International Journal of Engineering Education Vol. 32, No. 1(B), pp. 513–520, 2016
Printed in Great Britain

0949-149X/91 $3.00+0.00
© 2016 TEMPUS Publications.

Gamifying an Artificial Intelligence Course in Engineering Education*

RAMON MAS-SANSÓ** and CRISTINA MANRESA-YEE**
University of Balearic Islands, Crta. Valldemossa, km 7.5, 07122 Palma, Spain. E-mail: {ramon.mas, cristina.manresa}@uib.es

Students' motivation and engagement are important factors for actual learning. By using game-thinking and game elements in an education context we can attract learners' attention, increase their motivation and engage them to participate more actively in their learning process. In this work we present a gamified experience to teach Artificial Intelligence to computer science engineers. The inclusion of a competition and other game elements in the course has proved to be fun for the students, they attend more to class, they look for the continuous improvement of their work, no one copies and the level of assignments handed in is high.

Keywords: artificial intelligence education; gamification; motivation; students' competition

1. Introduction

Teaching and learning should not only address student cognition, but also the affective components involved in the learning process [1]. Within these affective components, students' motivation and engagement are important factors in their actual learning from academic tasks. These factors can emerge as a result of a gamified experience.

While no standard definition yet exists for gamification [2], most sources agree that it can be defined as the process of game-thinking and game mechanics to engage users and solve problems in a non-game context [3, 4]. Gamification uses elements of games for purposes other than their normal expected use as part of an entertainment game. Besides entertainment, it looks for joy of use, engagement and improvement of the user experience [3].

There are different game elements that can be included in a gamified application or experience. Reeves and Read [5] listed 10 important ingredients of great games such as self-representation with avatars, ranks, levels and competition under rules that are explicit and enforced. Deterding et al. [3] presented a taxonomy of game design elements by level of abstraction which comprises, among others, game interface design patterns (e.g. badge, leaderboard, level), game design patterns and mechanics (e.g. limited resources), game design principles and heuristics (e.g. clear goals), game models (e.g. challenge) and game design methods (e.g. playcentric design). Zichermann and Cunningham [4] also commented game mechanics to design for engagement such as points, levels, leaderboards, onboarding, challenges and quests, dashboards or customization.

Although nowadays gamification has become a hot topic in study and practice, drawing the attention of academics, practitioners and business professionals in diverse domains including education [6], the use of game-elements in learning started long ago. We find research works from the '80s, where researchers analyzed features that made computer games captivating and used them to make learning interesting [7]. Mining the literature, we find positive results of diverse experiences in this field that prove that gamification can serve to increase the engagement and class attendance, strengthen the learner's motivation, improve the retention of learners' attention and interest, promote the acquisition and understanding of knowledge as well as developing skills, increase the number of voluntary tasks completed, and offer students the opportunity to "Learn while enjoying and Enjoy while learning" [8–11]. Furthermore, students have a positive learning effect when working with game-related learning resources [12].

In this work, we will apply gamification to an Artificial Intelligence (AI) course in an undergraduate engineering degree. The goal of this course is to present a foundation in the principles and technologies that underlie many facets of AI, including knowledge representation, problem solving and logic and reasoning methods from a computational perspective. Specifically, we are interested in autonomous, problem-solving computational entities capable of effective operation in dynamic and open environments, that is, intelligent agents, as they are being used in a wide range of applications in industry such as military, aerospace or online sales [13]. Therefore, computer science engineers are expected to acquire a knowledge base in intelligent agents in the bachelor degree.

To increase the engagement and motivation in the practical part of developing intelligent agents, we

** Both authors contributed equally to this work.

* Accepted 11 August 2015.

will work with a real-application which is a video game. In order to include excitement and challenge to make more appealing the course to the students, we include a competition among the students and the results influence the students' grades.

