information

MDPI

Systematic Review

A Systematic Literature Review on Adaptive Supports in Serious Games for Programming

Pavlos Toukiloglou and Stelios Xinogalos *

Department of Applied Informatics, University of Macedonia, GR-54636 Thessaloniki, Greece; toukiloglou@uom.edu.gr
* Correspondence: stelios@uom.edu.gr

Abstract: This paper reviews the research on adaptive serious games for programming regarding the implementation of their support systems. Serious games are designed to educate players in an entertaining and engaging manner. A key element in terms of meeting their educational goals is the presentation of the learning content through a support system. Recent developments in artificial intelligence, data analysis, and computing made the development of support systems that adapt to player individual characteristics possible. A systematic literature review is necessary to evaluate the efficiency of adaptive supports and examine the implementation approaches. This review identified 18 papers reporting evidence about the efficiency of the provided support and methods of development. A variety of techniques for presenting educational content was found, with text being the preferred type. Researchers employed data-driven approaches to model student knowledge levels and behavior such as Bayesian networks and questionnaires, with fuzzy logic being utilized most frequently. The efficiency of the supports, when compared with non-adaptive or traditional methods of teaching, was mostly positive, although this is not a decisive conclusion. Some papers did not provide empirical evidence or concluded no difference in efficiency. The limited number of articles in the field, together with the lack of a standard evaluation methodology, leads to the conclusion that further work needs to be carried out in the area.

Keywords: serious games; adaptive support; systematic literature review; artificial intelligence; programming

Citation: Toukiloglou, P.; Xinogalos, S. A Systematic Literature Review on Adaptive Supports in Serious Games for Programming. *Information* **2023**, *14*, 277. https://doi.org/10.3390/info14050277

Academic Editors: Mohd Nor Akmal Khalid, Chu-Hsuan Hsueh, Ting-Han Wei and Hiroyuki Iida

Received: 30 March 2023
Revised: 29 April 2023
Accepted: 2 May 2023
Published: 8 May 2023

1. Introduction

A serious game (SG) is defined as a software system that combines game elements with a non-entertaining purpose [1]. SGs are designed to train players on a subject and they have a wide range of applications in politics, the military, healthcare, and art [2]. Nonetheless, one of the most important domains of application is the education field [3], as SGs can assist teachers and students by providing the learning material in an engaging manner. Learning programming is a multi-layer ability that requires strenuous effort from students to develop problem-solving skills and abstract thinking [4]. Therefore, many SGs have been developed to assist educators and students in the learning process of programming.

SGs use several methodologies and frameworks in their design to facilitate their entertaining and educational role. Elements such as attractive graphics, engaging narrative, and interactivity increase student motivation and active participation [5]. The pedagogical objectives are fulfilled with mechanisms that provide educational content to students. Feedback and support systems are integrated into the game environment, presenting new knowledge and promoting educational goals.

In the context of an SG, support can be defined as any type of supplementary information, guidance, or assistance that is provided to aid players in acquiring and achieving the learning objectives of the game. The design and utilization of various support types can vary depending on the game's design and intended use case [6]. Text support is a common form of support that provides written instructions, explanations, or feedback messages. It can appear during specific game events or between levels. Textbook support is a variant of text support that consists of relevant passages from textbooks or other educational materials that provide additional background information on the learning content. Hint support, another form of text support, offers brief notifications that provide clues or prompts to guide the player toward the correct answer or solution [7]. Image support presents visual aids or diagrams that illustrate the learning content, making it easier for players to understand. Video support provides short videos containing additional explanations or demonstrations of the learning content. Finally, working examples of support demonstrate how the learning content can be applied in similar situations to those presented in-game [8].

We argue that support is the most important factor for the success of an SG, as it can greatly impact the player's learning experience. It is critical since it offers learners the necessary resources and assistance to facilitate their learning process. Without it, learners may become disengaged, resulting in a lack of interest and motivation to learn. The availability of support builds confidence, reduces stress and anxiety, and encourages learners to experiment and learn from their mistakes. Given that learning is a process that takes time and effort, support encourages learners to persist despite setbacks, mistakes, or challenges. Furthermore, its various forms can accommodate the diverse learning needs of players, resulting in more effective knowledge acquisition.

Nonetheless, every student has a unique learning style and strategy while studying or encountering a new problem [9]. Educators in classrooms acknowledge these different learning preferences and adjust their teaching methods to handle the individual needs of their students. Following this model, an SG has the potential to significantly enhance learning efficiency by providing support customized to the unique requirements of each student. Through personalized support, an SG can help students overcome their specific learning obstacles. Emulating the interaction between a teacher and student in a classroom, individualized support in an SG can help bridge the gap between the varying learning styles and preferences. This will ultimately lead to a more productive and satisfactory learning experience.

Artificial intelligence (AI) applications in game-based learning environments have been studied in the past, but recent improvements in AI and data analysis have led to a significant research interest in adaptive learning [10]. Adaptive SGs can provide tailored content and experiences to their users by adjusting challenges and activities, maintaining motivation and engagement. Adapting feedback requires modeling the student's behavior by monitoring and assessing knowledge levels during gameplay. At the core of this process are AI models that collect user data from in-game actions, analyze them, and provide support according to educational goals. These models utilize pre-trained data of expert knowledge or data-driven approaches that track student development in real-time.

Adaptive supports have a great potential for fostering educational content in SGs and improving learning efficiency. Adaptive SGs can assist educators in overcrowded classrooms where help is not easily accessible, and/or act as self-teaching tools. This paper surveys the research on adaptive support in SGs for programming, reviews their design process, discusses the learning results, and presents the latest advancements in the field. It is structured as follows: Section 2 presents a literature review regarding adaptive systems in SGs. Section 3 defines the methodology and scope of the survey. Next, Section 4 presents the results of the research question analysis. Finally, Sections 5–7 summarize and conclude the paper.

2. Related Work

The field of adaptive support is still in its early stages. To our knowledge, there has not yet been a systematic literature review (SLR) in this field for the topic of serious games for programming. Nonetheless, adaptive systems in serious games and learning applications have been studied in terms of player engagement and feedback. We identified five reviews published in recent years with a relevant context in education and adaptivity, as shown in Table 1.

Table 1. SLR on adaptive systems in serious games and learning applications.

Study	Type	Year	Quality Assessment of Primary Studies	Number of Primary Studies/Number of Data Sources	Period	Context
[11]	SLR	2022	No quality assessment	26/3	2000–2022	Adaptation components in serious games
[12]	Survey	2022	No quality assessment	Not specified/5	2014–2018	Automated programming hint generation
[13]	SLR	2021	Based on journal article reporting standards by the American Psychological Association (2020b)	10/6	Not specified	Adaptivity within educational video games
[14]	SLR	2020	Four-process adaptive cycle by Shute and Zapata-Rivera (Shute and Zapata-Rivera, 2007)	62/5	Not specified	Educational games with adaptivity
[15]	SLR	2019	No quality assessment	21/5	2008–2017	Data-driven approaches to player modeling in educational games
[16]	SLR	2019	Based on five criteria	16/6	2012–2018	Adaptive gamification in education

Hooshyar et al. [15] conducted a review of data-driven approaches for player modeling in educational games. They considered data-driven techniques an optimal solution when modeling player knowledge and behavior in adaptive games. This approach relies less on expert authoring, and it captures a variety of player data while the game is being executed. They discovered that the main objective of data mining in adaptive educational games was behavior modeling, followed by goal recognition and procedural content generation methods.

In the work of Lopes et al. [16], 16 papers were presented in a review on adaptive gamification strategies for various learning goals, researching how the adaptive features work. Gamification applies game thinking and game elements such as badges, leaderboards, and virtual points to promote learning by strategically combining them according to the education field or situation. Adaptive gamification dynamically customizes the game elements for each user by categorizing their preferences, in contrast to the "one-size-fits-all" approach adopted in most gamified environments. The survey concluded that most adaptive gamification implementations use existing player typologies to categorize users as a base for further adaptation of their gamification elements. It suggests that the adaptive gamification process in learning environments should also consider continuous player-type profiling and adaptability to learning topics as strategies for optimal effectiveness in gamification element integration.

