A modular middleware approach for exergaming

Martina Eckert, Ignacio Gómez-Martinho, Juan Meneses, José F. Martínez Ortega
Research Center on Software Technologies and Multimedia Systems for Sustainability (CITSEM)
Technical University of Madrid, Spain
Martina.eckert@upm.es

***Abstract*—This paper presents the design of a new exergaming environment consisting of a modular middleware tool aimed at serving for intelligent adventure games. The middleware provides a modular and user-adaptive interface for data exchange between different devices (to date it supports a motion capture camera, a mobile phone, and a VR headset) and Blender. The target group is formed by young people between ages 6 to 26 with different physical diseases (muscular dystrophy, cerebral palsy, accidents, etc.). The gaming environment focuses especially on user awareness, immersion, and adaptability to special needs.**

Keywords—gaming; rehabilitation; physical exercises; virtual reality, animation; Blender; Motion Capture; user awareness; NUI

I. Introduction

A large number of applications has been created for physical exercises based on VR (Virtual Reality) games. Nevertheless, the developments are still mainly taking part in laboratories, and few can be found that are part of daily life. The reasons seem to be, that most applications are developed for special cases (i.e. stroke, Parkinson, etc.) and are not commonly applicable [1]. On the other side, several applications have been found which aim at serving for everybody, but they are not accessible for patients with non-frequent problems, e.g. if they have limited motor abilities, [2] analyzes general requirements for motor rehabilitation. Also, although most authors report good acceptation and fun, many games are quite repetitive and too evidently focused on the objective, such that the patients are quite aware of the movements to perform and notice their limitations, which could lead to discouragement. Here, a lot of self-motivation and discipline is required, which is already difficult for adults and even more for children and youngsters. Last but not least, much work is needed to control and evaluate the exertion of the exercises adequately to avoid overexertion [1].

Therefore, an ideal exercise game should fulfill the following requirements:

Allow **individual configurations** according to the patients' physical limitations and necessities, i.e. allow the selection of body parts to use or to exclude, the programming of exercises, the adjustment of parameters like frequency, duration or number of repetitions of certain movements, etc.

Automatic and intelligent user adaptability during gameplay: understand what the person needs and likes and respond by changing the task (exercise), adjusting the level of difficulty (lowering it in case of tiredness and rising it in case of success or learning) or providing positive feedback in form of sounds, game prizes, bonus points, etc.

Immersion and inclusion: The game should involve the users in a way that they don't realize they are doing exercises, and what's more, it gets them hooked in a good way, such that they want to integrate the play in their daily leisure activities. Such a game would fulfill also an important psychological function by enabling inclusion: disabled children desire to be integrated as much as everyone, but often cannot play commercial games together with their friends.

To cope with all the above mentioned requirements, the gaming environment must obtain, store, and evaluate a huge amount of data, such as quality of performed movements, performance over time (in the same session or comparing different ones), the user's state (animated, bored or exhausted), possible overexertion, variety of movements (to avoid boredom, discouragement and overexertion, distinguish weakness from boredom) etc.

This work presents a proposal for such a new form of combining games and exercises. It is composed of a modular middleware which can handle different inputs (to date Mocap (motion capture) camera, mobile phone, and VR glasses) and passes the data to a VR game, implemented with the open source Blender software. A first state of the work in progress was previously presented in [3].

II. System Description

A. Middleware

The centerpiece of the system is a modularly designed program that sends data obtained from any connected NUI (Natural User Interface) device to an animation software (in this work Blender), see Fig. 1. The middleware is necessary since Blender cannot interact with the SDKs and libraries that manufacturers provide to obtain data from their devices.

The user interface is built with tabs to access the different modules which contain the graphic controllers for the devices. Currently, it supports a Mocap camera, a mobile phone, and a VR headgear; more devices could be added easily at any time.