The main aim of the work is to describe a gamified experience conducted at the University of Balearic Islands when teaching AI to computer science engineers. The paper is arranged as follows: Section 2 compiles related works regarding AI teaching and using competitions in educational courses. Section 3 describes the elements of the gamified experience: the course of AI taught at the Computer Science Engineering degree, the project and the framework given to the students and the competition rules. Section 4 reports the results of the experience. Finally, the last section concludes and discusses the findings.

2. Related work

In order to enhance interests of students in learning IA, we can find diverse teaching approaches. Weidong et al. [14] suggested a research-based approach where students would need to look through materials to give answers to chosen topics. Teachers would import cases from the real world and they would give students useful guidance and resources to encourage them to propose innovative views based on the understanding of the existing knowledge.

Other educators included physical agents to add excitement to the AI course. Different works extended the software agent-centered approach to a robot perspective. Students worked in a simulated environment with physical agents in the real world. Students believed that robot projects help them learn the underlying AI concepts [15, 16]. There are also experiences mixing virtual laboratories and robots. Mckee [17] presented an internet-based laboratory environment where students could work and get involved adding a level of intelligence to control a toy device that was accessible remotely over the network.

Programming video games are a classical field of application for concepts of AI. DeNero and Klein [18] taught foundational concepts of AI using the video game Pac-Man. Students worked on four projects and each project required them to implement general-purpose AI algorithms and inject domain knowledge about the PacMan environment using search heuristics, evaluation functions, and feature functions. They found that the Pac-Man theme added consistency to the course, as well as tapping in to students' excitement about video games. McGovern et al [19, 20] presented a set of graphical games developed in Java that enabled students in an introductory AI course to immediately apply and visualize the topics from class. They found these games to be a very effective tool in enabling the class to be a significant learning experience. Holder and Cook [21] developed a web-based visual integrated simulation environment where students had to program an agent who explored an NxN grid world while avoiding a creature known as the Wumpus. Preliminary results indicated that their students benefited from using the tool in terms of subject interest, confidence in the material, and ability to understand and utilize the presented techniques.

Using video games can be fun and students can see the AI concepts implemented in a real application, but we take the programming of the video games approach one step further. In our experience, we introduce a competition to make the course more appealing to the students. Although, different types of competition can be more engaging than others for different groups of students such as direct or indirect competitions [22], active learning through competition has proven to be a captivating learning factor [16] and this learning approach is focused on a student-centered perspective where the process is more important than the outcome [23].

Carpio Cañada et al. [23] reviewed open competitions where participants had to show their skills in programming and AI knowledge when solving a problem. In some cases, like the Cosmobot or Facebook Hacker Cup, the competition was open to participants both professionals and students. We focus on those works that present competitions at a classroom level, where learners are new to the subject of AI, all have the same base and it is up to them to research and dedicate efforts to improve their solutions out of the given material. DeNero and Klein [18] and McGovern and Trytten [22] included optional competitions in their courses and only used the results for a limited amount of extra credit. Wallace and Margolis [24] presented an experience with open-ended projects, built around a game-like environment and involved different forms of competition (a challenge against the instructor's solution and a round-robin tournament with other classmates). They observed that projects were both engaging and relevant to students: students expressed interest in the projects and a number of individuals were inspired to pursue the projects well beyond the requirements sufficient to earn full credit. Palomo-Duarte et al. [25] presented a rule-based expert system where students develop the strategies to win the game. In order to pass the course, a strategy only needed to defeat a naive (sparring) team that was included in the system. This way, students could be confident they would pass this part of the course, even if the team ended up last in the competition.

3. Gamification of the course

In this section we describe the AI course, the project and the framework given to the students to implement intelligent agents and the competition rules.

3.1 The artificial intelligence course

The 6-ECTS AI course is compulsory for third-year students and is taught during a semester, four hours per week (15 laboratory sessions and 45 theory lessons). Following the competence-based learning (European Higher Education Area, 2014), this subject has three generic competences and a subject-specific one. On the one hand, students will: (1) improve their skills and strategies to acquire new knowledge autonomously, (2) find new resources and manage information in the computer science field and (3) communicate computer science concepts in a written and oral way in different contexts. On the other hand, the specific competence covers that students will gain knowledge about intelligent agents and they will be able to apply the fundamental principles and basic techniques related to them.

