A Gesture Recognition Approach for Wheelchair Users in an Exergame

Alexandre Greluk Szykman, André Luiz Brandão, and João Paulo Gois
Center of Mathematics, Computing and Cognition
Federal University of ABC (UFABC)
Santo André, Brazil
Email: alezuky@gmail.com, andre.brandao@ufabc.edu.br, joao.gois@ufabc.edu.br

***Abstract*—Studies indicate that playing exergames, games that use technologies of gesture recognition to incentive exercising, can improve multiple skills of the players. However, when players are wheelchair users, interactive gesture recognition is an issue not sufficiently explored. There are already exergames explicitly developed for wheelchair users, but until 2016, as far as we know, there was no game that explored the users' gesture movements in the digital game characters. In this work, we present our approach for gesture recognition of wheelchair users. It uses a depth sensor to track specific movements from wheelchair users. We employ those movements to interact with the game character. Results suggest that our technique, despite its simplicity, provides interactive frame rates while being also efficient.**

***Keywords*-Natural User Interface, Wheelchair Users, Digital Games, Exergames, Microsoft Kinect V2**

I. INTRODUCTION

Multidisciplinary research teams have observed that the practice of exergames – exercises with digital games – leads to the improvement of social, physical and psychological aspects of wheelchair users [1], [2], [3], [4], [5]. In this class of digital games, it is common the use of Natural User Interfaces (NUI), where the players perform "natural" actions that are familiar to them to control the game inputs, for instance shaking their arms, talking or blinking.

In 2013, Gerling et al. [6] presented an approach that considers wheelchair users interaction in digital games. As a consequence, the authors presented an essential sequence of studies on this topic [7], [8], [9], [5]. These works include the development of algorithms for gesture recognition, the design of digital games applied the gesture recognition algorithms and the application of Participatory Design (PD) [10], [11], [12], [13] principles for developing and evaluating the games.

However, up to our knowledge, none of such previous works focused on replicating or mimicking the wheelchair user's movements into game characters. According to Szykman et al. [14], the representation of the characteristics of people with disabilities in a game, including wheelchair users, leads to a vital manner to empower those characteristics in real life. This fact motivated us to approach wheelchair users' movements in a game.

In our context, the exergame uses the Microsoft Kinect V2 depth sensor to track wheelchair movements. To accurately map the wheelchair movements into our exergame, we took into account PD practices [13], where technical developers (game designers, programmers, etc.) and potential game players (wheelchair users) work strictly together to the design of the game.

The wheelchair users (we call them as *designers*) actively contributed to defining design elements of the game as well as the movements they would prefer to perform when playing the game. They argued for the importance the game interaction must provide similar movements to those that they would do in their daily activities. According to the wheelchair users, that aspect increased their confidence to enroll in exercising activities in real life.

As a consequence of the designers' requirements, we face the need to define not only an effective and fast algorithm to track the movements of the wheelchair users, but also to investigate further details, for instance, the best position of the sensor to track the wheelchair users. We developed the user interface of the game with inputs that referenced movements that they would perform to use a wheelchair in real life. In this work, we then detail about those inputs and the algorithm that we developed to implement them in a game. Our approach includes the use of a State Machine to identify each of the selected gestures to be developed. This State Machine also does the transitions from one gesture to another.

The main contributions of this paper include the following:

- A 3D gesture-based approach for the wheelchair with a State Machine to manage the identification and execution of the gestures;
- The application of the gesture-based approach in an exergame we developed that considers a wheelchair user as the main game character.

We organize this paper as follows: Section II presents studies related to the input of wheelchair users movements in exergames. We briefly describe the game that we developed for wheelchair users in Section III. In Section IV, we detail our approach to employ wheelchair users' movements in games. Section V presents details of the implementation and as well as evidences the limitations of our algorithm and future research that potentially addresses those limitations. Finally, Section VI presents our conclusions.

II. Related Work

Depth sensors have been understood as the main technology to track user's movements in games [15], [16], [17], [18], [19], [20]. Shotton *et al.* and Sharp *at al.* presented part of the first studies that applied the Kinect sensor on gesture-based interaction. Plagemann *et al.* and Ganapathi *et al.* include geodesic distances for real-time results. Brandão *et al.* also employed geodesic distances and compared classification algorithms to pose recognition. Nevertheless, these studies did not focus on gesture-based interaction for wheelchair users.

Until 2017, the most relevant studies focusing on exergames with wheelchair users referred to works of Gerling *et al.* [6], [21], [4], [7], [9], [5]. Some studies of those authors examined the value of collaboration between design experts and powered wheelchair users for developing games with Participatory Design [5].

According to Gerling *et al.* [5], representing or not the character of the game as a wheelchair user influence on how other wheelchair users accept the game. This representation can be implicit or explicit. Gerling *et al.* [9] recommend the representation being implicit. However, an explicit representation can also provide good acceptance if the wheelchair is considered as a way to overcome challenges rather than just a tool for mobility. In general, wheelchair users best accept the game when the wheelchair is represented in a manner that empowers the user [22], [14].

Gerling *et al.* [7], [6] performed a specific study to develop efficient NUI tracking for wheelchair users with the Kinect sensor. Afterward, with the tracking system structured, the authors applied it in a game developed by Participatory Design with wheelchair users [5]. Even though Gerling *et al.* [7], [6], [5] could get relevant conclusions about the application of the system, the NUI inputs did not necessarily replicate users movements in the game. Up to our knowledge, there was no related work referring to the development of user interaction for exergames that reflected the movements of wheelchair users.

