Article

A First Approach to Co-design with Pre-service Teachers a Multimodal Pedagogic Conversational Agent to Teach Programming in Primary Education

Diana Pérez-Marín * , Raquel Hijón-Neira , Celeste Pizarro

Posted Date: 27 November 2023

doi: 10.20944/preprints202311.1679.v1

Keywords: Basic education; Computer programming; Higher education; Scientific information; Online systems

Article

A First Approach to Co-Design with Pre-Service Teachers a Multimodal Pedagogic Conversational Agent to Teach Programming in Primary Education

Diana Pérez-Marín [1*], Raquel Hijón-Neira [2] and Celeste Pizarro [3]

[1] Computer Science Department, ETSII, Universidad Rey Juan Carlos; diana.perez@urjc.es
[2] Computer Science Department, ETSII, Universidad Rey Juan Carlos; raquel.hijon@urjc.es
[3] Applied Mathematics, Materials Science and Engineering and Electronic Technology, ESCET, Universidad Rey Juan Carlos, celeste.pizarro@urjc.es
* Correspondence: diana.perez@urjc.es

Abstract: Pedagogic Conversational Agents (PCAs) are interactive systems that engage the student in a dialogue to teach some domain. They can have the role of teacher, student, or companion, and adopt several shapes. In our previous work, a significant increase of students' performance when learning programming was found when using PCAs in the teacher role. However, it is not common to find PCAs used in the classrooms. In this paper, it is explored whether teachers would accept PCAs to teach programming better if co-designed with them. An experiment with 35 pre-service Primary Education teachers was carried out during the 2021/2022 academic year to co-design a robotic PCA to teach programming. The experience validates the idea that involving the teachers in the design of the PCA facilitates that they integrate this technology in their classrooms. 97% of the pre-service teachers answered in a survey that they believed that the robot PCA could help children to learn programming and 80% answered that they would like to use them in their classrooms.

Keywords: basic education; computer programming; higher education; scientific information; online systems

1. Introduction

Pedagogic Conversational Agents (PCAs) can be defined as "lifelike autonomous characters that cohabite the learning environment creating a rich interface face-to-face with student" [1]. Agents can ask students by voice, chat, gestures or with touch panel screens [2], and students can answer using any of these communication modes. Moreover, agents can be software and cohabitate an educational application, or they can be autonomous robots (hardware) to create the most natural possible communication experience with the student.

Agents, both software and hardware, can have multiple shapes, such as having a human body, being animals, characters, or robots. They can also have several roles in their interaction with the students [3]: teachers, who are the source of knowledge; students, who need to be taught by the students, called Teachable Agents; and, companions, who serve as peer students for emotional support, called Pedagogic Agent as Learning Companion (PAL).

Some benefits that have been published of using PCAs are the following: the Persona effect [4], according to which the presence of the agent has a positive influence in the students' perception of their learning experience; the Proteus effect [5], according to which students want to become like their agents and this is a source of motivation to interact more with them; and, the Protégé effect [6] (only for Teachable Agents), according to which students make greater efforts to teach their agents than to learn themselves.

In [7], a recent systematic review on the topic can be found. According to this review, and despite the multiple benefits of using PCAs for education that have been published, agents are still not able to communicate fluently on natural language. We believe that this problem could be overcome by taking advantage of non-verbal communication using gestures in the case of PCAs focusing on a certain domain instead of trying a general dialogue. Moreover, to take advantage of the Persona and Proteus Effect, Ocaña et al. [8] devised a companion agent, called Alcody.

Alcody's domain was knowing how to program with basic concepts such as input/output, conditionals, and loops in p-code with the agent. Learning how to program at early age seems to have multiple benefits such as the development of Computational Thinking [9-10] and other cognitive skills [11-12]. However, it is still a young research topic which needs more studies to find out which methodologies and technologies could be used given the lack of abstraction of the children and the complexity of the concepts to be taught [13-14].

Up to our knowledge, Alcody was the first PCA that was used to teach programming to children as a learning companion. Having a learning companion seems to help children to make the learning more tangible and adaptable to their needs as a significant increase of their learning was registered as well as high satisfaction levels [15]. However, the use of PCAs to teach programming in Primary Education has not been integrated in the classrooms. It is our hypothesis that the reason could be found in the lack of acceptance of such technology by the teachers as they could ignore what PCAs are, and they have not been asked about how PCAs should be. Therefore, upon the experience of Alcody and teaching programming to children, in this paper, we present an experience in which a co-design with pre-service Primary Education teachers to create with them a PCA to teach programming to children.

The paper is organized into four sections: Section 2 reviews the related work including PCAs and teaching programming in Primary Education; Section 3 describes the co-design method used with the pre-service Primary Education teachers; Section 4 proposes the PCA as result of the co-design procedure; finally, Section 5 ends the paper with the main conclusions and lines of future work.

2. Related work

2.1. Pedagogic Conversational Agents

Pedagogic Conversational Agents (PCA) can be defined as a software that can interact with the student in natural language (Johnson et al. 2000). It is very popular the simplest version of PCAs that are chatbots without artificial intelligence but able to make turns with a human user to achieve certain goal. Some samples are Alexa to listen to music or buy, the Google Assistant, Siri of Apple or the more innovative chatGPT with some AI for Education [16].

In the educational domain, PCAs have been investigated since 2000 [17] with body, including certain intelligence in the dialogue and social support [18]. Some samples are Doroty to teach Computer Science [19]; AutoTutor to teach Operating Systems [20]; or Betty to teach Natural Science [21]. The results achieved have been promising in these school domains. More samples can be found in [3].

However, despite the good results achieved in the literature, it is not common to find PCAs used in the classroom. It has been investigated the reason for this. It could be found in the lack of knowledge of this technology by the teachers or knowing the technology that teachers may dislike it because it does not follow the design principles, they would like that it had. A survey was taken by 82 teachers (52.4% men; 47.6% women) with 24 questions regarding the knowledge of PCAs by the teachers; how they would like PCAs in terms of friendliness, gestures with face/body, how to motivate and give advices to students, how to indicate to the student that has made a mistake and if the agent should remember students' previous choices; and, the most adequate frequency of use of the agent by the students [22].

The results of the survey revealed that although 78% of teachers used the computer as a support for their work by looking for information, no one knew of PCAs. It was necessary to show them several images of PCAs so that they could complete the survey as they did not know what a PCA was. Once they understood the educational technology, they highlighted that to use the PCA it should be friendly, able to give advice to the students, encourage the students to keep studying and indicate whether the students had made some mistake. The combination of these features would motivate teachers to use PCAs in their classrooms once they knew of the existence of this educational technology [22].