Similar to other universities, the course covers the following topics: concepts of AI, solving problems in AI, intelligent agents, state space search, constraint satisfaction problems, planning and the implementation of agents. In this work, we will focus on the last topic that is carried out in five laboratory sessions: the practice and application of the theoretical concepts into the implementation of an intelligent agent.

At the very beginning of the course in the theory class, agents are introduced and categorized. In order to reinforce the knowledge on these concepts from a practical point of view, in the laboratory sessions students work in a project to improve their practical skills implementing intelligent agents.

3.2 The project

McGovern and Fager [19] listed the next six criteria for projects to succeed at the goal of creating a significant learning experience in an introductory AI course:

1. Be enjoyable enough to stimulate and maintain student interest.
2. Be flexible enough to illustrate many of the possible topics that can be covered in an AI class. The project should focus on the AI aspects of each task and not require a significant amount of external knowledge.
3. Be extensible so that stronger students can explore alternatives to the main solution approach while weaker students can still complete the project.
4. Be sufficiently challenging to invest students in creating a good AI solution but not allow trivial answers.
5. Be realistic enough that students gain an appreciation for working on real-world applications of AI.
6. Be feasible to maintain student interest.

Based on these criteria, we present a project consisting on a detailed description and a large amount of scaffolding code with clear directions for students on where to add their implementations. Students are encouraged to work in pairs and although it is not mandatory, few students decide to work alone.
The educational objectives of the project are:

- Define and apply domain knowledge to particular problems.
- Implement an intelligent agent into a real-problem game-based application.
- Acquire knowledge and skills to implement intelligent agents.

DeNero and Klein [18] defined five design goals for projects in computer science programming that they specifically applied to AI projects. In order to support our learning objectives, we pursued those design goals in our project:

- Engagement: student motivation is increased if they face challenging and fun problems.
- Scaffolding: supplying a set of Java classes and methods to the students ensures that most effort is put to implement IA behaviors rather than implementing additional utilities.
- Visualization: results and behavior clues are visualized in early IA development stages.
- Consistency: even if it is intended for agent development the interface and utility classes can also be used to program algorithms supporting other IA topics.
- Grading: sharing a common development environment makes grading easy, as we only have to focus on the portion of code designed to solve the IA problem.

The framework provided to the students is a Java codebase consisting on a set of classes and methods that describes a world populated with bugs. Students are asked to use this framework to implement an intelligent agent simulating the behavior of a bug; so basic Java knowledge is required. Nevertheless, this poses no additional difficulties as this course is addressed to third-year computer science students, who are already familiar with the Java programming language. The framework also provides both perceptions and actions mechanisms for the agents.

Bugs are mainly interested in getting resources to

survive but have to face several obstacles: physical walls which are not transparent and other bugs that are also looking to fulfill their needs for survival.

The world where these bugs live is a closed scenario with randomly generated walls. Such a dynamic environment introduces a challenging target, as every new simulation implies a completely different scenario. There is a limited quantity of resources randomly distributed around the world that the bugs can collect. Such resources are the only way to ensure bugs survival (food, energy and bullets).

Although a bug is usually a peaceful creature, it can become very aggressive when it needs resources. Instinctively a bug takes all the resources that it can see even if that means confronting other bugs.

Morphologically, the agents are described as customizable squares. They are identified using a personalized icon and a name (both defined by the student). Functionally, the agents have several devices (Fig. 1):

- Variable speed mechanisms for displacement and rotation;
- a system of three configurable viewers;
- a device to jump into hyperspace (a limited number of times before overheating) and
- a gun of limited range to shoot bullets.
- The agent actions are programmed changing the attributes of these actuators. So, we can control:

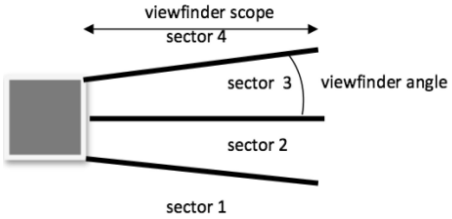

Fig. 1. Agent morphology.