During execution, the devices continuously provide information about the position and movements of the player: the smartphone's accelerometer detects the orientation relative to the ground and the speed at which it is swung, this information can be used by the game for accurate hand-tracking. The Mocap camera processes images from the player's complete body and tracks a certain set of joints and bones. Its output allows to recreate a 3D model of her/his 'skeleton' in the gaming software. The VR headgear allows a

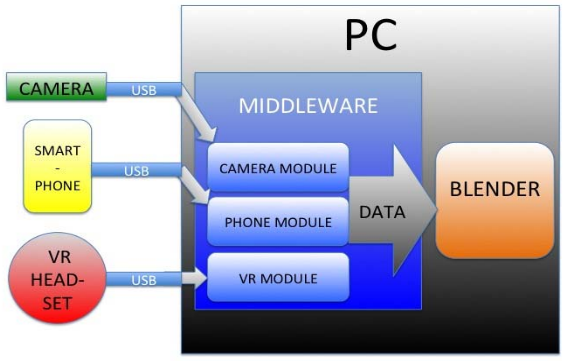

Fig. 1. Middleware design

much more accurate measure of her/his head's position and provides the user with a much more immersive experience.

Each module implements its own communication system, defined by the API of each particular device, and the data is transmitted to the middleware at a regular rate via USB. Several communication channels are established with Blender and data is sent in response to requests. The delivery of data has been implemented with the free library Python-OSC [4], which codes its messages accordingly to the Open Sound Control protocol, a variant of MIDI that runs on local UDP.

B. Gaming environment

The receiver in Blender is implemented through a Python add-on. This add-on allows the developer to insert a receiver object in the 3D environment, which collects data in real-time during gameplay and registers the position of the user in each frame of the game. The receivers are 3D models that represent the sending devices: the mobile phone is represented by a cube, the HMD (head mounted device) is implemented through an in-game camera, and a set of points and lines represent the player's joints and bones. Fig. 2 shows the initial skeleton and two boxes for mobile phone and headset. Every instant, when new data (rotation angles and spatial positions) is received, the script updates the position of the models, achieving thus an accurate depiction of the user's movement.

The Virtual Reality effect is achieved by translating the player's head movement to a virtual stereoscopic 3D camera, associated with the head of the avatar. The result is a first person view with epipolar images for each of the user's eyes. When displayed on the screens of the HMD, the player feels immersed in the game.

As mentioned in the beginning, the purpose of this project is to develop video games for physical exercises, which would be useful for an everyday fitness or to pass through a special rehabilitation program. Concretely, the patients are aimed to play an action-adventure game controlled with their bodies, which will be composed of many different tasks that are specially configured and selected for their needs. The game itself does not reveal which specific training is performed at each instant; the players will only be driven by the motivation to fulfill a mission, to reach some score or goal without noticing that they realize the correct exercises to achieve it. Here, the crucial part of the system is the reliable detection of corporal movements with the help of a Mocap camera. The captured motion can then be used in different ways:

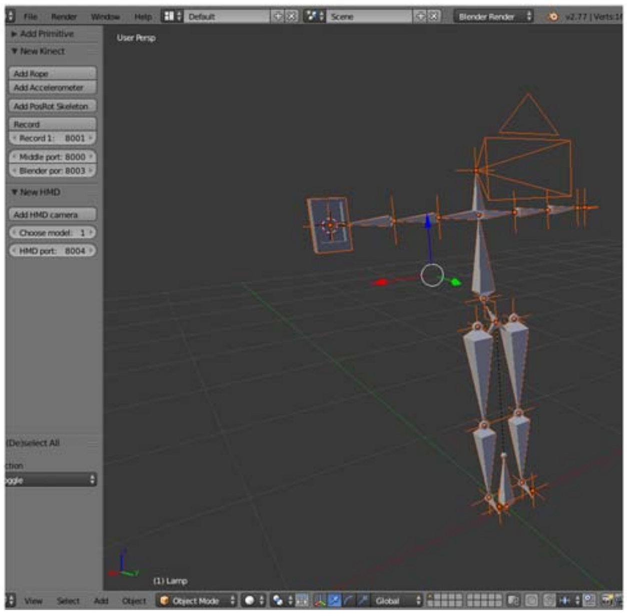

Fig. 2. Blender scene showing the basic skeleton, near the head the HMD object and near the right hand the mobile object. On the left add-ons for Mocap camera and HMD

- As a **simple copy** – which means that the avatar is doing the same movements as the player.
- As a **command** – which means that certain movements are not copied to the avatar but used to trigger some actions, e.g. a foot-shaking to make the character jump or a clap of hands for pausing the game.
- As a **boosted copy** – which means that the avatar does nearly the same movement as the player, but "better," i.e. in a game-like, illusive, comic-like or imaginary way if desired, or in a totally natural way, when the patient is not able to perform it due to some disability or accident.

