesauro, G. 1995. *Temporal Difference Learning and TD-Gammon*, Communications of the Association for Computing Machinery, vol. 38, No. 3.