- The agent's speed: using public methods to set both the linear and angular velocity.
- The agent's motion: using methods to move forwards, backwards, turn right or left and to jump to hyperspace.
- The range of vision: setting the angle (Fig. 2) and scope (Fig. 3) of the viewfinders.
- The activation of the gun. A bug is designed in order to resist just four impacts, then it explodes.

To gather perceptions, a status structure that is continuously updated is provided. The student can programmatically check this structure to see a detailed description of the current status of the world.

An agent can perceive:

- How many impacts it has received.
- Its current angular and linear speeds.
- The remaining jumps to hyperspace.
- The number of available resources.
- Whether it collides with an obstacle.
- The position (in pixel coordinates) and orientation (in degrees) of the agent.
- The status of each of the viewfinders (if an object is detected, in which sector, the distance to the object and which kind of object it is: wall, resources, enemy).

The agents have some limitations that students have to be aware of:

- No motion is allowed when there are no resources left.
- Resources are spent proportionally to linear and angular velocity.
- Jumping to hyperspace increases resources consumption.
- A limited amount of resources is available at the beginning of the simulation.

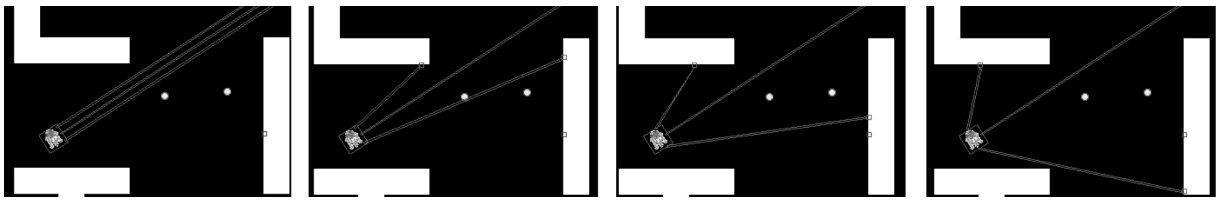

Fig. 2. Setting the angle of the viewfinders.

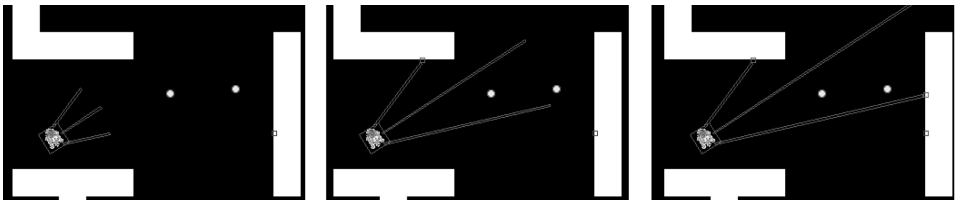

Fig. 3. Setting the scope of the viewfinders.

- Collecting resources gives extra fuel, bullets and protection.

In order to get used to the agents, the students have the possibility to manually interact with the application. Every action and every perception can be manually triggered and gathered using the keyboard, so testing behaviors becomes simple. Debugging is also easy using the visual clues that can be enabled in the framework Fig. 4). The viewfinders are drawn and intersecting points with the walls and the other agents are also displayed.

3.3 The competition

To motivate the students, we organize a competition where the agents will try to beat each other. The competition, informally called "the final battle", takes place the last day of the course and it is open to the public. The most recent competition involved thirty agents.

Due to the continuous increase in the number of students (from 8 groups the first year to 30 groups in the last competition), the competition format evolved from a round-robin or all-play-all competition at the beginning to a group stage and knockout phase when the number of groups was already substantial.

