2008 American Control Conference
Westin Seattle Hotel, Seattle, Washington, USA
June 11-13, 2008

Automated Modeling of the Guidance of a K-9 *

Winard Britt †, David M. Bevly ‡, and Gerry Dozier§

Abstract

This paper attempts to automate and replace human guidance in the control of a K-9 unit by modeling that guidance from observation. The ultimate research goal seeks to contribute toward the autonomous command of a trained K-9 unit by analyzing the movement and the behavior of the dog as it responds to command tones. Specifically, GPS and command signal information (from a human trainer) is recorded as a canine follows (or fails to follow) instructions as it moves toward a destination. The data is then processed into training instances and used as training data for a General Regression Neural Network (GRNN). Then, the network is used to classify previously unseen test instances to determine if the behavior at that moment is normal or anomalous (in need of correcting tones). Both representation of training instances and the system parameters of the GRNN are optimized using a simple Evolutionary Hill-Climber (EHC). Given even fairly limited initial data for training, the system performs well, producing relatively few false positives and false negatives in classification.

1. INTRODUCTION

Trained canines have historically proven very effective in a wide range of potentially dangerous security applications such as tracking and the detection of people, drugs, and explosives. However, canine units generally require the guidance of humans with some expertise in order to perform their tasks. Specifically, most trained canines act as an augmentation to existing human teams, rather than autonomous units in and of themselves. Relatively autonomous units, such as robots, suffer from a number of deficiencies such as limited sensors and often weak navigation intelligence. Canines have sophisticated senses and automatically perform their own obstacle avoidance.

The ultimate goal of this research is to develop algorithms which utilize information available from sensors on-board the canine to provide command and control signals for the purpose of autonomously directing the dog to waypoint(s). Once the dog is deployed for a given path, all command information should be produced using data collected on the system, without the need for human guides or trainers. The specific contribution of this work is to attempt to automatically identify anomalous behavior in the canine (as a human trainer would do), using GPS and sensor information.

In essence, determining whether or not a canine is behaving correctly at any given time can be considered a modeling problem, where the model is that of a human trainer. A human trainer can quickly identify the heading, location, distance to destination, and other parameters and make a decision whether or not the canine needs to adjust course. In order to automatically determine whether or not the dog is behaving correctly, a control algorithm needs to be able to process information regarding location, heading, and movement behaviors much like a human observing the dog would. In this work, a GRNN [1] was utilized on processed data gathered on-board the canine and then used to classify future, unseen behaviors by modeling the commands given by a human trainer.

Neural Networks have been used for a wide variety of control and modeling problems, both in traditional vehicles [2] and in animals [3, 4]. The GRNN has proven to be capable at classifying accurately even with relatively few training instances and with incomplete data (in this case, since the range of possible locations and behaviors of the dog is infinite, the data will always be incomplete).

The remainder of this paper is organized as follows. In Section 2, our methods of data collection from the canine are discussed, as well as information concerning the training of the canines themselves. In Section 3,

*This work was supported by the US Office of Naval Research

†Winard Britt is a student in the Department of Computer Science and Software Engineering, Auburn University, Auburn, AL 36849, USA brittwr@auburn.edu

‡David M. Bevly is a faculty member of the Department of Mechanical Engineering, Auburn University, Auburn, AL 36849, USA bevlydm@auburn.edu

§Gerry Dozier is a faculty member of the Department of Computer Science, North Carolina A & T State University, Greensboro, NC 27411, USA doziegv@auburn.edu

the GRNN classifier algorithm is discussed in some detail, along with issues regarding their implementation. In Section 4, the optimization method (an Evolutionary Hill-Climber) used to find the best parameters for the classifier is discussed. Data representation is explained in Section 5. In Section 6 and Section 7, experiments and discussion are given to validate the approach. Finally, in Section 8 some conclusions are drawn and some areas for our continuing work are given.