This last type of moves are the most interesting one for the here presented investigation, as it widens the utility of the game enormously. On one hand, virtual movements achieve a much better immersion of the user in the game than the exact copy, because a game is an elusive environment and the players want to forget about their normal world and transform into different personalities. In this way, they could imagine being e.g. different creatures with abilities they do not have in the real world. This is especially interesting for handicapped people, e.g. wheelchair users dream to walk and run, children with muscle weakness want to shoot a ball like their friends or stroke patients want to coordinate both body parts in the same way like before to play golf or similar. Here, the gaming environment will apply enhancement methods to achieve a multiplication of the real movements in a way that the patients imagine doing it by themselves.

C. Implementation

The translation of movements captured by the camera to the Blender skeletons can be realized in different ways. The add-on allows the developer to define four types of receiver skeletons, differentiated by the specific type of position data they process:

- The "**Rotation Skeleton**" receives only one type of data, which is the rotation of each bone related to its root bone. This skeleton allows determining precisely the pose of the user but ignores his or her actual body type and size, since the bones in this skeleton come with predetermined lengths that may or may not resemble the real ones. This may be inconvenient when determining the amplitude of the movements the patient is requested to perform in the game.
- The "**Position Skeleton**" receives the real distance between the players joints in x,y,z-coordinates, recreating their bodies and movements in the real space perfectly. The bones are just connections between joints without receiving the rotation data: in this way, the rotation of the bone along its axis is randomized, which means that the pose of the avatar is not always the one of the user and can even look aberrant.
- The "**PosRot Skeleton**" or combined skeleton, as presented in Fig. 2, solves the problems of the previous two by processing both kinds of data: the bones receive rotation information and are bound to the joints, which receive their positions in x,y,z-space. This skeleton reproduces both, the pose and the anatomy of the user perfectly, at the cost of being much more complex than the other skeleton types.

The combined skeleton, however, still presents a challenge to be resolved, which is the handling of the joint's coordinates, as they are defined related to the Mocap camera. This can be a serious problem when establishing reference points for the exercises, since users will never play at the exactly same position in the room, nor face precisely towards the camera. Here, the "Rotation Skeleton" solves this problem, receiving exclusively directional data for the bones while ignoring the "global alignment" of the users. This means, if the rotation of the bones is expressed related to each bone's root bone, the skeleton appears to be always facing front and standing at the (0,0,0) position.

Finally, to determine the bones' length, a "Position Skeleton" is applied in the configuration phase. The measured lengths are stored in a file, and the "Rotation Skeleton" reads it every time it is executed. It does not matter if it does not receive updates about the joint's positions because the length of the bones will always remain equal. The combination of the fixed lengths with the updated rotation information in the joints produces a fairly accurate representation of the player's position.

The formerly mentioned amplifying functionality to realize a "boosted copy," which is above all useful for patients with physical limitations, is realized as follows:

- During the configuration phase, the maximum motion range is measured for each limb. This information leads to the multiplying factors α, β, γ in x, y and z directions that are necessary to magnify the movements captured during the play.
- During the game, the received motion data is multiplied by the formerly obtained factor to achieve normal movements of the game character even if the user can only make small gestures:

$$\begin{bmatrix} \acute{x} \\ \acute{y} \\ \acute{z} \end{bmatrix} = \begin{bmatrix} \alpha \cdot x \\ \beta \cdot y \\ \gamma \cdot z \end{bmatrix} \qquad (1)$$

In Blender, this is implemented with an amplifier object, which marks three points in space, as illustrated in Fig. 3 using colored balls as visualization. The object must be placed in point 1 (red ball), which is the position of the tip of the avatar's limb when it is in the resting position. This point is always constant in relation to the spatial position of the Skeleton. Point 2 (green ball) is constantly moving while tracking the position of the tip of the user's leg or arm. Point 3 (yellow) is the one that applies the amplification: α multiplies the difference of the x-coordinates of points 1 and 2, and the resulting value marks the distance in x between points 1 and 3. The same operation is performed with y and z coordinates.