Ramli *et al.* [23] developed an asynchronous wheelchair navigation system using a hybrid of EEG signal and EOG artifacts embedded in EEG signals. The wheelchair navigation system is designed to move forward and backward in a total of six different directions. The authors modeled the system as a finite state machine with three modes, each containing three states.

We applied an explicit representation of wheelchair users by a wheelchair user character. In our Participatory Design process, the wheelchair users defined the movements to be performed in the game. We developed a State Machine for gesture recognition based on the three dimensions of the detected body parts of the wheelchair users. In the following, we describe the Wheelchair Jecripe game.

Figure 1. House Stage: Movements from the player (bottom right) are replicated in the character (bottom left).

III. The Wheelchair Exergame

In the game, we tested our gesture-based approach to move the main character through four different stages: *House, Street, Sea* and *Sky*. Each stage contains different scenarios and goals.

In the *House*, the player handles a wheelchair to move through internal and external environments of the house. The scenario of the *House* Stage has elements of the users' daily life, *e.g.*, chairs, tables, couch, etc. All these elements are treated with collision detection so that players can interact with them. These elements do not interfere with the game flow; they work as artifacts to entertain the player. The *House* Stage scenario contains a bus stop and a car. Players can ask for the bus at the bus stop or start driving the car by standing next to it. The *House* Stage scenario and background music aim to provide a friendly atmosphere to the players. The referred atmosphere evidence to the player that the *House* Stage is not a competitive scenario. The scenario works as a menu to select the stage to be played. Figure 1 presents the *House* scenario with the explicit representation of a wheelchair user as the main character.

The *Street* Stage reproduces a city street. The players control their cars and must deviate from obstacles as boxes or lampposts. The obstacles have a large space among them to make easier the movement. Figure 2 depicts the gameplay of the *Street* Stage.

The *Sea* Stage is similar to the *Street* Stage. However, the player controls a jet-ski, and we represent the obstacles by boats or buoys and are closer to each other, requiring greater attention from the players. The *Sea* Stage also contains the item *turbo*, which increases the speed of the players, also increasing the difficulty. Figure 3 shows the gameplay of the *Sea* Stage.

The *Sky* Stage has a scenario with different characteristics from the *Street* and *Sea* Stages. The players start in the center of the scenario and must reach a finish line. The scenario does not encompass a defined route. The players are free to ride a paraglider around an open 3D scenario. We created four different possibilities for the locations of the finish line. The finish line is randomly placed in one of the possible locations at the start of the Stage. The *Sky* scenario does not contain obstacles. The goal in that scenario is to produce a more pleasant scene than *Street* and *Sea*. The feeling that the scenario might provide to players originates from the freedom of being flying instead of the adrenaline of riding vehicles. Figure 4 demonstrates the gameplay *Sky* Stage.

It is worth to mention that Street, Sea and Sky stages are non-competitive, although two players control the game independently.

In the House Stage, we evidence that the main character is a wheelchair user. For this stage, we fully employed our gesture-recognition approach, detailed in the next sections. To move the character, we will show that it involves both body movements and hands gestures of the player simultaneously. To the remained stages, the movements of the characters are more straightforward, focusing mainly on the inclination of the player to the left and the right either with their arms closed to the torso (Car and Sea Stages) or with wide open arms (Sky Stage). Thus, we concentrate our explanation on the movements for the House Stage, as they are more complex and generalize the gestures of the other stages.

IV. THE GESTURE-BASED RECOGNITION APPROACH

Based on the information that we collected from the designers in the PD sessions, we proposed a user interaction approach in which one can move the main character of the game mainly using movements of controlling a wheelchair. Namely, simulating the turning of the left or the right wheel of the wheelchair makes the character turn to the right or left, respectively. Simulating the turning of both wheels to the front and reclining the body to the front makes the character move forward. The movement of pulling back with the arms makes the character move back. Figure 5 displays the inputs of the game.

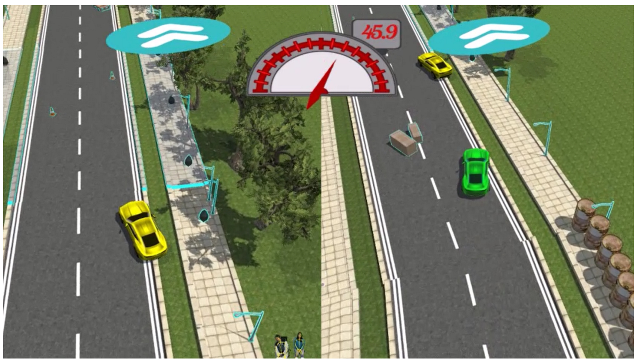

Figure 2. Street Stage: the user controls a car.

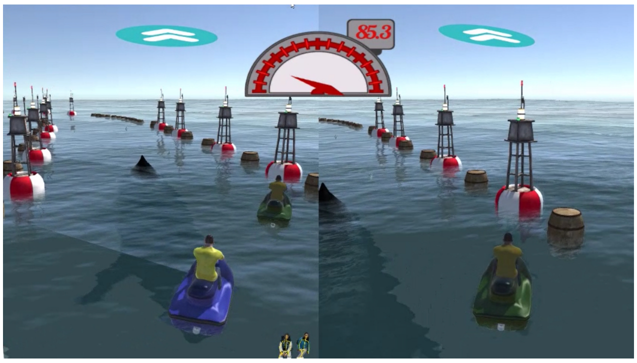

Figure 3. The Sea Stage: the player controls a character that rides a jet-ski.