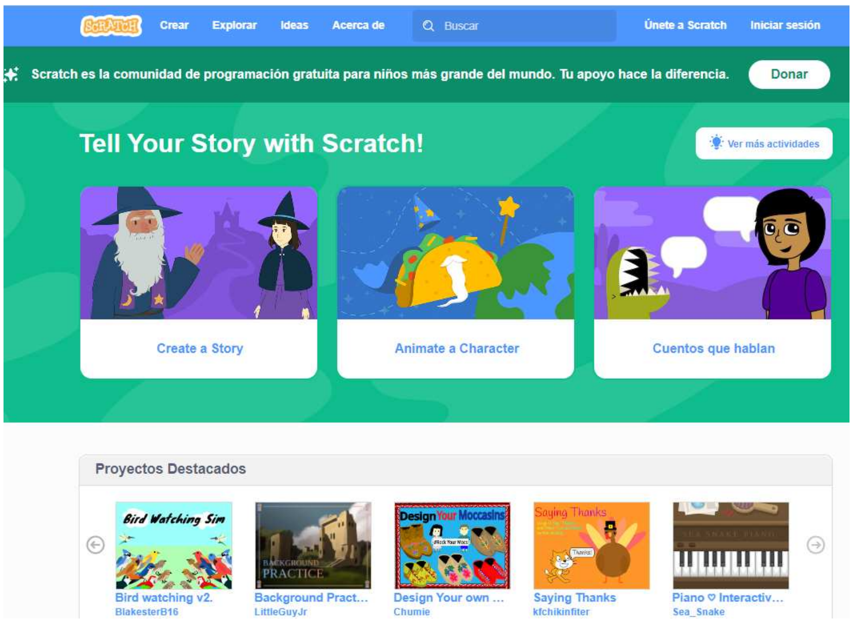
However, up to our knowledge, no survey has been carried out to find out if these features are the same that teachers would expect in the specific case of a PCA to teach programming in their Primary Education classrooms.

2.2. Teaching programming in Primary Education

The idea of teaching programming in Primary Education is not new. In fact, it dates back to the 80s with Papert's work with LOGO [23]. Papert believed that in order to teach programming to children they need to build an object to think with following a constructionism pedagogy [24]. In the LOGO environment, student interacted with a turtle, which responded to the students' commands as if they were talking. Children should also learn that there is no one correct solution, but they should test different possibilities until they work (i.e. the output of the program is as expected) [25].

Since then, there have been times in which the research of teaching programming was stopped given the lack of training of teachers and the complexity of the domain. However, in the last decades, the research has resurged from different approaches such as using multimedia software to program using blocks with the Scratch language with worldwide interest [26]; the use of robots [12] and unplugged exercises without using technology [27].

According to the survey carried out by [12], it was commonly agreed that Primary Education students should learn programming skills with a program such as Scratch. It combines the construction idea to learn by creating a program, with gamification as usually the program is a game which motivates them [28]. Figure 1 shows a sample image of Scratch. As can be seen the idea is to imagine the program, create it and share with other people. The program can have as goal to create a story, animate a character or a game among other possibilities. The programming language is simple without syntax. The instructions are block that must be moved together forming a puzzle to execute the program with a cat as can be seen in Figure 2 for the basic program to say Hello. One the "say hello" instruction is executed, the cat says "Hello".



Scratch es la comunidad de programación gratuita para niños más grande del mundo. Tu apoyo hace la diferencia.

Tell Your Story with Scratch!

Figure 1. A snapshot of the Scratch web version (see https://scratch.mit.edu/).

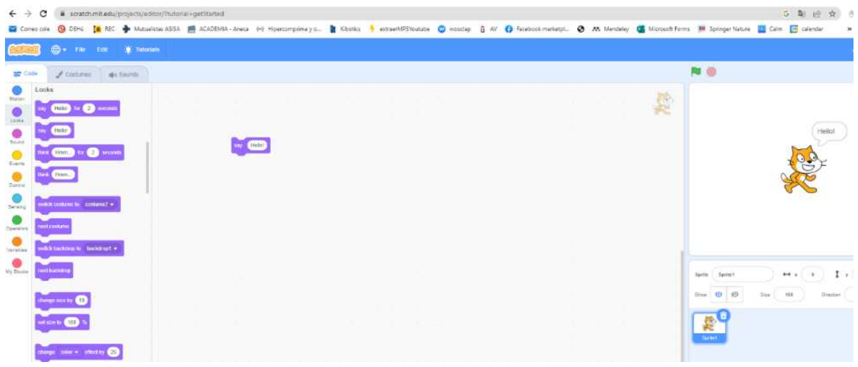


Figure 2. A basic program in Scratch to say "Hello".

Scratch has many tutorials to guide students and teachers. A debugging project was also carried out to help students when the program does not work [29]. However, it has limitations as students required to have some device to run Scratch and the execution of the program is always on the device. Therefore, other alternatives have been devised such as executing visual programming languages on robots such as LEGO Mindstorms [30] or unplugged approaches [27] with games such as moving the students or objects around the class according to the execution of the program or writing programs on paper. Recently, the use of Pedagogic Conversational Agents has also been explored to teach programming to Primary Education students with the companion role [15] and emotional support [31].

Some PCAs that have been reviewed in the literature to teach programming are Jeppy [32] and Alcody [8]. Jeppy is an emotional PCA in the shape of a dog with gestures to learn how to correct syntax mistakes. Jeppy gives advice and affective messages to guide students to solve the exercises following a constructivist approach. Figure 3 shows several Jeppy's gestures.