2. Data Collection

Experimental studies through the Auburn University Canine and Detection Research Institute have already been performed on the ability of a human to command the K-9 to specific locations. In these experiments, a single tone was issued over a radio to inform the dog that the direction being pursued was correct (normal behavior heading toward a pre-determined goal). Once the dog made a wrong turn, the tone would be deactivated (hereafter referred to as "anomalous behavior") and the dog would begin to pursue other directions until the tone was reactivated. When asked to make one direction change during the course (either to turn left or to turn right), the K-9 made the correct decision 76 out of 79 times. When asked to make multiple direction changes (in one path) while following the static tone to a designated location and to then return to home base also following the static tone, the K-9 correctly completed 189 out of 206 total trials (one trial consisted of multiple direction options and direction changes). The results from this current work have revealed that dogs can be trained to obey audible direction-oriented commands relayed wirelessly over long distances consistently and reliably for relatively lengthy periods of time with no human contact or intervention.

In later experiments, a GPS/INS radio in conjunction with an IMU, Magnetometer, and Accelerometer located on the K-9 unit (shown in Figure 1) wirelessly communicated to a laptop data on acceleration, velocity, latitude, longitude, and heading. This information was recorded as the dog was commanded by a human trainer from a start position to a known goal. Each time, the path was in the same general area (an open field), but the paths were not the same (differing start locations and destinations). The pathways were fairly straight and had only limited obstacles (changes in elevation, holes in the ground, etc.) A corresponding tone ("on" for normal or "off" for anomaly) that was being produced by the trainer was recorded alongside the GPS/INS data. An example trial is shown in Figure 2, where the x-marks represent the dog going off path and

Figure 1. Remotely Commanded K-9 with Radio and GPS/INS Receiver

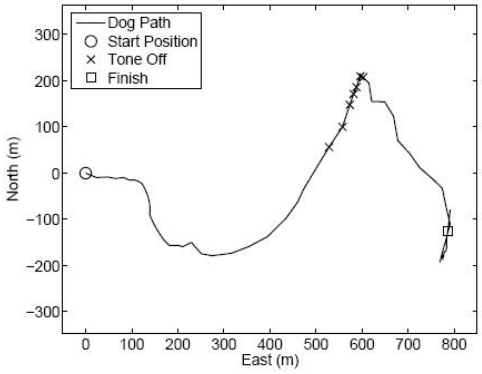

Figure 2. Example of a K-9 Path and Command Tones

the tone being turned off. This was repeated for a total of seven trials (each trial represents a single path from start to destination). The data collected from these trials was collected into text files which were processed for the purposes of training data in a neural network. In essence, the algorithm is modeling the human trainer in order to determine whether or not a canine's behavior is normal (in need of no change in tone) or anomalous (needs a change in tone).

3. The General Regression Neural Network (GRNN)

Neural networks in general attempt to emulate the biological neural networks in the brain, which depend on massive parallelism and interconnectivity in order to process information quickly.

A GRNN is a one-pass learning algorithm which attempts to approximate a continuous variable that is dependent on many representative vector training instances. This is advantageous over many other neural network strategies which require iterative learning, since they often require many iterations in order to produce workable solutions [1]. In practice, any analysis system to be deployed on an embedded system with relatively low computational resources should be made to be as simple and efficient as possible.

A complete description of the GRNN would be outside of the scope and contribution of this paper, but the interested reader is referred to [1]. A brief description follows. Elements of training instances $x_1 ... x_n$ are passed into the network and collected into pattern units (where each unit represents a single training instance). The output from the pattern units is collected in the summation units (as described below) and then the output of the summation units is ultimately collected to provide an estimate for $\hat{Y}$ corresponding to an unseen input vector X.

