applied sciences

Article

Proteo: A Framework for Serious Games in Telerehabilitation

Giuseppe Bernava [1], Andrea Nucita [2,*], Giancarlo Iannizzotto [2], Tindara Caprì [3] and Rosa Angela Fabio [3]

[1] Institute for Biomedical Research and Innovation (IRIB-CNR), Via Torre Bianca, Mortelle, Istituto Marino, 98164 Messina, Italy; giuseppe.bernava@irib.cnr.it
[2] Department of Cognitive Sciences, Psychological, Educational and Cultural Studies, University of Messina, 98122 Messina, Italy; ianni@unime.it
[3] Department of Clinical and Experimental Medicine, University of Messina, 98121 Messina, Italy; tcapri@unime.it (T.C.); rafabio@unime.it (R.A.F.)
* Correspondence: anucita@unime.it

Abstract: Within the context of telerehabilitation, serious games have a significant role, but creating software for serious games is resource demanding. We present Proteo, a modular and open-source framework for developing serious games from scratch. We also present two serious game implementation examples with analysis of end user and therapist/researcher satisfaction. By involving a group of 11 specialized therapists and 9 end users we analyzed the Proteo's user satisfaction. We found that both groups scored high for the level of involvement, and the therapists scored also high for the level of suitability. More in depth, both groups showed significant differences between positive and negative feelings, with positive feelings scoring higher than negative ones. Finally, the user level of suitability was reported as high while the difficulty of the system and the difficulty of the task were reported as low. Proteo has proven to be a useful tool to develop serious games for telerehabilitation and has been well accepted by the users involved in the evaluation tests.

Keywords: telerehabilitation; serious games; human–computer interaction

Citation: Bernava, G.; Nucita, A.; Iannizzotto, G.; Caprì, T.; Fabio, R.A. Proteo: A Framework for Serious Games in Telerehabilitation. *Appl. Sci.* **2021**, *11*, 5935. https://doi.org/10.3390/app11135935

Academic Editors: Inaki Diaz and Jorge Juan Gil

Received: 11 May 2021
Accepted: 21 June 2021
Published: 25 June 2021

1. Introduction

Telerehabilitation (TR) can be defined as delivering rehabilitation services to the patient's home through a remote connection with the therapist, possibly enhanced by additional multimedia technologies [1–3]. TR can be provided using a variety of technological tools and modalities, for example, real-time visits with audio, video, or both; asynchronous e-visits, telephone assessments; closed-circuit television to video conferences; internet-based communication software (e.g., teleconference platform) with headsets, webcams, speakers, handheld cameras, microphones, and internet; serious games; or remote evaluations of recorded videos or images [1].

Several systematic reviews have argued that TR is an effective tool for patients with different disabilities, from musculoskeletal conditions, multiple sclerosis, neurological disorders, and stroke to autism and genetic syndromes [4–9]. Moreover, it was found that TR services are effective both for children and adults with disabilities, and professionals have also reported high levels of satisfaction and acceptance of TR services [5,10–12]. However, it is expected that advancements in technology can increase the usability and effectiveness of TR, but current TR systems do not usually allow for therapists to customize or adapt these systems to the characteristics of the patient and setting, and they are limited to specific patient conditions [13–18]. This can be due to different factors such as the complexity of the process, the technical aspects of TR service development, and the heterogeneity of patients with disabilities. Thus, it is necessary to develop and test the suitability and safety of TR modalities that can be customized by the therapist across subgroups of patients.

The last year, which has been deeply impacted by the COVID-19 pandemic, has led to a significant increase in the use of remotely guided rehabilitation and therapy interventions [19–23]. However, patient conditions vary significantly, and the need for a framework to define customized TR systems that can improve the engagement of the involved patients arises, with the aim of facing the enormous variability in patient specificity and needs. Moreover, such a framework should have a modular structure, such that a therapist can add support for specific hardware devices (i.e., eye trackers) or software capabilities (i.e., computer vision, machine learning) which would be useful in allowing the patients interact with the system and to acquire data for further analyses. Consequently, the implementation of TR leads to a double challenge: motivating patients and ensuring an effective system [24–26].

In this scenario, we propose Proteo, a framework designed to give therapists and researchers without specific skills in software development the capability of defining TR service customization and developing rehabilitation-oriented serious games, starting from game templates. Moreover, the framework can be applied in rehabilitation centers that could easily develop suites of remotely administered serious games, focusing only on a simplified game implementation, exploiting the framework modules for client/server communication, user management, human–computer interaction, or deep learning techniques.

This framework aims at: (a) the creation of personalized serious games for TR systems; (b) monitoring of the therapeutic goals of the rehabilitation process; (c) simplifying the interaction schemes between the patients and serious games. Moreover, Proteo is open-source and cross-platform, and it is distributed under the MIT license.

In the next sections, after presenting Proteo, we discuss two use cases to analyze the practicability of the serious games creation process and to discuss user satisfaction. In this paper, we present experiments involving therapists and researchers, while we plan to apply Proteo in telerehabilitation programs involving people with disabilities as part of future work.

2. Materials and Methods

Several libraries and frameworks have been proposed to facilitate the programming of visual/artistic applications (e.g., openFrameworks.cc) and games (e.g., stencyl.com accessed on 11 May 2021, defold.com accessed on 11 May 2021, www.yoyogames.com/en/gamemaker, accessed on 11 June 2021, https://gdevelop-app.com/, accessed on 11 June 2021). Libraries such as openFrameworks can benefit from using the C++ language, which has various modules that are specifically designed for artistic purposes with computer vision support, a crucial tool when dealing with serious games. However, openFrameworks is intended for C++ developers, which can be a barrier to those who have no programming skills. On the other hand, applications and libraries for game development (even with a simplified block programming interface) are missing significant tools necessary in the field of telerehabilitation, such as specific computer vision tools aimed at pose estimation, face detection, facial expression recognition etc.

The aim of the proposed platform, Proteo, is hence to give therapists and researchers without specific skills in software development, the capability of defining serious game customizations starting from game templates. Given a game template, which defines the game characteristics, Proteo enables therapists/researchers to define the game logic and the user interaction in a simplified manner, focusing only on what she/he wants to analyze, overcoming technological barriers and concentrating only on defining tasks and collecting data.

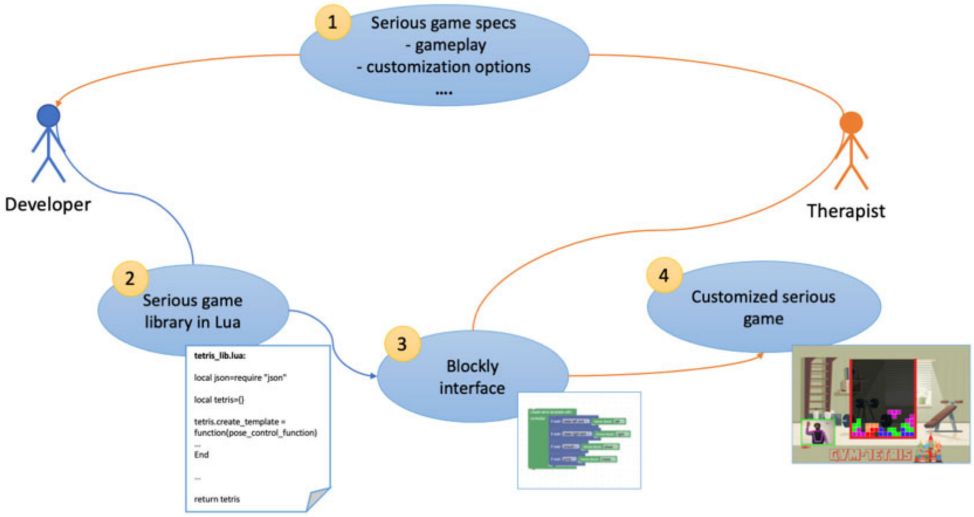
Moreover, the framework can be applied in rehabilitation or research centers with software development skills that could easily develop suites of remotely administered serious games, focusing only on a simplified game implementation, exploiting the framework modules for client/server communication, user management, a database for collecting data, human–computer interaction, or deep learning techniques, thus potentially gathering a large amount of data that can enable the exploitation of machine learning techniques [27].

In the following, we present Proteo, an open-source and cross-platform scriptable platform that is highly modular with built-in support for many useful modules for serious game implementation. We analyze two use cases testing Proteo with two different kinds of users: therapist/researchers and end users. In the first case, we ask the therapist/researcher group to customize a serious game after being given a game template through the platform. In the latter, we ask a group of end users to play a serious game developed with Proteo. In both cases we collected data regarding user satisfaction, which will be discussed in the Results section. Moreover, the technical details of the platform are presented in Appendix A.

2.1. Serious Game Implementation with Proteo

The process of implementing a serious game with Proteo involves a multidisciplinary team. As mentioned previously, the aims of such a platform are: (i) to facilitate the implementation of a serious game by providing specific tools and libraries to developers and (ii) to give therapists the possibility to customize serious games in terms of user interaction, data to acquire etc.

Starting from an initial idea, developing a serious game with Proteo involves a precise workflow, as shown in Figure 1.



```
tetris_lib.lua:

local json=require "json"

local tetris={}

tetris.create_template =
function(pose_control_function)
...
End

...

return tetris
```

Figure 1. Serious game implementation workflow.