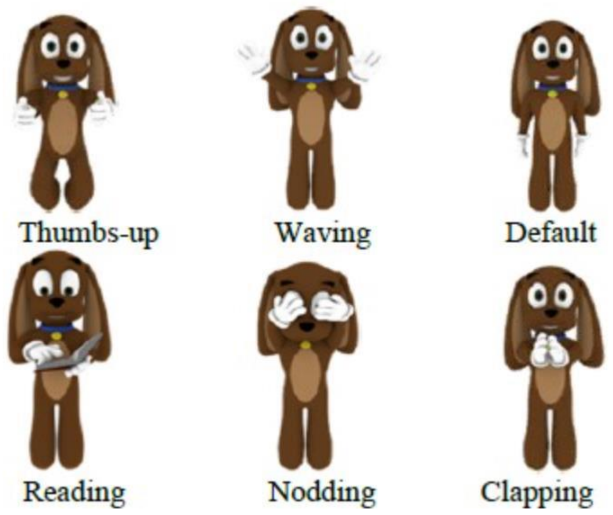
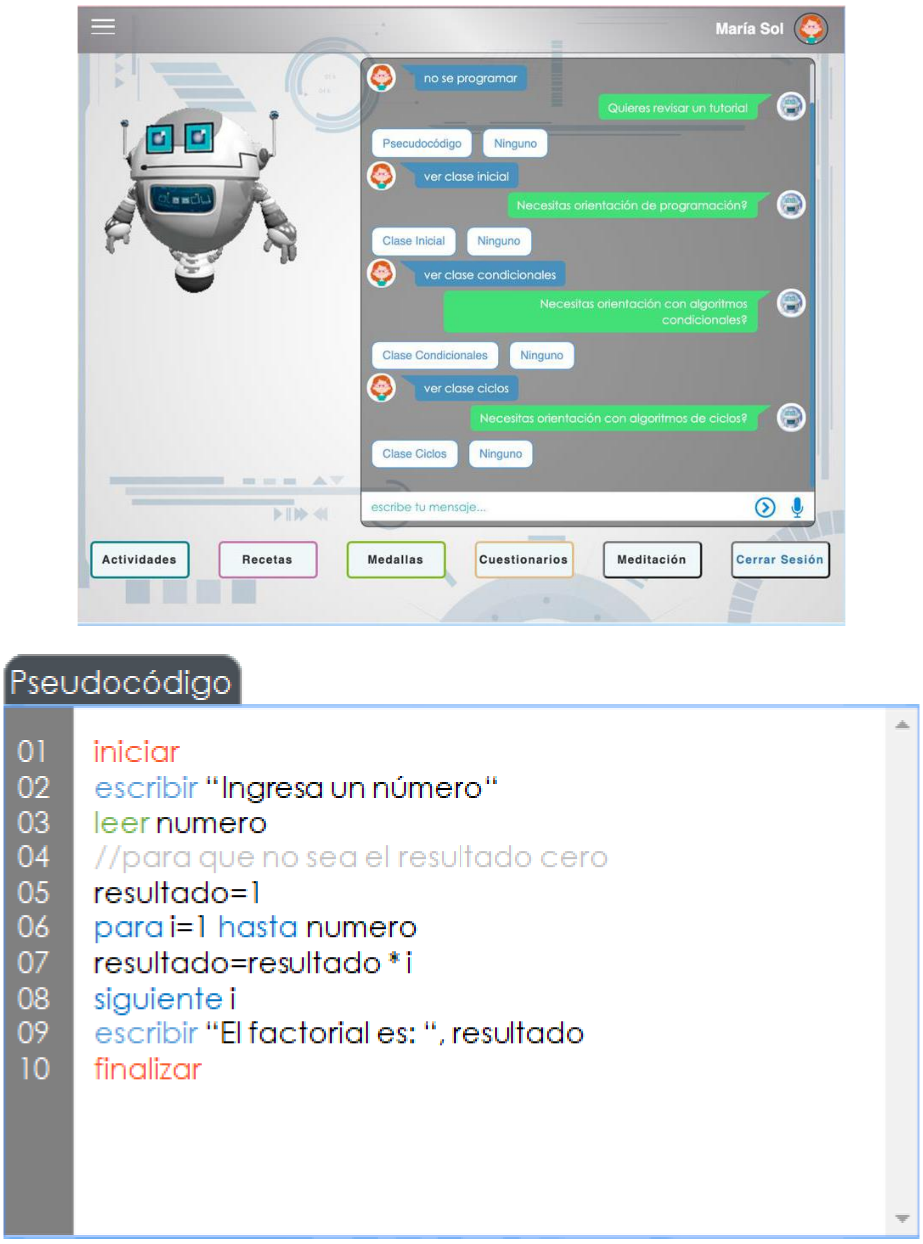


Figure 3. Some gestures of the PCA Jeppy to teach programming.

In an experiment carried out with novice programming students, it was found out that 66.7% of the students used Jeppy's advice to find the solution and 41.5% found the messages given by Jeppy clear and useful.

Alcody is an emotional learning companion to teach p-code to children following both a constructionism approach [23] and gamification [33]. The idea is to build a program by talking to the student and providing emotional recommendations to reach the optimum mental state to program. Moreover, Alcody has some games to motivate students to keep programming and win medals. Figure 4 shows an image of the Alcody agent.



no se programar

Quieres revisar un tutorial

Psecudocódigo | Ninguno

ver clase inicial

Necesitas orientación de programación?

Clase Inicial | Ninguno

ver clase condicionales

Necesitas orientación con algoritmos condicionales?

Clase Condicionales | Ninguno

ver clase ciclos

Necesitas orientación con algoritmos de ciclos?

Clase Ciclos | Ninguno

escribe tu mensaje...

```
iniciar
escribir "Ingresa un número"
leer numero
//para que no sea el resultado cero
resultado=1
para i=1 hasta numero
resultado=resultado * i
siguiente i
escribir "El factorial es: ", resultado
finalizar
```

Figure 4. A snapshot of the Alcody emotional learning companion to teach p-code.

As can be seen in Figure 4, Alcody has several tutorials to teach students how to program input/output, conditional and loops concepts. Moreover, it has activities in which students have to solve programming exercises such as to calculate the factorial of a number given; recipes to play cooking as programming; and medals as students are able to complete activities with Alcody [8, 31].

3. Materials and Methods

3.1. Goal

Given that in the literature, only general principles to design PCAs have been found and few PCAs to teach programming have been created, the goal here is to find out how teachers would like PCAs to teach programming to integrate them in their classrooms.

Moreover, the goal is to co-design with them a first prototype of PCA to teach programming according to the teachers' needs and preferences.

3.2. Materials and Procedure

To achieve the goal proposed, two questionnaires were created to be filled in by a sample of teachers. 44 teachers were recruited from the Primary Education degree of a Spanish university. They were future teachers and were chosen because they had a subject in which they had to learn how to program, and how to teach programming to children. Thus, they had knowledge about programming and programs such as Scratch. However, they did not know of the existence of PCAs to teach any subject. Therefore, initially they were taught what a PCA is and their features. We have the ethical approval of the University Ethical Committee (ID 19012022202722).

The initial questionnaire can be found on-line at this link. It starts with some general demographic questions such as age, gender, occupation, level of experience with technology and which technologies they use, the frequency and purpose of use. To focus on which applications and apps they know to learn programming, and their knowledge of PCAs. Given that it was expected they did not have any knowledge of PCAs to teach programming, students were shown a video of a sample PCA to be able to answer the questions. Students filled in the questionnaire anonymously and voluntarily in February.

After that they took their lessons on how to teach programming to children for two months. Meanwhile, the prototype was being implemented as requested in the initial questionnaire. In April, students were shown the second final prototype, and they were asked to fill in a final questionnaire to find out their opinion of the resulting PCAs, and whether they would like to use it to teach programming. Moreover, they were asked to evaluate that final prototype with questions regarding each screen and the navigation between screens.

4. Results

4.1. Initial Questionnaire

The analysis of the data gathered in the initial questionnaire reveals that all 44 students (pre-service teachers) that participated were between 18-25 years old (70% female, 30% male). 43.2% of the students indicated that their digital competence is between medium-high as they are used to work with smartphones, computers, and tablets (98% used their smartphones daily, 70% used their computers daily, 63.6% used their tablets daily).

70% claimed having used applications to learn programming before the course. The reason could be found in the teaching of Scratch that was common in Secondary Education in Spain before entering the University. However, 91% of the students indicated that they have never used an educational PCA before. This is why it was necessary to explain the concept of educational PCA before asking about how they would like to design such educational technology to teach programming.

Figure 5 shows the answers to the question about what the appropriate support would be to run the app of the PCA in the classroom. As can be seen, there is not a clear consensus among students, although it could be chosen tablets as the most voted option.