For moving the character back, the most natural movement should be “turning of both wheels to the back”. However, it conflicts with the moving forward, as the sensor and our approach cannot differentiate them. At this version of our approach, it was an agreed solution in the PD sessions and in the development phases that minimized issues related to precision and performance of the technique.

A. Sensor Setup

The NUI inputs of the game are gestures, and they required the tracking of body parts in the z-axis (Figure 5). For this reason, the mounting setup that optimized motion capture was by placing the sensor on a tripod on a table so that the total height from the floor was 2 meters and the distance from the players was 1.5 meters. The defined distances optimized the motion capture of the spatial coordinates because it avoided occlusions of the players' body parts in the z-axis. Figure 6 displays the setup for the revised inputs.

Figure 4. The Sky Stage: the player controls a character that rides a paraglider.

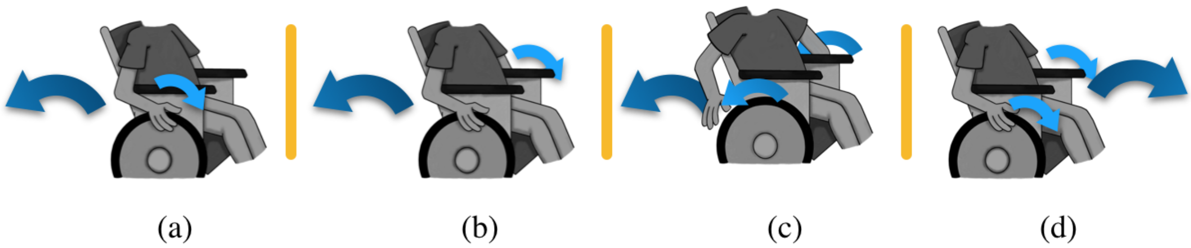

Figure 5. Body movements and the inputs in game for controlling: (a)-(b) to turn right (or left), the player reclines himself back and mimics to turn the left (or right) wheel; (c) to move backward, the player reclines himself back and pulls back both arms; (d) to move forward, the player reclines himself forward and mimics to turn both wheels.

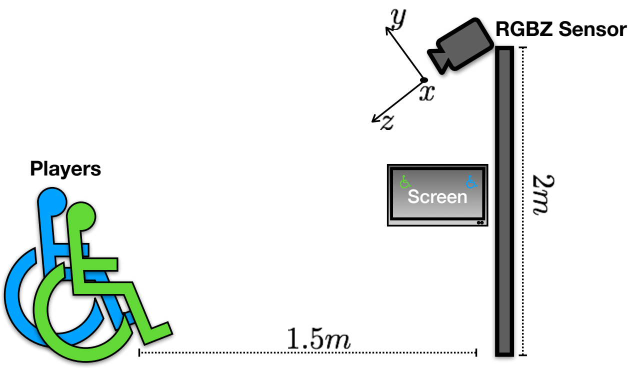

Figure 6. Kinect mounting setup for the revised inputs. The xyz-coordinates for the game follows the sensor's direction. This configuration reduces occlusions of body parts.

B. Motion Capture Approach

Our method recognizes the gestures of handling a wheelchair and reproduces the outputs of those gestures in the game, as described in Figure 5. We specify the motion capture features as the following [24]: Kinect V2 sensor; depth Camera Resolution with 512 × 424 pixels; capturing frequency with 30 Hertz; field of view of 70 × 60 degrees; range with 0.5 to 4.5 meters; and color camera with 1080 pixels, 30 Hertz (15 Hertz in low light).

To perform each gesture recognition of wheelchair users, we implemented a state machine with two different states: (1) Waiting for the Gesture; and (2) Performing the Gesture.

A set of conditions adjusts each state. The state machine initializes in State 1. The conditions of State 1 focus on defining whether a specific gesture started being either recognized or not. The conditions of State 2 focus on defining whether the recognized gesture in State 1 is either performed or not. When all the conditions of State 1 are satisfied, the machine moves to State 2. The machine remains in State 2 while the set of conditions of that state are satisfied, and returns to State 1 whenever they stop being satisfied. We check the conditions for each frame of the gameplay. There are four different gestures to be recognized: *Turning Right, Turning Left, Moving Forward* and *Moving Backward*. Therefore, we implemented four state machines, each one with its two states. Figure 7 depicts the representation of the implemented state machine.

Our method uses the data provided by the sensor. Specifically, we explore the body joints coordinates of the tracked users and their variation for each frame. For instance, the left hand of a tracked user contains (x, y, z) values in the coordinate system of the sensor at the current recorded frame. If the user moves the left hand to the right, the (x, y, z) coordinates will be modified to new values in the next frame. Therefore, the variation in the (x, y, z) coordinates of the user's left hand from the starting to the final frame can define the movement: *Move left hand to the right*. To define the conditions that ruled each state machine of our approach, we employed a set of variations from the joints of wheelchair users during gameplay.

