ELEKTROTEHNIŠKI VESTNIK 90(1-2): 45–50, 2023
ORIGINAL SCIENTIFIC PAPER

A Nature-Inspired Implementation of the Computer Opponent in the Arcade Computer Game of Darts

Nina Tovornik, Iztok Fister Jr., Vili Podgorelec, Lucija Brezočnik

University of Maribor, Faculty of Electrical Engineering and Computer Science, Intelligent Systems Laboratory
Koroška cesta 46, 2000 Maribor, Slovenia

E-mail: nina.tovornik1@student.um.si, iztok.fister1@um.si, vili.podgorelec@um.si, lucija.brezocnik@um.si

Abstract. Some three billion people play video games worldwide, which makes the video games industry an integral part of modern life. Such demand requires game developers to provide increasingly more complex games. Lately, to tackle the problem, many researchers have utilized approaches inspired by the Darwinian theory of biological evolution. The paper presents an overview of optimization techniques and their use in game development, and proposes a method for the computer opponent in the game of darts. Using the proposed computer opponent, the effect of different difficulty levels on its performance is shown.

Keywords: Evolutionary Algorithms, Genetic Algorithms, Swarm Intelligence, Computer game of darts, Unity game engine

Implementacija računalniškega nasprotnika v arkadni računalniški igri pikado po vzoru algoritmov iz narave

Približno tri milijarde ljudi po vsem svetu igra video igre, zato lahko industrijo video iger umestimo med sestavne dele sodobnega življenja. Toda takšen izkazani interes od razvijalcev iger zahteva, da zagotavljajo vedno zapletenejše igre. V zadnjem času so mnogi raziskovalci za reševanje tega problema uporabili pristope, ki jih je navdihnila Darwinova teorija biološke evolucije. Posledično je v članku najprej predstavljen pregled optimizacijskih tehnik in njihova uporaba pri razvoju iger. Sledi predstavitev predlagane metode za razvoj računalniškega nasprotnika pri igri pikada in prikaz tega, kako uporaba različnih težavnostnih stopenj vpliva na njegovo delovanje.

1 INTRODUCTION

In the latest report by Newzoo [1], the leader in video games and gamer data, it is shown that some three billion people worldwide play video games. That means that some 37% of the world population consists of gamers. Consequently, along with the positive growth trend, we can undoubtedly say that the video games industry is an integral part of modern life.

With the increasing complexity of the game development, and, consequently, games themselves, researchers and developers started to utilize different Artificial Intelligence techniques in game development. Because of the immense amount of algorithms in this field, we will limit ourselves only to those inspired by the Darwinian theory of biological evolution [2]–[6].

The focus of our paper is on the use of evolutionary approaches in computer games. The goal is to present their functionality in developing an arcade computer game of darts [7]. The main contributions of the paper are:

- a novel method is proposed for the computer opponent in the arcade computer game of darts;
- the effect of different difficulty levels of the proposed computer opponent on its performance is evaluated.

2 NATURE-INSPIRED ALGORITHMS IN COMPUTER GAMES

Nature is our most valuable asset. It's our first teacher and continues to teach, surprise, and inspire us to this day. It offers us a daily insight into its ecosystem, which is precisely what inspires scientists worldwide to create inventions that simplify and improve the lives of humanity.

Solutions inspired by the nature behavior can be seen in various engineering fields, such as Medicine, Robotics, Software Engineering, and Machine Learning [8]–[12]. Lately, a concept inspired by nature has also been introduced to the computer game development, mainly through Evolutionary Algorithms (EAs).

The EAs dynamic follows the basic principle of the Darwin theory of evolution [13]–[16]. The idea is based on the concept that only the most successful or fittest individuals from a population survive, while the

Received 19 December 2022
Accepted 10 February 2023

others are removed from the population. Therefore, only the strong ones that adapt well to their environment reproduce. EA uses three main operators, i.e., selection, mutation, and crossover [9], [17]. The EAs subfields include:

- Genetic Algorithms (GA),
- Genetic Programming (GP),
- Evolutionary Strategies (ES),
- Evolutionary Programming (EP), and
- Differential Evolution (DE).

The focus of the paper is on GA due to the concept of operations that it offers.

3 GENETIC ALGORITHM

GAs are one of the most general approaches to solutions optimization, as they provide the most straightforward mapping from the natural process of evolution to a computer system. GA is capable of finding solutions to complex problems, and is, at the same time, relatively easy to use. Therefore, it can be used for various optimization problems [9].