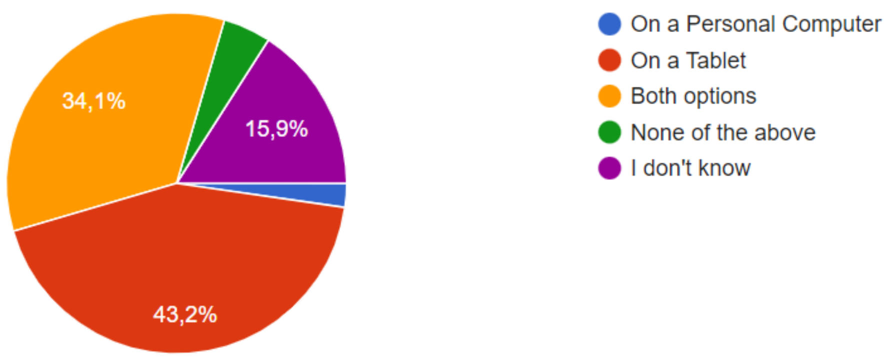


Figure 5. Technology to run the educational robot to teach programming**.**

75% of the students asked for a how-to guide to learn how to use such PCA. 63.6% of the students requested to be able to manage several classes and 59.1% indicated that Primary Education students usually work in teams from two to four. 70.5% of the students claimed that it would be of great help to have some student' tracking progress including at least some information about the number of correct and wrong answers, and the time needed per question.

Figure 6 shows the elements that are considered helpful when creating activities for the PCA to teach programming.

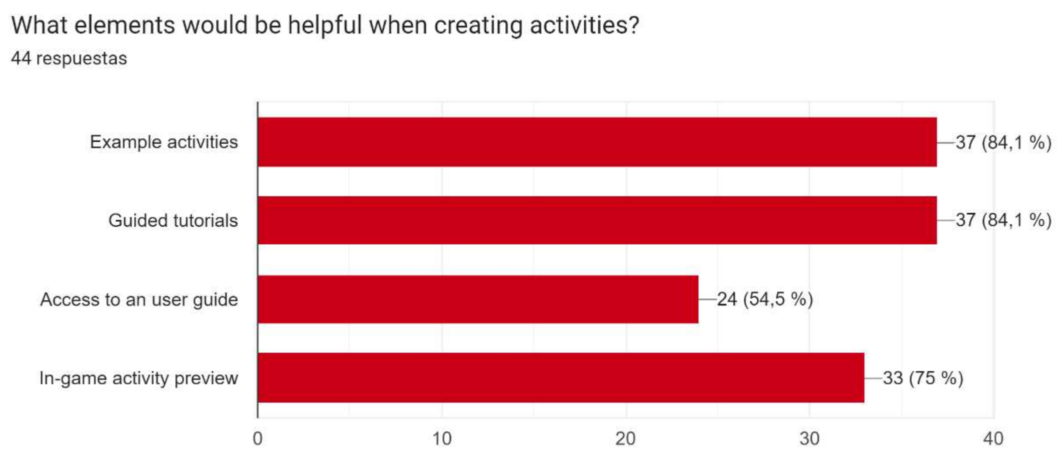


Figure 6. Useful elements to create activities in the PCA.

Figure 7 shows the elements that should be customizable.

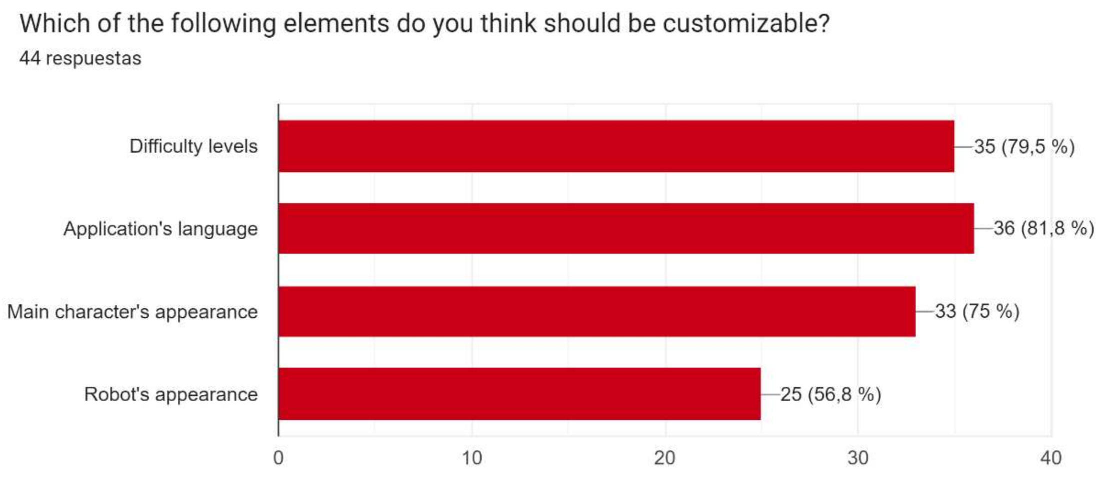


Figure 7. Elements to be customizable.

Regarding the language, most of the participants answered that the PCA should use simple short direct sentences in the students' mother language with an informal, kind, and friendly tone. The vocabulary should be easy for the students to understand. 65.9% of the participants also asked for the use of metaphors to teach programming as can be seen in Figure 8, and 63.6% of the participants considered that images should be presented with descriptive text.

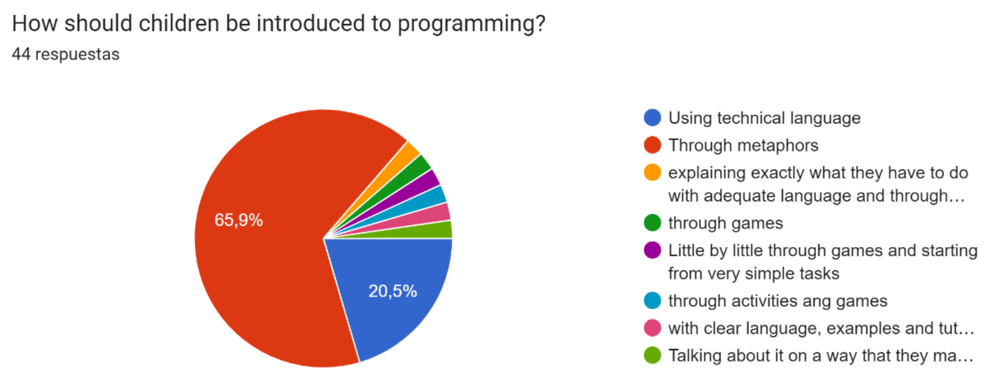


Figure 8. How to introduce children to programming.

If a child makes a mistake, 45.5% of the participants considered that children should be redirected towards the correct solution through clues as shown in Figure 9.