The overall project is discussed between IT experts and clinicians, defining the gameplay and which parameters the clinicians want to customize. Depending on the proposed activity, these parameters can affect different aspects of the game (e.g., audio, user interaction, acquired data etc.) thus determining which Proteo module is needed. For example, in GymTetris, which we present in the following section, it is necessary to match each of the four possible movements of the tetrominoes with a specific body movement. In this case, the module for the pose estimation is required.

Once these parameters are selected, the developer implements the serious game application in Lua. In addition, the developer implements a simplified Blockly interface so that therapists can customize the serious game in a simplified manner, without changes to the Lua code.

To test Proteo and to collect preliminary regarding about the usage of the framework, we prepared two serious game examples, namely GymTetris and Problem Solving Task, which are presented in the following section.

2.1.1. GymTetris

GymTetris is based on the popular Tetris game (https://en.wikipedia.org/wiki/Tetris, accessed on 11 May 2021) and was conceived to encourage physical exercise. This serious game follows the dynamics of the classic game, but the user moves or rotates the tetrominoes by means of body movements. A scheme of the Proteo modules used to develop the game is shown in Figure 2.

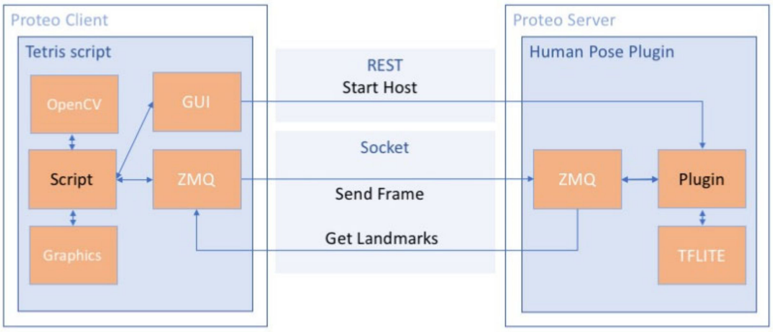
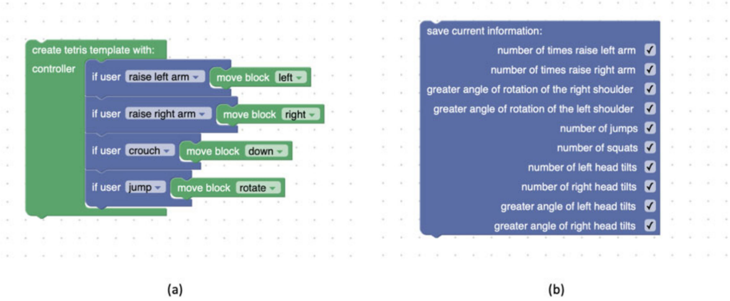


Figure 2. GymTetris scheme.

As previously discussed, the game template provides a base that the therapist can customize while considering the needs of the user to which the game will be deployed. In the case of GymTetris, the game can be customized in terms of user interaction and data collection: the therapist can choose which body movements must be performed by the user to interact with the game by using the Blockly [28] editor (Figure 3a). Moreover, the therapist can decide which user data to collect during the game sessions (Figure 3b).



```
create tetris template with:
controller  if user raise left arm ▾   move block left ▾
            if user raise right arm ▾  move block right ▾
            if user crouch ▾           move block down ▾
            if user jump ▾             move block rotate ▾
```

(a)

```
save current information:
                        number of times raise left arm ☑
                       number of times raise right arm ☑
greater angle of rotation of the right shoulder ☑
  greater angle of rotation of the left shoulder ☑
                                   number of jumps ☑
                                   number of squats ☑
                           number of left head tilts ☑
                          number of right head tilts ☑
                      greater angle of left head tilts ☑
                     greater angle of right head tilts ☑
```

(b)

Figure 3. Blockly interface to customize the correspondence between user movements and game actions (**a**) and to select data to store during game sessions (**b**).

Figure 4 shows a screenshot of the game (you can also find the game implementation on GitHub: https://massimobernava.github.io/proteo/, accessed on 11 June 2021) (for more information, see Supplementary Materials section). To detect user movements, the game incorporates a pose estimation model [29].

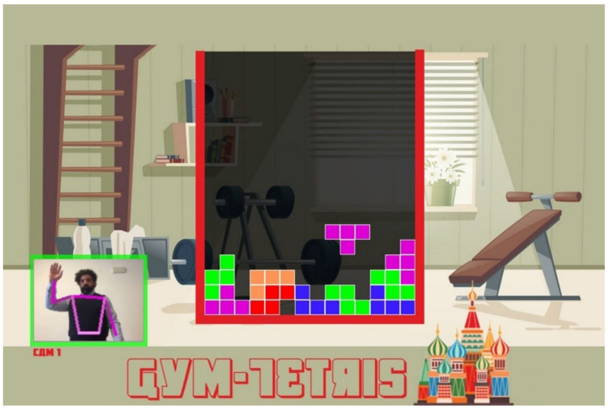


Figure 4. GymTetris game screenshot. Users can move blocks by moving her/his body. To detect user movements, the game incorporates a pose estimation model.

2.1.2. Problem Solving Task

Problem Solving Task is a game in which the user has to solve cognitive tasks, such a series completion. Moreover, the user can interact with an avatar represented by an animated character that speaks to foster the user's attention. In Figure 5, a scheme of the Proteo modules involved in the game is shown, while in Figure 6, a game screenshot is presented.

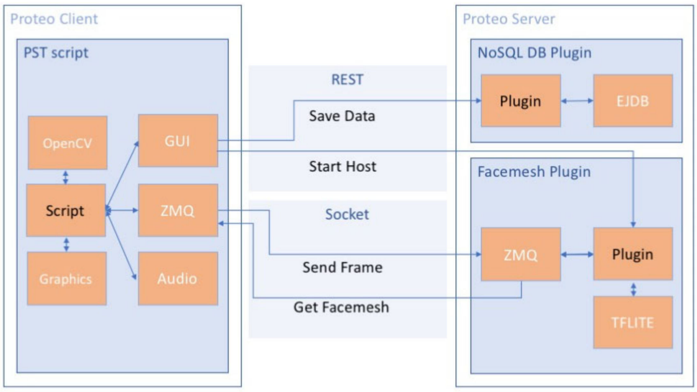


Figure 5. Problem solving task game scheme.

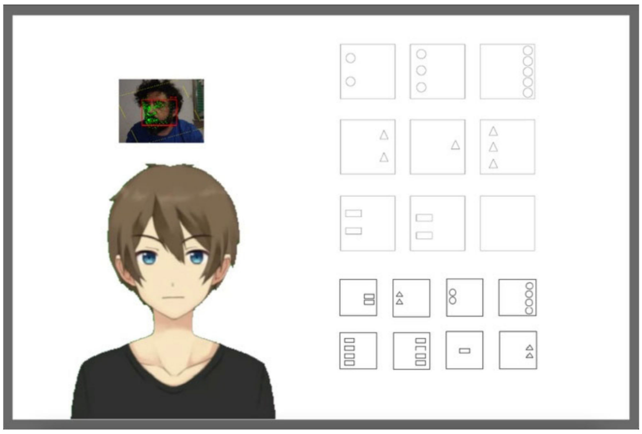


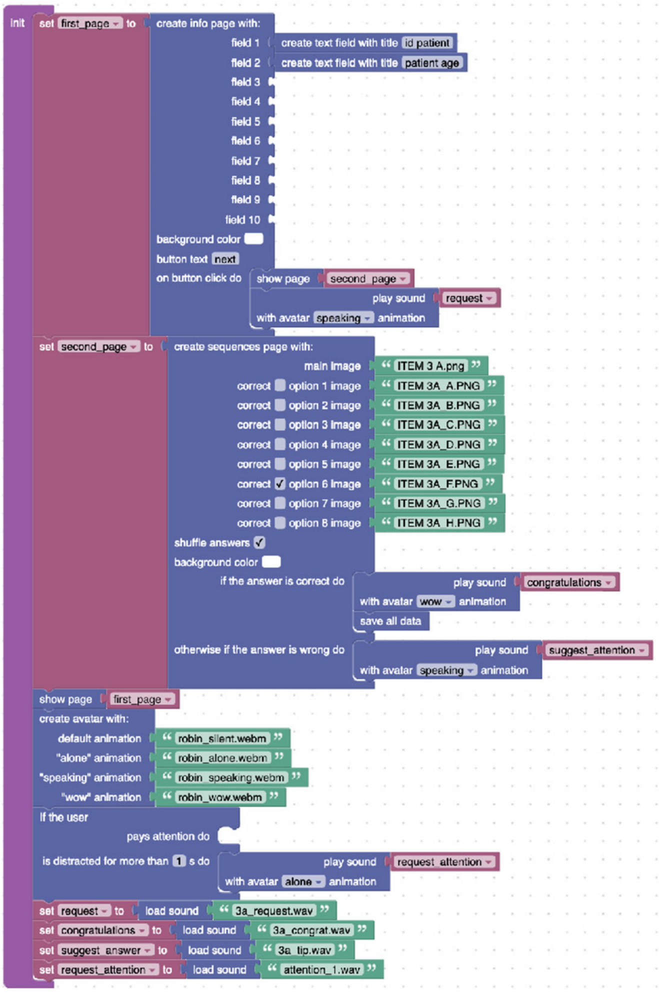
Figure 6. Problem solving task game screenshot. The avatar interacts depending on the user's attention. The user's attention is monitored through a face detection algorithm.

In Problem Solving Task, four main blocks are provided for game customization (Figure 7):

- Questionnaire page creation;
- Cognitive task page creation;
- Avatar creation;
- User attention trigger.