Because the inputs for our algorithm were defined by mimicking the movements of turning the left or right wheels of the wheelchair, we are interested in the angular variation of left (right) hand considering the center of the left (right) wheel. We named the 2D vector estimated from the hands' coordinates to the wheel's center coordinates as (1) *Wheelchair Vector*. To measure that variation, we defined other two variables, which we named: (2) *Time Stamp* and (3) *Angle Rate*. Figure 9 displays these variables and the relationship between them. All these three variables resulted from relations among the (x, y, z) coordinates of specific body joints captured by the sensor. Figure 8 depicts the used joints coordinates and how we labeled them in this work. In our description, we use the notations $RightHand.x$ to refer to the x coordinate of the right hand in the sensor coordinate system (Figure 6), $RightHand.y$ to refer to its y coordinate, etc. In the next sections, we detail each key variables and its influence in the motion capture.

1) Wheelchair Vector: Considering the yz-plane, as depicted in Figure 6, the *Wheelchair Vector* is a two-dimensional vector defined from the player's hand to the center of the wheel of the wheelchair in the yz-plane while the player is performing the movement of turning the wheel. The following projection computes it:

$$WheelchairVector = \Pi_{yz}\left(RightHand.xyz - RightWheelCenter.xyz\right)$$

for turning the right wheel, whereas:

$$WheelchairVector = \Pi_{yz}\left(LeftHand.xyz - LeftWheelCenter.xyz\right)$$

for turning the right wheel, where Π_{yz} refers to the projection of the pointed vectors in the yz plane.

The Kinect V2 provides a precise response for capturing body joints coordinates. However, we noticed that the wheelchair interferes in recognition of coordinates of the lower parts of the body. As a consequence, we did not have a precise response in tracking the center of the wheelchair, located in the lower part of the body. Figure 10 displays the tracking areas with a roughly accurate response from the sensor. To tackle this issue, we estimated the wheelchair center coordinates based on trivial notions of anatomy. To perform the estimation, we created a point, named *Wheel Center*. Such a point was composed of the x and z coordinates from the user's center hip joint and a modified y coordinate. To define the new y coordinate, we subtracted half of the user's torso size from the center hip original y coordinates. We measure the torso size by the distance between the user's shoulder and the hip y coordinate. The resulting point was a satisfactory estimate of the wheel center. The estimate of the wheel center also presented a satisfactory tracking precision, because it was composed only of upper body joints. Figure 9 shows the composition of the *Wheel Center* point. The estimate is computed by:

$$RightWheelCenter = \Pi_{yz}\left(RightHip - \left(0, \frac{TorsoSize}{2}, 0\right)\right),$$

for turning the right wheel, or

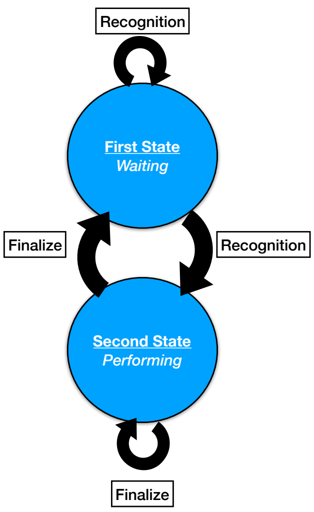

Figure 7. An implemented machine state for motion capture of wheelchair users.

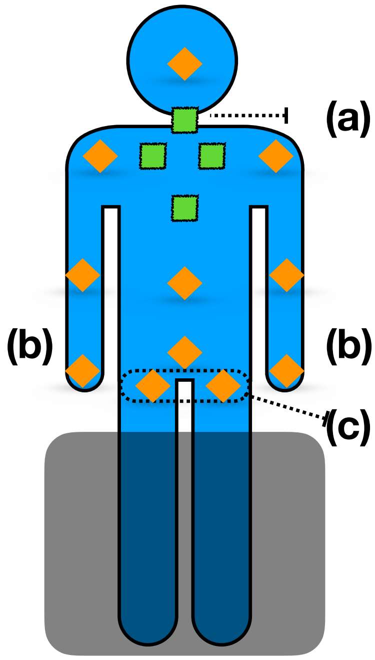

Figure 8. Scheme of the utilized joints coordinates for motion capture. The image reproduces the avatar for animation in the software that we used to develop the game: (a), (b) and (c) corresponds to the Spine Shoulder, Hands and Hip joints, respectively.

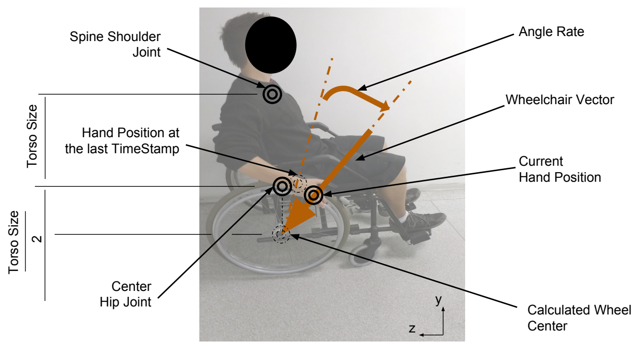

Figure 9. Representation of the used variables for motion capture with a human model and the relationship among them.

$$LeftWheelCenter = \Pi_{yz}\left(LeftHip - \left(0, \frac{TorsoSize}{2}, 0\right)\right),$$

for turning the left wheel, where

$$TorsoSize = SpineShoulder.y - CenterHip.y.$$

2) Time Stamp: The variable *Time Stamp* represents the period that the *Wheelchair Vector* is re-computed with updated users' joints coordinates. In our work, we defined the *Time Stamp* variable to 0.2s, because this value provided a satisfactory performance to the algorithm. Therefore, during