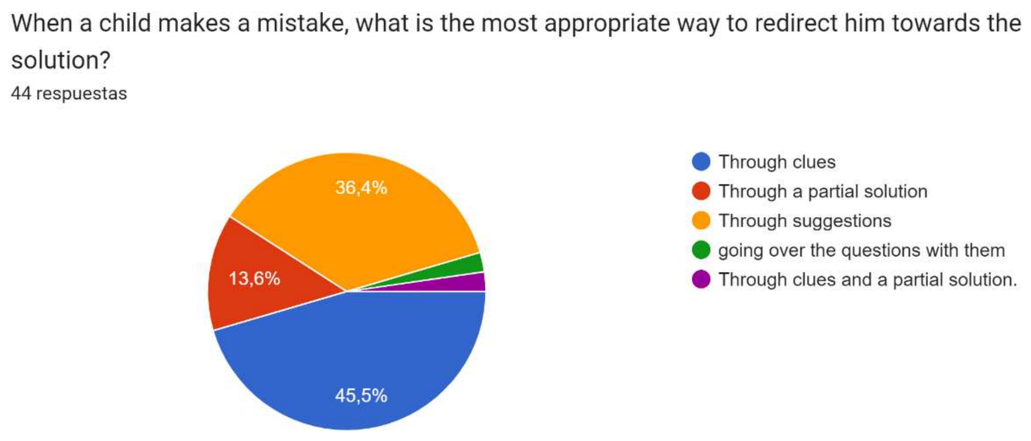


Figure 9. How to guide children towards the solution.

4.2. Prototype

The scenario indicated to the participants was a Primary Education classroom with 25 children paired in 2-4 groups. Each row of the classrooms had one PCA as shown in Figure 10. The PCA can work in the teacher mode to create new activities or in the student mode to show the activities so that students can start learning how to program. The teacher can have either a tablet or a digital whiteboard to create the activities and show the progress of the students. The advantage of a digital whiteboard is that teachers could show how to solve an activity to all the students in case that they all have doubts and did not know how to continue interacting with the PCA.

The use of the tablet could also be beneficial in case that the teacher would like to track the progress of a certain group without the rest of the class knowing it given that the teacher is the only one looking at the tablet.

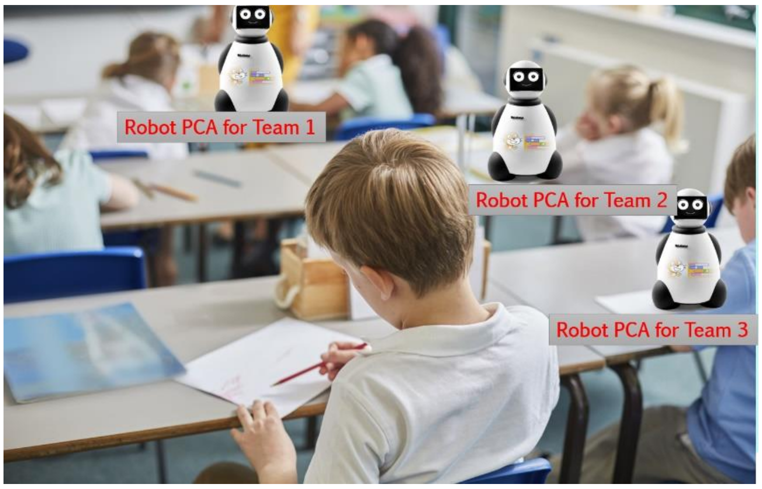


Figure 10. Scenario to use the robot PCA in the classroom.

Figure 11 shows the how-to guide to give to the teacher. It helps the teacher to follow the steps to set up the PCAs. For both modes (create activities or do activities, see Figure 12), the first step is to turn the digital whiteboard on to show the digital PCA. Next, if the teacher wants to create activities, the creation mode should be chosen to access the authoring tool. Or, if the goal is to do activities, one student per row is responsible to turn their robot PCA too. The robot PCA has a Bluetooth connection to the digital PCA in the whiteboard.

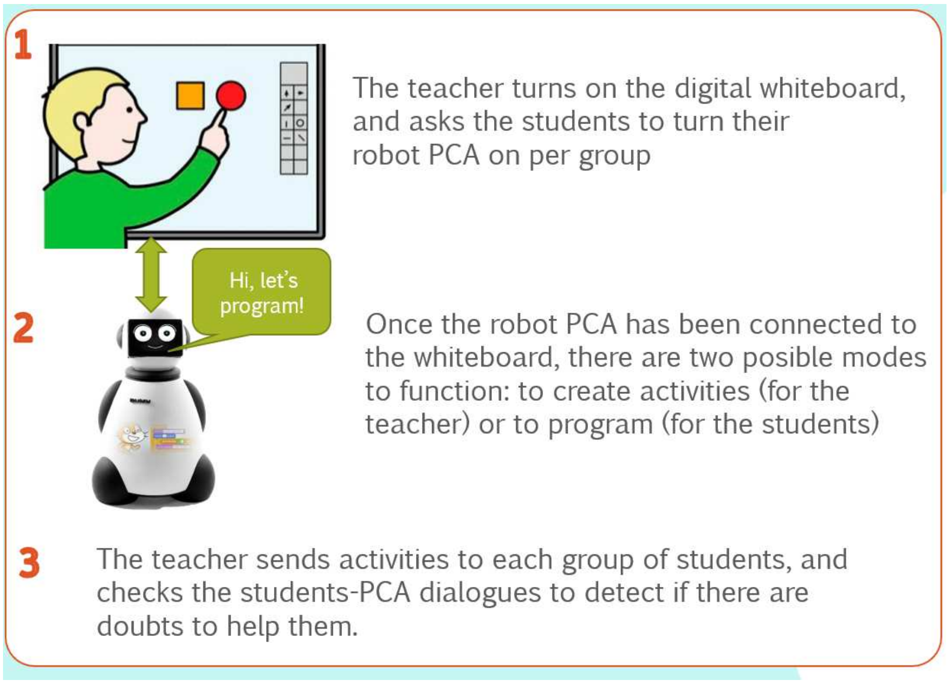

1
The teacher turns on the digital whiteboard, and asks the students to turn their robot PCA on per group
2
Hi, let's program!
Once the robot PCA has been connected to the whiteboard, there are two posible modes to function: to create activities (for the teacher) or to program (for the students)
3
The teacher sends activities to each group of students, and checks the students-PCA dialogues to detect if there are doubts to help them.

Figure 11. How-to guide to the teacher.