The questionnaire page is a page where different types of questions can be entered and contains a button that can trigger a customized action (e.g., continue to the next page or save the data and conclude the test).

The task page includes the presentation of a main image and nine images from which the user must choose the correct image to complete the exercise. The block allows the therapist to select which answers are correct and optionally shuffle the answers before displaying them. Depending on the correct or incorrect answer, the therapist selects the consequent action (e.g., an animation by the avatar).



```
init  set first_page to  create info page with:
                              field 1  create text field with title id patient
                              field 2  create text field with title patient age
                              field 3
                              field 4
                              field 5
                              field 6
                              field 7
                              field 8
                              field 9
                              field 10
                         background color
                         button text next
                         on button click do  show page  second_page
                                                          play sound  request
                                             with avatar speaking animation
      set second_page to  create sequences page with:
                                              main image  " ITEM 3 A.png "
                          correct □ option 1 image  " ITEM 3A A.PNG "
                          correct □ option 2 image  " ITEM 3A B.PNG "
                          correct □ option 3 image  " ITEM 3A_C.PNG "
                          correct □ option 4 image  " ITEM 3A_D.PNG "
                          correct □ option 5 image  " ITEM 3A E.PNG "
                          correct ☑ option 6 image  " ITEM 3A_F.PNG "
                          correct □ option 7 image  " ITEM 3A_G.PNG "
                          correct □ option 8 image  " ITEM 3A H.PNG "
                          shuffle answers ☑
                          background color
                             if the answer is correct do            play sound  congratulations
                                                          with avatar wow animation
                                                          save all data
                          otherwise if the answer is wrong do      play sound  suggest_attention
                                                          with avatar speaking animation
      show page  first_page
      create avatar with:
         default animation  " robin_silent.webm "
         "alone" animation  " robin_alone.webm "
      "speaking" animation  " robin_speaking.webm "
           "wow" animation  " robin_wow.webm "
      If the user
                      pays attention do
       is distracted for more than 1 s do          play sound  request_attention
                                          with avatar alone animation
      set request to  load sound  " 3a_request.wav "
      set congratulations to  load sound  " 3a_congrat.wav "
      set suggest_answer to  load sound  " 3a_tip.wav "
      set request_attention to  load sound  " attention_1.wav "
```

Figure 7. Blocky interface to customize the Problem Solving Task game.

Through the "show page" block, the first page to be displayed is selected, and it is possible to create a sequence of pages to be shown.

The avatar has four possible animations. The first animation is performed cyclically by the avatar (where there is no activity). The other three animations can be activated according to the therapist's settings by combining them with sounds or spoken sentences.

The last block allows the therapist to select which actions to take if the user is distracted or attentive (e.g., an animation by the avatar).

In this case, the therapist can choose the images needed for a specific task and how to analyze user performance and collect data but can also decide how and if the avatar should interact with the user. Here, the user interaction to choose the correct image to complete the series is conventionally based on conventional mouse interaction, however, the actions of the avatar can be activated by more natural input modalities, for example, by the user's level of attention. This kind of avatar interactivity has indeed been proven to be very important to enhance human–computer interaction [30], especially for users with an attention deficit [31].

In the following section, we present our findings of the User Satisfaction Evaluation Proteo Questionnaire, both from a group of therapists and end users.

2.2. Sample and Recruitment

A convenience sample of specialized therapists was recruited from the Department of Medicine and in the hospital health units at the University of Messina and the San Raffaele Hospital of Milan to test the Problem Solving Task example for both involvement and suitability. The inclusion criteria were to have at least 3 years of experience with rehabilitation training. No attention was paid to previous experience with the use of computers or of computer apps. Twenty potential participants were approached personally or by phone. Nine of them refused to participate, and eleven were included. The final sample was composed of 11 specialized therapists in cognitive and motor training for people with disabilities. They had a mean age of 31.48 with a range of 26–43; ten participants were female and one was a male. Another convenience sample of end users was also recruited to test the Proteo modular framework to test GymTretis. The nine end users had a mean age of 32.08 with a range of 25–44; four participants were female and five were male.

2.3. Outcome Measures

In this study, the User Satisfaction Evaluation Proteo Questionnaire (USEPQ) was used. USEPQ is based on USEQ [32], but it was specifically designed to test the Proteo system (see Table 1). Essentially, USEPQ extends USEQ, including questions to get responses from therapists on items related to Proteo. The Core Module measures the users' previous knowledge and experience with applications such as Proteo through different components (see Table 1), the level of involvement with the components (with negative affect and positive affect), and the level of suitability of the system and the tasks, with a total 18 items.

Table 1. The USEPQ (User Satisfaction Evaluation Proteo Questionnaire).

Question			Response		
			Not at All–Very Much		
Previous knowledge/experience					
Q1. Please indicate your level of computer experience	1	2	3	4	5
Q2. Indicate your level of experience with the apps on your computer	1	2	3	4	5
Q3. Have you ever used a system similar to Proteo?		No		Yes	
Q4. Indicate your level of knowledge of serious games	1	2	3	4	5
Q5. Indicate your level of knowledge in creating serious games	1	2	3	4	5
Q6. Indicate your level of knowledge of the security/privacy	1	2	3	4	5
Level of involvement					
Q7. I enjoyed trying Proteo	1	2	3	4	5
Q8. I felt involved	1	2	3	4	5
Q9. I felt tired	1	2	3	4	5
Q10. I felt bored	1	2	3	4	5
Level of adequacy/suitability					
Q11. Were you successful using the system?	1	2	3	4	5
Q12. Were you able to control the system?	1	2	3	4	5
Q13. Is the information provided by the system clear?	1	2	3	4	5
Q14. Do you think that this system will be helpful for telerehabilitation?	1	2	3	4	5
Q15. How difficult did you find the system?	1	2	3	4	5
Q16. How difficult did you find the task?	1	2	3	4	5
Q17. Did you feel discomfort during your experience with the system?	1	2	3	4	5
Q18. If yes, why?					

USEQP for therapists includes 18 questions, 16 of them with a response graded on a 5-point Likert Scale (from 1 = "not at all" to 5 "very much") and a final open question. Following the USEPQ scheme, the first six questions measure the users' previous knowledge and experience with apps such as Proteo, with computers, the knowledge of serious games, and the capacity to create and develop software. The next four questions measure the involvement of the user (if he enjoys using Proteo, if he feels involved, if he feels bored or tired). The third group, composed of eight questions, evaluates the level of suitability of Proteo. In this group, the first question (Q1) evaluates the successfulness in using Proteo, the second (Q2) evaluates the level of control, the third (Q3) evaluates the clearness, Q4 evaluates the level of helpfulness, Q5 and Q6 evaluate the perceived difficulty of the task and of the system, Q7 evaluates discomfort, and the last question, (Q8), requires a final explanation open answer. The global score of the first subscale ranges from 5 (poor previous knowledge) to 25 (excellent previous knowledge). Before calculating the global score of the second subscale, Q9 and Q10 have to be reversed, and the global score ranges from 4 (poor level of involvement) to 20 (high level of involvement). Before calculating the global score of the third subscale, Q15, Q16, and Q17 have to be reversed, and the global score ranges from 7 (poor suitability) to 35 (excellent suitability).

The USEPQ for end users only includes nine questions (Q1–Q4, Q6–Q10).

2.4. Data Analysis

Because of the ordinal nature of the measures, non-parametric statistical tests were used for data analyses. Hence, the median was used as a measure of central tendency and the interquartile range (IQR) for dispersion. To test for differences across conditions, the Wilcoxon signed-rank test was used for pairwise comparisons, with significance values adjusted by the Bonferroni correction. The effect size was calculated by dividing the absolute (positive) standardized test statistic z by the square root of the number of pairs:

$$d = \frac{z}{\sqrt{n}}.$$

Data were analyzed using IBM Statistics for Mac, Version 20.0 (IBM Corp., Armonk, NY, USA).

3. Results

Results are discussed first with reference to the therapists and later with reference to the end users. The three sections of Proteo related to the users' previous knowledge, the involvement of the user, and the level of suitability were analyzed for the therapists, and the two sections on previous knowledge and involvement were analyzed for end users.

3.1. Users' Previous Knowledge

Concerning Users' previous knowledge, comparisons with the Wilcoxon signed-rank test revealed that Serious game creation knowledge was significantly lower than Computer experience (z = −2.99; $p < 0.011$, d = 0.84), Apps experience (z = −3.02; $p < 0.001$, d = 0.94) and Security rules knowledge (z = −2.77; $p < 0.011$, d = 0.77).

In Table 2 and Figure 8, all conditions (Computer experience, Apps experience, Serious game knowledge, Security rules knowledge) were reported as presenting an intermediate level of Knowledge (out of 3 points). Their level of Serious game creation knowledge was considered low.

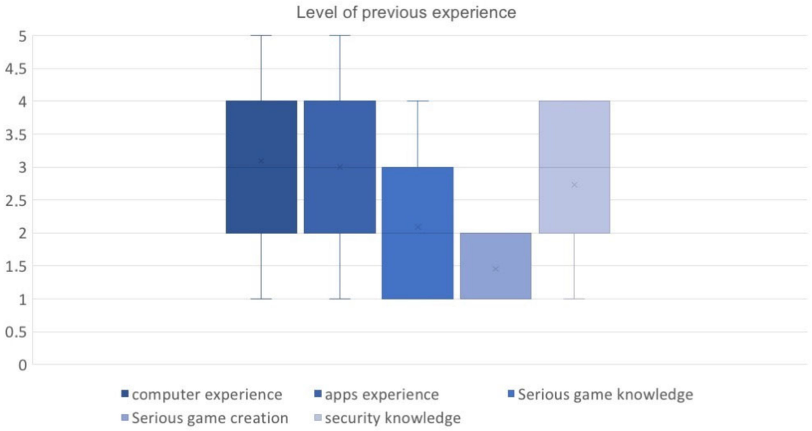


Figure 8. Boxplots of the components of the level of previous experience.