The GA structure and operations follow the pseudocode in Algorithm 1.

Algorithm 1 Genetic Algorithm pseudocode.

```
Create a random population of individuals and name it rand_pop.
while termination_condition_not_met do
    Create a new population of individuals new_pop;
    while new_pop is empty do
        Select two individuals from rand_pop, preferably those that have a greater probability of higher fitness function.
        Cross these individuals to create a new generation.
    end while
    Let each individual in new_pop have a random chance to mutate.
    Replace rand_pop with new_pop.
end while
For a solution, select the individual with the highest fitness function from new_pop.
```

4 PROPOSED METHOD

The pseudocode presented in Algorithm 1 represents the basis for developing a computer opponent. The main character in the game of darts is the target. The computer moves its target randomly across the playing area, i.e., the dartboard. If the target is positioned at good coordinates for shooting arrows, it will give the computer opponent higher points and a higher chance of winning. Therefore, with each generation, the coordinates that demonstrate the upgraded position of the target are improved. In a certain way, those coordinates represent the area limit of the target movement.

Therefore, the task of the proposed computer opponent is crosshair aiming. The current position of the crosshair can be presented mathematically with Eq. 1

$$A = (x_1, y_1, z_1) \tag{1}$$

and can move (up, down, left, and right) in the following manner:

$$f(x) = \begin{cases} B = (0, y_2, 0) & \text{if } x = 0 \\ B = (0, -y_2, 0) & \text{if } x = 3 \\ B = (x_2, 0, 0) & \text{if } x = 2 \\ B = (-x_2, 0, 0) & \text{if } x = 1 \end{cases} \tag{2}$$

where:
B - movement of the crosshair.

The new position is then calculated as:

$$\begin{aligned} AB =& (x_2 - x_1, y_2 - y_1, 0 - z_1) \\ |AB| =& \sqrt{(x_2 - x_1)^2, (y_2 - y_1)^2, (-z_1)^2} \end{aligned} \tag{3}$$

With every round in the game (after the last shot of an arrow), the computer-released arrows are collected and saved in a directory. That represents our population of individuals. Besides collecting arrows at the end of each computer round, we also start running our GA algorithm. First, we begin by carrying out the selection in GA. We find the best arrow collected up from the directory. The current best has the highest fitness function, or, in other words, it is the arrow that carries the highest points. After that, we crossover and mutate the fittest arrow with each of the three currently released arrows to get a new generation. More specifically, we crossover and mutate the coordinates from the fittest arrow and the coordinates from the presently shot arrows, in order to get a new generation.

Algorithm 2 gives an overview of the implementation of GAs and shows how selection, crossover, and mutation are used to create new generations. The new generations give us the coordinates for the limits of the computer target.

When creating a new generation, we assign it to a random operation. Therefore, the crossing takes place so that some coordinates from the released arrows are preserved and some are changed. For example, the x-coordinate of the released arrow is preserved, and the y-coordinate of the fittest arrow replaces the y-coordinate. A new generation is created in such manner. Afterward, the new generation replaces the previous one

and becomes our current generation to be used in the game.

As a result, the current generation represents the reset limits that define new and upgraded coordinates of limits for the computer target. In theory, the higher the number of the new generations, the more accurate the target is, and the closer the computer opponent can get to scoring higher and higher points. So, the newly released arrows will carry a higher fitness function value.

Algorithm 2 Genetic Algorithm used in the game of darts.

1: Create a population that holds all the so far released arrows and name it $arrow\_pop$.
2: Get a population of the currently released arrows and name it $curr\_pop$.
3: Create a population of coordinates that represents limits and name it $curr\_limits$.
4: Find the current highest points from the $arrow\_pop$ - the currently best-valued fitness function, and call it the fittest.
5: **while** termination_condition_not_met **do**
6: Cross the x and y coordinates of the arrows from $curr\_pop$ with the x and y coordinates of the fittest so that a new generation is created. Name the new generation new_limits;
7: Let each individual in $curr\_pop$ have a random chance to mutate.
8: Replace $curr\_limits$ with new_limits.
9: **end while**
10: Use population of new_limits to achieve better results for the movement of the target.

When approaching the end of the game, we reach a situation where we can no longer rely only upon GA. For example, if a computer opponent has only five points to nullify, but it would be throwing at the maximum of points, such as 60, or even 25 points, it would have little to no chance of winning.