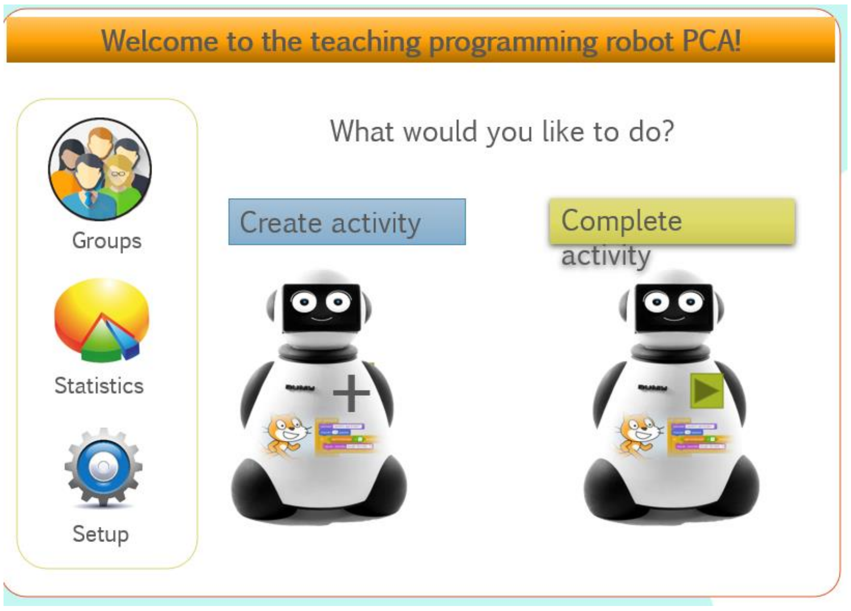
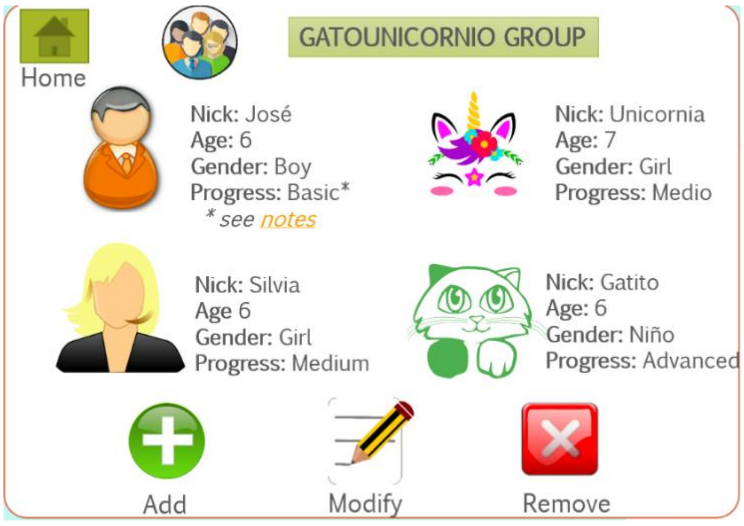

Welcome to the teaching programming robot PCA!
What would you like to do?
Groups
Statistics
Setup
Create activity
Complete activity

Figure 12. Main menu.

The first step is to create the students' groups. Teachers should press the "Groups" button and as can be seen in Figure 13, they can add, modify, and remove students per groups and the info per student. Secondly, once the groups have been created, teachers can create the activities as shown in Figure 14.



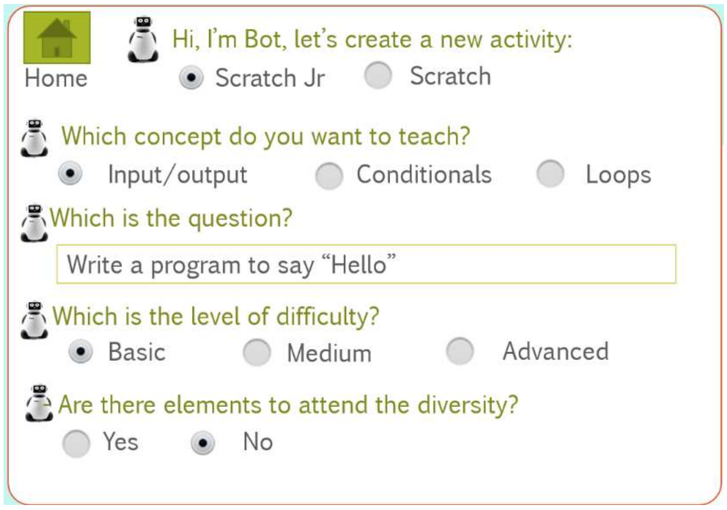
Nick: José
Age: 6
Gender: Boy
Progress: Basic*
* see notes

Nick: Unicornia
Age: 7
Gender: Girl
Progress: Medio

Nick: Silvia
Age 6
Gender: Girl
Progress: Medium

Nick: Gatito
Age: 6
Gender: Niño
Progress: Advanced

Figure 13. Menu to manage students' groups.


Home

Hi, I'm Bot, let's create a new activity:

[✓ Scratch Jr] Scratch

Which concept do you want to teach?

[✓ Input/output] Conditionals Loops

Which is the question?

Write a program to say "Hello"

Which is the level of difficulty?

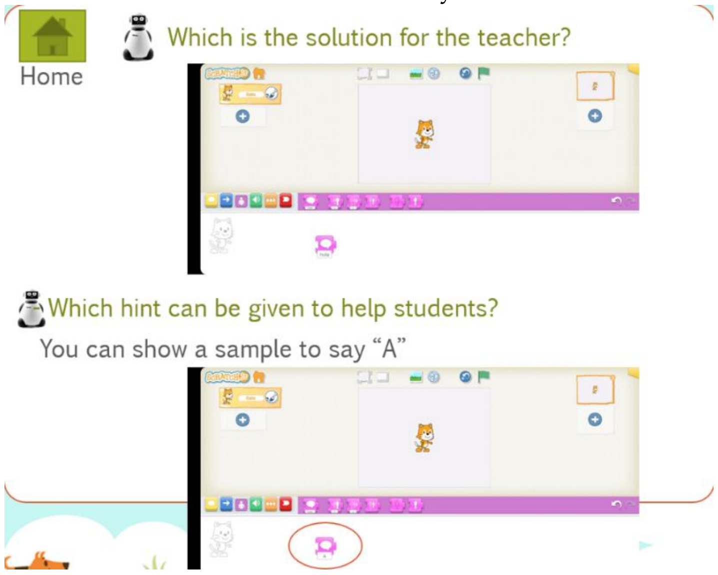
[✓ Basic] Medium Advanced

Are there elements to attend the diversity?

Yes [✓ No]

Figure 14. Creation of the activities for the PCA.

For each activity, teachers should choose the programming language which could be Scratch Jr or Scratch depending on the age of the students. They should also indicate the concept to teach: input/output, conditionals or loops; the statement of the question and some possible solution as shown in Figure 15, the level of the difficulty and if there is any element to attend the diversity. Once the teacher filled in the activity, a team can interact with the bot as shown in Figure 16. In this case, the team did not know how to program the activity, and thus the PCA provides them with a sample. With the sample, the team is now able to solve the activity.

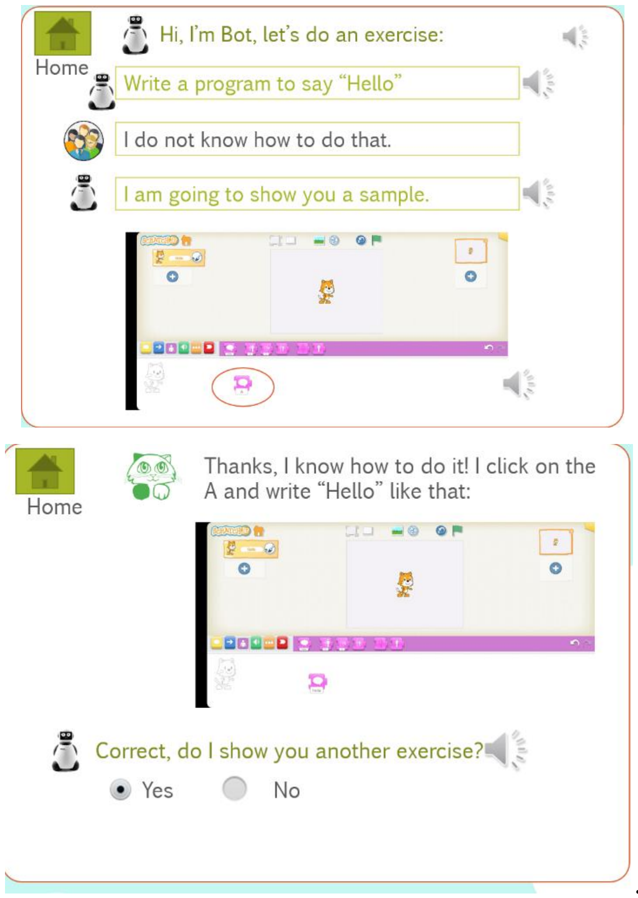

Home

Which is the solution for the teacher?