A systematic analysis [14] was conducted on the integration of adaptivity in educational games using bibliometric, qualitative, thematic, and meta-analysis methods. Liu et al. [14] identified 62 publications, and concluded that adaptivity in games does not positively contribute to learning and game performance compared with non-adaptive games. According to the research, these conditions were more likely to be affected by factors such as randomization, game genre, and gameplay length rather than adaptivity design. However, this has also resulted in adaptivity having a positive effect on user engagement and focus on learning.

A study by Ninaus and Nebel [13] reviewed methods of acquiring data from serious games and how these analytics were utilized to adapt their learning environments. They found 10 relevant studies focused mostly on natural sciences with their integrated adaptive mechanisms set to improve learning. The majority of implementations relied on data-driven approaches instead of cognitive frameworks, and most of them resulted in positive outcomes. The authors reported a lack of standardized methods and theoretical foundations to analyze the effects of adaptations. Although they identify an increased interest in the field, the low number of papers indicates that more research is needed.

Another survey by McBroom et al. [12] provided an overview of methods for automatic hint generation as support for programming exercises. The paper surveyed the research on the numerous developed techniques by analyzing them as a series of smaller steps. Most hint-producing approaches were based on a next-step selection from past data, goal identification, or utilizing program features. Other techniques provided feedback by automatically repairing syntax errors in user programs either by using machine learning or search algorithms to find possible corrections. The authors proposed their own framework called HINTS (Hint Iteration by Narrow-down and Transformation Steps) for describing hint generation techniques and identifying relationships to facilitate future comparisons.

Lastly, another study by Aydin et al. [11] examined the adaptation components in educational games. Reviewing 26 articles revealed that adaptive game design is applied in various fields, although most of them refer to teaching programming. The study revealed that games adapt their educational content, item behaviors, and interface, with educational content being the preferred adaption element. They concluded that the most frequent method for adapting the learning content was by adjusting its difficulty. In addition, it was found that adaptation in the games was implemented both pre-game and in-game with a range of methods such as deep learning, Bayesian networks, and decision trees.

Previous research on adaptive methods in serious games focused on identifying various techniques and strategies for their implementation. This review shares the common research goal of classifying the adaptive methods and player behavior modeling, as demonstrated in the works of Aydin et al. [11], Hooshyar et al. [15], and Lopes et al. [16]. Additionally, some of the previous studies reported a similar research objective to the present work, which is the examination of the learning effect of adaptivity [13,14]. Although past research has explored the adaptivity of various game elements such as content and difficulty, the present study is distinct in its explicit focus on adaptive support methods. While one previous study, conducted by McBroom et al. [12], focused on support, it solely referred to hints, the present study includes all types of support. Lastly, this study stands out in its focus on teaching programming, as it is the only study to explicitly investigate the use of SGs for this purpose. By filling this research gap, this study contributes to the broader understanding of the potential benefits and challenges of using SGs for programming education. This paper provides an evaluation of provided support in terms of learning efficiency and discusses the implications of the design choices. This review is important for researchers and designers of serious games about programming who are considering implementing adaptive support to increase the learning results of their students. In addition, researchers can familiarize themselves with the up-to-date adaptive approaches and appliances of support in the field.

3. Methodology

A systematic literature review answers clearly defined research questions in a field by following a search protocol. This protocol specifies the criteria and the search strategy plan before the study is carried out to ensure unbiased results that other researchers can reproduce. This systematic literature review follows the PRISMA framework [17] to identify the search strategy and quality of papers. A central feature of the framework is the flow chart, which consists of four stages: identification, screening, eligibility, and inclusion. We aimed to review all empirical and theoretical studies regarding the adaptivity of supports in serious games for teaching programming. This review can be considered systematic as it attempted to include all existing published literature on the research questions. The search process was documented and will be reported in the following sections.

3.1. Research Questions

Our primary research question is as follows:

"What is the state of the art of adaptive support in serious games for programming?"

This primary question can be divided into three subsidiary questions. Firstly, we investigated what aspect of the support is adapted in terms of educational goals, context, rate of appearance, or the extent of the assistance (RQ1). Secondly, we identified the theoretical frameworks, methods, or models used to implement the adaptivity (RQ2). Finally, we examined the papers that provided evidence through empirical studies, reports, or questionnaires, and classified the results in terms of effectiveness by achieved learning, game progress, and user preference (RQ3). The subsidiary research questions can be summarized as follows:

RQ1: What adaptive supports are offered?

RQ2: Which methods are used to generate adaptive supports?

RQ3: What are the results of adaptivity in terms of effectiveness in learning?

3.2. Data Sources and Selection Criteria

The following four electronic databases were selected as relevant in this SLR research: Scopus, IEEE Xplore, SpringerLink, and ACM. Google Scholar was not used, as most of the results overlapped with the above sources. The articles were selected from the areas of computer and social sciences to exclude literature irrelevant to programming and education. The period covered by the review was from 2000 to 2022 (inclusive), referring to journal papers and conference proceedings in the English language. To address the topic of research questions, we used keywords and related synonyms of the main concepts, as presented in Table 2. Subsequently, we grouped the search terms to define the following search query:

("serious game" OR "game-based learning" OR "edutainment" OR "educational game" OR "learning game") AND (adapt* OR intelligen*) AND ("support" OR "help" OR "hint" OR "system" OR assistan* OR "feedback" OR "agent" OR tutor*) AND (program*)

Table 2. Search terms.

Topic	Keywords/Search Terms
Serious games	serious game OR game-based learning OR edutainment OR educational game OR learning game
Adaptiveness	adapt* OR intelligen*
Support	support OR help OR hint OR system OR assistan* OR feedback OR agent OR tutor*
Subject	program*

The query was adjusted to comply with each database's syntax and Boolean rules and then applied to perform a full-text search. Additionally, quotation marks were used wherever allowed to indicate exact phrases and proximity operators to limit word distance to 15 characters. In order to limit the search scope to only relevant results according to the research questions, we set the boundaries for this systematic review with inclusion and exclusion criteria, as presented in Table 3. Criteria C2 to C7 were used through the database interface during the search process, while criterion C1 was applied throughout article screening. We decided to include only papers that refer to serious games with the subject discipline of programming. These serious games needed to incorporate some form of adaptivity in their learning content or support method. Articles with theoretical models on adaptivity or implementation plans without an actual developed game were excluded.

Table 3. Inclusion and exclusion criteria.

ID	Criterion	Inclusion	Exclusion	Rationale
C1	Relevance for a review question	Relevant	Not relevant	Answering the review questions or addressing the topic of adaptive support in the context of serious games about programming
C2	Language	English	All except English	English can be considered the universal language for academic publications
C3	Time period	2000–2022	Before 2000	To ensure high quality standards, only the most recent papers will be included
C4	Scientific field	Computer Science, Education	All other scientific disciplines	This is the scientific discipline in which the review questions are located
C5	Publication Type	Journals, conference articles, or book chapters	Early-access articles, posters (<four pages)	To maintain a quality standard
C6	Media format	PDF, Word	Videos, images	Only formally structured documents are reviewed
C7	Availability	Available online as full text	Limited preview	The text should be available for review

3.3. Data Selection

The database search resulted in 930 articles, as shown in Table 4. After removing 88 duplicates, a three-phase screening process was initiated following the PRISMA guidelines. Articles were excluded in the first two phases based on title and abstract relevance according to the eligibility criterion C1. The remaining 83 articles were retrieved and evaluated taking into account criterion C1, which examines the relevance of the article with the research questions. After a full-text read, 18 papers referring to 10 serious games were selected, and they are presented in Table 5. The article selection process is shown in Figure 1, which depicts the flow of information through the screening procedure.

Table 4. Database results.

Database	Number of Articles Identified by Search Terms and Selection Criteria C2–C7
Scopus	313
IEEE-Xplore	142
SpringerLink	58
ACM	417
Total	930

Table 5. Serious games for programming with adaptive supports.