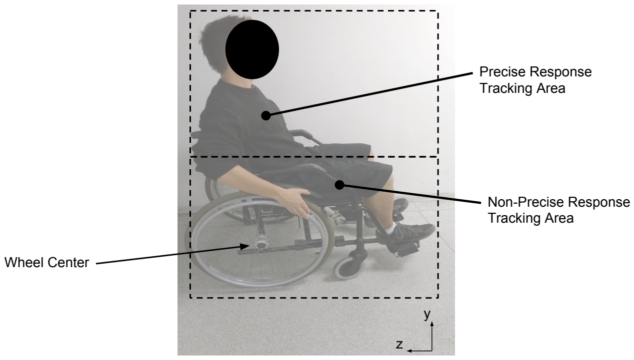

Figure 10. Precision of joints recognition through Kinect for people seated in wheelchairs. The upper part of the body provided a tracking area with a better response.

gameplay, for each 0.2s period, a new value is computed and attributed to the *Wheelchair Vector*.

3) Angle Rate: The *Angle Rate* variable represents the angle between the current *Wheelchair Vector* and the last computed *Wheelchair Vector* in the period delimited by the *Time Stamp*. Figure 9 displays the representation of the *Angle Rate* variable with respect to the other variables. The *Angle Rate* is the output of the State 2 for all the state machines in our study. Therefore, during gameplay in our algorithm, the *Angle Rate* is recomputed after each 0.2s period and a new output is generated.

When a state machine computes a new output, the game engine increases the linear or angular velocity of the character by multiplying the *Angle Rate* variable value by a constant. As a consequence, the greater values attributed to the *Angle Rate* variable, the faster speed to the character. The algorithm increases the linear velocity if the recognized gesture is *Moving Forward* or *Moving Backwards* and increases the angular velocity whenever the recognized gesture is *Turning Left* or *Turning Right*. The value of the constant that provided the best performance of the algorithm was 0.1. Therefore, during the gameplay with our algorithm, the linear or angular velocity is increased by the value $AngleRate \times 0.1$. Table I shows the resulting mechanics from each activated state machine. In the next, we present the functionality of each state of the state machines.

4) State Machines: There are two state machines that control all the movements. In the first state of each movement recognition, the algorithm waits for a set of conditions to be satisfied. The purpose of the conditions of the State 1 is to check if the player is in a pose that implies that the gesture to be recognized is starting to be performed. The logic of the conditions varies according to each movement to be recognized. For instance, a starting pose for the recognition of the gesture *Turning Right* can be composed by the right hand standing behind the player's hip, in a determinate (x, y, z) coordinate that implies that the player will start rotating the wheel of the wheelchair. When all the conditions of a certain movement are satisfied, the state machine moves to the second state. Tables II–V list the conditions of State 1 for each recognized movement. The thresholds in the conditions of the algorithm are measured in millimeters and were defined so that the response of the game was optimized after several tests.

In the second state of each movement recognition, the algorithm remains to produce a new output at every time, defined by the variable *Time Stamp*. The output is the current value of the variable *Angle Rate*. Each state embraces a set of conditions in State 2. Those conditions are different from the ones defined in State 1. The purpose of the conditions in State 2 is to check if the player is performing the gesture that started in State 1. For instance, to ensure that the player is performing the gesture *Turning Right*, the algorithm checks if the player's right hand is in a certain, fixed x coordinate and performing an arc movement with the dimensions of the wheel of the wheelchair in the yz-plane. If any of the conditions stop being satisfied, the state machine returns to State 1. Table VI details the conditions of State 2 for each recognized movement. The thresholds in the conditions' algorithm are measured in millimeters and were defined so that the response of the game was optimized after several tests. Figure 11 depicts the general flowchart of the algorithm through the machine states.

V. RESULTS AND DISCUSSION

We developed the exergame as well as our gesture recognition approach in the *Unity3D* Engine, using the C# Language of Unity3D. We used the sensor *Kinect V2* with a third-party asset from *Unity3D On-Line Asset Store* to support the Natural User Interface (NUI) [25], [26], [13].

For respecting smoothness and coherence, we set the frame update to 0.2 seconds, after several empirical tests. This value was the best trade-off among all the tested performance.

Once we had implemented our approach, three wheelchair designers tested the game developed [13]. They played all the stages of the game, that took from 15 to 20 minutes. In the end of the game, we asked them to provide feedback about the facility to learn and interact with the game.

Despite its simplicity, we observed that proposed technique provides precision for the movement detection. Specifically, in previous user tests, the results showed that the users were capable of playing and using the sensor with reasonable facility, as demonstrated by Szykman *et al.* [13]. Therefore, the input approach was successful in capturing users' movements without tiring them. A video of the game in the House Stage can be found at `https://youtu.be/YHFdyun7Twk`.

We structure an approach that explores movements of wheelchair users with the support of the *Microsoft Kinect V2* sensor. However, we believe that it is possible to improve our approach in two different aspects. First, we believe that we

Table I
TRANSLATION OF THE VARIABLE *Angle Rate* TO CHARACTER'S MECHANICS.