The agents are split into groups of four teams. Two teams from each group proceed to the knockout phase alongside the best third-placed teams to complete the sixteen agents that will compete in this phase. From now on, agents are directly eliminated when they lose.

When two agents compete, the one who collects more resources wins. Destroying the rival is allowed, so in this case the winner is the one left alive. Agents have to be careful about spending all the resources they have, as they would not be able to move anymore and could be easily destroyed.

Initial surveys showed that some students complained about the disadvantage that they could have with a random scenario, so to minimize random bias, an agent must be defeated three times in five bouts. Anyhow, the referee (the lecturer) can decide to generate another random world if a position is much more favorable to one of the contenders.

The results of the competition are considered to grade the project of the students (to an extent of a 30%). The students have to deliver their Java classes a week prior to the contest. The required documentation must include a brief (pecha kucha style) presentation where the students detail their strategy, so the correctness of the implementation can be verified during the contest.

4. Results and discussion

The experience was conducted during five years with over 160 students grouped into 84 teams. In the third-year of the degree most students know each other, and including a competition-based game-context is fun, motivating and engaging. Most students have never had this kind of approach in

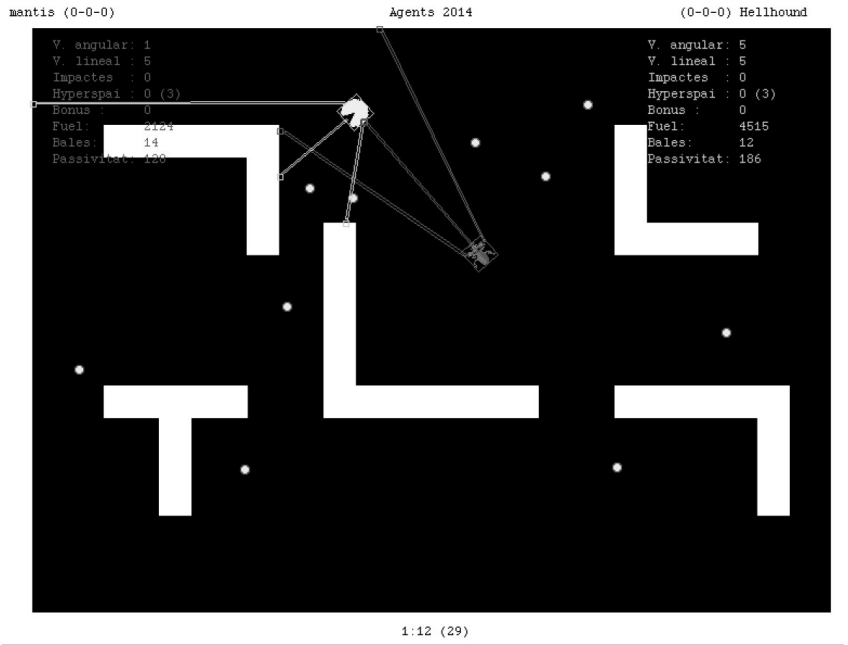

Fig. 4. The framework showing the status of the simulation. See the identification of the agents on the top of the window and the personalized icon shown for each agent.

any other course, so for all is a new and different experience in a practical assignment.

From informal comments and discussion with students it is clear that they enjoyed the project and the experience. When asked anonymously an open question in a questionnaire about the experience, most of the comments compiled were positive such as: ''I've liked competing with my classmates'', ''I really enjoyed the lab sessions'', ''It has been an interesting and entertaining experience'' ''It has been fun'', ''I've liked the fact that we are programming a video game''. Furthermore, in the lab we observed that students were engaged with the task and excited about the day where the final battle would take place as they joked among them and with the lecturer about who was going to win and how their bugs were going to ''destroy'' all the other ones. The day of the final battle, students spontaneously clapped when agents performed good battles or one of them performed a really good strategy.