References	Game	Genre	Subject Discipline	Type of Support	Methodology
[8,18]	NanoDoc	FPS Puzzle	K12 programming	Working examples	Fuzzy logic
[19]	FuzAd_Escape	Adventure	HTML	Text, items, NPCs	Fuzzy knowledge state definer
[7,10,20]	AutoThinking	Puzzle	Programming, computational thinking	Text, video images, hints	Bayesian network
[21]	Quiz Time!	Quiz	C# programming language	Chat	Fuzzy logic
[22–24]	HTML Escape	Adventure	HTML	Text, items, NPCs	Fuzzy knowledge state definer
[25]	OGITS	Board	Programming concepts	Textbook material, webpage links	Bayesian network
[26,27]	Minerva	Puzzle	K12 programming	Text, image, video, hints	Questionnaire

Table 5. *Cont.*

References	Game	Genre	Subject Discipline	Type of Support	Methodology
[28,29]	BOTS	Puzzle	K12 programming	Hints	Interaction network (hint factory algorithm)
[30]	Aurora toolset from Neverwinter Nights 2	RPG	SQL (SELECT)	Dialog text	Questionnaire, dialog interactions
[31]	ENGAGE	Puzzle adventure	Programming	Hints	Dynamic Bayesian network

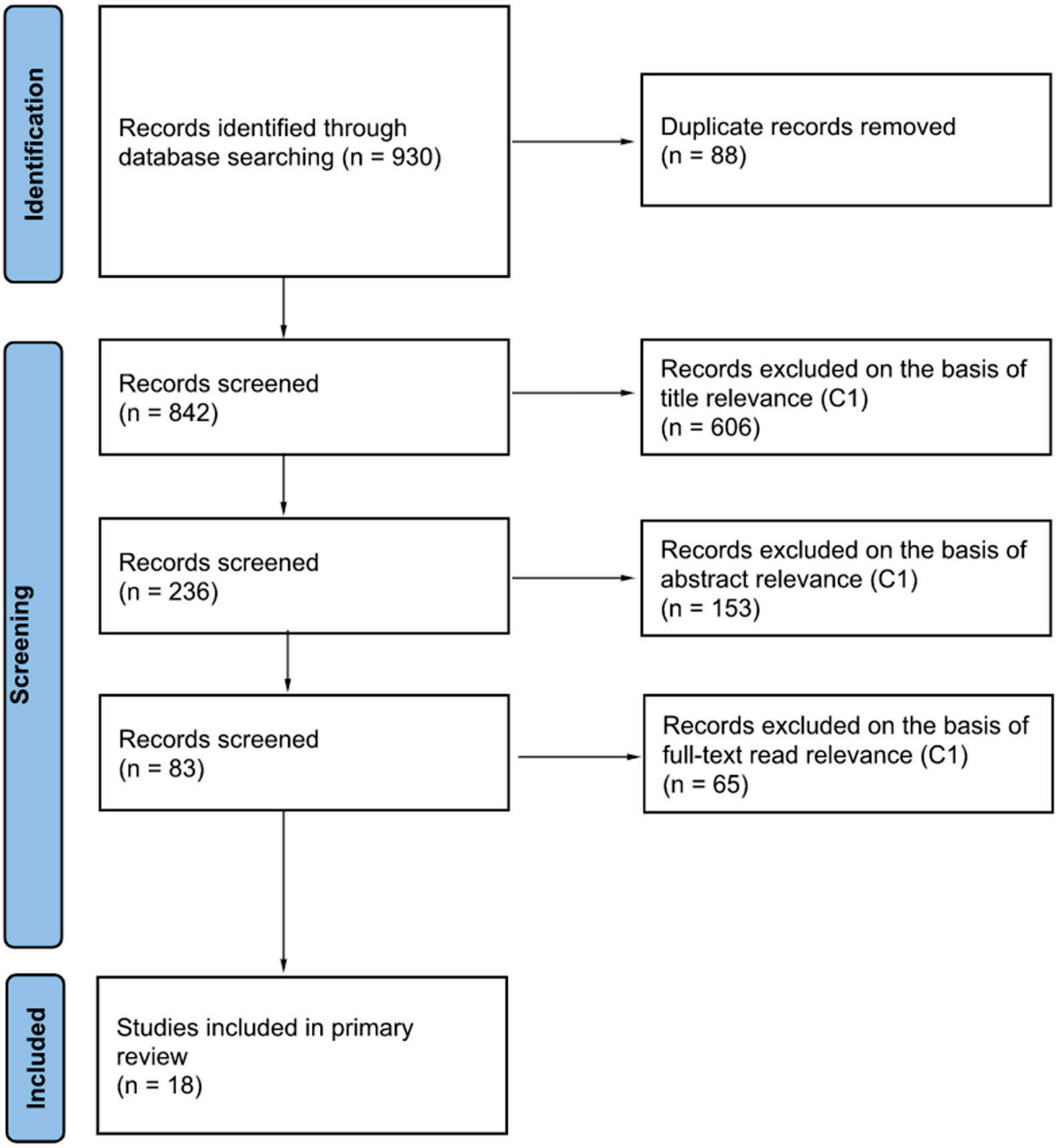

Identification
Records identified through database searching (n = 930)
Duplicate records removed (n = 88)
Screening
Records screened (n = 842)
Records excluded on the basis of title relevance (C1) (n = 606)
Records screened (n = 236)
Records excluded on the basis of abstract relevance (C1) (n = 153)
Records screened (n = 83)
Records excluded on the basis of full-text read relevance (C1) (n = 65)
Included
Studies included in primary review (n = 18)

Figure 1. PRISMA Flowchart.

4. Results Analysis

A quantitative analysis classified the papers by year (Figure 2) and by type of publication (Figure 3). The majority of papers were published after the year 2016, with journals and conferences being the main source of representation.