Each training instance in the GRNN consists of a vector X (consisting of training elements or "attributes" $x_1 ... x_n$) and a desired output Y. In order to provide an estimate of the value of $\hat{Y}$ for an unseen instance X is calculated in (1):

$$\hat{Y}(X) = \frac{\sum_{i=1}^{n} Y_i * exp(-\frac{C_i}{\sigma})}{\sum_{i=1}^{n} exp(-\frac{C_i}{\sigma})} \quad (1)$$

where n is the number of all training instances, Y_i is the desired output (in other words, the actual value observed in training) for a given training instance vector, σ is a constant parameter of the GRNN, and the value of C_i is the Euclidean distance between X and a given training instance vector, as shown in (2):

$$C_i = \sum_{i=1}^{p} |X_j - X_{ij}| \quad (2)$$

where X_j represents a single element of the training instance vector, X_{ij} represents the corresponding element in the instance to be classified, and p is the number of elements in the vector.

The effectiveness of a Neural Network depends on [5]:

1. The presence of learnable characteristics in the training instances
2. The number of training set instances
3. The representativeness of the training set
4. System parameters of the Neural Network

Our classifier attempts to carefully address each of the above factors. Learnability is believed to be present given that our devices attempt to capture the same information that a human uses to guide the canine visually. While limited, the current training data consists of a number of instances of both normal and anomalous behavior. Further, since the data comes from actual canines attempting to follow paths, the data seems likely to be representative of the problem. The system parameters (including which attributes to use and the system parameter for the GRNN) are optimized using a simple evolutionary method (described in Section 4) in a fashion that would be done off-line in a real system. In other words, the system would be optimized prior to deployment to avoid the computational cost in an actual embedded system.

3.1. Rationale

Although GRNNs can be more memory intensive than other neural network strategies (since each training instance essentially forms one "neuron" and all training instances are stored and used in the classification of new instances), the training data in this case remained compact enough to be viable for practical use. Even before optimizing the training data based on attributes (as described in Section 4), the worst case for memory usage during classification was approximately $memory = traininginstances * numberelements * elementsize$. In concrete terms, this was a worst case of about 36,400 bytes (the embedded device that will be used for the controller has about 400K of memory free for applications). Post-optimization, the memory usage dropped to roughly half of that worst-case value. The primary motivation for choosing a GRNN in this early phase was to train the classification algorithm quickly (a major advantage of the GRNN) and to discover which attributes available from the sensors were most useful (as described in Section 4). GRNNs also tend to perform reasonably well even when presented with fairly limited training data (they still perform regression well, as shown in [1]). In our case, the physical devices used to collect the training data was no longer available for us until a new sensor pack unit for the dog was fabricated. This meant that for the time being, the modelling effort had to be performed with existing data. In the continuing work for this project, other machine learning techniques are being developed and applied.

Figure 3. Evolutionary Hill-Climber

```
t = 0
Initialize candidate solution i
Evaluate i
while (i's fitness < desired fitness)
    choose a step location
    evaluate step location
    if step yields better fitness
        replace i with new individual
```

4. The Evolutionary Hill-Climber Algorithm (EHC)

In order to discovery an appropriate value of the GRNN system parameter σ and to determine which attributes from the training instances should be included (in other words, what elements comprise the vector X in Section 3), a simple Evolutionary Hill-Climber was utilized.

4.1. Algorithm Description

Traditional deterministic Hill-Climbing methodologies tend to inadequately deal with complex search spaces and instead get stuck at local minima [6]. Instead, stochastic methods of search often prove preferable in instances where there exists no guarantee of smoothness or of singular maxima in the function to be optimized [7]. In optimizing σ and the attributes for the GRNN, there are no such guarantees. The Evolutionary Hill-Climber (EHC) as outlined in Figure 3, based loosely on the Random-Mutation Hill-Climber described in [8], first randomly creates a single candidate solution consisting of a chromosome with a number of values corresponding to parameters being optimized (in this case, the σ value and whether or not to include each of the thirteen possible attributes described later). Initial values are chosen from within the allowable range dictated by the problem type (in this case, the range of values for the σ was chosen from the range (0,1.0] and each of the attributes could either be considered "included" or "not included"). This candidate solution is then evaluated by a fitness function and the fitness is assigned to that individual.