Activated State Machine	Resulting Character Mechanics
Turning Right	Rotate Character to the Right (increases angular velocity)
Turning Left	Rotate Character to the Left (increases angular velocity)
Moving Forward	Translate Character Forward (increases linear velocity)
Moving Backwards	Translate Character Backwards (increases linear velocity)

Table II
GESTURE: TURN LEFT

Condition Description for Turn Left	**Condition Algorithm**
SpineShoulder can not be in front of Hips. This condition differs the gesture from the conditions of Moving Forward	$SpineShoulder.z > CenterHip.z - 0.2$
Hand behind hips to start rotating the wheel	$RightHand.z > RightHip.z + 0.05$
Hand between hips and the wheel in the x direction	$RightHand.x > RightHip.x$ AND $RightHand.x < RightHip.x + 0.7$
Hand at the height of the wheel	$WheelchairVector > TorsoSize - 0.3$ AND $WheelchairVector < TorsoSize + 0.3$

Table III
GESTURE: TURN RIGHT

Condition Description for Turn Right	**Condition Algorithm**
SpineShoulder can not be in front of Hips. This condition differs the gesture from the conditions of Moving Forward	$SpineShoulder.z > CenterHip.z - 0.2$
Hand behind hips to start rotating the wheel	$LeftHand.z > LeftHip.z + 0.05$
Hand between hips and the wheel in the x direction	$LeftHand.x < LeftHip.x$ AND $LeftHand.x > LeftHip.x - 0.7$
Hand at the height of the wheel	$WheelchairVector > TorsoSize - 0.3$ AND $WheelchairVector < TorsoSize + 0.3$

Table IV
GESTURE: MOVE FORWARD

Condition Description for Move Forward	**Condition Algorithm**
SpineShoulder have to be in front of Hips This condition differs the gesture from the conditions of Turning to the Sides and Moving Backwards	$SpineShoulder.z < CenterHip.z - 0.1$
Hands behind hips to start rotating the wheel	$RightHand.z > RightHip.z - 0.2$ AND $LeftHand.z > LeftHip.z - 0.2$
Hands between hips and the wheel in the x direction	$RightHand.x > RightHip.x - 0.1$ AND $RightHand.x < RightHip.x + 0.3$ AND $LeftHand.x < LeftHip.x + 0.5$ AND $LeftHand.x > LeftHip.x - 0.7$
Hand at the height of the wheel.	Right $\|WheelchairVector\| > TorsoSize - 0.3$ AND Right $\|WheelchairVector\| < TorsoSize + 0.3$ AND Left $\|WheelchairVector\| > TorsoSize - 0.3$ AND Left $\|WheelchairVector\| < TorsoSize + 0.3$

can improve the accuracy of the algorithm by better tune the values in Tables II, III, IV, V, and VI. For instance, we can improve our approach by making use of Machine Learning approaches such as Classification Algorithms. Second, even though the Kinect V2 has shown an acceptable accuracy for tracking the movements, it still presented some limitations in terms of false positives and missed tracking. Thus, it is relevant testing the algorithm with different sensors and architectures.

We believe that the algorithm would benefit from the inclusion of two new functionalities. First, we noticed that the algorithm presented low efficiency for wheelchair users with Cerebral Palsy. We encourage the prospective of versions of the algorithm that better understand and include the interaction features for those people in the approach. Second, the wheelchair users that supported us as co-designers asked for the movement of pulling back the arms as the input of moving the character back. It is possible to see this movement in Figure 5 sub-image (c). We understand that the movement of moving the wheel of the wheelchair back would better represent moving the character back. Also as a future work we can compare our approach to other techniques that considers the gesture-based interaction. The comparison with other techniques would give us more

Table V
GESTURE: MOVE BACKWARD

Condition Description for Move Backward	Condition Algorithm
Hands in front of spine	$RightHand.z < SpineShoulder.z - 0.1$ AND $LeftHand.z < SpineShoulder.z - 0.1$
Hands above the shoulder	$RightHandIndex.y > SpineShoulder.y$ AND $LeftHand.y > SpineShoulder.y$

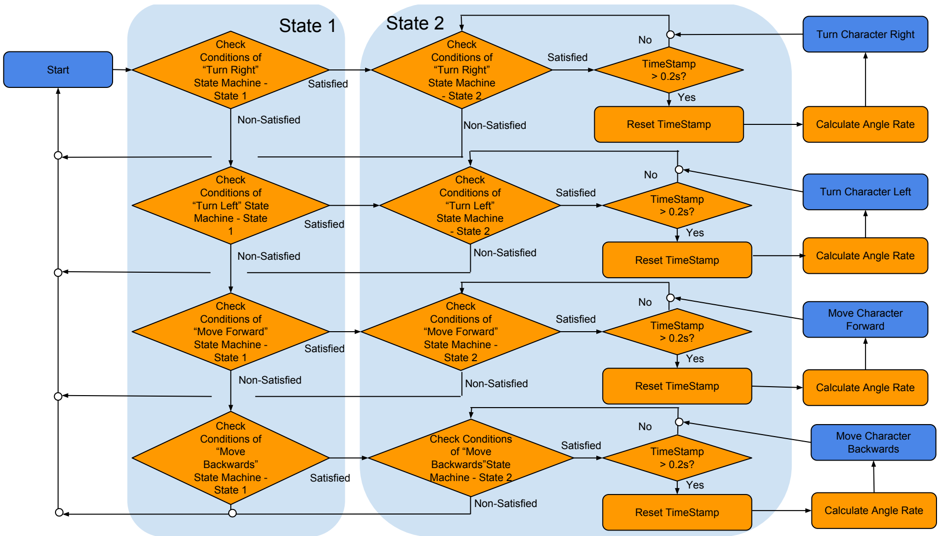

Figure 11. Flowchart of the motion capture algorithm through the states machines.

information about the performance and improvements that we need to do in our approach.