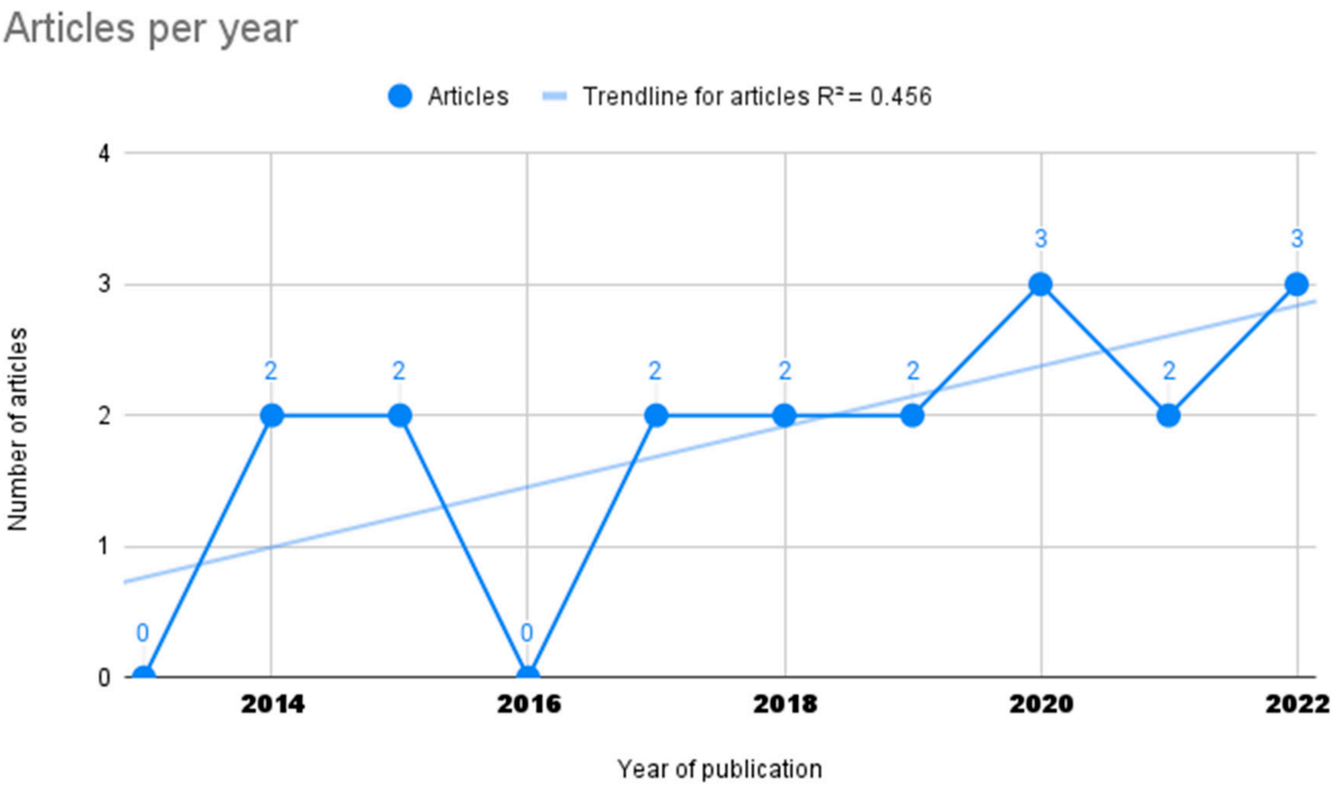


Figure 2. Distribution of papers per year.

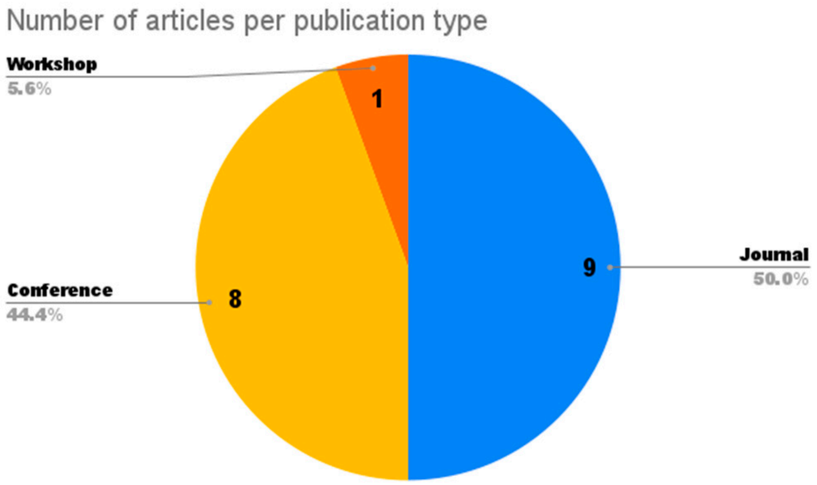


Figure 3. Distribution of papers per type of publication.

4.1. Adaptive Supports (RQ1)

A synopsis of all the adaptive support methods of the reviewed SGs is presented in Table 6. The majority of the SGs in this study focused on text and hint support methods. Two games (AutoThinking and Minerva) utilized multiple media formats as support, and only NanoDoc provided working examples. None of the games that were reported implemented audio as an adaptive support method. Figure 4 showcases screenshots of the SGs where the retrieval of images was possible, incorporating their adaptive support methods.

Table 6. Presentation methods of adaptive support (✔ when the type of support is applied).

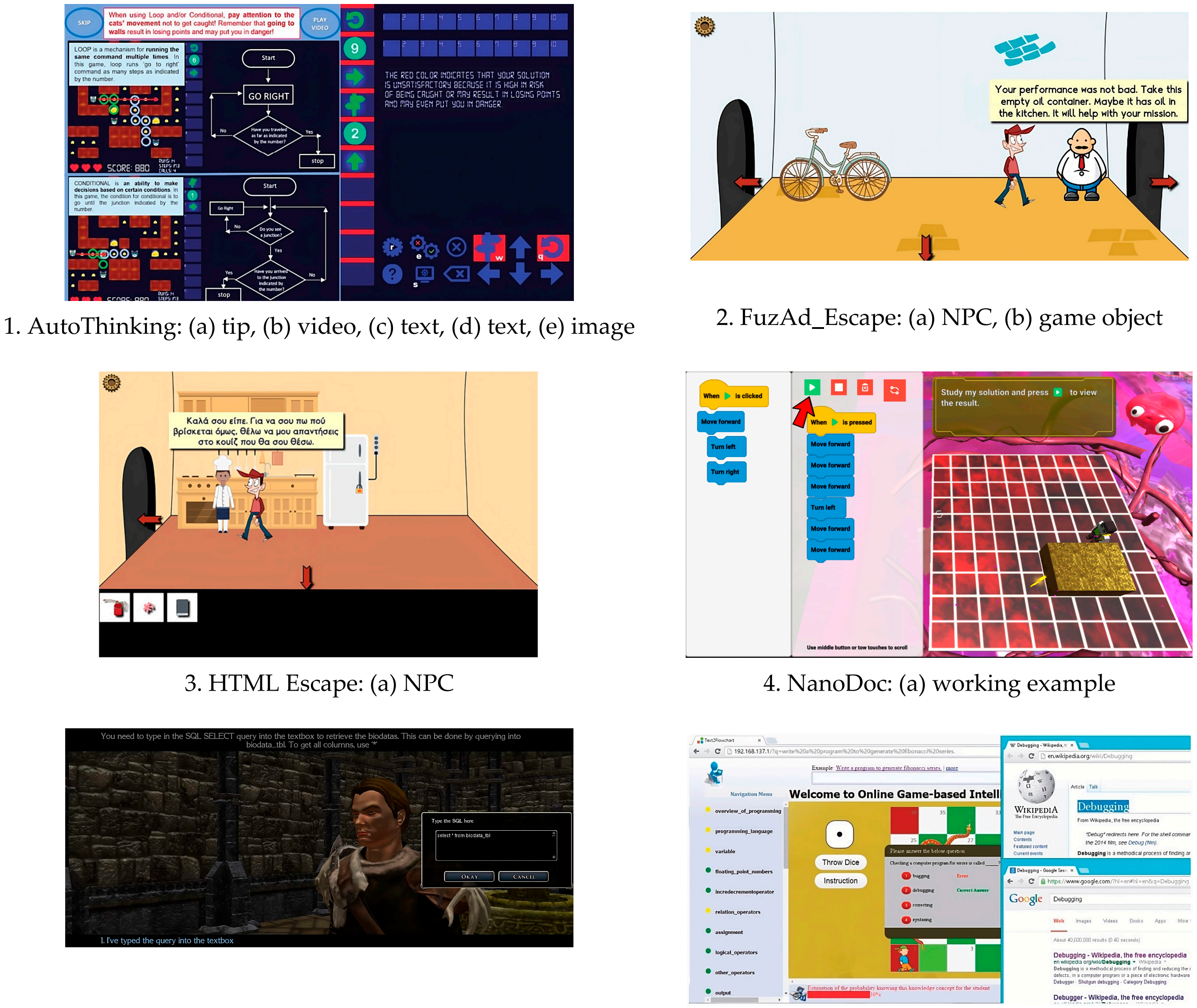
Game	Text	Hints	Images	Video	Working Examples
AutoThinking	✔	✔	✔	✔	X
BOTS	X	✔	X	X	X
ENGAGE	X	✔	X	X	X
FuzAd_Escape	✔ (NPC, game objects)	X	X	X	X
HTML Escape	✔ (NPC, game objects)	X	X	X	X
Minerva	✔ (Learning topic)	X	✔	✔	X
NanoDoc	X	X	X	X	✔
Neverwinter Nights	✔ (NPC dialog)	X	X	X	X
OGITS	✔ (Learning material, web links)	X	X	X	X
Quiz Time!	✔ Chat	X	X	X	X
Total	7	3	2	2	1

HTML Escape and its expansion FuzAd_Escape are adventure games that teach HTML. Both games adapt the scenario according to the player's knowledge level by adding new assets to the world corresponding to the player's learning needs. They provide support as text through non-player characters and objects such as notes placed in the dynamically created rooms. A game with no name mentioned in the paper by [30] was built using the aurora toolset from Neverwinter Nights 2, a well-known commercial role-playing game. It teaches the database programming language SQL (structured query language) and presents support to the player with the in-game dialog system. BOTS is another puzzle game designed to teach programming fundamentals. Players solve puzzles by creating small programs with graphical pseudo-code commands, and the game provides support with auto-generated hints.