On each iteration, a single Uniform Mutation (taken from the range [-1,1]) is added to the σ value, multiplied by the mutation amount δ (chosen to be 0.25 for these experiments) and the starting value of the gene itself. There is small chance (0.15, for each attribute) that an attribute bit will be reversed (effectively deleting or adding an attribute from consideration). This new candidate solution is then evaluated. If the new candidate solution has better fitness than the previous, the previous is replaced. However, if the new candidate solution has worse fitness than the previous, it is rejected.

The overhead associated with the EHC is quite low on each iteration, requiring only a handful of elementary operations. The primary motivation for adding or deleting attributes is to generate a minimal set of attributes which will provide the most accurate results. Having fewer attributes has the added benefit of reducing the complexity of the GRNN each time it is used to classify.

4.2. Fitness Function

In order to evaluate the fitness of a candidate solution, the GRNN is run with the indicated attributes and value of σ on test instances that were *not* used as training data. The fitness formula is given by:

$$fitness = C - (3 * f_n + f_p) \tag{3}$$

In (3), C is the number of instances correctly classified by the GRNN, f_n is the number of false negatives (anomalies incorrectly identified as normal behavior), and f_p is the number of false positives (normal behaviors incorrectly identified as anomalies). The fitness function was biased against false negatives in order to promote networks that more effectively identified anomalous behavior in the canine.

In order to break ties in cases where the success rates for two candidate solutions were equal, the average distance from the desired output to the resultant output over all the test cases was used.

5. Representation of Instances

What information and how that information is represented obviously effects the ability of the system to correctly classify behaviors. This section discusses the raw data from the device on the canine and the processing done to that data to make it most useful to the GRNN.

5.1. Raw Data

Many times a second (approximately 60 Hertz), the mobile system reports the measurements (the entire set of which is referred to as an "entry"), as shown in Table 1.

For the purposes of offline training, only about one in ten measurement entries were taken to be processed as training data for the GRNN, essentially reducing the sample rate from the sensor data to 6 Hertz. In practice, the changes in the recorded values was low enough

Name	Units
ax, ay, az	g
gx, gy, gz	(deg/s)
velocity	(m/s)
heading	(deg)
latitude, longitude	(deg)
signal	unitless

Table 1. Raw Measurements

due to the dog's relatively low speed that it was unnecessary to record at the maximum rate possible. Further, it was undesirable to record large amounts of redundant data due to memory concerns. It is also worth noting that even if the dog's behaviors could be classified faster than this, it is undesirable to issue the command tones given to the dog too rapidly because it will cause the K-9 confusion.

In all, there were 7 slightly different paths (each of which took between 20 and 35 seconds for the dog to travel) that were measured which produced between 200 and 350 raw entries (differing paths differed slightly in length). Of those seven, two contained no anomalous behavior while the other five all contained some anomaly which caused the tone to be removed while the dog found the new path.

5.2. Data Processing

In order to produce training instances, the raw data was processed into a series of derived metrics. The transformations were made using only data available at the point of the measurement (no future knowledge) and were generally differences between two raw data entries to illustrate potential anomalies through unusually large or small changes. For example, very large changes in velocity might be significant. Additionally, three derived metrics were utilized. The first is the Distance (D) in degrees between the current latitude and longitude and the coordinates of the destination (the value could be converted to meters, but since the data will ultimately be normalized this conversion would make little difference and require more computation). The second is the Readings Since Improvement (RSI), which is the number of readings that have passed since the Distance has decreased, which provides some indication of whether or not the dog is making progress toward the known destination. Finally, the Deviation from Desired Heading (DEV) in degrees shows the difference between the current heading and the ideal heading which would lead the dog to the goal (calculated using the current coordinates and the destination coordinates). A summary of all 13 metrics is provided in Table 2, where the Δ symbol indicates the attribute is the absolute value of the difference between the attribute in the current raw data instance and the attribute in the previous raw data instance.

