Control of Four-wheeled Vehicle on Ice Surface Using Attention-gated Reinforcement Learning (AGREL)

Marek Lapko, Rudolf Jakša

Center for Intelligent Technologies, Department of Cybernetics and Artificial Intelligence, FEI, TU Košice, marek.lapko@tuke.sk, jaksa@neuron.tuke.sk

Abstract: This paper describes the control of the mobile robot (four-wheeled vehicle) on a slippery surface. The aim of this control is to follow the given path causing minimal errors. We compared two approaches of the control in this paper: inverse control (supervised offline learning), and control based on reinforcement learning (online learning, AGREL). The main focus of this work is on a recently introduced type of reinforcement learning algorithm AGREL [1] designed for classification. We tried to adapt it to for control tasks.

Keywords: AGREL, reinforcement learning, four-wheeled vehicle, control of mobile robot

1 Introduction

An advantage of approaches based on artificial neural networks, especially on reinforcement learning, is their ability to adapt to new and unexpected situations. Perhaps, one of many reasons why to study biologically plausible approaches of control is their adaptability as it is exemplified in [3]: *A gazelle calf struggles to its feet minutes after being born. Half an hour later it is running at 20 miles per hour.*

This work is focused on such a problem. The aim of it is to implement (and compare it with direct inverse control) a new type of reinforcement learning (AGREL) in a control task and to examine its ability to control four-wheeled vehicle in unexpected situations such as driving on a slippery surface. This approach is not meant to be universal solution for difficult control problems, but the one of variety of tools that is required [4].

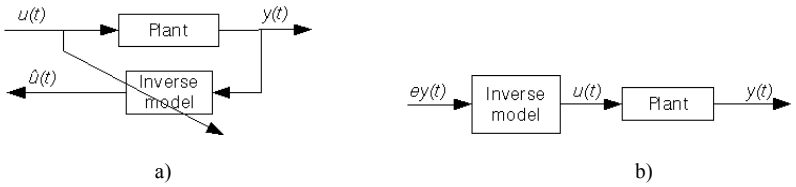


Figure 1

a)Inverse model identification of the plant
b) Control of the plant by inverse controller

An intelligent control was proposed by Fu in 1971 as an alternative to classical control applying artificial intelligence [5]. The exact definition of the intelligent control has not been given yet [6].

Intelligent control is a branch combining features of four other branches: artificial intelligence, control theory, computer science and operations research [7]. The main contribution to the intelligent control is from artificial intelligence and control theory. If we consider main features of these approaches we will get an important feature of intelligent control as mentioned in [8]: the ability to improve its performance in the future, based on experiential information it has gained in the past, through closed-loop interactions with the plant and its environment.

One of the possible implementation of intelligent control is the use of artificial neural network (ANN). Such a control is also called neural control. According to Katic and Vukobratovic [9] neural control of mobile robots is divided into three approaches also known from learning ANN:

- supervised learning,
- unsupervised learning,
- reinforcement learning.

1.1 Direct Inverse Control

The main idea of the direct inverse control is to identify the inverse model of the controlled plant (Figure 1a) in the way that the input of the model is the output of the plant and the output of the model is the input of the plant. Then we can use the inverse model (as is depicted in Figure 1b) in the way that the desired output is the input of the inverse model and its output is the input of the plant [2].

To identify (and control) plant we can use ANN. Training data, needed to train ANN, contains description of current state and a random action taken in this state. When training ANN, input is a state at the time t and desired state at the time $t+1$. Output is the action taken at the time t. Then, if the inverse model of the plant is a function, we can train ANN to take an action that leads to the desired state.

1.2 Attention-gated Reinforcement Learning (AGREL)

In reinforcement learning, we do not have a desired output, as in supervised learning, we do not know what to do. We just have information, whether the taken action leads to the desired state or not (reward or penalty). Definition of reinforcement learning proposed by Barto and Sutton [3] claim that reinforcement learning is learning what to do, how to map situations to actions, so as to maximize a numerical reward signal, then the learner is not told which actions to take, but instead must discover which actions yield the most reward by trying them and finally any action may affect not only the immediate reward but also the next situation and, through that, all subsequent rewards.