The OGITS (Online Game-based Bayesian Intelligent Tutoring System)'s objective is to introduce programming (C++) to students through board games. Using a web-based environment, students can play the board games "Snakes and Ladders" and Tic-Tac-Toe while they answer quiz questions about programming concepts. They interactively explore the educational content as a Bayesian network is monitoring their progress and modeling prerequisite information to customize guidance with learning material and web links. In Minerva, a puzzle game for teaching programming to elementary school students, the adaptive support is static. It registers the player's learning style at the start of the game and provides assistance based on that. Support includes videos on how the game is played, images explaining the puzzle functions, and text about the learning topic.

AutoThinking is a puzzle game that assists students in developing computational thinking and programming skills. Players move inside a maze by creating small programs with an icon interface, collecting items, and avoiding enemies. Each time players submit a solution, the game adapts the support learning material according to the current state of the maze. It presents the player with text, images, and video tutorials or highlights game features as hints. ENGAGE is a game for teaching computer science principles where the support is based on the knowledge and learning activities of the students. Players explore the game world and solve problems with a visual programming interface. The game presents several interactive activities introducing the learning concepts while supporting students with screen hints and animated vignettes.

In Quiz Time!, a mobile game for teaching the C# programming language, students play cooperatively in pairs by answering questions in quizzes. They use the integrated chat system to discuss incorrect answers, thus encouraging student engagement and promoting collaboration. In case of a wrong answer, the game provides students with personalized help based on their knowledge profile. The support is created adaptively from an advice generator and is presented in chat form as text. Finally, NanoDoc also provides adaptive support based on the student's domain knowledge profile. It is a hybrid first-person shooter where players move in a 3D maze and solve puzzles with a block-based coding interface. It teaches the structures of sequence and iteration to novice users through a series of puzzles related to the programming concepts. It provides support depending on the knowledge level of players by presenting working examples of similar problems during the solution construction phase.



When using Loop and/or Conditional, **pay attention to the cats' movement** not to get caught! Remember that **going to walls** result in losing points and may put you in danger!

LOOP is a mechanism for **running the same command multiple times**. In this game, loop runs 'go to right' command as many steps as indicated by the number.

THE RED COLOR INDICATES THAT YOUR SOLUTION IS UNSATISFACTORY BECAUSE IT IS HIGH IN RISK OF BEING CAUGHT OR MAY RESULT IN LOSING POINTS AND MAY EVEN PUT YOU IN DANGER.

CONDITIONAL is **an ability to make decisions based on certain conditions**. In this game, the condition for conditional is to go until the junction indicated by the number.

1. AutoThinking: (a) tip, (b) video, (c) text, (d) text, (e) image

Your performance was not bad. Take this empty oil container. Maybe it has oil in the kitchen. It will help with your mission.

2. FuzAd_Escape: (a) NPC, (b) game object

Καλά σου είπε. Για να σου πω πού βρίσκεται όμως, θέλω να μου απαντήσεις στο κουίζ που θα σου θέσω.

3. HTML Escape: (a) NPC

Study my solution and press ▶ to view the result.

4. NanoDoc: (a) working example

You need to type in the SQL SELECT query into the textbox to retrieve the biodatas. This can be done by querying into biodata_tbl. To get all columns, use '*'

Type the SQL here

select * from biodata_tbl

1. I've typed the query into the textbox

5. Neverwinter Nights: (a) NPC dialog

Welcome to Online Game-based Intell

Please answer the below question

Checking a computer program for errors is called ____?

1 bugging Error
2 debugging Correct Answer
3 correcting
4 syntaxing

Estimation of the probability knowing this knowledge concept for the student 30%

Debugging is a methodical process of finding and reducing the number of bugs, or defects, in a computer program or a piece of electronic hardware

6. OGITS: (a) learning material, (b) web links

SKIP
PLAY VIDEO
Start
GO RIGHT
Have you traveled as far as indicated by the number?
No
Yes
stop
SCORE: 880
RUNS: 14
STEPS: 113
CALLS: 4
Go Right
Do you see a junction?
Have you arrived to the junction indicated by the number?
9
2
6
1
When ▶ is clicked
Move forward
Turn left
Turn right
Use middle button or tow touches to scroll
OKAY
CANCEL
Text2Flowchart
192.168.137.1/?q=write%20a%20program%20to%20generate%20fibonacci%20series.
Example: Write a program to generate fibonacci series. | more
Navigation Menu
overview_of_programming
programming_language
variable
floating_point_numbers
incredecrementoperator
relation_operators
assignment
logical_operators
other_operators
output
Throw Dice
Instruction
Debugging - Wikipedia
en.wikipedia.org/wiki/Debugging
WIKIPEDIA The Free Encyclopedia
Article
Talk
Debugging
From Wikipedia, the free encyclopedia
Main page
Contents
Featured content
Current events
Debugging - Google Search
https://www.google.com/?hl=en#hl=en&q=Debugging
Google
Web
Images
Videos
Books
Apps
More
Debugging - Wikipedia, the free encyclopedia
Debugger - Wikipedia, the free encyclopedia

Figure 4. Screenshots from SGs implementing various support methods.

4.2. Methods Used to Generate Adaptive Supports (RQ2)

Table 7 summarizes the methodological approaches used in the games to adapt their learning content. Overall, five types were identified: questionnaire, fuzzy logic, Bayesian network, hint factory, and dialog responses. Most of the games adapt their support during gameplay by processing player input in real time, with the exception of two games that adapt it statically with input taken prior to starting.

Table 7. Implementation methods for adaptive support.

Adaptation Type	Method	Game	Input Data	References
Static	Questionnaire	Minerva	Questionnaire answers (learning style)	[27]
		Neverwinter Nights	Questionnaire answers (learning style)	[30]
Dynamic	Fuzzy Logic	NanoDoc	Player programs	[18]
		FuzAd_Escape	Quiz answers	[19]
		Quiz Time!	Quiz answers, frequency of misconceptions, previous knowledge	[21]
		HTML Escape	Quiz answers	[22]
	Bayesian Network	AutoThinking	Player programs	[20]
		OGITS	Quiz answers	[25]
		ENGAGE	Player programs	[31]
	Hint Factory	BOTS	Player programs	[29]
	Dialog	Neverwinter Nights	Conversation selections (learning style)	[30]

Quiz Time!, NanoDoc, HTML Escape, and its expansion FuzAd_Escape utilize fuzzy logic to represent the student knowledge level and cognitive state of the player. Fuzzy logic [32] is a heuristic approach that can mimic real-life conditions with partial truth statements. This multivalued logic is in contrast to binary systems and provides partial values of “true” or “false” conditions, allowing decision-making with imprecise estimates. Through fuzzy logic, the aforementioned games approach the uncertainty of describing the knowledge level of a student on a domain using partial levels instead of distinct values. For example, the knowledge level of a student on the iteration control structure could be 0.6 insufficiently known and 0.4 known. These values are calculated from membership functions that take input from player actions. HTML Escape and FuzAd_Escape use quiz results from the player as input, while NanoDoc utilizes the result of player-created programs. Quiz Time! also receives the player’s answers to quiz questions to calculate the current knowledge, but additionally requires two more inputs. These are the previous knowledge of players in computer programming and the frequency of misconceptions made by players.

The Neverwinter Nights-based game modifies the text of the learning materials that appear in the conversation system between the player and non-player characters (NPCs). This text is adapted according to the learning style of the player and is based on the Felder–Silverman model [33], which indicates areas of personality that contribute to learning. The game has three modes, one non-adaptive and two adaptive depending on the support presented. In the first adaptive mode, the characteristics of the player are identified with a questionnaire of 44 questions that are filled out before starting the game. These characteristics remain static throughout the game session, and player actions are recorded passively without altering the support presentation. In the second adaptive mode, the student’s learning style is analyzed in real time according to the player’s interactions with the game. The support presentation is based on the evaluation of player responses in each conversation. Moreover, it provides an option to change the presentation type manually during conversation and records the selection as additional information for the calculation of the next NPC interaction.