Therefore, we choose a different approach and develop a hybrid algorithm combining GA concepts with an Ant Colony Algorithm (ACO). The ACO technique is a widely used metaheuristic based on Swarm Intelligence [18].

Due to the possibility of a rapid reduction of the current points that the computer opponent must nullify, we execute a hybrid algorithm after each throw of an arrow (not after the whole round). So, the updated coordinates instruct the target precisely where to shoot next in order to win. Thus, the game introduces a hybrid algorithm after the computer opponent is left with sixty or fewer points to nullify.

In the hybrid algorithm operation, the best results, i.e. the released arrow with the most points, are no longer important. Instead, the released arrows that carry the closest points to the currently remaining points of the computer opponent come to the forefront. Now, the closest arrows carry the higher fitness function.

Algorithm 3 shows that, in each iteration, an arrow, is selected with points that are equal to or smaller, but the closest to the current computer points. Based on the selected or the fittest arrow, its x and y-coordinates are used to create a new generation. Therefore, ACO is included in the selecting part of the hybrid algorithm, and GA is included in the crossover and mutation, which are the same as in Algorithm 2.

Algorithm 3 Use of the Hybrid algorithm in the game of darts.

1: Create a population that holds all the so far released arrows and name it $arrow\_pop$.
2: Get a population of the currently released arrows and name it $curr\_pop$.
3: Create a population of coordinates that represents limits and name it $curr\_limits$.
4: Find the currently best valued fitness function of an arrow from the $arrow\_pop$, which points are equal or smaller, but the closest to the current points that the computer opponent must nullify.
5: **while** termination_condition_not_met **do**
6: Cross the x and y coordinates of the arrows from $curr\_pop$ with the x and y coordinates of the fittest so that a new generation is created. Name the new generation new_limits;
7: Let each individual in $curr\_pop$ have a random chance to mutate.
8: Replace $curr\_limits$ with new_limits.
9: **end while**
10: Use population of new_limits to achieve better results for the movement of the target.

To make the game more interesting we make the computer opponent smarter in order to challenge the player. To accomplish this, we introduce three difficulty levels or game modes, e.g., easy, normal, and hard. The player can choose the preferred game mode in the game menu, as shown in Figure 1.

The development and operation mentioned above represent an easy mode. The main difference between the easy and normal modes is in the directory. In addition to collecting the computer arrows, the normal mode also collects the player arrows. In theory, such circumstances give the computer the ability to learn faster.

Nonetheless, our goal is to develop a hard mode that would be able to make the computer opponent learn fastest. Therefore, the hard mode contains all the attributes from the previously mentioned modes. Additionally, the hard mode also follows the direction arrow developed to make the game of darts more challenging to play. The direction arrow rotates its position on the

y-coordinate constantly. When the arrow of the player or the computer opponent is released, the current position of the direction arrow is added to the y-coordinate of the released arrow. So, players must pay extra attention to it. That is why the hard mode tracks the direction arrow current position carefully and fires arrows when the condition is the most suitable.

The easy and normal modes do not consider the direction arrow rotating position. Because of that, the two game modes are more likely to miss the desired points their target is leading them to. In general, this means that the hard mode should be more precise and narrow regarding firing its arrows than the other two game modes.

5 Results

The game of darts is implemented using the Unity game engine [19] and written in the C# programming language. The Blender tool [20] is used to render all objects of the game, i.e., dartboard, arrows, and scoreboard.

For the experiment, twenty measures are taken at each difficulty level. Though the differences between difficulty levels are minor, different results are obtained. For a better presentation of the played games at all levels, their average performance is shown in Figure 2. The graph shows that the average game in the hard game mode is the most successful.

Based on the results, the average number is calculated of the newly created generations needed to finish the game in a particular game mode. The average number of the generations needed for the:

- easy mode is 40;
- normal mode is 29, and
- hard mode is only 26.

The results confirm that the computer opponent in the hard mode learns the fastest. The easy and normal modes are left to chance at times resulting in the so-called anomalies due to ignoring the direction arrow. Occasionally, this coincidence leads them either to very good or very bad results.

Undoubtedly, incorporating the player data of released arrows in the collected-arrows directory gives the normal and hard modes a significant advantage over the easy mode.

Figure 3 shows the differences between the game difficulty levels. The games played in each game mode and the number of newly created generations needed to win the game are shown. Their results are given in four intervals displaying the number of created generations. The results show how many times each result is higher or lower than the set intervals.

