Postprint of: Li G., Xu Y., Zhang H., Szczerbicki E., Towards Knowledge Sharing Oriented Adaptive Control, CYBERNETICS AND SYSTEMS Vol. 53, iss. 5 (2022), pp. 520-528, DOI: 10.1080/01969722.2021.2018550

Towards Knowledge Sharing Oriented Adaptive Control

Guixian Li [a], Yufeng Xu [a], Haoxi Zhang [a b], Edward Szczerbicki [c]

[a] School of Cybersecurity, Chengdu University of Information Technology, Chengdu, China; [b] Advanced Cryptography and System Security Key Laboratory of Sichuan Province, Chengdu, China; [c] Faculty of Management and Economics, Gdansk University of Technology, Gdansk, Poland

Abstract. In this paper, we propose a knowledge sharing oriented approach to enable a robot to reuse other robots' knowledge by adapting itself to the inverse dynamics model of the knowledge-sharing robot. The purpose of this work is to remove the heavy fine-tuning procedure required before using a new robot for a task via reusing other robots' knowledge. We use the Neural Knowledge DNA (NK-DNA) to help robots gain empirical knowledge and introduce a Knowledge Adaption Module (KAM) utilizing the deep neural networks (DNN) for knowledge reuse. The initial experiment shows that the target robot can adapt to the inverse dynamic model of the source robot via our KAM and reuse the knowledge shared by the source robot.

Address correspondence to Haoxi Zhang, School of Cybersecurity, Chengdu University of Information Technology, No. 24 Block 1, Xuefu Road, Chengdu, China, 610225. E-mail: haoxi@cuit.edu.cn

Keywords: knowledge sharing, Neural Knowledge DNA, Deep Neural Networks, inverse dynamic model, control system.

INTRODUCTION

Nowadays, machine learning is becoming increasingly popular for solving robot control problems (Zhou et al., 2019). It helps robots perform various tasks well in uncertain and complex conditions (Bokeno et al., 2018; Liang, 2019; Sajedi & Liang, 2019). However, training a robot for particular tasks are still time-consuming and expensive (Ding et al., 2016; Liang et al., 2018).

In order to reduce the costs of the learning process, extensive efforts have been made (Dai et al., 2008; Taylor & Stone, 2009). Approaches such as learning invariant features (Gupta et al., 2017) and manifold alignment (Ammar et al., 2015; Daftry et al., 2016) are introduced. These methods aim to improve the efficiency of robot learning by sharing knowledge among robots. Additionally, transferred knowledge can accelerate the training of target robots and help improve the target robot's performance in untrained tasks (Taylor & Stone, 2009). Knowledge sharing or migration learning problems have been studied in various fields (Bocsi et al., 2013). Knowledge sharing in robotics is divided into two directions: (i) transfers across robots and (ii) transfer across tasks. The former is about transferring the collected knowledge to another robot (Devin et al., 2017; Gupta et al., 2017; Pereida et al., 2018). In contrast, the latter focuses on

transferring knowledge learned from an old task to a new task on the same robot (Cavallo et al., 2014; Wang et al., 2008). Nevertheless, existing approaches almost always encounter the problem that the target robot does not adapt well to the source robot's knowledge in practice (Adlakha & Zheng, 2020; Kim et al., 2020; Taylor & Stone, 2009). Therefore, it remains challenging to enable the target robot to adapt to other robots' inverse dynamics. This paper introduces a novel approach to address the adaption problem in sharing knowledge between robots with different dynamics.

The rest of the paper is organized as follows: In Section 2, we introduce the idea of the NK-DNA. Section 3 presents the proposed Knowledge Adaptation Model (KAM), including its architecture and methodology. The experiment and results are explained and discussed in Section 4. Finally, Section 5 concludes the paper.