BOTS provides intelligent feedback in the form of personalized hints. It extends the hint factory [34] which is a data-driven method for creating hints by tracing student states to a predefined solution graph. The graph contains data on previous student user interactions, and when a solution path is found, a hint is generated by suggesting a potential next step, which is the next node of the solution graph. However, BOTS uses the output of players’ programs instead of player actions as data, reducing the size of the solution graph, which would otherwise be immense for its open-ended puzzle problems.

The OGITS models student progress with a Bayesian network [35] and adapts the support material accordingly. Bayesian networks are probabilistic graphical systems that employ directed acyclic graphs to describe a set of variables and their conditional dependencies. The game implements a Bayesian network that contains a directed acyclic graph where each node represents a programming concept. Nodes that require knowledge of previous concepts are connected with them. Employing conditional dependencies between joints and taking into account the student answers to quiz questions, each node is labeled "known" or "unknown". Thus, with the Bayesian network, the game estimates what prerequisite concepts were not learned and directs students to the required resources. Minerva utilizes a questionnaire to identify student learning styles based on the models of Mumford's learning style questionnaire [36] and Bartle's player types [37] to handle students both as players and as learners. The questionnaire has 32 Likert scale statements that players answer before entering the game, mapping them as activists, theorists, pragmatists, or reflectors. The detection stage occurs only once and remains static throughout the course of gameplay. Mapping players with a corresponding learning style allows the game to alter the viewing order of the learning content, therefore adapting the support.

AutoThinking creates a cognitive model of the player with a Bayesian network allowing a real-time non-invasive assessment. The model consists of a directed acyclic graph where each skill is represented as a node and prerequisite relationships are connected with directed edges in a parent–child manner. A conditional probability set of distributions is defined for each node depending on its parent. The Bayesian network uses the input from the player solution, classifying it as satisfactory, normal, or unsatisfactory, and adapts the support content. This process can occur in two phases, in the debug mode before submitting the solution or after executing it.

ENGAGE also uses two Bayesian networks, named outer loop and inner loop, to support students. The first one is an adaptive problem-solving task strategy that selects tasks related to topics where students have knowledge gaps. A dynamic Bayesian network models student knowledge from their submitted solutions and quantifies their skill with binary variables on specific learning concepts. When it detects that a concept is not successfully learned, the game uses a scaffolding approach and dynamically reallocates players to intermediate tasks in order to master the concept. The second Bayesian network generates adaptive hints by suggesting the next steps of a solution in a problem task. Additionally, it displays hints to enhance learning outcomes when it predicts that a concept is not fully learned. The article mentions that both Bayesian networks are planned but not yet implemented.

4.3. Adaptivity Effectiveness in Learning (RQ3)

In Table 8, a summary of the results for learning effectiveness and the adaptivity effect of the reviewed games that provide evaluation data are presented.

Table 8. Results in terms of effectiveness in learning and adaptivity effect.

Game	**Effectiveness in Learning**	**Adaptivity Effect**	**Method of Testing**	**Participant Age Group**
AutoThinking	Positive	Not tested	Pre/post test, a control group	Elementary school
FuzAd_Escape	Positive	Not tested	Questionnaire survey, *t*-test, a control group	University
HTML Escape	Positive	Not tested	Questionnaire survey	University
Minerva	Not affected	Not tested	Post-game test, control group, interview	Elementary school
NanoDoc	Positive	Positive	Control group	Elementary school
Neverwinter Nights	Not affected	Not affected	Pre/post test, control group, interviews	University
Quiz Time!	Positive	Positive	*t*-Test, control group	University

HTML Escape collected feedback with a questionnaire survey. The researchers reported positive results as students stated that the game helped them overcome difficulties in the learning content. FuzAd_Escape used questionnaires for teachers and students and in accordance with their answers, the game improved the learning results in HTML. Additionally, FuzAd_Escape conducted a T -test on two groups of students, with only one group playing the game. After a teaching period of three weeks, a test about HTML was given to both groups, and their grades were compared. It was found that students who played the game had better scores on the test.

The Neverwinter Nights-based game evaluated its adaptive methods by comparing the results of a pre-test and post-test on SQL with four groups of students. The first group (control) studied the learning material only from a textbook and the second group played the game in a non-adaptive mode. The third group used the adaptive mode with data taken in advance of playing the game, while the last group played the mode where support is automatically adapted to the learning style of the player. It was concluded that students who played the game performed better on the test compared to those who studied with the textbook. The adaptive mode group had shorter completion times than all other groups and also the highest means of learning effectiveness, although the latter was proven to be not statistically significant. Interviews were conducted with students, and quantitative and qualitative data were gathered to detect the contribution of the game to student programming skills. The analysis showed that students improved their comprehension of programming concepts and that the game helped them to recognize their weak spots. Moreover, it was noted that the game succeeded in teaching students without a teacher.

The Minerva game's adaptive model was assessed with a formative evaluation with elementary school students. The researchers compared the learning outcomes of a post-test questionnaire between a group of students who played the game and a control group who studied with a textbook. Additionally, semi-structured interviews were conducted, and game log data were analyzed. It was concluded that learning efficiency was equally effective for both groups, although the game was shown to facilitate engagement among the students. A similar approach to the evaluation was followed for the AutoThinking game with an experimental and a control group of students. The computational thinking knowledge was estimated with a pre-test that all participants took. The control group were taught the learning content with traditional teaching, while the experimental group played the game for an equal amount of time. Statistical analysis of the post-test revealed that students who used the game had better scores than the control group. It was concluded that the game improved the computational thinking of students related to both conceptual knowledge and skills.

For Quiz Time!, an evaluation of the game's learning effectiveness was conducted using 80 university students. The population was split into two equal-sized groups, and students used the game for an academic semester. A different version of the game was given to each group. The first one had a conventional version of the game that lacked the dynamic advice generator and knowledge assessment features, while the second had a fully adaptive application. Statistical analysis with a *t*-test revealed significantly better performance in terms of quiz grades for the group of students who used the dynamic advice generator support. For NanoDoc, researchers conducted an empirical study on 102 elementary school students divided randomly by the game into two even groups. The game provided the same support method for both groups but with different types of accessibility. In the first, players could manually choose when to receive assistance, while in the second group, the support was activated adaptively according to the knowledge level of the student. Statistical analysis of the comparison in player programs revealed higher educational efficiency of the group with adaptive support.

Three of the studies examined in this review did not provide any empirical evidence of the impact of their games (BOTS, ENGAGE, and OGITS) on learning efficiency. On the other hand, four SGs, namely HTML Escape, FuzAd_Escape, Minerva, and AutoThinking, conducted experiments involving control groups who received traditional teaching or textbook-based instruction alongside experimental groups who played the game. Although these studies reported positive effects on learning efficiency, none of them explicitly measured the effect of game adaptivity on learning effectiveness. However, given that the primary feature of these games was the adaptive support, it is plausible that this was the cause of the positive learning outcomes. Nonetheless, such claims cannot be made with certainty. In contrast, the studies on three SGs, namely Quiz Time!, NanoDoc, and Neverwinter Nights, directly compared the learning outcomes of groups who received adaptive support with those who received non-adaptive support. These studies reported positive learning outcomes for adaptive support groups, except for Neverwinter Nights, which reported no impact.

5. Discussion

The vast majority of reviewed games in the survey were not publicly available, so it was not possible to examine them and verify their adaptive support methods and mechanics. It would be beneficial and effective for future research if serious game developers provided public access to their games. Moreover, there was a lack of a standard methodology in the presented articles, since various types of quantitative and qualitative techniques were applied. As indicated in similar studies [38], there is a need for common methodological practices to be established for the evaluation of serious games.