Table 2. Medians and interquartile range in the Proteo experience questionnaire components.

Previous Experience	Median	Interquartile Range
Computer experience	3	2–4
Apps experience	3	2–4
Serious game knowledge	2	1–3
Serious game creation	1	1–2
Security rules knowledge	3	2–4
Level of involvement		
Level of enjoyment	4	4–5
Level of involvement	4	4–5
Level of tiredness	2	1–3
Level of boredom	1	1–2
Level of suitability		
Successfulness	4	3–4
Ability to control	3	3–4
Clarity	4	3–4
Helpfulness	4	3–5
Difficulties with the system	3	2–3
Difficulty with the task	2	2–3

3.2. *Involvement of the Therapist Users*

Regarding the user's level of involvement (Figure 9), the conditions of positive feeling (level of enjoyment and level of involvement) were reported as high (out of 4 points) while the conditions of negative feeling were reported as low (1 or 2). Concerning engagement (Table 1), in all conditions, pairwise comparisons revealed that enjoyment and involvement were significantly higher than tiredness and boredom (enjoyment vs. tiredness: (z = −2.725; $p < 0.009$, d = 0.86), enjoyment vs. boredom: (z = −2.971; $p < 0.009$, d = 0.94), involvement vs. tiredness: (z = −2.625; $p < 0.009$, d = 0.83), involvement vs. boredom: (z = −2.873; $p < 0.004$, d = 0.90).

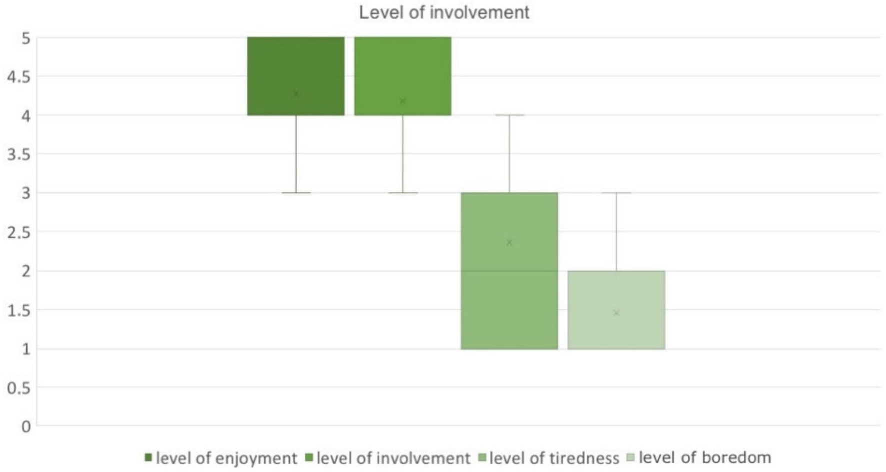


Figure 9. Boxplots of the components of the level of user's involvement.

3.3. *Level of Suitability*

Concerning the user level of suitability (Figure 10), the conditions of successfulness, ability to control, clarity, and helpfulness were reported as high (between 3 and 5), while the difficulty of the system and the difficulty of the task were reported as low. Pairwise comparisons with the Wilcoxon signed-rank test revealed that the first four conditions were significantly lower than the last two: difficulty of the task with successfulness, ability to control, clarity, and helpfulness, respectively: z = −1.951; $p < 0.09$, d = 0.51; z = −1.54; $p < 0.07$, d = 0.49, z = −1.799; $p < 0.05$, d = 0.59 and z = −2.36; $p < 0.009$, d = 0.75, and the difficulty of the system with successfulness, ability to control, clarity, and helpfulness, respectively: z = −1.83; $p < 0.09$, d = 0.53; z = −1.49; $p < 0.08$, d = 0.49, z = −1.83; $p < 0.03$, d = 0.61 and z = −2.41; $p < 0.009$, d = 0.81.

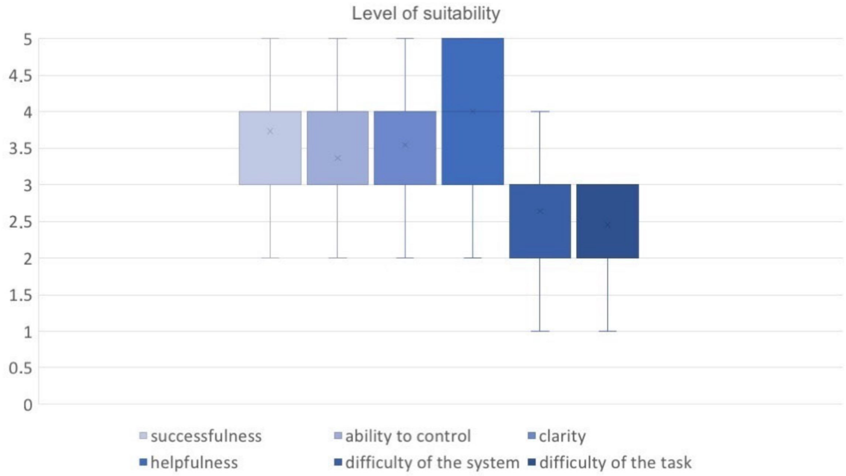


Figure 10. Boxplots of the components of the level of user's level of suitability.

3.4. *End Users' Previous Knowledge*

Concerning pervious user knowledge (Table 3 and Figure 11), all conditions (computer experience, apps experience, serious game knowledge, and security rules knowledge) were reported as presenting a high level of knowledge (out of 4/5 points). The level of serious game knowledge was low (2 points). Pairwise comparisons with the Wilcoxon signed-rank test revealed that serious game knowledge was significantly lower than computer experience z = −2.51; $p < 0.009$, d = 0.83.

Table 3. Medians and interquartile range in the Proteo experience questionnaire components.

Previous Experience	Median	Interquartile Range
Computer experience	5	3–5
Apps experience	4	1–5
Serious game knowledge	3	1–5
Security rules knowledge	2	3–5
Level of involvement		
Level of enjoyment	4	4–5
Level of involvement	3	3–5
Level of tiredness	1	1–1
Level of boredom	2	1–3

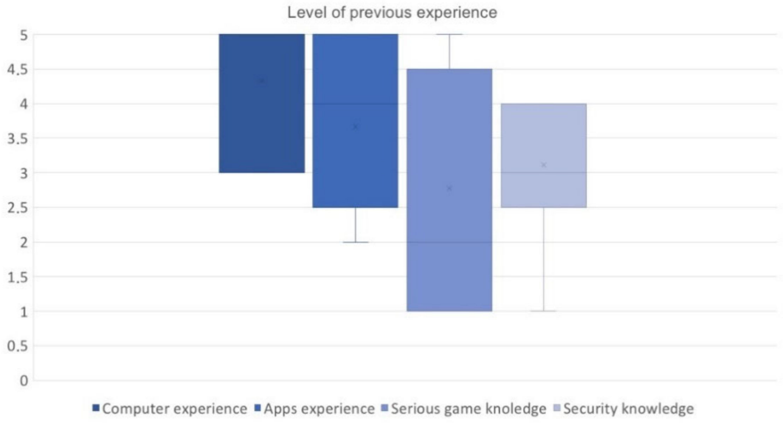


Figure 11. Boxplots of the components of the level of the user's experience.

3.5. Involvement of the User

Regarding the level of user involvement (Table 2 and Figure 12), the conditions of positive feeling (the level of enjoyment and level of involvement) were reported as high (out of 4 points) while the conditions of negative feeling (boredom and tiredness) were reported as low (1 or 2). Concerning engagement (Table 2), in all conditions Pairwise comparisons revealed that enjoyment and involvement were significantly higher than tiredness and boredom (enjoyment vs. tiredness: (z = −2.99; $p < 0.001$, d = 0.94), enjoyment vs. boredom: (z = −2.771; $p < 0.009$, d = 0.87), involvement vs. tiredness: (z = −2.92; $p < 0.009$, d = 0.90), involvement vs. boredom: (z = −2.673; $p < 0.004$, d = 0.88).

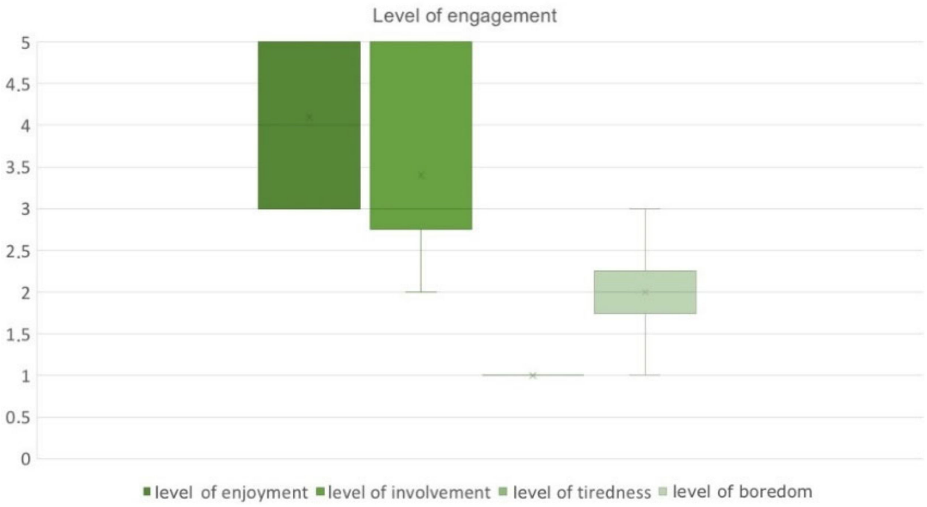


Figure 12. Boxplots of the components of the end user's level of engagement.