AGREL as a reinforcement learning algorithm was proposed by Roelfsema and Ooyen in 2005 in [1]. One of the AGREL's main feature is that, it is more biologically plausible than others, especially than supervised learning methods. Biological plausibility of AGREL is given by its reinforcement learning character, global error signal δ (similar to signal computed by dopamine neurons of midbrain) and an attentional feedback signal [1].

Aim of this kind of ANN (AGREL) is to speed up as well as simplify reinforcement learning. AGREL is a type of associative reinforcement learning algorithm, where we consider just spatial credit assignment, so it is learning how to map states to actions [3]. It is also possible to extend AGREL to the sequence learning (actor - critic), considering both spatial and temporal credit assignment.

Signal from pattern in the input layer is propagated through network in the same way as in BP. As activation function is used logistic function (1)

$$y_i = f(in_i) = \frac{1}{1 + e^{-\alpha in_i}}, \tag{1}$$

where in_i is input to i-neuron and:

$$in_i = \sum_{j=0}^{N} w_{ij} x_j \ . \tag{2}$$

Weight w_{i0} is a bias of *i*-neuron (x_0 is equal to *1*). In the output layer, there is used stochastic softmax rule (3) to determine the probability. According to this probability, by *Winner Takes All* (WTA) strategy, neurons have activation equal to 0 except for winning neuron with activation equal to 1.

$$\Pr(y_k = 1) = \frac{e^{in_k}}{\sum_{k'=1}^{M} e^{in_{k'}}} \tag{3}$$

In other words, input is classified into the one of the *M* classes, where *M* is the number of neurons in the output layer.

Weights are adapted by Hebbian rule. For adaptation of weights between output layer and hidden layer is used equation (4). We made one change in adaptations rule of AGREL as well as topology of ANN. In adaptation rules, we do not consider feedback connections weights as in [1]. There was used just one set of weights for forward signal propagation as well as feedback signal propagation (learning).

$$\Delta w_{ij} = \beta y_i y_j f(\delta) \tag{4}$$

Weights between second hidden layer and first hidden layer are adapted according to equation (5)

$$\Delta w_{ij} = \beta y_i y_j f(\delta)\left[w_{js}(1-y_j)\right] \tag{5}$$

where *s* is the winning neuron in output layer. Weights between input layer and hidden layer are adapted according to equation (6).

$$\Delta w_{ij} = \beta y_i y_j f(\delta)\left[fb_{y_j}(1-y_j)\right], \tag{6}$$

with

$$fb_{y_j} = \sum_{k=1}^{L} y_k(1-y_k)w_{ks}w_{jk} \,. \tag{7}$$

On rewarded trials, to count *delta* we use

$$\delta = r - E(r)\,, \tag{8}$$

where E(r) equals $\Pr(y_k = 1)$. When the trial is punished, then δ is set to -1. To stress unexpected trials, which are very valuable in learning, there is used f(δ) in adaptation equations.

$$f(\delta) = \begin{cases} \frac{\delta}{1-\delta} & \delta \geq 0 \\ \delta & \delta = -1 \end{cases} \tag{9}$$

2 Design and Implementation

All experiments were conducted in simulated mode using Open Dynamics Engine libraries. Task complexity was scaled by changing friction coefficient as well as gravity acceleration with values several times lower than the real ice friction or, on the other hand, gravity of Earth. For the sake of increasing complexity, there was also simulated fault (blocked rear wheel). The path was defined by ten points in all

conducted experiments. Length of the path was 1700 m. Average speed was around $1ms^{-1}$. Weight of the vehicle was set to 1500 kg, which is the same as the weight of a real car.

ODE parameters of vehicle and environment were set as following:

LENGTH	5.5
WIDTH	2.5
HEIGHT	1.0
RADIUS	0.5
CMASS	1500
WMASS	10
FMAX	50
GRAVITY	1.8
FRICTION ICE	0.005
FRICTION DRY	20.0
ERP	0.8
CFM	0.0001

2.1 Direct Inverse Control of Vehicle by ANN

As mentioned above, we need to describe states and actions by measurable characteristic. There, due to increasing accuracy of control, we can use more than just desired state in inverse control. Especially in this proposal, there were used current and one or more desired states. Current state of the vehicle was represented by:

- information whether the wheel is slipping or not,
- velocity,
- change of the angle of velocity,
- the angle of rotation (yaw).