Which hint can be given to help students?

You can show a sample to say "A"

Figure 15. Possible solution for an activity.


Home

Hi, I'm Bot, let's do an exercise:

Write a program to say "Hello"

I do not know how to do that.

I am going to show you a sample.

Home

Thanks, I know how to do it! I click on the A and write "Hello" like that:

Correct, do I show you another exercise?

[✓ Yes] No

Figure 16. Sample interaction between a team and the PCA.

Teachers can also see how many exercises students have passed in global (all the class) or per team. They can also set up the difficulty level of the activities, the language of the PCA, and the shape both digital and physical PCA as a teddy bear or robot.

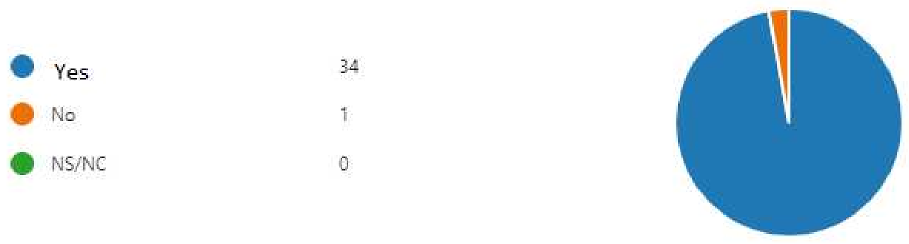
4.3. Validation Questionnaire

35 participants completed the questionnaire. 97% answered that they believed that the robot PCA could help children to learn programming as shown in Figure 17 and 80% of the participants answered that they would like to use them in their classrooms as shown in Figure 18.

Some more data gathered are the following:

- 89% appreciated the How-to guide provided for the teacher shown in Figure 11.
- 71% considered that the sample interaction between the PCA and the team shown in Figure 16 is as they expected it should be. They also considered that the navigation between screens is adequate.

Table 1 shows the evaluation of the screens in a scale from 1 (dislike) up 5 (like).



doi:10.20944/preprints202311.1679.v1

Figure 17. Answers to the question if a PCA robot as the one shown can be used to teach programming (NS/NC are the students who have not answered).

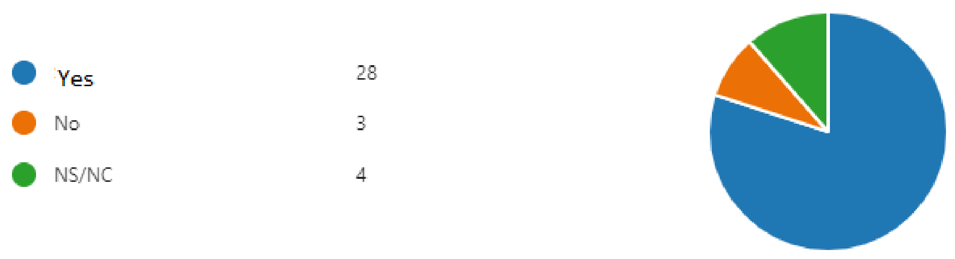


Figure 18. Answers to the question if the future teachers will have enough robots, if they would use them in their lessons (NS/NC are the students who have not answered).

Table 1. Evaluation of the prototype screens in the Likert scale (1-dislike up to 5-like).

Screen	Figure(s)	Value
Main menu	12	3.4
Groups	13	3.7
Activity creation	14	3.6
Students' activities	16	4.2
Attention to the diversity	12-16	4.2
Use of language	12-16	3.7
Intuitiveness	12-16	3.8
Friendliness	12-16	3.7
Easiness of use	12-16	3.6

Finally, when the future teachers were given the opportunity of writing any comment they would like to freely share, some of their answers were the following:

- "I think that this approach to teach programming is appropriate and can help children integrate technology into classes and make them more enjoyable and didactic".
- "I think it's a great way to teach children, as it's fun and entertaining at the same time."
- "I find it very useful to make lessons more interactive and motivate children."

5. Discussion

The use of PCAs could be helpful to teach in general, and in the case of teaching programming to Primary Education students the co-design with pre-service teachers seems to be the motivator for the future teachers to integrate such technology into their classrooms. 97% of the survey respondents indicated that PCAs could help children to learn programming and 80% answered that they would like to use them in their classrooms. Some recommendations for any researcher or teachers that would like to integrate the use of robot PCAs to teach programming in their classrooms are the following:

1. It is necessary a detailed step-by-step how-to guide to help teachers in case they have some doubts about the PCA and the activities to teach programming.
2. The associated application to the PCA should allow teachers to manage several teams (each team should have 2-4 students). The profile per student should at least include his/her name, age, class, and progress. The progress should register at least the number of correct and incorrect activities solved and the time needed to complete them.
3. When creating an activity for the PCA to teach programming it should include samples, tutorials, and a preview of how to solve the activity.
4. There should be a set up menu to modify the difficulty level, language, and shape of the digital (the one in the tablet or digital whiteboard) and physical PCA (the one used per students).

5. The PCA language should be adapted to the age of the students with short, simple, fun, nice, friendly, motivating, and respectful sentences so that they can understand it and are interested in interacting with it.
6. It is also advisable to use metaphors to teach programming concepts given that it could facilitate the children to understand the concepts.
7. If a team is unable to solve a programming activity, the PCA could provide hints, suggestions, and partial solutions to help them towards the successful completion of the activity.
8. Multimodality can help children to understand the tasks. Thus, images should always be accompanied by text to provide the same information in different channels. Audio could be helpful if children can have earphones to avoid disturbing the rest of teams working in the same classroom at the same time.

Author Contributions: Conceptualization, Diana Pérez-Marín and Raquel Hijón-Neira; methodology, Celeste Pizaro; validation, all authors; investigation, Diana Pérez-Marín; resources, Raquel Hijón-Neira; data curation, Celeste Pizarro; writing—original draft preparation, Diana Pérez-Marín; writing—review and editing, Raquel Hijón-Neira and Celeste Pizarro; funding acquisition, all authors. All authors have read and agreed to the published version of the manuscript. Please turn to the CRediT taxonomy for the term explanation.

Funding: This research was funded by the project with reference number M3035 of Universidad Rey Juan Carlos and the project "Novel interactive approaches based on collaborative and visualization systems aimed at learning block-based programming" of the Spanish Ministry.

Data Availability Statement: Data is unavailable due to ethical restrictions.

Conflicts of Interest: The authors declare no conflict of interest. The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript; or in the decision to publish the results.