Therefore, we can see that the easy mode dominates the intervals where the results are higher than the fifty and thirty new generations that are required to win. The

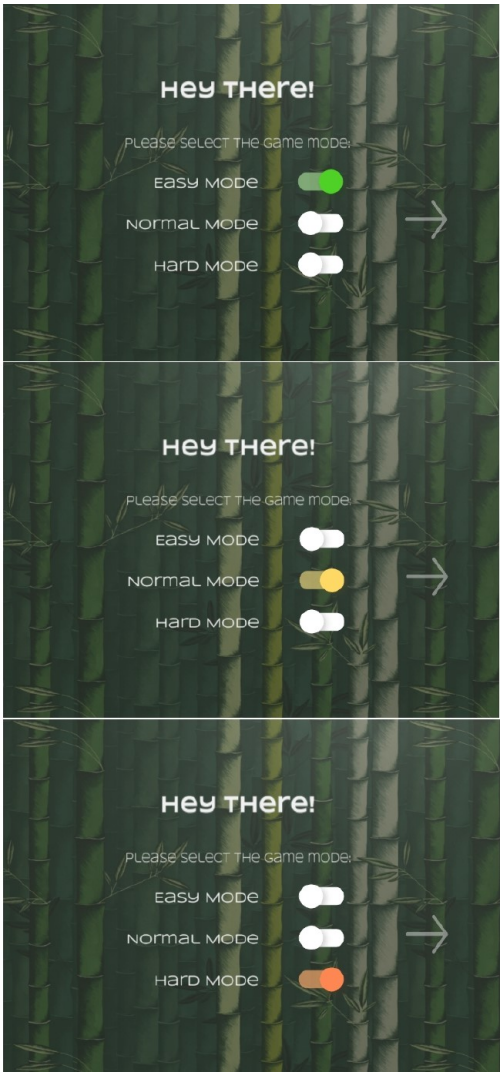

Figure 1. Entry menu of the game of darts. Three images are displayed. They show (from top to bottom) the easy game mode (green color of the toggle switch), the normal game mode (yellow color of the toggle switch), and the hard game mode (red color of the toggle switch). The player can choose either of the represented switches to be played by the computer opponent.

easy difficulty level is represented by the yellow bars. The hard mode, represented by the red bars, dominates intervals that are lower than the fifty and thirty new generations. This mode holds most of the results that are closer to the game minimum (minimum number of newly created generations).

Indeed, the minimum of this rate would decrease with a larger number of experiments. However, these results already give a very good presentation of the functioning of individual difficulty levels.

The above experiments determine the necessary number of iterations (new generations) of self-adaptation

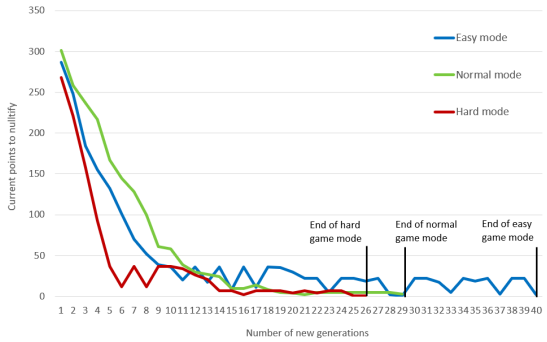

Figure 2. Average game played in each game mode.

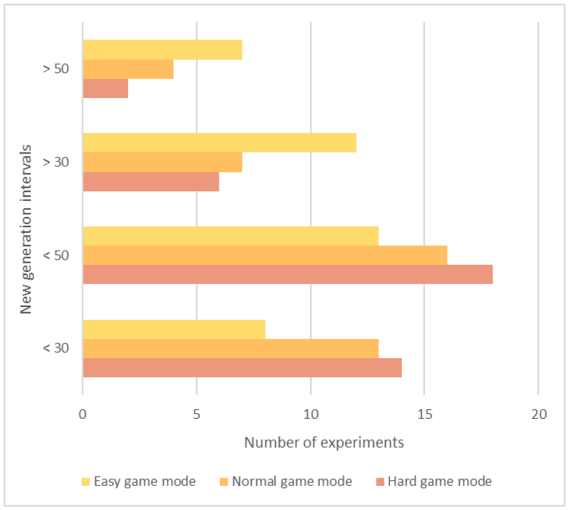

Figure 3. Interval representation of all results of the experiments on the graph.

and show the improvement of the computer opponent to win the game. The adjusting and upgrading is shown in Figure 4.