VI. CONCLUSIONS

We developed an approach for tracking and exploring the movements of wheelchair users in a gesture-based game. We used the *Microsoft Kinect V2* sensor to track the wheelchair users and their stakeholders in Participatory Design Sessions. The four basic movements were proposed and implemented movements: *Turn Right, Turn Left, Move Forward,* and *Move Backwards*. Results pointed that the algorithm leads to the game with acceptable accuracy in the inputs. That is because participants are familiar with the gestures and, then, get used to the sensor with reasonable facility.

This study is included in the context of the Jecripe Project [27], [28] which the main goal is to develop digital games to help people with disabilities. In this study, we presented an alternative to Brandão et al.'s [20] work that also considers the application of the Kinect for movement recognition. As we presented in Brandão et al. [29], developing games for people with disabilities can encourage studies in different Computer Science fields. In this context, an option for future work is to apply classifications in the context of wheelchair users, similar as we have already done in Brandão et al. [19].

As future work, we propose improving the algorithm' accuracy by testing it with different sensors, including new inputs and developing versions for wheelchair users with different needs, such as Cerebral Palsy or more severe physical or psychological limitations. In this work, we also presented the exergame which is the result of the Participatory Design of wheelchair users in the development of this gesture-based system.

ACKNOWLEDGMENT

We specially thank the *CR Tennis Academy*, the participants of the design process and the evaluation, as well as the *PARAJECRIPE Team* for providing us with artistic assets for the game. This work was financially supported by *Federal University of ABC* – UFABC and the Coordination for the Improvement of Higher Education Personnel (CAPES, Brazil) Masters Scholarship. The authors also thank São Paulo Research Foundation (FAPESP, Brazil), proc. nº 2014/11067-1, for the equipment support.

REFERENCES

[1] M. Plow and M. Finlayson, "A qualitative study exploring the usability of Nintendo Wii Fit among persons with multiple sclerosis," *Occupational therapy international*, vol. 21, no. 1, pp. 21–32, 2014.

Table VI
CONDITIONS FOR MOTION CAPTURE IN STATE 2. THE PURPOSE OF THE CONDITIONS IS TO CHECK IF THE PLAYER IS IN A POSE THAT IMPLIES THAT THE GESTURE TO BE RECOGNIZED IS STARTING TO BE PERFORMED. ALL THE NUMBERS IN THE TABLE ARE MEASURED IN MILLIMETERS.

Gesture	Condition Description	Condition Algorithm
Turn Right	Hand between hips and the wheel in the x direction.	$RightHand.x > RightHip.x - 0.1$ AND $RightHand.x < RightHip.x + 0.7$
	Hand movement is conformed by the contour of the wheel.	$-0.2 < WheelChairVector-$ First Captured $WheelchairVector$ <0.2
	Hand always moves forward.	$AngleRate > 0$
Turn Right	Hand between hips and the wheel in the x direction.	$RightHand.x > RightHip.x - 0.1$ AND $RightHand.x < RightHip.x + 0.7$
	Hand movement is conformed by the contour of the wheel.	-0.2 <WheelChairVector - First Captured WheelchairVector <0.2
	Hand always moves forward.	$AngleRate > 0$
Move Foward	Hand between hips and the wheel in the x direction.	$RightHand.x > RightHip.x - 0.5$ AND $RightHand.x < RightHip.x + 0.7$ AND $LeftHand.x < LeftHip.x + 0.5$ AND $LeftHand.x > LeftHip.x - 0.7$
	Hands movement is conformed by the contour of the wheel.	-0.3 <Right WheelChairVector - First Captured Right WheelchairVector <0.3
	Hands always move forward.	Right $AngleRate > 0$ AND Left $AngleRate > 0$
Move Backwards	Hands in front of spine.	$RightHand.z < SpineShoulderI.z - 0.1$ AND $LeftHand.z < SpineShoulder.z - 0.1$ AND
	Hands above shoulder.	$RightHand.y > SpineShoulder.y$ AND $LeftHand.y > SpineShoulder.y)$
	Hands always move backwards.	Right $AngleRate < 0$ AND Left $AngleRate < 0$

[2] G. Alankus, R. Proffitt, C. Kelleher, and J. Engsberg, “Stroke Therapy Through Motion-based Games: A Case Study,” in *Proceedings of the 12th International Conference on Computers and Accessibility*. ACM, 2010, pp. 219–226.

[3] A. Forsberg, Y. Nilsagard, and K. Bostrom, “Perceptions of using videogames in rehabilitation: a dual perspective of people with multiple sclerosis and physiotherapists,” *Disability & Rehabilitation*, vol. 37, no. 4, pp. 338–344, 2014.

[4] K. Gerling, I. Livingston, L. Nacke, and R. Mandryk, “Full-body motion-based game interaction for older adults,” in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, ser. CHI ’12. ACM, 2012, pp. 1873–1882. [Online]. Available: http://doi.acm.org/10.1145/2207676.2208324

[5] K. M. Gerling, C. Linehan, B. Kirman, M. R. Kalyn, A. B. Evans, and K. C. Hicks, “Creating wheelchair-controlled video games: Challenges and opportunities when involving young people with mobility impairments and game design experts,” *International Journal of Human-Computer Studies*, vol. 94, pp. 64 – 73, 2016.