THE NEURAL KNOWLEDGE DNA

The Neural Knowledge DNA (NK-DNA) is proposed to store and represent knowledge captured in intelligent systems that use artificial neural networks as the central power of its intelligence (Zhang et al., 2017). It utilizes the ideas underlying the success of deep learning (LeCun et al., 2015) to the scope of knowledge representation.

The NK-DNA is constructed similarly to DNA formed (Sinden 1994): built up by four essential elements. As the DNA produces phenotypes, the Neural Knowledge DNA carries information and knowledge via its four fundamental elements: States, Actions, Experiences, and Networks (see Figure 1).

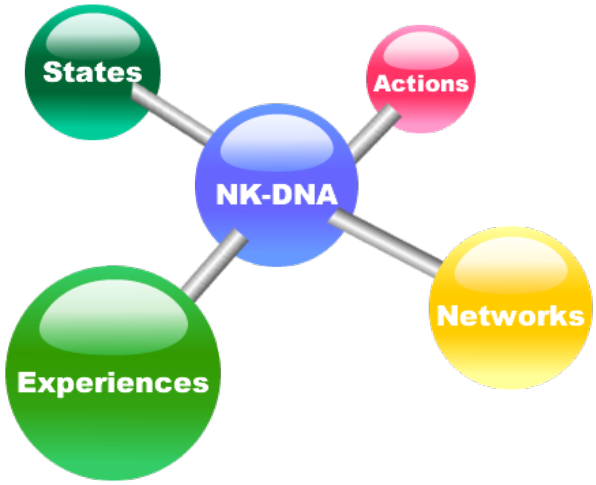

Fig. 1. The conceptual structure of the NK-DNA.

The NK-DNA's four-element combination is designed to carry detailed information of decisions: States are situations in which a decision or a motion can be made or performed. Actions are used to store the decisions or activities the domain can select. While Experiences are the domain's historical operation segments with feedbacks from outcomes, And Networks hold the description of neural networks for training and using such knowledge, such as the network structure, weights, bias, and deep learning framework used.

Generally, knowledge is acquired as models after training in deep learning systems. The model usually stores information about weights and biases of the connections between neurons of the neural network and the hierarchy of the neural network in detail. Once the neural network has been trained, it will give results straightforward through the computation of its network layers after feeding it with inputs—similarly, the NK-DNA stores knowledge using the same idea. Figure 2 shows the concept of knowledge carried by the NK-DNA.

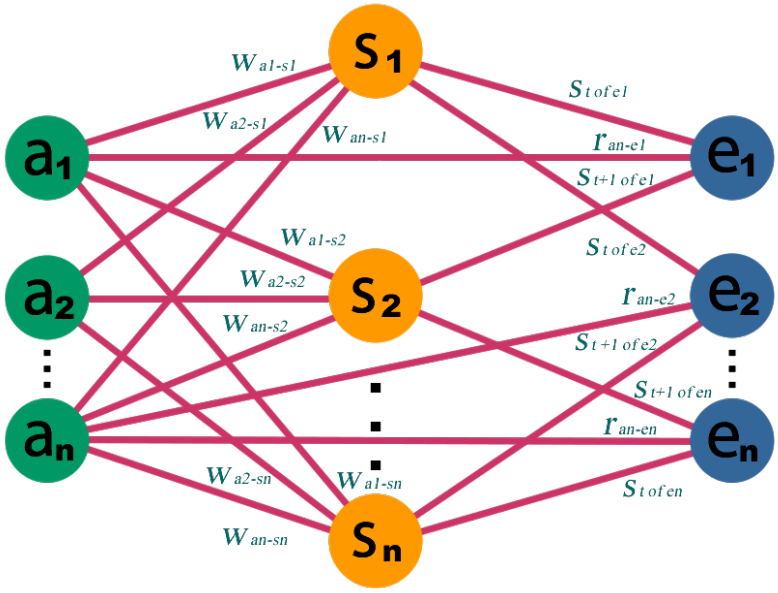

Fig. 2. Concept of the NK-DNA-carried knowledge.