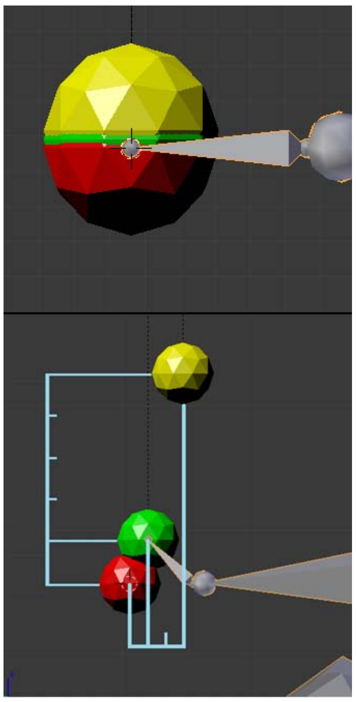

Fig. 3. Top: Hand-bone with 3 point amplifier object in resting position. Bottom: Hand bone is pointing upwards. Red: origin, green: real current position, yellow: the avatar's hand's position through amplification.

On the top image the bone, representing the right hand of the user, is shown in a resting position (for a person in a wheelchair, the hand would be pointing forwards). On the tip of the bone we placed the mentioned three points, overlapping because the hand is not moving. The red ball represents the origin (point 1), the green one represents the current position of the hand (point 2), and the yellow one represents the amplified position (point 3).

On the bottom half, the hand is shown pointing slightly upwards. As can be seen, the red ball has not moved, while the green one has followed the movement of the bone. The yellow ball represents the amplification of the movement: in this case, the amplifier has been set to multiply the displacement along the z-axis (vertical) by factor 5, and the displacement along the y-axis (horizontal) by factor 3. For the sake of simplicity, the motion along the x-axis is multiplied by one, such that the yellow and the green ball are exactly at the same x-position.

The magnified movements are employed by the avatar as follows: instead of directly copying the pose of the skeleton's joints, it tracks the displaced virtual object and inverse kinematics are applied. As a result, a very simple gesture, e.g. raising the tip of the hand, could be translated to lifting the whole arm of the avatar. In future stages of the project, this amplification will be enhanced to vary dynamically according to the user's progress with the rehabilitation.

Fig. 4 demonstrates this process: the hand is in the same position as in Fig. 2, but here with an amplifying vertical factor of 20. It can be seen how the avatar is now pointing to the yellow ball, while the green one (representing the real movement) is barely a few centimeters above the resting point. The inserted photo shows the real movement of the user.

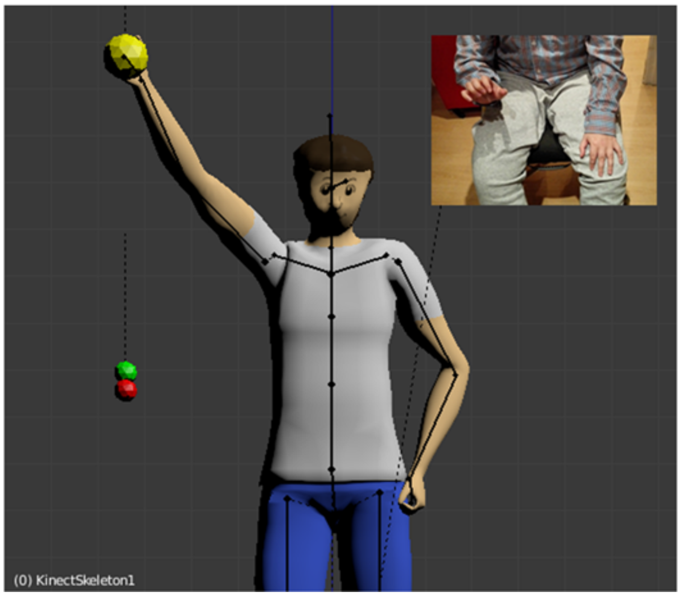

Fig. 4: The avatar is executing a boosted movement. The green ball represents the real position of the user's hand (see photo).

A further problem to solve is to treat with the different resting positions and pointing directions of the hands of a standing avatar and a sitting user (e.g. in a wheelchair). The resting position of the player would be much above the one of the avatar and, while the player's hand is pointing forwards, the resting hand of a standing person 8here the avatar) should be hanging down. This issue has been fixed by adding an offset to the amplifier.

Fig. 5 illustrates a case using this offset: there are a red and a green ball (origin and moved position) showing the real resting position and the real displacement of the player's hand, and a copy of each at the corresponding site within the avatar's range of movements. In the top image, the user points with the hand slightly upwards, while in the bottom image it is resting on the armrest of the wheelchair.