Another observation was that students attended most laboratory sessions compared to other courses given by the same lecturer. They tried to improve constantly their solutions asking for advices and suggestions from the lecturer and thinking in creative solutions. In such a competitive framework, we detected some students who explored IA techniques beyond the scope of the course to use them against the rival agents as they were highly motivated to win. All students tested continuously the new agents developed. For this purpose, the framework makes it is very easy to set up challenges between two bugs, so students can test their previous version with the new one making them battle.

The experience promoted teamwork, as students were encouraged to work in pairs to discuss solutions together. But, one of the drawbacks of this competition context is that students did not share information among teams, as they wanted their agent to be the best. Information sharing and analyzing other solutions can help students to learn. However, this fact had also a positive effect, as no one cheated or copied other students' solutions.

One aspect to highlight is that most students finished their agent for the battle, so the number of handed in projects is high. This means that students have applied many theoretical concepts in the implementation of the agent, and helps them to understand the theory when sitting for the exam. The 30% of the grade may also have a huge influence in meeting the deadlines.

A feature students enjoyed, was the possibility of customizing the external appearance of the agents. Students could give a name and an icon to their agent, which helped increasing the user experience.

One year after the last experience, we joined a group of 20 students that participated in it. They answered anonymously a questionnaire with 12 questions regarding the experience. Eleven questions were scored on a 10-point Likert scale that went from strongly disagree (1) to strongly agree (10) and there was one open-question for students to comment the positive and negative aspects of the experience. Responses to these questions are summarized in Table 1.

Analyzing the results, we can observe that all answers (except question 5) have a percentage above 50% in the highest points of the Likert scale. It is important to highlight that 67% of the students felt that the competition-based project was a significant learning experience (question 1) and 61% thought that it help to consolidate their knowledge on AI agents (question 7). Therefore, we achieved not only a fun, entertaining and motivating experience (see questions 2, 3 and 4), but also a significant learning experience, ultimate objective of this project.

Although 50% rated with high points question 5, 50% of the students answered with 5, 6 or 7 points that they did not feel that the project increased their participation. This fact can be understood first because this project was not a voluntary assignment, and second, one student left a sidenote in the questionnaire commenting that he or she would have actively participated anyway, so other students might have thought that too. Even though, based on the 83% with answers above 6 of question 8, students got involved in the project.

Table 1. Questionnaire and results to the questions in %

Questions	<5	[5–7]	>7
1. I think the competition-based project was a significant learning experience	5	28	67
2. I think the competition-based project was entertaining	0	11	89
3. I think the competition-based project increased my motivation	5	17	78
4. I've liked the experience	0	6	94
5. I think the competition-based project increased my participation	0	50	50
6. I like the fact that the results of the competition influence the grades	11	28	61
7. I think this system has helped me to consolidate my knowledge on AI	17	22	61
8. I think it is important to work with a real application to apply the knowledge	0	33	67
9. I had a high involvement in the development of the project	0	17	83
10. The framework was easy to use	0	17	83
11. The possibility of configuring the bug has improved my user-experience	0	28	72

Students rated high the ease of use of the framework (question 10) and definitely enjoyed being able to configure the bug (question 11). In the open-question, several students agreed in including more customizable aspects.

In the open question, although some students included explicit positive comments on the competition and its relation with the grades such as "I like that if I've worked more than my classmates, this is reflected in my grades", some criticism was also received such as "The competition is fine, but if you lose, your agent looks very bad" or "Sometimes winning was a question of luck". To be as fair as possible, we included some mechanisms to avoid "luck" as commented in sub-section 3.3.

5. Conclusions and future work

In this work we described an experience of a gamified AI course for computer science engineers. The presented project fulfills McGovern and Fager six criteria for projects to succeed at the goal of creating a significant learning experience in an introductory AI course. First, the project is enjoyable as it is based on video games and competition which it has been proved to be suitable for AI and adds components of fun, engagement and motivation. Furthermore, students commented positively on the experience. The framework is flexible enough to work with other AI topics. The project is extensible for those talented students who want to explore more creative solutions when programming the agent and they can solve other characteristics of the problem. The inclusion of a competition makes the experience challenging. Students know that a trivial AI solution is not enough to win the competition and many students a highly motivated in winning the final battle. By using a video game, we present a realistic context, and students apply domain knowledge to a particular problem that can be found in real-life. And finally, by combining different game-elements into the learning process, it is feasible to maintain the student's interest.