Desired state was represented by *change of the angle of velocity* in time $t+1$ up to $t+n$.

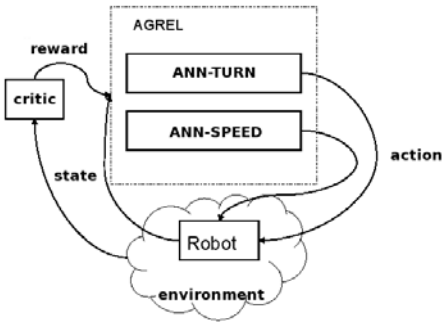


Figure 2
Architecture of control with AGREL

Action was represented by:

- turn of front wheels,
- velocity,
- brake.

After getting inverse model of plant, it was used to control such a plant in the way described above.

As a structure to represent inverse model was used feedforward neural network with two hidden layers and was adapted by backpropagation learning algorithm (BP). Input into ANN was current state (7 neurons) and as the desired state was used just the change of the angle of velocity in next n steps. There were x neurons in first hidden layer, y neurons in second hidden layer and tree neurons in output layer (action).

2.2 Reinforcement Learning (AGREL) Control of Vehicle

We designed two approaches to control vehicle. Control of the vehicle by one ANN. In this case we control just turn of the front wheels. Or we can control vehicle by two ANN when we control turn of the front wheels as well as desired speed of vehicle (Figure 3).

Input (state) was represented by:

- information whether the wheel is slipping or not,
- the angle between vector position of the vehicle - destination and the velocity vector,
- change of the angle between vector position of the vehicle - destination and the velocity vector,
- the angle of rotation,

- change of the angle of rotation,
- the angle between the vector of rotation vehicle and the velocity vector,
- change of the angle between the vector of rotation of the vehicle and the velocity vector,
- the angle between the vector position of the vehicle - destination and the vector of rotation of the vehicle,
- velocity,
- change of velocity,
- action taken in *t-1*.

In both cases was state described with the same parameters. On the other hand, when we control vehicle by one ANN, output was represented by *x* neurons so there were *x* possible position of steering wheel and speed was set to constant value. In the case of control by two ANN were used *x* neurons for control turn as well as speed.

ANN for turn control was rewarded every action after turning the velocity vector to the target in all following experiments. Analogously, punished in other case. ANN for speed control was rewarded in the case the ANN for turn control was rewarded and speed increased or was punished in turn control and speed decreased. Punished in other case.

2.3 Integration of Inverse Control and AGREL

This approach includes advantages of both described approaches. In this case, control based on AGREL is used to accurate inverse control, as well as this approach is able to adapt to unexpected situations.

There was included design of both approaches into one. AGREL was used to correct action taken by inverse control, so the output of AGREL was presenting just the correction of the action (Δx) instead of the real action (x).

3 Experiments

3.1 Direct Inverse Control of Vehicle

There were conducted some experiments with different topology of ANN (parameters x, y and n). Experiments were compared according to the measured values such as the average speed, the length of trajectory and the average distance from given trajectory as well as subjective value.

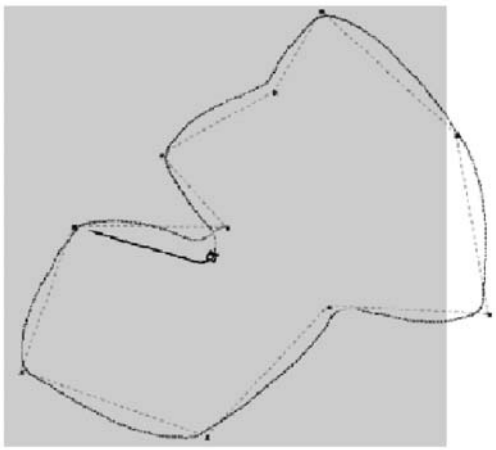


Figure 3

Inverse control of vehicle. Gray field stands for ice.
Dashed line is prescribed trajectory while solid lines is actual path of vehicle.