Name	Units
Δax, Δay, Δaz	g
Δgx, Δgy, Δgz	(deg/s)
Δvelocity	(m/s)
Δheading	(deg)
Δlatitude, Δlongitude	(deg)
D	(deg)
RSI	unitless
DEV	(deg)

Table 2. Processed Metrics

All of the metrics were normalized (using the highest known values of a given attribute as the maximum) for input into the GRNN.

6. Experiments

In order to evaluate the classifier system, the data from the 7 trials was divided into two groups: a training set and a larger testing set. The training set consisted of the results of a single trial (340 processed instances, which included some anomalous behavior) and the testing set consisted of the remaining 6 trials (roughly 250 processed instances each, some trials did not contain anomalies). Two different experiment types were run: one using trial-specific parameters and the other using general parameters optimized over all the trials.

6.1. Trial-Specific Parameters

In these experiments, the EHC was run to optimize the attributes and parameters for specific trials. In other words, parameters were discovered for each of the 6 trials in order to minimize classification error (as described in Section 4.2). Optimizing over a specific path has the benefit of providing a nice improvement in accuracy, as would be expected from tailoring the parameters. However, it has the disadvantage of losing generality. In other words, using path-specific parameters on a different path will generally perform badly. In many applications, however, path-specific parameters would be the ideal choice. For example, if a canine were to be used to routinely check a path around an airport for drugs, then path-specific parameters would be preferable since generally the goal would be to minimize error even at the cost of generality.

In order to optimize the GRNN parameters, the EHC was given 500 cycles. In Table 3, the results of the

Trial	σ	attributes	total	correct	SR	FN	FP	fit
2	0.1488	Δax,Δgy,Δgz,Δlat.,Δlong.,ED,RSI	250	228	0.912	1	21	204
3	0.0379	Δvelocity, Δlat.,Δlong.,D	215	183	0.851	13	19	125
4*	0.0262	Δheading, Δlong.,D,RSI,DEV	204	204	1.0	0	0	204
5	0.1091	Δlong.,D,RSI,DEV	284	246	0.866	17	21	174
6*	1.0	Δlong.,D,RSI,DEV	322	322	1.0	0	0	322
7	0.0970	Δax, Δaz, Δgy,Δgz,Δvelocity,Δlong.,RSI,DEV	230	210	0.913	9	11	172

Table 3. Path-Specific Parameter Results

optimization, alongside the results of running the optimized GRNNs are shown. The "Trial" column indicates which path was used as the test set (there is no other significance to the trial number). An asterisk (*) indicates that the given trial did not contain anomalies (hence, none should be detected). The σ column indicates what value of σ passed to the GRNN yielded the results shown. The "attributes" column shows which attributes were included in the GRNN calculations. Columns "total" and "correct" indicate the total number of instances and the raw number that were classified correctly. The value of "Success Rate," "FN," and "FP" indicate the percentage of correct classifications, the number of false negatives, and the number of false positives, respectively. The value of "fit" is given by (3).

In general, the method gives fairly good results given the limited amount of training data available. Even in the worst cases (Trials 3 and 5), the accuracy was still over 0.85 with relatively few false positives and false negatives - in both those trials, enough anomalous instances were identified such that a control system could recognize it as something other than an outlier. In Trials 2 and 7 (both with anomalies), the accuracy improved even more, exceeding 0.90, and there were very few false negatives. An example test trial (Trial 7) is shown in Figure 4, where the thick line indicates the path that the dog actually traveled along, the small crosses indicate the points where the human trainer ended the tone (indicating the dog needed to correct itself), and the small x's indicate where the GRNN indicated that the tone should be dropped. The GRNN makes only a few relatively isolated errors near the end of the dog's path, which could likely be resolved with increased training data or by only changing the tone given a certain number of anomaly's detected. In the remaining trials (which contained only normal behavior), no false positives were detected at all.