6 Conclusion

This paper shows that it is irrelevant in which game mode we play or have it played by the computer. Sooner or later the computer opponent will win.

Using GA and ACO the computer opponent learns independently and adapts successfully to its playing ground.

There is still room for an improvement in the game of darts. The conducted experiments reveal some possibilities. One of the improvements can be a more accurate releasing of arrows. A directory can be created to store more precise coordinates of individual fields on the dartboard to be used in a normal and hard game mode. Thus, the normal mode would differ from the easy mode in two ways, and the normal and hard mode would improve the game further, as they would most likely bring the computer opponents even better results.

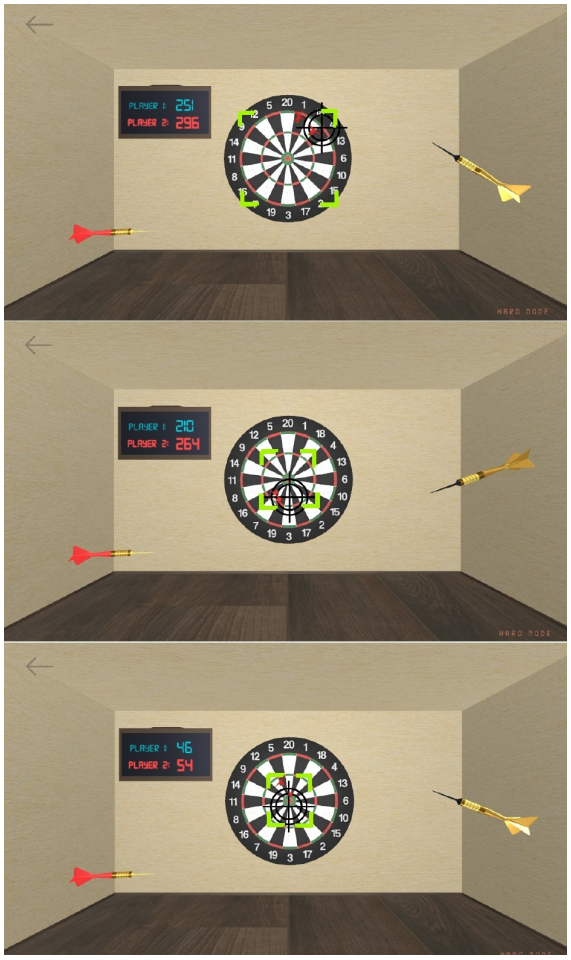

Figure 4. The demonstration of playing with a computer opponent in the hard game mode. The green frames in the figure represent the computer opponent adjusting and upgrading the coordinates to improve the limits for its target.

Acknowledgement

The authors acknowledge the financial support by the Slovenian Research Agency (Research Core Funding No. P2-0057).

References

[1] N. Tom Wijman, "The games market's bright future: Player numbers will soar past 3 billion towards 2024 as yearly revenues exceed $200 billion." https://newzoo.com/insights/articles/the-games-markets-bright-future-player-numbers-will-soar-past-3-billion-towards-2024-as-yearly-revenues-exceed-200-billion, 2021. Online, accessed 15. 11. 2022.

[2] K. O. Andrade, R. C. Joaquim, G. A. Caurin, and M. K. Crocomo, "Evolutionary algorithms for a better gaming experience in rehabilitation robotics," *Computers in Entertainment (CIE)*, vol. 16, no. 2, pp. 1–15, 2018.

[3] A. Fernández-Ares, A. M. Mora, J. J. Merelo, P. García-Sánchez, and C. Fernandes, "Optimizing player behavior in a real-time strategy game using evolutionary algorithms," in *2011 IEEE congress of evolutionary computation (CEC)*, pp. 2017–2024, IEEE, 2011.

[4] R. D. Gaina, S. Devlin, S. M. Lucas, and D. Perez-Liebana, "Rolling horizon evolutionary algorithms for general video game playing," *IEEE Transactions on Games*, vol. 14, no. 2, pp. 232–242, 2021.

[5] V. Kraner, I. Fister, and L. Brezočnik, "Procedural content generation of custom tower defense game using genetic algorithms," in *International Conference "New Technologies, Development and Applications"*, pp. 493–503, Springer, 2021.

[6] W. L. Raffe, F. Zambetta, and X. Li, "A survey of procedural terrain generation techniques using evolutionary algorithms," in *2012 IEEE Congress on Evolutionary Computation*, pp. 1–8, IEEE, 2012.