There seemed to be two different cases in conducted experiments. On the one hand vehicle was driving with a big distance from given trajectory where direction was changed (*n=1 or x=15, y=10* and *n=5*0), on the other hand, it took longer and there was a bigger average distance (*x=45, y=30 and n=50*).

The best solution according to the time was the experiment with x=15, y=10 and n=50, according to the average distance experiment with x=45, y=30 and n=1 and according to the length of trajectory experiment with x=45, y=30 and n=50 (Figure 4). One of the interesting results was attempt of ANN to choose action after which vehicle was not slipping.

3.2 Reinforcement Learning (AGREL)

3.2.1 XOR Problem

First of all, we tested AGREL on benchmark XOR problem. We made some modifications in AGREL algorithm before all experiments were conducted. We replaced penalty value -1 with value varying from -0.1 up to -0.5. ANN was able to classify all patterns for XOR problem after 80 iteration.

3.2.2 Control of the Vehicle by One ANN

We used ANN with topology 16-33-13-x and value of punishment set to -0.35. Speed was equal to 1.0 or 1.4 and learning rate was equal to 0.2, 0.5 and 1.0 depending on experiments. According to conducted experiments better solutions were that with more (5) neurons in the output layer. It is also better to use more

neurons due to better stability of ANN. We got better results with learning rate equal from 0.5 to 1.0. It was important to set the higher learning rate in order to decrease adaptation time. Another conclusion was that the controller (ANN - AGREL) was not able to control vehicle on slippery surface with higher velocity acceptably.

3.2.3 Control of the Vehicle by Two ANN

In this case, there were controlled both the turn and the speed. Punishment was set to -0.65 or -0.45, learning rate was equal to 0.2, 0.5 or 1.0 and rate of weight change when rewarded was equal to 0.05, 0.15 or 0.45 depending on experiments. We got the best results with lower learning rate. It is also better to set lower learning rate due to better stability of ANN. In experiments with higher learning rate, there was lower error till weight reinitialization.

3.2.4 Integration of Inverse Control and AGREL

In these experiments, we used the same ANN for inverse control like in previous experiments and also with the same settings for AGREL, but with output suitable for correction of action taken by inverse control. Punishments were equal to -0.64 or -0.45 and learning rate was equal to 0.2 or 0.4 depending on experiments.

The best results were achieved with learning rate equal to 0.2. Generally, there were achieved better results then in individual approaches (compare Inv. and I+A in Table 1).

Table 1
Measured values of the best experiment of each approach of control focused on higher speed. I+A stands for integration of inverse control and control based on learning AGREL.

Type	Speed[ms^{-1}]	Length[m]	Av. dist. E.[m]
Inv.	1.43	1876	15.7
AGREL 1NS	1.35	1931	15.3
AGREL 2NS	1.26	1876	13.4
I + A	1.45	1866	14
I + A	0.79	1936	13.5

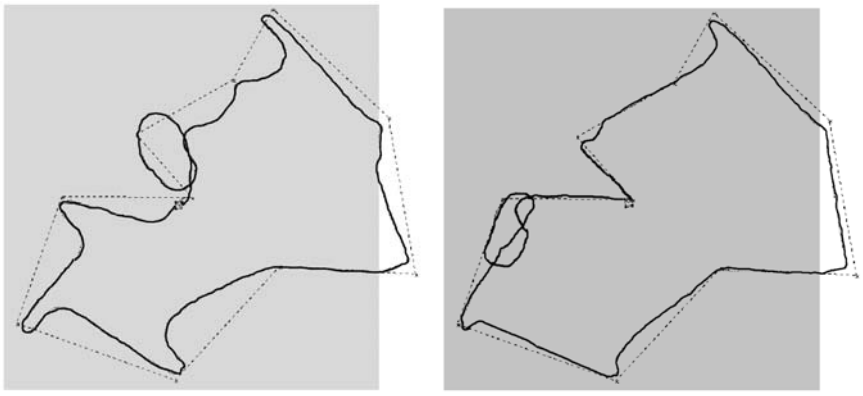
3.2.5 Control of the Vehicle with a Fault

These experiments were the most interesting as the problem, that was solved, was the most complex and maybe more interesting for practice than other. There was blocked one rear wheel as a simulation of possible fault. All parameters of control were set to the same values as in the best previews experiments.