4. Discussion

We then tested the process of game customization by therapists and analyzed the levels of experience, involvement, and suitability for both therapists and end users. Data from the different components of the User Satisfaction Evaluation Proteo Questionnaire assessed the impact that the two game experiences had on participant engagement. Overall and irrespective of the game mode, participants reported low feelings of boredom and tiredness, i.e., low levels negative affect and highs level of positive affect (involvement and enjoyment). Irrespective of previous experience level, the suitability with the conditions of successfulness, ability to control, clarity, and helpfulness were also reported as high.

More in depth, with reference to the therapists, before starting the study, we wanted to refer to a sample of therapists who were not very experienced in the use of the PC or in the use of the different apps; the results concerning the therapists' previous knowledge confirmed these data, as the therapists show an intermediate level of knowledge and, as we expected, their experience in creating a serious game was very low. With reference to the therapists' level of involvement, the condition of positive feeling was reported as high while the condition of negative feeling was reported as low. With reference to users' previous knowledge, the sample we contacted to participate in the research had a high level of knowledge, and their level of involvement was also very high. We believe that this positive impact of the system on the subjects' feelings was mainly due the easiness and the fun of the system. This was confirmed by the data related to the level of suitability described by the therapists. As we underlined, the therapists were able to finish the assigned task, and they reported a strong feeling of being in control of the process and that the Proteo app was clear and helpful.

In clinical practice, frameworks such as Proteo can enable simultaneous training of physical, cognitive, and psychological aspects and show great potential to improve patient motivation and adherence, mainly due to the possibility of gamification and personalizing the feedback from TR modalities that can be customized by the therapist across subgroups of patients. In order to analyze the suitability of the system, our results are promising; in order to analyze the safety of Proteo app, we have no data because nobody replied in an affirmative way to the question: "Did you feel discomfort during your experience with the system?" For this reason and because no therapist reported problems during the task, we believe that Proteo is safe. However, in order to systematically adopt Proteo in routine clinical practice, the ease with which the therapists completed the assigned task and the pleasure of creating new tasks are not sufficient for the success of the therapy. It is necessary that the therapists are well-trained. They must be competent when determining what feedback to give and at what moment, they and must be aware of the hierarchy of learning objectives, the transfer of them, and all of the factors that can lead to the autonomy the patient. The therapists also need to be careful not to give patients more help than necessary.

This study has some limitations that should be acknowledged. Regarding the data analysis, our sample size is relatively small for some of the statistical analyses performed, and the results should be considered with caution. However, sample size calculations determined that the sample size was sufficient to evaluate the study objectives.

5. Conclusions

Telerehabilitation systems have been widely used in the recent years, showing their crucial importance during the COVID-19 pandemic. In the field of telerehabilitation, serious games play an important role since they can foster user engagement and hence, enhance rehabilitation outcomes. However, telerehabilitation systems based on serious games are often suited for specific patient conditions, so implementing new games or even customizing existing ones is resource demanding and requires software development skills. Moreover, several libraries exist to ease game development, but they lack in offering advanced user interaction methods, such as those that use computer vision algorithms or eye-tracking devices.

In this paper, we presented Proteo: an open source, modular framework for developing serious games from scratch, providing a high-level interface for serious game development and customization. Proteo provides many built-in modules to cover most common features that a serious game may need, such as client/server communication, user management, data management, advanced human-computer interaction, computer vision, or deep learning techniques.

Proteo has proven to be a useful tool to develop serious games for telerehabilitation and has been well accepted both by the therapists and by the users we involved in the evaluation of suitability questionnaire.

As for future work, we plan to conduct more extensive tests involving a larger group of therapists and rehabilitation researchers. Moreover, we plan to use Proteo in structured telerehabilitation programs involving people with disabilities.

Another step is to explore other varieties of games, such as complex task units (such as with math problems). Finally, we consider it important to understand if the obtained results will be similar with larger group sizes and will also investigate the applicability of Proteo to different settings such as in families, schools, and hospitals.

Supplementary Materials: Proteo is available on GitHub under the MIT License at https://massimobernava.github.io/proteo/, accessed on 11 June 2021.

Author Contributions: Conceptualization, G.B. and A.N.; methodology, G.B., A.N., R.A.F. and G.I.; software, G.B.; validation, G.B., A.N., R.A.F. and G.I.; writing—original draft preparation, G.B., A.N., R.A.F. and T.C.; writing—review and editing, A.N., G.B., R.A.F. and T.C.; visualization, G.B. and A.N.; All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Ethical review and approval were waived for this study, because the authors did not involve patients in a telerehabilitation program. In this study only therapists and researchers were involved to assess the suitability of the presented software.

Informed Consent Statement: Informed consent was obtained from all subjects involved in the study.

Data Availability Statement: Publicly available datasets were analyzed in this study. This data can be found here: https://github.com/massimobernava/proteo/tree/master/data accessed on 11 May 2021.

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

Appendix A

Appendix A.1. Proteo Architecture

Developing a cross-platform, scriptable solution for multimedia applications even with computer vision and deep learning support is a challenging task since many different components have to be connected and tuned to reach a robust yet performative system. To tackle the complexity of the needed features, Proteo has a highly modular structure in which every aspect can be easily and independently modified.

Proteo consists of two main applications, the Proteo Server and the Proteo Client, both of which are written in C using libraries that can be ported to the main hardware and software platforms. At the heart of the system lies a Lua interpreter (LuaJIT.org) with an API designed specifically for Proteo. The use of pure C and LuaJIT allows for considerable execution speed. Furthermore, on the server side, it is possible to extend the functionality of the system by installing Lua modules through package managers such as LuaRocks (LuaRoks.org).

Each client-side application consists of a script, which may require various libraries that are also loaded remotely. The script connects to the server via one or more plugins. Both the scripts and the plugins are written in Lua. The client and server communicate through an API Rest interface and sockets. A typical message exchange between the client and server is shown in Figure A1.

Proteo architecture supports both multi-client and multi-server configuration. The multi-server system has a star architecture, with a main server as the center of the star and several ancillary servers that are periodically queried by the main server. When a client logs in, the main server checks which resources are required and responds to the client by specifying which server (including itself if necessary) is needed to address each request. In order to manage the workload independently, the main server uses a simple ticketing system: when querying the secondary servers, the main server queries each server for available tickets. By considering which server has the higher value of available tickets and load distribution, the main server assigns the resource request to a secondary server and sends the client the address of that server. Due to the scriptable nature of Proteo, the ticketing system can be easily modified if a more complex algorithm is needed.

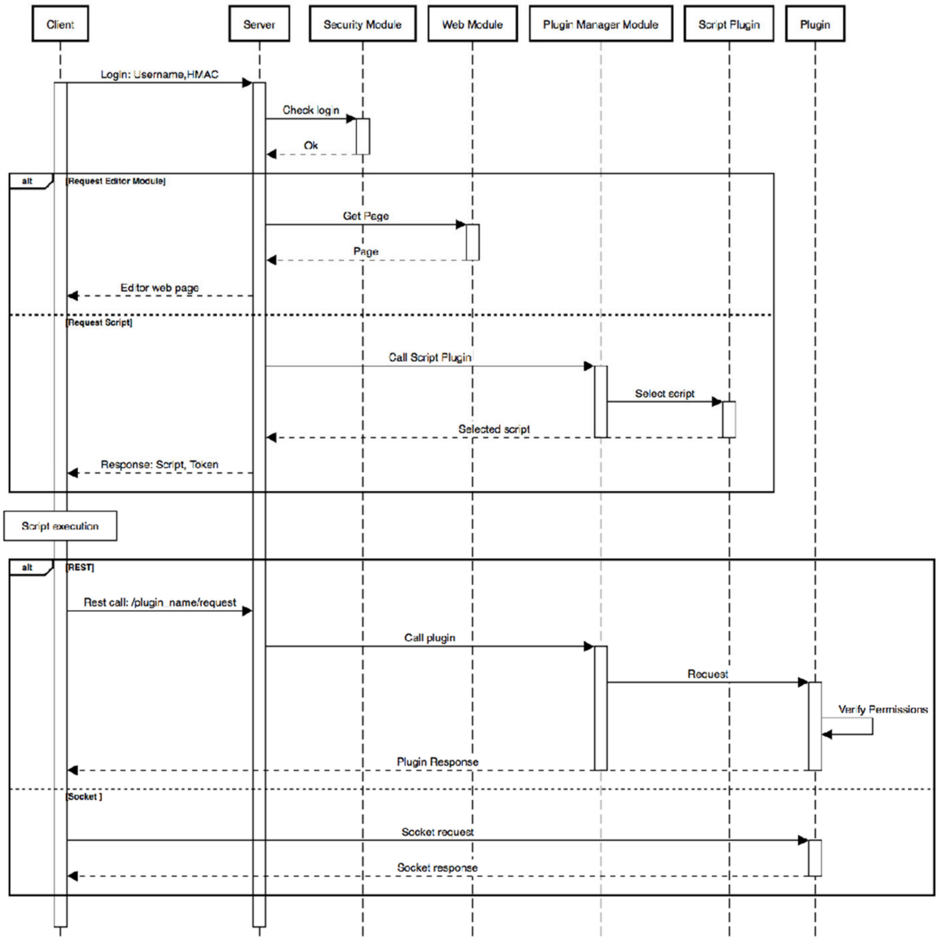


Figure A1. A typical message exchange between the client and server involving several Proteo modules.