We include diverse game-elements to the project. First of all, it is a video game which already includes points, limited resources and a challenge with clear goals. Furthermore, the competition environment adds leaderboards, scores and levels (each round of the competition increases the difficulty as agents are better). And finally, students can customize different aspects of the agent. The relation between the classification and the grades, also incorporates a gamification-style reward system.

Using this gamified context, a competition-based project, to improve students' practical skills implementing intelligent agents, we observed the following insights:

- Students attend to most laboratory lessons and they ask for suggestions from the lecturer to try improving their agents.
- Students try continuously to improve the algorithms. Once they have modified an agent, they confront their previous version with the new one, trying to get the best agent for the battle. By the final due date of the project, a number of students have gone beyond the methods presented in class.
- Enabling the possibility of customizing the external appearance of the agents (name and image) increases the user experience. And students demand more aspects to customize.
- No one copies. Students want their agent to be the best one, so no one lets their classmates copy the algorithms. However, the competition decreases the collaboration and information sharing among teams.
- Students are engaged with the task and they are excited about the day where the final battle will take place. Meanwhile, students joke among them about who is going to win.
- Most students finish their agent for the battle, so the number of presented works is high. This means that students have applied many theoretical concepts in the implementation of the agent, and helps them to understand the theory when sitting for the exam.

Future work lines will be to include more projects using the same framework to train other AI topics. Although the framework already has sound effects (when agents are destroyed, hyperspace is activated, walls are hit or bullets are shot) it could be extended to include additional features that appear in video games such as different scenario backgrounds and more customizations. This does not contribute to the educational objectives, but can increase the user experience and increase the positive behavior of the student towards the project. Finally, before the battles, students should describe their agents and explain how they implemented them in order to show other solutions and minimize the effect of not sharing information in the development phase. This will contribute to their knowledge and understanding of the AI concepts.

Acknowledgement—This work is supported by the University of Balearic Islands.

References

1. H. Tuan, C. Chin and S. Shieh, The development of a questionnaire to measure students' motivation towards science learning, *Int. J. Sci. Educ.*, **27**(6), 2005, pp. 639–654.
2. K. Seaborn and D. I. Fels, Gamification in theory and action: A survey, *Int. J. Hum.-Comput. St.*, http://dx.doi.org/10.1016/j.ijhcs.2014.09.006, 2014.
3. S. Deterding, D. Dixon, R. Khaled and L. Nacke. From game design elements to gamefulness: defining "gamifica-