In the NK-DNA, a neural network is used to carry the relation between actions and states: as we can see in Figure 2, each state (represented as S_1, S_2 … S_n) can have connections with a set of actions (defined as a_1, a_2, … a_n). If an action is connected with a state, the connected action is available in that state; in other words, the agent can choose the action to perform if it is in that state. The trained neural network provides the knowledge of which action is the best choice for a specific state. The states here are the inputs, which can be the raw sensory data or data describing the agent's current situation.

Another essential feature of this approach is that the NK-DNA uses previous decisional experience as the primary source to collect and expand intelligence for future decision making. Experience in the NK-DNA is stored as the Set of Experience Knowledge Structure (SOEKS) (Sanin & Szczerbicki, 2006). Usually, the agent transitions from one state to another during its operation. It makes decisions (picks

actions) in each state and receives feedback from its operation; these states, actions, feedbacks, and transitions make up the so-called 'experience'.

THE KNOWLEDGE ADAPTION MODULE

A. The System Overview

Our research concentrates on the problem of transfer across robots, which is to share one robot's knowledge about one specific task with another robot. In order to ensure the target robot's performance in reusing the source robot's knowledge, we design an add-on module, which we call the knowledge adaption module (KAM), to the target robot's control system to enhance the adaptability of the NK-DNA-based knowledge reusing. Together with the NK-DNA module and the target robot's controller, these three parts comprise the knowledge sharing oriented system (see Figure 3).

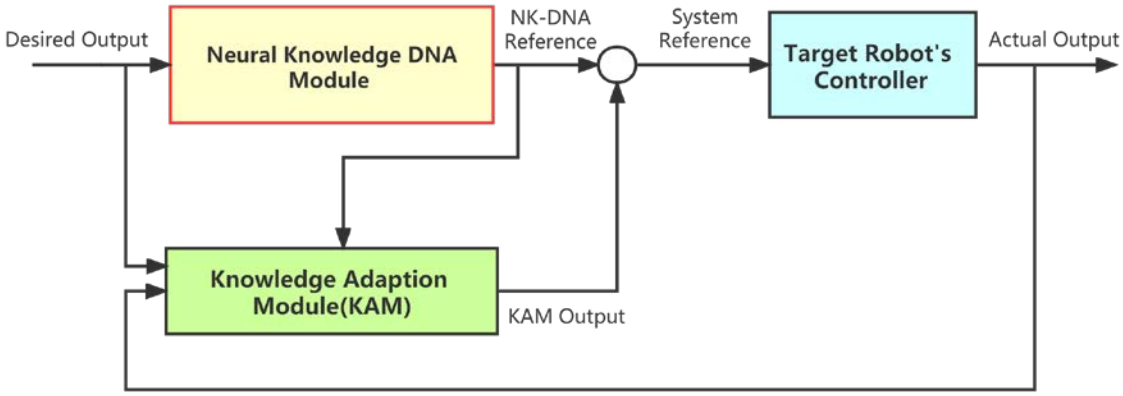

Fig. 3. The architecture of the knowledge sharing oriented system.

The NK-DNA module is used to store and infer the knowledge received from the source robot. It also enables the target robot to learn new knowledge through its new

experience (Zhang et al., 2017). When a desired output is received, the system sends it to the NK-DNA module and the KAM. The NK-DNA module reuses the source robot's knowledge to give a control reference. Then KAM adjusts this reference according to the desired output and the robot's current state.