Although this survey includes articles from 2000–2022, it is reported that the first study about adaptive serious game support for programming was published in 2014. After that year, there has been growing interest in the field and a steady increase in the creation of new applications. Recent artificial intelligence and data analysis developments allowed researchers to produce new methods for personalization and adaptivity [39]. Additionally, advancements in the game industry and computing reduced the costs of game development and offered new tools for designing, prototyping, and authoring a game. These improvements overcome technical challenges of the past such as real-time data processing and the creation of complex immersive worlds, allowing more challenging and sophisticated games to be implemented.

According to the results of RQ1, the most common type of educational content presentation through support was text. Although support was offered with other means too such as images, videos, or working examples, the text method had a constant presence in almost all studies. Support via text is easy to implement and provides an effective and direct way of communication with the user. The text was presented in the games utilized in the studies with a diversity of forms such as tips (FuzAd_Escape, HTML Escape, Minerva), hints (BOTS, ENGAGE, AutoThinking), textbook material (OGITS), dialog (Neverwinter Nights, OGITS, FuzAd_Escape, HTML Escape), or in a chat form (Quiz Time). Some authors attempted to integrate support in the game context by presenting it over a game element such as a non-player character. However, in most cases, although the content of the message was personalized and adapted to cover the learning insufficiencies for a specific user, it was not delivered as a game mechanic. Integrating learning material through a game mechanic instead of plain text was reported to produce better learning results [40].

This review presented several techniques that were applied in games to generate adaptive support content (RQ2). Fuzzy logic was most frequently used (NanoDoc, FuzAd_Escape, Quiz Time, HTML Escape) followed by Bayesian networks (AutoThinking, OGITS, ENGAGE), questionnaires (Neverwinter Nights, Minerva), and the hint factory algorithm (BOTS). All of them were based on data-driven approaches and did not apply models that require training before execution such as artificial neural networks and deep learning. Data-driven methods tend to be easier to develop and cost-effective when compared with model-based approaches such as intelligent tutoring systems [41]. The most common utilization of collected data in games mentioned in the survey was to model student knowledge and behavior through user interactions and progress.

The majority of the studies reviewed in this analysis provided data on how the use of support affects learning effectiveness, as assessed by research question 3 (RQ3). The data in

Table 8 show that in most cases, the impact of support on learning was positive. However, three of the SGs (BOTS, ENGAGE, and OGITS) did not present any empirical evidence on how their support implementation affected players, while two SGs (Neverwinter Nights and Minerva) reported no changes in learning efficiency. It should be noted that the adaptive support methods used by Neverwinter Nights and Minerva were static, based on data collected from questionnaires prior to gameplay (as shown in Table 7). Conversely, all of the real-time adaptive support methods that provided evaluation data demonstrated positive results in terms of learning effectiveness. This suggests that static adaptive methods may perform worse than methods that can adapt to player feedback during gameplay. Notably, three SGs (Quiz Time!, NanoDoc, and Neverwinter Nights) directly compared adaptive and non-adaptive support, with two of them (Quiz Time!, NanoDoc) reporting positive results in terms of learning efficiency.

All the participants in the reviewed studies (Table 8) were novices in programming (elementary schools) or new to the discipline subject (universities). This suggests that researchers tend to assess adaptive support using students with no previous experience in programming. The logic behind this decision is probably because this group of students requires constant assistance due to the difficulties of learning programming during the early stages [4]. An additional confirmation of the above argument is the design decision of developers to create mostly puzzle games (Table 5), which is the most popular game genre in teaching introductory programming [42]. However, secondary school students or people of older ages (age > 25) were not included in the surveys. Further study is needed to investigate adaptive support models on various knowledge levels of programming and how it differentiates according to age.

6. Limitations

The current study was conducted under a number of limitations and the results should be reviewed accordingly. Firstly, all non-English studies, early-access papers, and posters have been excluded from the search procedure. Furthermore, the search results were limited by the query syntax and the number of databases used. Although the query included many keywords and operators, some relevant studies might have been missed. However, a detailed citation tracking of the reviewed papers did not reveal any additional studies that failed to be discovered by the search strategy. Finally, the review focused specifically on adaptive support methods for programming in serious games and did not target broad gamification-related fields.

7. Conclusions

This paper describes a scientific literature review on adaptive supports for programming in serious games. It covers the time period 2000–2022 and it is the only review to date that has been conducted on this subject. The review follows the PRISMA guidelines, and the screening process resulted in 18 papers, which are based on 10 distinct games. The adaptive supports of these games were compared in terms of their type, utilized methodology, and effectiveness in learning. The findings indicate that developers prefer data-driven methods for adaptive support implementations and text is the most common form of presentation. Regarding learning effectiveness, not all researchers agreed on positive results, nor did every game provide evaluation data. Due to these facts, we can not conclude decisively that adaptive support is beneficial in terms of learning effectiveness in serious games for programming. However, most of the games reported positive results, although their subjects were exclusively novices. The limited number of relevant publications indicates that adaptive support for serious games is a new approach in the field. Consequently, taking into consideration the results of this review, additional research is needed in the field, as are future reviews with a significantly larger pool of studies.

Author Contributions: Conceptualization, methodology, validation, formal analysis, data curation, writing—original draft preparation, writing—review and editing, visualization, P.T. and S.X.; supervision, S.X.; project administration, S.X. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: All data are included in the article.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Felicia, P. *Handbook of Research on Improving Learning and Motivation through Educational Games*; iGi Global: Hershey, PA, USA, 2011. [CrossRef]
2. Susi, T.; Johannesson, M.; Backlund, P. Serious Games—An Overview. 28. 2007. Available online: https://www.diva-portal.org/smash/record.jsf?pid=diva2:2416 (accessed on 5 December 2022).
3. Bellotti, F.; Kapralos, B.; Lee, K.; Moreno-Ger, P.; Berta, R. Assessment in and of Serious Games: An Overview. *Adv. Hum.-Comput. Interact.* **2013**, 1–11. [CrossRef]
4. Cheah, C.S. Factors Contributing to the Difficulties in Teaching and Learning of Computer Programming: A Literature Review. *Contemp. Educ. Technol.* **2020**, *12*, ep272. [CrossRef] [PubMed]
5. Avila-Pesántez, D.; Rivera, L.A.; Alban, M.S. Approaches for Serious Game Design: A Systematic Literature Review. *ASEE Comput. Educ. (CoED) J.* **2017**, *8*, 12.
6. Schrader, C. Serious Games and Game-Based Learning. In *Handbook of Open, Distance and Digital Education*; Springer: Singapore, 2022; Volume 1, pp. 1–14. [CrossRef]
7. Hooshyar, D.; Pedaste, M.; Yang, Y.; Malva, L.; Hwang, G.-J.; Wang, M.; Lim, H.; Delev, D. From Gaming to Computational Thinking: An Adaptive Educational Computer Game-Based Learning Approach. *J. Educ. Comput. Res.* **2020**, *59*, 383–409. [CrossRef]
8. Toukiloglou, P.; Xinogalos, S. Adaptive Support with Working Examples in Serious Games About Programming. *J. Educ. Comput. Res.* **2023**, 07356331231151393. [CrossRef]
9. Vermunt, J.D.; Vermetten, Y.J. Patterns in Student Learning: Relationships Between Learning Strategies, Conceptions of Learning, and Learning Orientations. *Educ. Psychol. Rev.* **2004**, *16*, 359–384. [CrossRef]
10. Hooshyar, D.; Lim, H.; Pedaste, M.; Yang, K.; Fathi, M.; Yang, Y. AutoThinking: An Adaptive Computational Thinking Game. In Proceedings of the Innovative Technologies and Learning: Second International Conference, ICITL 2019, Tromsø, Norway, 2–5 December 2019; pp. 381–391. [CrossRef]
11. Aydin, M.; Karal, H.; Nabiyev, V. Examination of adaptation components in serious games: A systematic review study. *Educ. Inf. Technol.* **2022**, *1*, 1–22. [CrossRef]
12. McBroom, J.; Koprinska, I.; Yacef, K. A Survey of Automated Programming Hint Generation: The HINTS Framework. *ACM Comput. Surv.* **2021**, *54*, 1–27. [CrossRef]
13. Ninaus, M.; Nebel, S. A Systematic Literature Review of Analytics for Adaptivity Within Educational Video Games. *Front. Educ.* **2021**, *5*, 611072. [CrossRef]
14. Liu, Z.; Moon, J.; Kim, B.; Dai, C.-P. Integrating adaptivity in educational games: A combined bibliometric analysis and meta-analysis review. *Educ. Technol. Res. Dev.* **2020**, *68*, 1931–1959. [CrossRef]
15. Hooshyar, D.; Yousefi, M.; Lim, H. A systematic review of data-driven approaches in player modeling of educational games. *Artif. Intell. Rev.* **2017**, *52*, 1997–2017. [CrossRef]
16. Lopes, V.; Reinheimer, W.; Medina, R.; Bernardi, G.; Nunes, F.B. Adaptive gamification strategies for education: A systematic literature review. *Braz. Symp. Comput. Educ. (Simpósio Bras. De Inf. Na Educ. SBIE)* **2019**, *30*, 1032. [CrossRef]
17. Moher, D.; Liberati, A.; Tetzlaff, J.; Altman, D.G. Preferred Reporting Items for Systematic Reviews and Meta-Analyses: The PRISMA Statement. *Ann. Intern. Med.* **2009**, *151*, 264–269. [CrossRef] [PubMed]
18. Toukiloglou, P.; Xinogalos, S. NanoDoc: Designing an Adaptive Serious Game for Programming with Working Examples Support. In Proceedings of the European Conference on Games Based Learning, Lisbon, Portuga, 6–7 October 2022; Volume 16, pp. 628–636. [CrossRef]
19. Chrysafiadi, K.; Papadimitriou, S.; Virvou, M. Cognitive-based adaptive scenarios in educational games using fuzzy reasoning. *Knowl. Based Syst.* **2022**, *250*, 109111. [CrossRef]
20. Hooshyar, D.; Malva, L.; Yang, Y.; Pedaste, M.; Wang, M.; Lim, H. An adaptive educational computer game: Effects on students' knowledge and learning attitude in computational thinking. *Comput. Hum. Behav.* **2020**, *114*, 106575. [CrossRef]
21. Troussas, C.; Krouska, A.; Sgouropoulou, C. Collaboration and fuzzy-modeled personalization for mobile game-based learning in higher education. *Comput. Educ.* **2019**, *144*, 103698. [CrossRef]
22. Chrysafiadi, K.; Papadimitriou, S.; Virvou, M. Fuzzy states for dynamic adaptation of the plot of an educational game in relation to the learner's progress. In Proceedings of the 2020 11th International Conference on Information, Intelligence, Systems and Applications, Piraeus, Greece, 15–17 July 2020; pp. 1–7. [CrossRef]