None of both individual approaches achieved satisfactory results with a fault. Control with AGREL was not able to deal with the fault in appropriate time and therefore we do not provide any results with this type of control in the Table 2.

Table 2
Measured values of the experiments (vehicle with fault). I+A stands for integration of inverse control and control based on learning AGREL. E.n is a number of experiment, Length is a length of the trajectory made by vehicle in one round, Av. dist. E. is an average distance error from given trajectory and Av.speed. is an average speed.

E.n.	Type	Length[m]	Av. dist. E.[m]	Av. speed[ms^{-1}]
1	Inv.	-	-	-
	I+A	2987	31.3	1.06
2	Inv.	2434	20.4	0.8
	I+A	2000	19.6	0.81
	I+A after 5 r.	2890	12.0	0.8
3	Inv.	-	-	-
	I+A	3459	46	1.1
4	Inv.	2040	24	0.82
	I+A	1936	13.5	0.79



a) b)

Figure 4

a)Inverse control of the vehicle with fault
b)Integration of inverse control and AGREL control of the vehicle with fault. Gray field stands for ice. Dashed line is prescribed trajectory while solid lines is actual path of vehicle.

In the case of inverse control, ANN with proper settings was able to control vehicle even with fault, but only in these cases, when it drove with lower speed and with lower average distance error in control without fault (E.n. 2 and E.n. 4 in Table 2) (Figure 5a). The only one approach that satisfied all requirements was integration of inverse control and AGREL. As is depicted in Figure 5b, we can see that except the one part of trajectory at the beginning, control was very accurate.

Conclusions

In conducted experiments we compared two types of control approaches: inverse control (supervised learning) and control based on reinforcement learning (AGREL). Inverse control achieved satisfactory results and also we can say that inverse control was better than control based only on AGREL. But on the other hand, results achieved by control based on AGREL were similar to these with inverse control. The biggest problem of AGREL is the quick rise of weights. The other problem is to set learning rate. If the learning rate is too small, adaptation is also slow, but we can store knowledge for longer time, on the other hand with bigger learning rate, adaptation is faster, ANN is able to store knowledge only for a short time and there is high probability that weights will rise. As the best control approach we can consider integration of both approaches, AGREL and inverse control. The biggest difference between integrated approach and the other was achieved in control of vehicle with fault, when the individual approaches did not reach final destination in satisfactory time and the integrated approach achieved results comparable with control in normal conditions.

References

[1] Roelfsema, P. R., Ooyen, A.: Attention-gated Reinforcement Learning of Internal Representations for Classification. In Neural Computation (2005), Vol. 17, MIT, pp. 2176-2214

[2] Jakša, R.: Neural Networks in Intelligent Control (in Slovak). PhD thesis, Technical University of Košice, 1999

[3] Sutton, R., Barto, A.: Reinforcement Learning: An Introduction. Cambridge, MA, MIT Press, 1998

[4] Miller III W. T., Sutton R. S., Werbos P. J.: Neural Networks for Control. Cambridge, MA, MIT Press, 1995

[5] Harris C. J.: Advances in Intelligent Control. Taylor Francis, 1994

[6] Zi-Xing C.: Intelligent Control: Principles, Techniques and Applications. Word Scientific Publishing, 1998

[7] Madarsz L.: Inteligentn Technológie a ich aplikácie v zložitých systémoch. University Press Elfa, 2005

[8] White D. A. Sofge D. A.: Handbook of Intelligent Control. New York: Van Nostrand Reinhold, 1991

[9] Vukobratovic M, Katic D.: Intelligent Control of Robotic Systems. Springer, 2003