B) Knowledge Adaption

We considered the knowledge adaption problem as the problem of trajectory tracking. Precisely, the target robot must track the trajectory given as the control reference of the NK-DNA Module in the system. We consider the dynamics of the source robot and the target robot can be represented by

$$u(t+1) = y\big(u(t)\big) + g\big(u(t)\big)\,x(t) \tag{1}$$

$$f(t) = p\big(u(t)\big) \tag{2}$$

$$f(t+k) = M\big(u(t)\big) + H\big(u(t)\big)\,x(t) \tag{3}$$

where the state of the system is $u \in \mathbb{R}^n$, the system's input (i.e. the control reference) and output are $x \in \mathbb{R}$ and $f \in \mathbb{R}$ respectively. The discrete-time index is $t \in \mathbb{R} \geq 0$, and *y(·)*, *g(·)*, and *p(·)* are smooth functions, $M\big(u(t)\big) = p \cdot y\big(u(t)\big)$ and $H\big(u(t)\big) = \frac{\partial}{\partial u}\, p \cdot y^{k-1}\big(u(t+1)\big)$. According to Equation 3, the ideal reference to track for the target robot controller is

$$x_{ideal}(t) = \big(f_{desired}(t+k) - M\big(u(t)\big)\big)/H\big(u(t)\big) \tag{4}$$

where $f_{desired}(t+k)$ represents the desired output the system received.

Based on SOEKS knowledge representation, the source robot's inverse dynamics model is learned through NK-DNA. Specifically, the robot's state changes after the

controller execute a control reference. The state-control relationship can be discovered using a deep neural network inside the NK-DNA. In the NK-DNA, the association of states and controls is linked through the *Experience* and represented as SOEKS. In order to have a sufficient amount of experience for learning and to improve the inverse dynamics modelling, we generate random control references and send them to the robot for experience collection. Every time the robot is given a control reference, we collect a specific experience of applying the reference and store them as $[x_t, u_t, f_t]$ where x_t is the control reference executed at that time and u_t is the original state at that time, and the f_t is the source robot's output. We use the experience to train an inverse dynamics model for the source robot.

Similarly, such experience can be collected by the KAM on the target robot. Then, the accumulated experience is used to train a DNN approximating the control law $f(\cdot)$ for accurate trajectory tracking and online adaption control. Specifically, in the KAM module, a 3-layer neural network is used to model the relationship between the desired output $f_{desired}(t)$, the NK-DNA's control reference $x_{nkdna}(t)$, actual output $f(t)$ with the adjusted control reference $x_{ideal}(t)$ (i.e. the system reference in Fig.3) based on Equations (1-4).

INITIAL EXPERIMENTS

To examine our proposed approach, we use KAM to learn about the inverse dynamics difference between the target robot and the source robot. Then, we use the

learned knowledge as an add-on module to the control system of the target robot. The dataset is collected as $D = \{u(t),\ f_{desired}(t),\ x_{nkdna}(t),\ f(t)\}$ where the current state is $u(t)$, the desired output is $f_{desired}(t)$, the control reference given by the source robot's knowledge is $x_{nkdna}(t)$, the actual output is $f(t)$. As shown in Figure 4, a total of approximately 10,000 datasets are collected.