Appendix A.2. Server Application

Each Proteo server exposes a REST interface with base endpoints (login/admin/edit module) and specific endpoints, depending on the active plugins. Each plugin operates independently but requires at least two GET endpoints: plugin_name/info and plugin_name/permissions.

The first provides service information: the plugin version, whether the plugin is working, and the number of tickets it offers.

The second returns an array of strings, with each string being a permission that the plugin exposes. The plugin computation is transparent to Proteo, which will only take care of assigning permissions to individual users. The server application is composed of several modules that will be briefly described in the following.

Security module. Proteo implements a security mechanism in every REST call that consists of two components:

- A call-based authentication system, based on keyed-hash message authentication codes (HMAC) and tokens;
- A plugin-oriented permission management.

During the login process, the client generates an HMAC using the password entered by the user, and the client tries to generate the same HMAC using the password stored locally; if the two HMACs match, the server generates a token and sends it to the client. All subsequent client requests must contain an HMAC generated with a common key known to both the client, the server, and the token (see Figure A2).

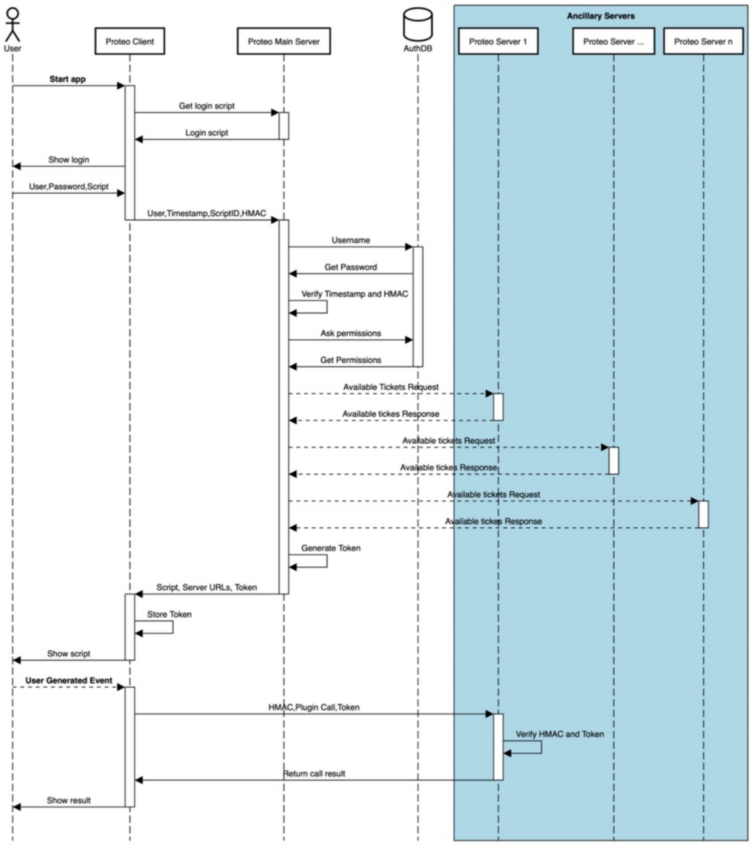


Figure A2. Proteo login sequence diagram.

The token is generated by encrypting (with a key owned by the server) information on the user’s identity and the token expiring date so that the client can connect to several servers without storing copies of the user’s authorization to each server.

Because of the scriptable nature of server-side plugins, it was necessary to develop a flexible structure for permissions. In practice, each plugin exposes a list of permissions to the server. These permissions are transparent to the server, but they are assigned to users via the admin module. At each REST call, the server passes the user ID to the plugin and the list of permissions that the user has. The plugin will then act appropriately according to the received permissions.

Login module. A login request from the client is made via the [POST/login] call, this call can have two results:

- If the credentials are missing (or incorrect), a Lua script is returned that allows a login page to be displayed with the possibility of registering or recovering a lost password. This interface is written using the client’s graphical API and can be modified to adapt it graphically and functionally to requirements;
- The login is successful, and the requested script is returned (together with the token, see security module).

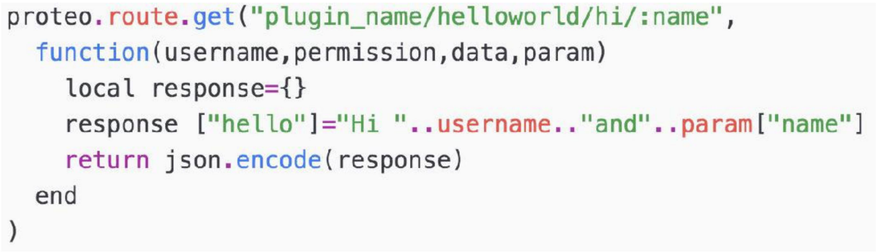
Admin module. An administration interface can be accessed through a [GET/admin] call or through a login request. This script-based interface can also be modified, but to keep the main part of the implementation constant, the administration functions are part of a library. In addition, the library makes it transparent to the plugin which method is used to save the administration information. In the current version, the administration database is stored via SQLite. Through the administration module it is possible:

- To activate or deactivate plugins that the server loads at start-up;
- To create and deactivate users;
- To change user passwords;

- To edit user permissions.

During installation, there is only one user 'admin' with a password 'admin' who can access the script 'admin'. As a result, after the first installation, it is possible to add users and manage permissions.

Plugin manager module. Each plugin consists of several access endpoints, such as the one shown in Figure A3:



```
proteo.route.get("plugin_name/helloworld/hi/:name",
  function(username,permission,data,param)
    local response={}
    response ["hello"]="Hi "..username.."and"..param["name"]
    return json.encode(response)
  end
)
```

Figure A3. Proteo endpoint example.

The routing system is very simple. Each endpoint is defined by a verb (in the example: GET), a path, and a function. The path can contain variables (in the example: name). The function arguments are the username, the user permissions, the data passed during the function call, and the parameters passed by the path. Conventionally, the function returns a JSON string, but any string is allowed.

Script plugin. A Proteo standard plugin is the plugin that manages the sending of scripts to clients. The main access points it exposes are:

- [GET proteo/permissions]: which returns the permissions needed to access the individual scripts;
- [GET proteo/script/:script]: which checks whether the user has permission to access the script and if so, returns it encapsulated with JSON;
- [GET proteo/lib/:lib]: which returns the requested library to the client. Libraries are not handled with permissions, but there is an array in the script that identifies which libraries a client can request.

Editor. There are three types of script files in Proteo: plugin, script, and lib. Every type of script is written in Lua and is stored in specific folders within the server root folder. Given the aim of making Proteo easy to program, in addition to being able to edit scripts manually, it is possible to act at a higher level of abstraction through visual programming.

Using Blockly [28], it is possible, for instance, to translate the function in the previous figure (Figure A4) into a block in the following figure:

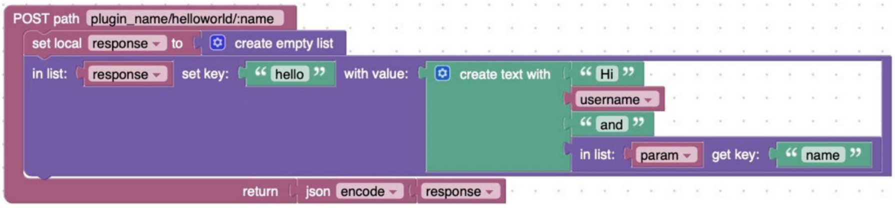
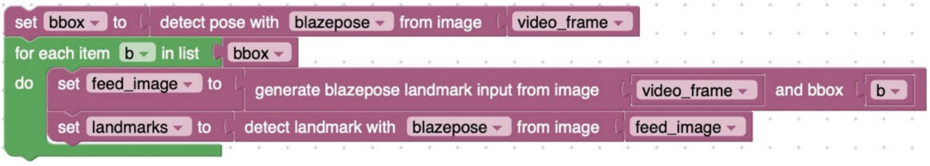


Figure A4. Blockly example for a simple function.

Proteo is equipped with an internal web-based editor to create or edit scripts and plugins. Through the endpoint [GET/admin], a user can authenticate (with editor permission) and can access the files for which he/she has reading/writing permissions. The script is immediately available once saved. To the contrary, saving a plugin restarts the server, so that it can be reloaded. Access and modification of script files is also managed by a server plugin.

All API blocks have been implemented but, in order to optimize visual programming, the aim is to further abstract the programming level through the use of libraries.

For example, to estimate the pose from a frame using the Blazepose algorithm [29], an appropriate library has been created, making the blocks extremely readable and easy to implement, as shown in Figure A5:



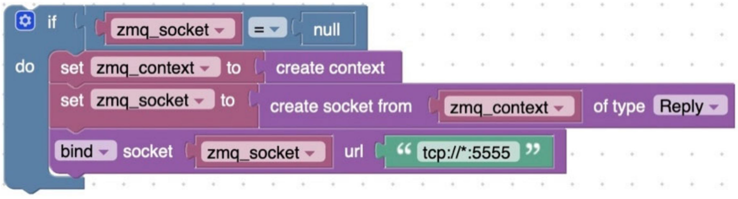
```
set bbox to detect pose with blazepose from image video_frame
for each item b in list bbox
do  set feed_image to generate blazepose landmark input from image video_frame and bbox b
    set landmarks to detect landmark with blazepose from image feed_image
```

Figure A5. Blokly example for pose estimation using Blazepose.