6.2. General Parameters

In these experiments, the EHC was run to optimize one set of attributes and parameters for all of the (non-training) trials together. Parameters were discovered only once to give the best average results over all trials. The benefit of choosing general parameters is that they generate better anomaly detection over a wider range of paths. However, the best general parameters will often perform worse on any individual path. General parameters would be preferable in a situation where either the canine must deal with a wide variety of paths, the path is dynamic in some way (such as an area with vehicles), or simply the path itself is unseen, other than the coordinates of the destination.

In Table 4, the results are broken down by trial for comparison purposes. The column headings have the same meaning, although it should be noted that the σ and attributes are the same for all trials. While performance is still fairly good, there is a roughly 0.05 drop in SR across nearly all of the trials. There is also a moderate increase in both false positives and false negatives. It is worth noting that the best performing settings used less than half of all of the available attributes, indicating that the classification is largely dependent on the information given by a few derived metrics.

7. Discussion

These results provide indication that the concept of an "anomaly" (as defined by the tones given by a human trainer) can be detected with a relatively high degree of accuracy. The authors believe that as more trials are gathered, the training data can be refined and expanded, allowing for even greater accuracy for a larger variety of paths. Further, it should be noted that in practice a single instance being classified as "anomaly" should not necessarily result in an immediate change of stimulus to the canine. Rather, several "anomaly" classifications in a short period of time should require a change in control signals. This would ameliorate the impact of scattered false positives and false negatives in practice. Another important observation is that even a human trainer is not completely accurate in identifying that the canine is not behaving correctly, so some error is inherent in the system.

With respect to performance, a single classification

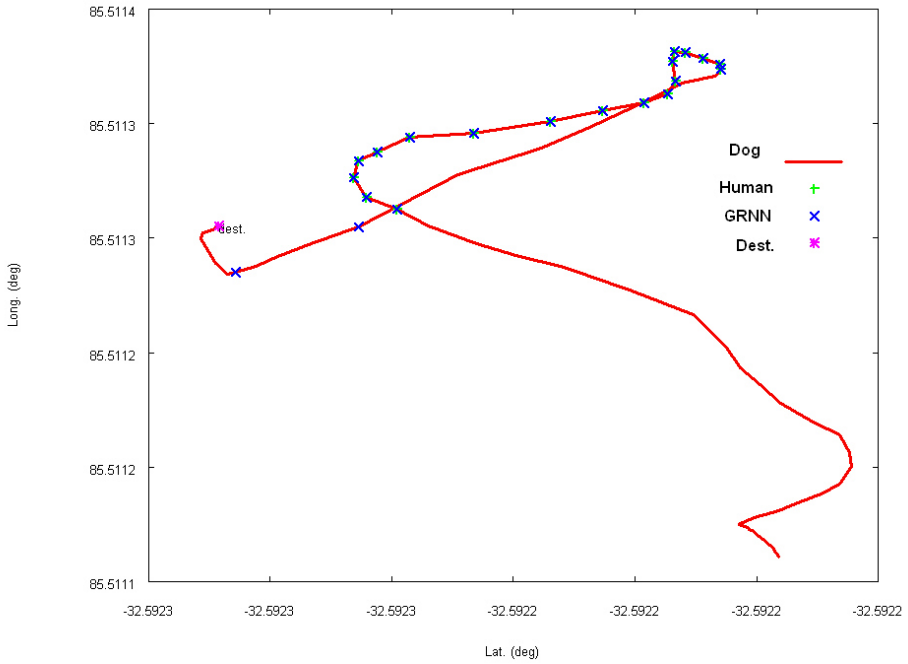

Figure 4. Sample Test Trial with both human trainer and GRNN Output

Trial	σ	attributes	total	correct	SR	FN	FP	fit
2	0.1234	Δaz,Δlong.,D,RSI,DEV	250	215	0.860	4	31	172
3	0.1234	Δaz,Δlong.,D,RSI,DEV	215	178	0.818	23	16	91
4*	0.1234	Δaz,Δlong.,D,RSI,DEV	204	197	0.966	0	7	190
5	0.1234	Δaz,Δlong.,D,RSI,DEV	284	230	0.810	27	27	122
6*	0.1234	Δaz,Δlong.,D,RSI,DEV	322	322	1.0	0	0	322
7	0.1234	Δaz,Δlong.,D,RSI,DEV	230	205	0.891	16	9	148