dataSet (1).csv

```
0.368121202,0.659485339,0.219526515,0.985377657,0.58198936,0.653367724,1.335180742,1.570541702
1.096458423,0.346971972,0.969183118,0.027594363,1.236597164,0.723109343,1.07570996,2.185574367
0.796004035,1.230080508,0.416391162,0.212613227,0.314159084,1.466002752,0.332235418,0.86367312
0.659918906,1.187232748,0.759616824,1.213546671,0.559139949,0.54078717,0.939044804,2.387457059
0.459472877,1.489826382,1.474435508,1.361897551,0.529329266,0.360362888,2.074513504,0.208586693
0.446756269,0.948652613,0.808456473,0.090822738,0.386880556,0.185127923,1.73036323,1.317301862
0.950379683,0.782314826,1.255055386,1.472023085,1.202357595,1.089347086,2.070180558,1.15719277
0.758235844,1.092777396,0.168671245,1.115609022,0.890992794,0.752646427,1.695358365,1.337978093
0.142638963,0.68252963,0.31230235,1.431944925,1.123790318,0.191767044,2.014408738,0.916442831
0.401462679,1.227428475,1.05421469,0.916836553,1.010776597,1.455680732,2.587413453,0.589216664
1.158263326,1.22528966,1.381693332,0.593644243,0.362249049,0.367056003,1.435086442,1.599288856
0.149314549,1.077735984,1.023043991,0.252857085,0.73872369,0.682289855,1.485330761,1.998017274
1.158900394,0.757486628,0.171977438,0.627228213,0.142591251,1.350764464,2.062121287,2.32300214
0.191926692,1.268687282,0.834160068,0.545539643,0.628001553,0.929359965,2.918562175,1.77876794
0.631328727,1.122192128,0.233999575,1.285221676,1.118691122,1.483744254,1.603298877,0.85136625
0.135279198,1.254484121,0.697782061,0.499994276,0.301564428,1.120228571,2.859858576,2.694194276
1.381303632,0.612089723,1.31858822,1.109351273,0.987240762,0.98955803,2.555642846,1.416428103
0.606887131,1.25531425,1.13076055,0.628604839,0.144587117,1.48585834,1.325013653,0.256638682
0.945735241,0.228047436,1.006921065,1.067911407,0.231960237,1.20464192,2.484607627,1.62384561
0.213063722,0.52726881,0.504892526,0.676504097,0.741743307,0.16627152,0.209986682,1.4575769
0.571993441,1.007952089,0.788657398,1.332355698,1.480383203,0.611269553,1.839582374,2.33674114
0.779783876,1.418744228,0.742164953,1.17080118,0.132095888,0.960388735,0.23110356,2.304469521
0.067389632,0.883160153,0.005840168,1.476498734,0.018794079,0.65641471,0.293379119,2.435645946
0.671023104,0.01421092,0.347243176,1.013391886,1.072970786,0.700043384,1.375491563,1.313647923
1.176308979,0.961958497,1.144571253,1.090066325,0.926804109,0.226812374,1.834671225,2.914167462
0.555788638,0.776849594,1.364786732,0.059352207,0.14992044,0.87792642,1.487125833,1.416941443
1.395269635,0.887681721,1.122936095,0.671250608,1.449493138,0.067419071,2.908477897,2.257461608
0.742201711,1.105514545,0.709591126,0.942865935,0.417382991,0.112292911,1.689464478,1.203705891
0.388179029,1.456872475,0.36209839,0.698655176,0.582990068,0.38288763,0.722761877,2.204125661
```

Fig. 4. The dataset collected on the target robot.

To get the adaption control reference (i.e. the system reference in Fig. 3), we calculate it as $x_{ideal}(t) = x_{kam}(t) + x_{nkdna}(t)$.We can optimize the actual $f(t)$ towards $f_{desired}(t)$ by tuning the $x_{kam}(t)$ through stochastic gradient descent during the KAM training process. The experiment results are illustrated in Figure 5, as we can see that without the KAM adjustment (the blue line), the original system can barely

follow the source robot's reference. In contrast, the target robot achieves high performance following the source robot's reference when the KAM is applied to its controller. The original system's root-mean-square (RMS) errors (without KAM) and use of the KAM are about 3.4509 and 0.1140, respectively. It illustrates how a trained KAM module can facilitate knowledge sharing among robots.