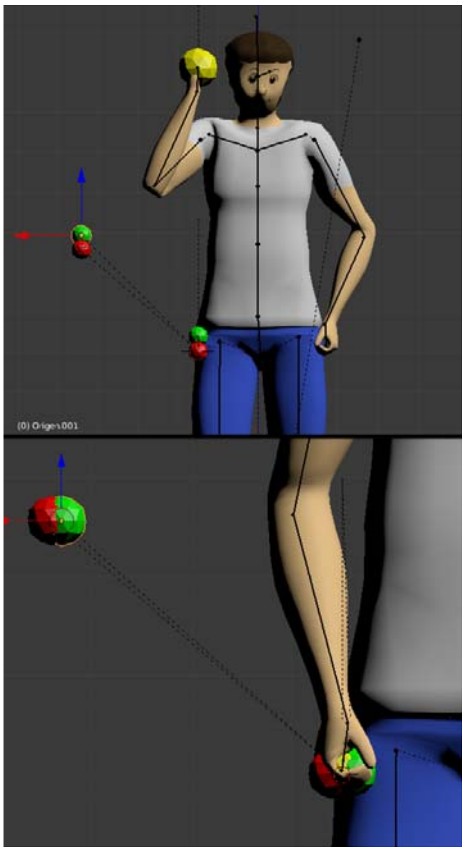

Fig.5: Illustration of the offset, when the user is sitting. Top: user's hand in motion. Bottom: User's hand is resting on the armrest of the wheelchair.

D. Application

The here described techniques are currently implemented in a demo game that will be used for functional tests with different types of persons, sane and handicapped, in wheelchair or not, mostly affected by neural muscle dystrophies.

The game will be of adventure type with multiple tasks to fulfill. Implementation is in the beginnings, and only few exercises have been created as e.g. rowing, climbing, flying and hitting moles, see Fig. 6. In all examples, the avatar's movements will be real although the users have different capacities. The final aesthetics will be completely different, and all exercises will be integrated into an imaginary adventure landscape to maximize immersion.

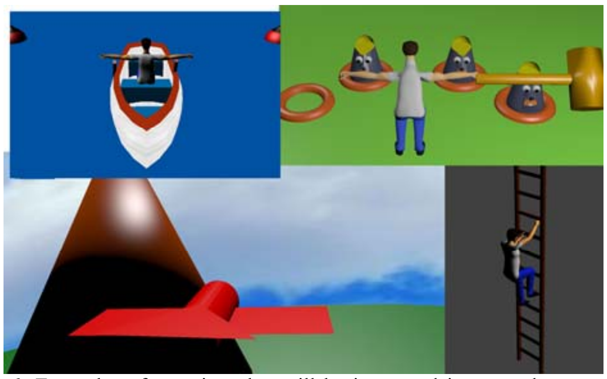
Fig. 6: Examples of exercises that will be integrated into an adventure-like gaming environment.

III. Conclusions

This paper presents a new approach of middleware, which integrates input data of different NUI devices for the realization of adaptive and immersive motor rehabilitation games for young people. We introduced a completely new technique for user integration with help of amplified user movements. This technique should help to improve immersion and the feelings of rehabilitation patients, as they feel more "normal". The system is still under development; only preliminary tests have been performed, first results with two groups of users (sane and affected) will be published in autumn 2016.

Acknowledgment

This work was sponsored by Spanish National Plan for Scientific Tech. Res. and Innov.: TEC2013-48453-C2-2-R.

References

[1] D. Webster and O. Celik, "Systematic review of Kinect applications in elderly care and stroke rehabilitation," *Journal of Neuroengineering and Rehabilitation*, vol. 11, Jul 3, 2014.

[2] R. C. Menezes, P. K. A. Batista, A. Q. Ramos and A. F. C. Medeiros, "Development of a complete game based system for physical therapy with Kinect," *IEEE 3rd International Conference on Serious Games and Applications for Health (SeGAH)*, Rio de Janeiro, 2014, pp. 1-6.

[3] M. Eckert, I. Gomez-Martinho, J. Meneses, and J.F Martinez Ortega, "A multifunctional plug-in for exergames", *IEEE International Symposium on Consumer Electronics (ISCE)*, 2015.

[4] Python Software Foundation, "Python-OSC 1.5", online: https://pypi.python.org/pypi/python-osc, [visited: April, 7th 2016].