tion", *Proc. MindTrek '11*. ACM, New York, NY, USA, 2011, pp. 9–15.
4. G. Zichermann and C. Cunningham, *Gamification by Design*, O'Reilly Media, Inc., 2011.
5. B. Reeves and J. L. Read, *Total Engagement: Using Games and Virtual Worlds to Change the Way People Work and Businesses Compete*, Harvard Business School Press, Boston, MA, 2009.
6. K. Seaborn and D. I. Fels, Gamification in theory and action: A survey. *Int. J. Hum.-Comput. St.*, http://dx.doi.org/10.1016/j.ijhcs.2014.09.006, 2014.
7. T. W. Malone, What makes things fun to learn? Heuristics for designing instructional computer games. *Proc. SIGSMALL'80*, ACM Press, New York, USA, 1980, pp. 162–169.
8. M. Minovic and M. Milovanovic, Gamification ecosystems, *Proc. TEEM 2014*, Salamanca, Spain, October, 2014, pp. 1–3
9. K. M. Kapp, *The Gamification of Learning and Instruction: Game-Based Methods and Strategies for Training and Education*, ISBN-10:1118096347, 2012.
10. E. Díaz San Millán and R. Gutiérrez Priego, *Learning by Playing: Is Gamification a Keyword in the New Education Paradigm?* F. García-Peñalvo, & A. Seoane Pardo (Eds.) *Online Tutor 2.0: Methodologies and Case Studies for Successful Learning*, Information Science, Hershey, PA, 2014, pp. 16–69.
11. A. Domínguez, J. Saenz-de-Navarrete, L. de-Marcos, L. Fernández-Sanz, C. Pagés and J. J. Martínez-Herráiz, Gamifying learning experiences: Practical implications and outcomes, *Comput. Educ.*, **63**, 2013, pp. 380–392.
12. B. J. Foss and T. I. Eikas, Game Play in Engineering Education: Concept and Experimental Results, *Int. J. Eng. Educ.*, **22**(5), 2006, pp. 1043–1052.
13. W. Brenner, R. Zarnekow and H. Wittig, *Intelligent Software Agents: Foundations and Applications*, Springer, 2012.
14. Z. Weidong, W. Haifeng and W. Anhua, Research-based teaching in artificial intelligence course, *Proc. IEEE ICCSE'09,* 2009, pp. 1756–1759.
15. A. Kumar, Three years of using robots in an artificial intelligence course: lessons learned, *J. Educ. Resour. Comput.*, **4**(3), 2004, Article 2.
16. J. Carpio Cañada, T. J. Mateo Sanguino, S. Alcocer, A. Borrego, A. Isidro, A. Palanco and J. M. Rodríguez, From classroom to mobile robots competition arena: An experience on artificial intelligence teaching, *Int. J. Eng. Educ.*, **27**(4), 2011, pp. 813–820.
17. G. T. McKee, The development of Internet-based laboratory environments for teaching robotics and artificial intelligence, *Proc. IEEE ICRA'02, 2002*, vol. 3, 2002, pp. 2695–2700.
18. J. DeNero and D. Klein, Teaching introductory artificial intelligence with Pac-man, *Proc. EAAI'11, 2011*, pp. 1885–1889.
19. A. McGovern and J. Fager, Creating significant learning experiences in introductory artificial intelligence, *Proc. SIGCSE 2007*, 2007, pp. 39–43.
20. A. McGovern, Z. Tidwell and D. Rushing, Teaching introductory artificial intelligence through java-based games, *Proc. AAAI Symposium EAAI, North America*, 2011.
21. L. B. Holder and D. J. Cook, A Client-Server Interactive Tool for Integrated Artificial Intelligence Curriculum, *Proc. FLAIRS Special Track on AI Education, 2001*, pp. 206–210.
22. A. McGovern and D. Trytten, Making in-class competitions desirable for marginalized groups, *Proc. FIE'13, 2013*, pp. 704–706.
23. J. Carpio Cañada, T. J. Mateo Sanguino, J. J. Merelo Guervós and V. M. Rivas Santos, Open classroom: enhancing student achievement on artificial intelligence through an international online competition, *J. Comput. Assist. Lear.*, **31**(1), 2014, pp. 14–31.
24. S. A. Wallace and J. Margolis, Exploring the use of competetive programming: observations from the classroom, *J. Comput. Sci. Coll.*, **23**(2), 2007, pp. 33–39.
25. M. Palomo-Duarte, J. M. Dodero, J. Tomás Tocino, A. García-Domínguez and A. Balderas, Competitive evaluation in a video game development course, *Proc. ITiCSE '12. ACM, New York, NY, USA, 2012*, pp. 321–326.

Ramon Mas-Sansó received his degree in Computer Science from the Autonomous University of Barcelona and his Ph. D. in Computer Science from the University of the Balearic Islands. He is currently a lecturer at the University of Balearic Islands. His research interests include human-computer interaction and computer graphics. He has been teaching AI for four years.

Cristina Manresa-Yee received her degree in Computer Science and her Ph. D. in Computer Science from the University of Balearic Islands. She is currently an Associate Professor at the University of Balearic Islands. Her research interests include human-computer interaction and usability engineering.