Even if Blockly is capable of reproducing each of the Lua language constructs, it is specifically intended for users with no development skills. In the context of telerehabilitation, we considered therapists or researchers that often need to set up task batteries (in the form of serious games), which is typically unfeasible without support from developers. With Blockly, starting from a serious game template, we give them the possibility to overcome technical skill barriers and to reach autonomy. In Section 2.2. we present some examples of the process of building a serious game starting from a template.

API. Proteo API works as a general-purpose glue and wraps together several libraries comprising all parts of memory management, garbage collection, and data interconnections transparent to the developer. There are system functions for accessing the disk, using timers, etc., which we will not go into detail about here, but we will instead concentrate on listing the parts of the library related to the development of serious games.

Net. Through the networking functions, it is possible to make REST calls to specific addresses, and it is also possible to carry atomic messages across TCP/UDP, with patterns like fan-out, pub-sub, and request-reply. To do this, Proteo incorporates the functionalities of the most popular networking libraries with a focus on portability: OpenSSL and ZeroMQ are just some of the libraries integrated into the Proteo API (Figure A6):



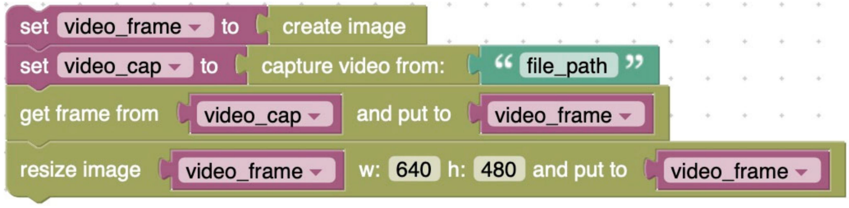
```
if  zmq_socket = null
do  set zmq_context to create context
    set zmq_socket to create socket from zmq_context of type Reply
    bind socket zmq_socket url " tcp://*:5555 "
```

Figure A6. Blockly example for networking functions.

Communication between the client and the server is conventionally via JSON-encoded strings, even if this is not an implementation constraint.

In the context of serious games for telerehabilitation, network support is crucial, for instance, for user authentication, multi-user support, and data collecting. Moreover, a tele-rehabilitation session may be intended as an autonomous activity by the patient that follows a specific protocol and that can generate data about users that are further analyzed by therapists. In this case, network support is useful for user authentication and data collection. Nevertheless, another kind of telerehabilitation session may involve the therapist, which monitors the ongoing user activities and can react in certain given circumstances. Proteo is designed to give support to both of these scenarios.

Computer Vision. In the field of computer vision, the reference libraries are OpenCV and FFMPEG. The use of Blockly simplifies coding (Figure A7).



```
set video_frame to create image
set video_cap to capture video from: " file_path "
get frame from video_cap and put to video_frame
resize image video_frame w: 640 h: 480 and put to video_frame
```

Figure A7. Blockly example for computer vision support.

There is a module in OpenCV that implements forward pass (inferencing) with deep networks, pre-trained using some popular deep learning frameworks, such as Caffe. FFMPEG is mainly used for codec functions.

The client can send a video stream to the server. This stream can be managed to create a group video chat, to be analyzed with a deep learning algorithm, or both at the same time. The support for computer vision can be on both the client and server sides.

Database. Proteo integrates two database systems, namely SQLite and EJDB2, so that it is straightforward to add support for data storage and manipulation in a serious game. However, support for other DBMS can be provided by adding specific modules.

Deep learning. One of the central aspects of Proteo is the possibility of using deep learning techniques in a simple way. For this reason, in addition to OpenCV, there is also a section of the API that allows inference via Tensorflow Lite. Using deep learning models in serious games is crucial to add advanced human–computer interaction support. Models for pose estimation, face recognition, or emotion recognition can be easily added by means of predefined functions. For instance, in the serious games presented in Section 2.1, we exploit these modules to add pose estimation and face recognition capabilities to the games.

Client Application

Proteo is designed following the cloud computing paradigm. A Proteo client can be an empty container: the interface, multimedia, and computational resources lie on the server. During login, the client receives a “task”. This task may depend on the request, but it may also depend on the user’s permissions or other conditions. In addition, the client also receives the addresses of the secondary servers from the main server, thus transparently distributing the workload in the cloud. APIs that require the presence of a media file also check for the presence of the file locally and if not found, download it from the server transparently.

However, even if Proteo relies on the cloud paradigm, some elaborations can be demanded from the client, if needed. This is particularly useful, for instance, when serious games need human–computer interaction based on computer vision algorithms. In general, client elaboration can be used when it can give a more reactive interaction experience.

Client–API. Following the logic of cloud computing, the client-side API and libraries are oriented towards the human–machine interface. Net and computer vision APIs are also present with the same implementation to make the dialogue between the client and server simple.

Client–GUI. Proteo has a set of standard GUI elements: label, text field, checkbox, button, list, dropdown, form, and container. The API is designed to be simple and efficient to use, and each GUI element can be created using a single function.

Client–Graphics. Proteo is oriented towards the development of serious games, which is why in addition to the standard elements of a graphic environment, such as images, lines, rectangles, ellipses, and text, the system also has more complex elements: sprites, shapes, and skeletons.

Sprites are used for 2D animations created through a sequence of frames. During creation, a sprite sheet is specified from which subsequent frames will be cut out. Sprites are a valid technique for simple animations, but for more complex animations, a skeletal animation is required.

A shape element is a vector structure made up of a combination of two elementary objects: polygon and Bezier, the first of which is for drawing figures composed of straight lines, and the second of which is for drawing curves. A shape object can also be imported from an SVG file, allowing Proteo to manage vector graphics.

To animate a shape, we borrowed the technique of PoseAnimator [33]. In Proteo, this system has been generalized to apply a skeletal animation to any type of vector figure.

In Proteo skeletal animation system, the object to be animated is composed of:

- Shape: the drawing vector;
- Skeleton: a set of interconnected bones used for animation.

In Proteo, the shape object is composed of a set of shape objects and polygon objects. Shape objects are therefore used to group several polygons into a single shape.

A shape object can be loaded from an SVG file. A polygon object consists of an array of points that are drawn to form a closed polygon. To allow Proteo to draw curved figures as well, it is possible to create polygon objects by composing Cubic Bézier curves. Each Bezier curve is characterized by four points, namely P_0, P_1, P_2, and P_3. While the curve passes through points P_0 and P_3, points P_1 and P_2 are used to determine directions attracting the curve going from P_0 to P_3.

A skeleton, instead, is composed of a set of points called joints, these points are linked together by bones. In a skeleton, it is possible to freely move each joint. This will consequently modify the bones that are afferent to the moving joint. The transformation applied to the bone will reverberate in a weighted modification to the points of the shape to which the bone itself is connected. It is possible to assign each bone to a group of belonging through a generic ID, in this way, it is possible to limit the influence that a single bone has on different parts of the shape.

It is possible to save and load a skeleton through a Json-based file.

Proteo uses one of the most used binding algorithms to deform the shape using bone structure. This algorithm is called Linear Blend Skinning (LBS). This algorithm blends the transformations controlled by each bone individually.

The complete skeleton-shape binding process requires the following steps:

- For each polygon belonging to the shape to be connected, the nearest bone is calculated;
- Of the nearest bone, all bones belonging to the same group are taken;
- Weights are calculated for each bone belonging to this group.

The weights are automatically generated by calculating the impact that bone J has on point i:

$$w_{i,j} = \frac{1}{d^2}$$

where d is the distance between point i and the bone calculated during the binding phase. At the end of the calculation, the weights are normalized so that the sum of the weights of all of the bones affecting a single point are always equal to 1:

$$\sum_{j=0}^{m} w_{i,j} = 1$$

During the animation, after having made all of the the necessary movements of the joints, through an update function, all of the weighted offsets of the points of each polygon are calculated, and the shape is then ready to be drawn.

Client–Audio. A first feature of the audio module of the Proteo platform is the management of an internal software mixer that allows polyphonic playback (and possibly recording). From a logical point of view, the audio API is divided into two sections: one section allowing the processing of audio files in recording and playback, and another section oriented towards streaming. The management of audio files is fundamental for serious games in which a sound compartment made up of music and polyphonic effects can foster user engagement. For the transmission of audio streams and, if necessary, for their analysis by the server, functions which allow the creation of a stream, the possibility of encoding it, packaging it, and sending it via a messaging service are required (see API-Network).

After succinctly presenting the key components of the Proteo framework, in the following section, we discuss two serious game examples developed with Proteo. We adopted these serious game examples to analyze user satisfaction data regarding Proteo usage, which will be discussed in the Results Section.