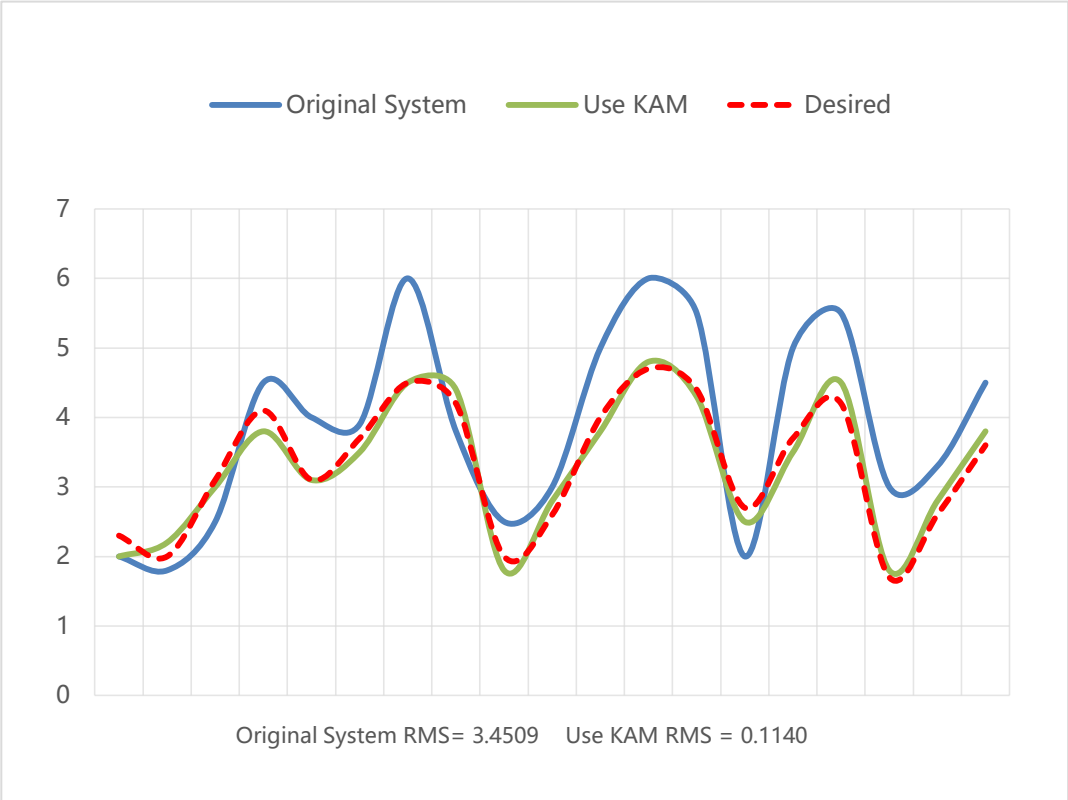

Fig. 5. The tracking control performance comparison.

CONCLUSIONS AND FUTURE WORK

In this paper, we propose an adaptive control method oriented to promote knowledge sharing between different robot systems. In order to ensure the target robot's performance in reusing the source robot's knowledge, we design an add-on module,

which we call the knowledge adaption module (KAM), to the target robot's control system to enhance the adaptability of the NK-DNA-based knowledge reusing. Our approach uses the NK-DNA to acquire knowledge of the robot's inverse dynamics and then reuse such knowledge in other robot's control systems by utilizing the KAM. The initial experiment shows that the proposed KAM is promising for knowledge sharing between different robots.

As this study is still at an early stage of research, there are further studies and improvements to be done, some of which are:

- Further optimization and design of the KAM.
- To examine the method under uncertain and complex dynamics.
- Improvement and further development of the KAM-based robots.

ACKNOWLEDGEMENT

The authors would like to thank the editors and anonymous reviewers for their valuable comments and suggestions on this paper. This work was supported by the Sichuan Science and Technology Program under Grant 2019YFH0185.

REFERENCES

Adlakha, R., & Zheng, M. (2020). An Optimization-Based Iterative Learning Control Design Method for UAV's Trajectory Tracking. 2020 American Control Conference (ACC), 1353–1359.

Ammar, H. B., Eaton, E., Ruvolo, P., & Taylor, M. E. (2015). Unsupervised cross-domain transfer in policy gradient reinforcement learning via manifold alignment. Twenty-Ninth AAAI Conference on Artificial Intelligence.

Bokeno, E. T., Bort, T. M., Burns, S. S., Rucidlo, M., Wei, W., & Wires, D. L. (2018). Package delivery by means of an automated multi-copter UAS/UAV dispatched from a conventional delivery vehicle. Google Patents.