23. Papadimitriou, S.; Chrysafiadi, K.; Virvou, M. FuzzEG: Fuzzy logic for adaptive scenarios in an educational adventure game. *Multimed Tools Appl.* **2019**, *78*, 32023–32053. [CrossRef]
24. Papadimitriou, S.; Virvou, M. Adaptivity in scenarios in an educational adventure game. In Proceedings of the 2017 8th International Conference on Information, Intelligence, Systems & Applications (IISA), Larnaca, Cyprus, 27–30 August 2017; pp. 1–6. [CrossRef]
25. Hooshyar, D.; Ahmad, R.B.; Wang, M.; Yousefi, M.; Fathi, M.; Lim, H. Development and Evaluation of a Game-Based Bayesian Intelligent Tutoring System for Teaching Programming. *J. Educ. Comput. Res.* **2017**, *56*, 775–801. [CrossRef]
26. Lindberg, R.S.N.; Hasanov, A.; Laine, T.H. Improving Play and Learning Style Adaptation in a Programming Education Game. In Proceedings of the 9th International Conference on Computer Supported Education, Porto, Portugal, 21–23 April 2017; pp. 450–457. [CrossRef]
27. Lindberg, R.S.N.; Laine, T.H. Formative evaluation of an adaptive game for engaging learners of programming concepts in K-12. *Int. J. Serious Games* **2018**, *5*, 3–24. [CrossRef]
28. Hicks, A.; Peddycord, B.; Barnes, T. Building Games to Learn from Their Players: Generating Hints in a Serious Game. In Proceedings of the Intelligent Tutoring Systems: 12th International Conference, ITS 2014, Honolulu, HI, USA, 5–9 June 2014; pp. 312–317. [CrossRef]
29. Hicks, D.; Dong, Y.; Zhi, R.; Cateté, V.; Barnes, T. BOTS: Selecting Next-Steps from Player Traces in a Puzzle Game. In Proceedings of the 8th International Conference on Educational Data Mining, Madrid, Spain, 26–29 June 2015. Available online: https://ceur-ws.org/Vol-1446/GEDM_2015_Submission_10.pdf (accessed on 5 December 2022).
30. Soflano, M.; Connolly, T.M.; Hainey, T. An application of adaptive games-based learning based on learning style to teach SQL. *Comput. Educ.* **2015**, *86*, 192–211. [CrossRef]
31. Min, W.; Mott, B.; Lester, J.C. Adaptive Scaffolding in an Intelligent Game-Based Learning Environment for Computer Science. In Proceedings of the Second Workshop on AI-Supported Education for Computer Science, Honolulu, HI, USA, 5 June 2014; pp. 41–50.
32. Zadeh, L.A. Is there a need for fuzzy logic? *Inf. Sci.* **2008**, *178*, 2751–2779. [CrossRef]
33. Graf, S.; Viola, S.R.; Leo, T. Kinshuk In-Depth Analysis of the Felder-Silverman Learning Style Dimensions. *J. Res. Technol. Educ.* **2007**, *40*, 79–93. [CrossRef]
34. Stamper, J.; Barnes, T.; Lehmann, L.; Croy, M. The Hint Factory: Automatic Generation of Contextualized Help for Existing Computer Aided Instruction. In Proceedings of the 9th International Conference on Intelligent Tutoring Systems Young Researchers Track, Raleigh, NC, USA, 29 June–2 July 2016; p. 9.
35. Heckerman, D. A Tutorial on Learning with Bayesian Networks. *arXiv* **2022**, arXiv:2002.00269. [CrossRef]
36. Honey, P.; Mumford, A. *The Learning Styles Helper's Guide*; Peter Honey Publications: Maidenhead, UK, 2000.
37. Bartle, R. Hearts, Clubs, Diamonds, Spades: Players Who Suit MUDs. *J. MUD Res.* **1996**, *1*, 28.
38. Miljanovic, M.A.; Bradbury, J.S. A Review of Serious Games for Programming. In Proceedings of the Serious Games: 4th Joint International Conference, JCSG 2018, Darmstadt, Germany, 7–8 November 2018; pp. 204–216. [CrossRef]
39. Streicher, A.; Smeddinck, J.D. Personalized and Adaptive Serious Games. In Proceedings of the Entertainment Computing and Serious Games: International GI-Dagstuhl Seminar 15283, Dagstuhl Castle, Germany, 5–10 July 2015; pp. 332–377. [CrossRef]
40. Molnar, A.; Kostkova, P. On Effective Integration of Educational Content in Serious Games: Text vs. Game Mechanics. In Proceedings of the 2013 IEEE 13th International Conference on Advanced Learning Technologies, Beijing, China, 15–18 July 2013; pp. 299–303. [CrossRef]
41. Hooshyar, D.; Lee, C.; Lim, H. A survey on data-driven approaches in educational games. In Proceedings of the 2016 2nd International Conference on Science in Information Technology, Balikpapan, Indonesia, 26–27 October 2016; pp. 291–295. [CrossRef]
42. Pelánek, R.; Effenberger, T. Design and analysis of microworlds and puzzles for block-based programming. *Comput. Sci. Educ.* **2020**, *32*, 66–104. [CrossRef]

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.