References

1. Putrino, D. Telerehabilitation and emerging virtual reality approaches to stroke rehabilitation. *Curr. Opin. Neurol.* **2014**, *27*, 631–636. [CrossRef]
2. Dovigo, L.; Caprì, T.; Semino, M.; Nucita, A.; Giannatiempo, S.; Iannizzotto, G.; Fabio, R.A. School's out. Interactive tele-education for Rett patients at the time of COVID. *Prof. Psychol. Res. Pract.* **2021**, in press.
3. Brennan, D.M.; Mawson, S.; Brownsell, S. Telerehabilitation: Enabling the remote delivery of healthcare, rehabilitation, and self management. *Stud. Health Technol. Inform.* **2009**, *145*, 231–248. [PubMed]
4. Agostini, M.; Moja, L.; Banzi, R.; Pistotti, V.; Tonin, P.; Venneri, A.; Turolla, A. Telerehabilitation and recovery of motor function: A systematic review and meta-analysis. *J. Telemed. Telecare* **2015**, *21*, 202–213. [CrossRef]
5. Caprì, T.; Fabio, R.A.; Iannizzotto, G.; Nucita, A. The TCTRS Project: A Holistic Approach for Telerehabilitation in Rett Syndrome. *Electronics* **2020**, *9*, 491. [CrossRef]
6. Caprì, T.; Nucita, A.; Iannizzotto, G.; Stasolla, F.; Romano, A.; Fabio, R.A. Telerehabilitation for Improving Adaptive Skills of Children and Young Adults with Multiple Disabilities: A Systematic Review. *Rev. J. Autism Dev. Disord.* **2020**, *8*, 244–252. [CrossRef]
7. Cottrell, M.A.; Galea, O.A.; O'Leary, S.P.; Hill, A.J.; Russell, T.G. Real-time telerehabilitation for the treatment of musculoskeletal conditions is effective and comparable to standard practice: A systematic review and meta-analysis. *Clin. Rehabil.* **2017**, *31*, 625–638. [CrossRef]
8. Grona, S.L.; Bath, B.; Busch, A.; Rotter, T.; Trask, C.; Harrison, E. Use of videoconferencing for physical therapy in people with musculoskeletal conditions: A systematic review. *J. Telemed. Telecare* **2018**, *24*, 341–355. [CrossRef]
9. Knutsen, J.; Wolfe, A.; Burke, B.L.; Hepburn, S.; Hepburn, S.; Lindgren, S.; Coury, D. A systematic review of telemedicine in autism spectrum disorders. *Rev. J. Autism Dev. Disord.* **2016**, *3*, 330–344. [CrossRef]
10. Benham, S.; Gibbs, V. Exploration of the effects of telerehabilitation in a school-based setting for at-risk youth. *Int. J. Telerehabilitation* **2017**, *9*, 39–46. [CrossRef] [PubMed]
11. Golomb, M.R.; McDonald, B.C.; Warden, S.J.; Yonkman, J.; Saykin, A.; Shirley, B.; Huber, M.; Rabin, B.; AbdelBaky, M.; Nwosu, M.E.; et al. In-home virtual reality videogame telerehabilitation in adolescents with hemiplegic cerebral palsy. *Arch. Phys. Med. Rehabil.* **2010**, *91*, 1–8. [CrossRef]
12. Huber, M.; Rabin, B.; Docan, C.; Burdea, G.; Nwosu, M.E.; Abdelbaky, M.; Golomb, M.R. PlayStation 3-based tele-rehabilitation for children with hemiplegia. In Proceedings of the Virtual Rehabilitation, Vancouver, BC, Canada, 25–27 August 2008. [CrossRef]
13. Vallejo, D.; Gmez-Portes, C.; Albusac, J.; Glez-Morcillo, C.; Castro-Schez, J.J. Personalized Exergames Language: A Novel Approach to the Automatic Generation of Personalized Exergames for Stroke Patients. *Appl. Sci.* **2020**, *10*, 7378. [CrossRef]
14. Pereria, F.; Bermudez, I.; Badia, S.; Jorge, C.; Cameirao, M.S. The use of game modes to promote engagement and social involvement in multi-user serious games: A within-person randomized trial with stroke survivors. *J. Neuroeng. Rehabil.* **2021**, *18*, 62. [CrossRef]
15. Antón, D.; Berges, I.; Bermudez, J.; Goni, A.; Illarramendi, A. A Telerehabilitation System for the Selection, Evaluation and Remote Management of Therapies. *Sensors* **2018**, *18*, 1459. [CrossRef]
16. Rozevink, S.G.; van der Sluis, C.K.; Garzo, A.; Keller, T.; Hijmans, J.M. HoMEcare aRm rehabiLItatioN (MERLIN): Telerehabilitation using an unactuated device based on serious games improves the upper limb function in chronic stroke. *J. Neuroeng. Rehabil.* **2021**, *18*, 48. [CrossRef]
17. Guillén-Climent, S.; Garzo, A.; Muñoz-Alcaraz, M.N.; Casado-Adam, P.; Arcas-Ruiz-Ruano, J.; Mejías-Ruiz, M.; Mayordomo-Riera, F.J. A usability study in patients with stroke using MERLIN, a robotic system based on serious games for upper limb rehabilitation in the home setting. *J. Neuroeng. Rehabil.* **2021**, *18*, 41. [CrossRef] [PubMed]
18. Salgueiro, C.; Urrùtia, G.; Cabanas-Valdés, R. Available Apps for Stroke Telerehabilitation during Corona Virus Disease 2019 Confinement in Spain. *Disabil. Rehabil. Assist. Technol.* **2021**. [CrossRef]
19. Aderonmu, J.A. Emerging challenges in meeting physiotherapy needs during COVID-19 through telerehabilitation. *Bull. Fac. Phys. Ther.* **2020**, *25*, 16. [CrossRef]
20. Prvu Bettger, J.; Resnik, L.J. Telerehabilitation in the Age of COVID-19: An Opportunity for Learning Health System Research. *Phys. Ther.* **2020**, *100*, 1913–1916. [CrossRef]
21. Christy, B.; Keeffe, J. Telerehabilitation during COVID-19: Experiences in service delivery from South India. *Indian J. Ophthalmol.* **2020**, *68*, 1489–1490. [CrossRef] [PubMed]
22. Nuara, A.; Fabbri-Destro, M.; Scalona, E.; Lenzi, S.E.; Rizzolatti, G.; Avanzini, P. Telerehabilitation in response to constrained physical distance: An opportunity to rethink neurorehabilitative routines. *J. Neurol.* **2021**. [CrossRef]

23. Da Silva, T.D.; da Silva, P.L.; Valenzuela, E.J.; Dias, E.D.; Simcsik, A.O.; de Carvalho, M.G.; Fontes, A.; Alberissi, C.; de Araújo, L.V.; Brandão, M.; et al. Serious Game Platform as a Possibility for Home-Based Telerehabilitation for Individuals with Cerebral Palsy During COVID-19 Quarantine—A Cross-Sectional Pilot Study. *Front. Psychol.* **2021**, *12*, 622678. [CrossRef] [PubMed]
24. Castelli, I.; Antonietti, A.; Fabio, R.A.; Lucchini, B.; Marchetti, A. Do Rett syndrome persons possess Theory of Mind? Some evidence from not-treated girls. *Life Span. Disabil.* **2013**, *16*, 157–168.
25. Fabio, R.A.; Magaudda, C.; Caprì, T.; Towey, G.E.; Martino, G. Choice behavior in Rett syndrome: The consistency parameter. *Life Span. Disabil.* **2018**, *21*, 47–62.
26. Gangemi, A.; Caprí, T.; Fabio, R.A.; Puggioni, P.; Falzone, A.M.; Martino, G. Transcranial direct current stimulation (TDCS) and cognitive empowerment for the functional recovery of diseases with chronic impairment and genetic etiopathogenesis. *Adv. Genet. Res.* **2018**, *18*, 179–196.
27. Nucita, A.; Bernava, G.M.; Giglio, P.; Peroni, M.; Bartolo, M.; Orlando, S.; Marazzi, M.C.; Palombi, L. A Markov chain based model to predict HIV/AIDS epidemiological trends. *Lect. Notes Comput. Sci.* **2013**, *8216*, 225–236. [CrossRef]
28. Blokly. Available online: https://developers.google.com/blockly (accessed on 1 May 2021).
29. Google AI Blog—On-Device, Real-Time Body Pose Tracking with MediaPipe BlazePose. Available online: https://ai.googleblog.com/2020/08/on-device-real-time-body-pose-tracking.html (accessed on 1 May 2021).
30. Iannizzotto, G.; Lo Bello, L.; Nucita, A.; Grasso, G.M. A vision and speech enabled, customizable, virtual assistant for smart environments. In Proceedings of the 2018 11th International Conference on Human System Interaction, Gdansk, Poland, 4–6 July 2018; pp. 50–56. [CrossRef]
31. Fabio, R.A.; Caprì, T.; Iannizzotto, G.; Nucita, A.; Mohammadhasani, N. Interactive avatar boosts the performances of children with attention deficit hyperactivity disorder in dynamic measures of intelligence. *Cyberpsychol. Behav. Soc. Netw.* **2019**, *22*, 9. [CrossRef]
32. Gil-Gomez, J.A.; Manzano-Hernández, P.; Albiol-Pérez, S.; Aula-Valero, C.; Gil-Gómez, E.; Lozano-Quilis, J.A. USEQ: A Short Questionnaire for Satisfaction Evaluation of Virtual Rehabilitation Systems. *Sensors* **2017**, *17*, 1589. [CrossRef]
33. TensorFlow Blog—Pose Animator—An Open Source Tool to Bring SVG Characters to Life in the Browser via Motion Capture. Available online: https://blog.tensorflow.org/2020/05/pose-animator-open-source-tool-to-bring-svg-characters-to-life.html (accessed on 11 June 2021).