References

1. Johnson, W., Rickel, J., y Lester, J. (2000). Animated Pedagogical Agents: Face-to-Face Interaction in Interactive Learning Environments. Journal of Artificial Intelligence in Education, 11, 47-78.
2. Graesser, A. & McNamara, D. (2010). Self-regulated learning in learning environments with pedagogical agents that interact in natural language, Educational Psychologist 45, 234-244.
3. Pérez-Marín, D. (2021). A Review of the Practical Applications of Pedagogic Conversational Agents to Be Used in School and University Classrooms. Digital, 1(1), 18-33.
4. Lester, J., Converse, S., Kahler, S., Barlow, S., Stone, B., & Bhogal, R. (1997). The persona effect: affective impact of animated pedagogical agents, Proceedings of the SIGCHI conference on Human factors in computing systems, ACM.
5. Yee, N., & Bailenson, J. (2007). The Proteus effect: The effect of transformed self-representation on behavior, Human Communication Research 33, 3, 271-290.
6. Chase, C., Chin, D., Oppezzo, M., & Schwartz, D. (2009). Teachable agents and the Protégé effect: increasing the effort towards learning, Journal of Science Education and Technology 18, 334-352.
7. Sikström, P., Valentini, C., Kärkkäinen, T., & Sivunen, A. (2022). How pedagogical agents communicate with students: A two-phase systematic review. Computers & Education, 104564.
8. Ocaña, J.M., Morales-Urrutia, E.K., Pérez-Marín, D., Pizarro, C. (2020). Can a Learning Companion Be Used to Continue Teaching Programming to Children Even During the COVID-19 Pandemic?, IEEE Access, vol. 8, pp. 157840–157861, doi: 10.1109/ACCESS.2020.3020007
9. Wing, J.M. (2006). Computational thinking, Communications of ACM, 49(3), 33–35. https://doi.org/10.1145/1118178.1118215
10. Pérez-Marín, D., Hijón-Neira, R., Bacelo, A., Pizarro, C. (2020). Can computational thinking be improved by using a methodology based on metaphors and Scratch to teach computer programming to children?, Computers in Human Behaviour, 105, https://doi.org/10.1016/j.chb.2018.12.027
11. Arfé, B., Vardanega, T., Ronconi, L. (2020). The effects of coding on children's planning and inhibition skills, Computers & Education, vol. 148, https://doi.org/10.1016/j.compedu.2020.103807
12. AlQarzaie, K. N., & AlEnezi, S. A. (2022). Using LEGO MINDSTORMS in Primary Schools: Perspective of Educational Sector. International Journal of Online & Biomedical Engineering, 18(1).
13. Bati, K. (2022). A systematic literature review regarding computational thinking and programming in early childhood education. Education and Information Technologies, 27(2), 2059-2082.

14. Bucciarelli, M., Mackiewicz, R., Khemlani, S. S., & Johnson-Laird, P. N. (2022). The causes of difficulty in children’s creation of informal programs. International Journal of Child-Computer Interaction, 31, 100443.
15. Ocaña, M. (2021). MEDIE_GEDILEC: propuesta de metodología para la creación de compañeros de aprendizaje para la enseñanza de la programación en Educación Primaria. Tesis Doctoral presentada en la Universidad Rey Juan Carlos.
16. García-Peñalvo, F. J. (2023). La percepción de la Inteligencia Artificial en contextos educativos tras el lanzamiento de ChatGPT: disrupción o pánico. Education in the Knowledge Society (EKS), 24, e31279. https://doi.org/10.14201/eks.31279
17. Cassell, J. (2000). Embodied conversational interface agents. Communications of the ACM, 43(4), 70-78.
18. Kumar, R., & Rosé, C. (2011). Architecture for building conversational agents that support collaborative learning. IEEE Transactions on Learning Technologies, 21-34.
19. Leonhardt, M., Dutra, R., Granville, L., & Tarouco, L. (2005). DOROTY: una extensión en la arquitectura de un ChatterBot para la formación académica y profesional en el campo de la gestión de redes. Conferencia Mundial IFIP sobre Computadoras en la Educación.
20. Graesser, A. C., Greenberg, D., Frijters, J. C., & Talwar, A. (2021). Using AutoTutor to track performance and engagement in a reading comprehension intervention for adult literacy students. Revista Signos. Estudios de Lingüística, 54(107).
21. Han, J. H., Shubeck, K., Shi, G. H., Hu, X. E., Yang, L., Wang, L. J., ... & Biswas, G. (2021). Teachable Agent Improves Affect Regulation. Educational Technology & Society, 24(3), 194-209.
22. Tamayo, S., & Pérez-Marin, D. (2017). ¿Qué esperan los maestros de los Agentes Conversacionales Pedagógicos?. Education in the Knowledge Society (EKS), 18(3), 59–85. https://doi.org/10.14201/eks20171835985
23. Papert, S. (1980). Mindstorms: Children, computers, and powerful ideas. New York, NY: Basic Books.
24. Papert,S., & Harel, I. (1991). Constructionism, Ablex Publishing, Norwood, NJ.
25. Papert, S. (1999). Logo philosophy and implementation. Logo Computer Systems, Highgate Springs.
26. Resnick, M., Maloney, J., Monroy-Hernandez, A., Rusk, N., Eastmond, E., & Brennan, K. (2009). Scratch: Programming for all, Communications of ACM, 12, 52(11), 60-67.
27. Brackmann, C., Barone, D., Casali, A., Boucinha, R., & Muñoz-Hernandez, S. (2016). Computational thinking: Panorama of the Americas. Presented at Computers in Education (SIIE), pp. 1-6.
28. Papavlasopoulou, S., Giannakos, M.N., & Jaccheri, L. (2019). Exploring children's learning experience in constructionism-based coding activities through design-based research, Computers in Human Behaviour, DOI: 10.1016/j.chb.2019.01.008, 2019.
29. Moreno-León, J., & Robles, G. (2015). Dr. Scratch: A web tool to automatically evaluate Scratch projects. In Proceedings of the workshop in primary and secondary computing education (pp. 132-133).
30. Zygouris, N., Striftou, A., Dadaliaris, A., Stamoulis, G., Xenakis, A., & Vavougios, D. (2017). The use of LEGO Mindstorms in elementary schools", EDUCON, 514-516, doi:10.1109/EDUCON.2017.7942895.
31. Morales-Urrutia, E. (2021). MEDIE_LECOE: propuesta de metodología para la integración de emociones en compañeros de aprendizaje para la enseñanza de la programación en Educación Primaria. Tesis Doctoral presentada en la Universidad Rey Juan Carlos.
32. Pérez, J. E., Dinawanao, D. D., & Tabanao, E. S. (2020). JEPPY: An interactive pedagogical agent to aid novice programmers in correcting syntax errors. International Journal of Advanced Computer Science and Applications, 11(2), 48-53.
33. Deterding, S., Dixon, D., Khaled, R., & Nacke, L. (2011). From game design elements to gamefulness: Defining “gamification”. Memorias del 15th International Academic MindTrek Conference: Envisioning Future Media Environments, (pp. 9–15).