Table 4. General Path Parameter Results

operates in less than a second, even when file I/O is required. In an actual implementation, many classifications could be performed in a second. The optimization routine takes considerably longer (on the order of minutes), but should be performed in an off-line fashion to discover parameters once and then reuse them.

8. CONCLUSIONS AND FUTURE WORKS

8.1. Conclusions

In this research, a method for classifying canine behavior as either normal or anomalous was presented. A General Regression Neural Network was used on a set of training data taken using a GPS/INS unit on-board a dog being directed by a human trainer and then used to predict the canine behavior in future trials. An optimization method, the Evolutionary Hill-Climber, was discussed as a means to determine which attributes to include in the training and to determine good system parameters for the GRNN. Both path-specific and general settings were presented and their results on several test sets were shown. In general, path-specific results were (unsurprisingly) better, but in many applications general results might be more useful. Overall, the results prove to be quite promising and in the continuing work the approach seems likely to yield higher accuracy in emulating a human trainer.

8.2. Future Work

Currently, the project is still in relatively early stages, but even preliminary results have been promising. Our continuing work includes:

- Conducting considerable additional trials to give

increased training data which should improve the accuracy of the network.

- Including paths with a series of waypoints in order to model more sophisticated paths
- Optimizing which training instances to use to remove unnecessary training data and improve efficiency
- Exploring alternate classification algorithms (Radial Basis Function Networks or Support Vector Machines) to reduce overhead and improve accuracy
- Predicting continuous variables such as heading and velocity based on the canines behavior and signal changes

9. ACKNOWLEDGMENTS

This work was supported by an ONR YIP award N00014-06-1-0518.

References

[1] D. Specht, A General Regression Neural Network, IEEE Transactions on Neural Networks 2 (1991) 568–576.

[2] X. Xu, H.-G. He, Neural-Network-Based Learning Control for the High-Speed Path Tracking of Unmanned Ground Vehicles, in: Proceedings of the 2002 International Conference on Machine Learning and Cybernetics, Vol. 3, 2002, pp. 1652–1656.

[3] D. Helweg, H. Roitblat, P. Nachtigall, Using a Biomimetric Neural Net to Model Dolphin Echolocation, in: Proceedings of the First New Zealand International Conference on Two-Stream Artificial Neural Networks and Expert Systems, 1993, pp. 247–251.

[4] D. Calvert, E. Bajar, D. Stacey, J. Thomason, Analysis of Equine Gait Through Strain Measurement, in: Proceedings of the 25th International Conference of Engineering in Medicine and Biology Society, Vol. 3, 2003, pp. 2370–2373.

[5] S. Haykin, Neural Networks: A Comprehensive Foundation, 2nd Edition, Prentice Hall, Upper Saddle River, New Jersey, 1999.

[6] D. Golberg, Genetic Algorithms in Search, Optimization, and Machine Learning, Addison-Wesley Publishing Company, Inc., Reading, Massachusetts, 1989.

[7] S. Russell, P. Norvig, Artifical Intelligence: A Modern Approach, 2nd Edition, Prentice Hall, 2003.

[8] S. Forrest, M. Mitchell, Relative building-block fitness and the building-block hypothesis, in: L. D. Whitley (Ed.), Foundations of Genetic Algorithms 2, Morgan Kaufmann, San Mateo, CA, 1993, pp. 109–126.