[7] N. Tovornik, "Using evolutionary algorithms in development of computer opponent in game of darts," Master's thesis, University of Maribor, Faculty of Electrical Engineering and Computer Science, 2022.

[8] L. Brezočnik, I. Fister, and V. Podgorelec, "Swarm intelligence algorithms for feature selection: A review," *Applied Sciences*, vol. 8, no. 9, 2018.

[9] F. Streichert, "Introduction to evolutionary algorithms," *Evolutionary Computation 1*, 2018.

[10] E.-G. Talbi, *Metaheuristics: from design to implementation*. John Wiley & Sons, 2009.

[11] C. Blum and A. Roli, "Metaheuristics in combinatorial optimization: Overview and conceptual comparison," *ACM computing surveys (CSUR)*, vol. 35, no. 3, pp. 268–308, 2003.

[12] K. Ksiazek, W. Masarczyk, and I. Nowak, "Heuristic approach to the game of darts by using genetic algorithm and ant colony optimization," in *Proceedings of the International Conference for Young Researchers in Informatics, Mathematics and Engineering (ICYRIME 2017), CEUR*, vol. 1852, pp. 33–38, 2017.

[13] K. De Jong, D. B. Fogel, and H.-P. Schwefel, "A history of evolutionary computation," *Handbook of Evolutionary Computation A*, vol. 2, pp. 1–12, 1997.

[14] A. P. Engelbrecht, *Computational intelligence: an introduction*. John Wiley & Sons, 2007.

[15] A. E. Eiben, J. E. Smith, *et al.*, *Introduction to evolutionary computing*, vol. 53. Springer, 2003.

[16] A. E. Eiben and J. Smith, "From evolutionary computation to the evolution of things," *Nature*, vol. 521, no. 7553, pp. 476–482, 2015.

[17] P. A. Vikhar, "Evolutionary algorithms: A critical review and its future prospects," in *2016 International conference on global trends in signal processing, information computing and communication (ICGTSPICC)*, pp. 261–265, IEEE, 2016.

[18] H. N. K. Al-Behadili, K. Ku-Mahamud, and R. Sagban, "Ant colony optimization algorithm for rule-based classification: Issues and potential solutions," *J. Theor. Appl. Inf. Technol*, vol. 96, no. 21, pp. 7139–7150, 2018.

[19] Unity, "Official Unity Website." https://unity.com/. Online, accessed 10. 11. 2022.

[20] Blender, "Official Blender Website." https://www.blender.org/. Online, accessed 10. 11. 2022.

Nina Tovornik received her B. Sc. degree from the Faculty of Electrical Engineering and Computer Science of the University of Maribor, Slovenia, in 2019, and her M. Sc. degree from the same faculty in 2022. Her research interest is in intelligent game design.

Iztok Fister Jr. received his B.Sc., M.Sc., and Ph.D. degrees in Computer Science from the University of Maribor, Slovenia. He is currently an Assistant Professor at the University of Maribor. He has published over 120 research papers in refereed journals, conferences, and book chapters. His research interests include Data Mining, Pervasive Computing, Optimization, and Sports Science. He has acted as a Program Committee member at some 30 international conferences and is a member of the editorial boards of three international journals.

Vili Podgorelec is a professor of computer science at the University of Maribor, Slovenia. His research interests include intelligent systems, data analytics, computational intelligence, medical informatics, and software engineering. He has been involved in many national and international research projects, both scientific and industrial R&D projects. He has worked as a visiting professor and/or researcher at several universities around the world, including University of Osaka, Japan; Federal University of Sao Paulo, Brazil; University of Nantes, France; University of La Laguna, Spain; University of Madeira, Portugal; University of Applied Sciences Seinäjoko, Finland. He is an author of over 70 peer-reviewed scientific journal papers, three books and several book chapters. He has performed 15 invited talks at major international conferences worldwide.

Lucija Brezočnik received her B.Sc. and M.Sc. degrees from the Faculty of Electrical Engineering and Computer Science of the University of Maribor, Slovenia, in 2014 and 2016, respectively. She is a teaching assistant and a Ph.D. candidate. She works on several projects in the industry and is the author of journal papers, over 20 conference papers, a book chapter, and is a reviewer for over 40 recognized peer-review journals. Her research interests include machine learning, computational intelligence, data science, swarm intelligence, software engineering, and agile software development. She is a recipient of several international and local awards and grants for her research activities. She is also an IEEE Slovenia Vice-Chair.