Cavallo, F., Limosani, R., Manzi, A., Bonaccorsi, M., Esposito, R., Di Rocco, M., Pecora, F., Teti, G., Saffiotti, A., & Dario, P. (2014). Development of a socially believable multi-robot solution from town to home. Cognitive Computation, 6(4), 954–967.

Daftry, S., Bagnell, J. A., & Hebert, M. (2016). Learning transferable policies for monocular reactive mav control. International Symposium on Experimental Robotics, 3–11.

Dai, W., Chen, Y., Xue, G.-R., Yang, Q., & Yu, Y. (2008). Translated learning: Transfer learning across different feature spaces. Advances in Neural Information Processing Systems, 21, 353–360.

Devin, C., Gupta, A., Darrell, T., Abbeel, P., & Levine, S. (2017). Learning modular neural network policies for multi-task and multi-robot transfer. 2017 IEEE International Conference on Robotics and Automation (ICRA), 2169–2176.

Ding, F., Wang, X., Chen, Q., & Xiao, Y. (2016). Recursive least squares parameter estimation for a class of output nonlinear systems based on the model decomposition. Circuits, Systems, and Signal Processing, 35(9), 3323–3338.

Gupta, A., Devin, C., Liu, Y., Abbeel, P., & Levine, S. (2017). Learning invariant feature spaces to transfer skills with reinforcement learning. ArXiv Preprint ArXiv:1703.02949.

Kim, J., Cauli, N., Vicente, P., Damas, B., Bernardino, A., Santos-Victor, J., & Cavallo, F. (2020). Cleaning Tasks Knowledge Transfer Between Heterogeneous Robots: A Deep Learning Approach. Journal of Intelligent & Robotic Systems, 98(1), 191–205. https://doi.org/10.1007/s10846-019-01072-4

LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. Nature, 521(7553), 436–444.

Liang, X. (2019). Image-based post-disaster inspection of reinforced concrete bridge systems using deep learning with Bayesian optimization. Computer-Aided Civil and Infrastructure Engineering, 34(5), 415–430.

Liang, X., Zheng, M., & Zhang, F. (2018). A scalable model-based learning algorithm with application to UAVs. IEEE Control Systems Letters, 2(4), 839–844.

Pereida, K., Helwa, M. K., & Schoellig, A. P. (2018). Data-efficient multirobot, multitask transfer learning for trajectory tracking. IEEE Robotics and Automation Letters, 3(2), 1260–1267.

Sajedi, S. O., & Liang, X. (2019). A convolutional cost-sensitive crack localization algorithm for automated and reliable RC bridge inspection. In Risk-based Bridge Engineering (pp. 229–235). CRC Press.

Sanin, C., & Szczerbicki, E. (2006). Using set of experience in the process of transforming information into knowledge. International Journal of Enterprise Information Systems (IJEIS), 2(2), 45–62.

Taylor, M. E., & Stone, P. (2009). Transfer learning for reinforcement learning domains: A survey. Journal of Machine Learning Research, 10(7).

Wang, Z., Song, Y., & Zhang, C. (2008). Transferred dimensionality reduction. Joint European Conference on Machine Learning and Knowledge Discovery in Databases, 550–565.

Zhang, H., Sanin, C., Szczerbicki, E., & Zhu, M. (2017). Towards neural knowledge DNA. Journal of Intelligent & Fuzzy Systems, 32(2), 1575–1584. https://doi.org/10.3233/JIFS-169151

Zhou, S., Helwa, M. K., Schoellig, A. P., Sarabakha, A., & Kayacan, E. (2019). Knowledge Transfer Between Robots with Similar Dynamics for High-Accuracy Impromptu Trajectory Tracking. 2019 18th European Control Conference (ECC), 1–8. https://doi.org/10.23919/ECC.2019.8796140