[6] K. M. Gerling, M. R. Kalyn, and R. L. Mandryk, “Kinect wheels: wheelchair-accessible motion-based game interaction,” in *CHI’13 Extended Abstracts on Human Factors in Computing Systems*. ACM, 2013, pp. 3055–3058.

[7] K. M. Gerling, R. L. Mandryk, M. Miller, M. R. Kalyn, M. Birk, and J. D. Smeddinck, “Designing wheelchair-based movement games,” *ACM Transactions on Accessible Computing (TACCESS)*, vol. 6, no. 2, p. 6, 2015.

[8] K. Hicks and K. Gerling, “Exploring casual exergames with kids using wheelchairs,” in *Proceedings of the 2015 Annual Symposium on Computer-Human Interaction in Play*, ser. CHI PLAY ’15. New York, NY, USA: ACM, 2015, pp. 541–546.

[9] K. Gerling, K. Hicks, M. Kalyn, A. Evans, and C. Linehan, “Designing movement-based play with young people using powered wheelchairs,” pp. 4447–4458, 2016. [Online]. Available: http://doi.acm.org/10.1145/2858036.2858070

[10] N. Hendriks, K. Slegers, and P. Duysburgh, “Codesign with people living with cognitive or sensory impairments: A case for method stories and uniqueness,” *CoDesign*, vol. 11, no. 1, pp. 70–82, 2015.

[11] M. J. Muller, J. H. Haslwanter, and T. Dayton, “Participatory practices in the software lifecycle,” *Handbook of human-computer interaction*, vol. 2, pp. 255–297, 1997.

[12] D. Schuler and A. Namioka, *Participatory design: Principles and practices*, 1993.

[13] A. G. Szykman, A. L. B. ao, and J. ao Paulo Gois, “Development of a Gesture-Based Game Applying Participatory Design to Reflect Values of Manual Wheelchair Users,” *International Journal of Computer Games Technology*, p. 19 pages, 2018.

[14] A. G. Szykman, J. a. P. Gois, and A. L. Brandão, “A perspective of games for people with physical disabilities,” in *Proceedings of the Annual Meeting of the Australian Special Interest Group for Computer Human Interaction*, ser. OzCHI ’15. ACM, 2015, pp. 274–283.

[15] J. Shotton, A. Fitzgibbon, M. Cook, T. Sharp, M. Finocchio, R. Moore, A. Kipman, and A. Blake, "Real-time human pose recognition in parts from single depth images," in *Proceedings of the 2011 IEEE Conference on Computer Vision and Pattern Recognition*, ser. CVPR '11. IEEE Computer Society, 2011, pp. 1297–1304. [Online]. Available: http://dx.doi.org/10.1109/CVPR.2011.5995316

[16] T. Sharp, C. Keskin, D. Robertson, J. Taylor, J. Shotton, D. Kim, C. Rhemann, I. Leichter, A. Vinnikov, Y. Wei, D. Freedman, P. Kohli, E. Krupka, A. Fitzgibbon, and S. Izadi, "Accurate, robust, and flexible real-time hand tracking," in *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, ser. CHI '15. ACM, 2015, pp. 3633–3642.

[17] C. Plagemann, V. Ganapathi, D. Koller, and S. Thrun, "Real-time identification and localization of body parts from depth images," in *Proceedings of the IEEE International Conference on Robotics & Automation (ICRA)*, 2010, pp. 3108–3113.

[18] V. Ganapathi, C. Plagemann, D. Koller, and S. Thrun, "Real-time human pose tracking from range data," in *Computer Vision – ECCV 2012*. Springer Berlin Heidelberg, 2012, vol. 7577, pp. 738–751.

[19] A. Brandão, L. A. F. Fernandes, and E. Clua, "A comparative analysis of classification algorithms applied to m5aie-extracted human poses," in *Proceedings of the XII Brazilian Symposium on Games and Digital Entertainment*, ser. SBGAMES '13, 2013.

[20] A. L. Brandão, L. A. F. Fernandes, and E. Clua, "M5aie: A method for body part detection and tracking using rgb-d images," in *Proceedings of International Conference on Computer Vision Theory and Applications - VISAPP*, 2014, pp. 367–377.

[21] K. M. Gerling, K. K. Dergousoff, and R. L. Mandryk, "Is Movement Better?: Comparing Sedentary and Motion-based Game Controls for Older Adults," in *Proceedings of Graphics Interface 2013*. Canadian Information Processing Society, 2013, pp. 133–140.

[22] Y. Hutzler, A. Chacham-Guber, and S. Reiter, "Psychosocial effects of reverse-integrated basketball activity compared to separate and no physical activity in young people with physical disability," *Research in developmental disabilities*, vol. 34, no. 1, pp. 579–587, 2013.

[23] R. Ramli, H. Arof, F. Ibrahim, N. Mokhtar, and M. Y. I. Idris, "Using finite state machine and a hybrid of eeg signal and eog artifacts for an asynchronous wheelchair navigation," *Expert Systems with Applications*, vol. 42, no. 5, pp. 2451 – 2463, 2015.

[24] Microsoft, "Body tracking in kinect for windows sdk 2.0, programming guide," Microsoft, 2014.

[25] RF-Solutions, "Kinect v2 Examples with MS-SDK," https://www.assetstore.unity3d.com/en/#!/content/18708, 2017.

[26] Alexandre Greluk Szykman, "Participatory Design with Wheelchair Users in the Development of a NUI based interactive systems," Master's thesis, Universidade Federal do ABC, Santo André,Brazil, 2016.

[27] A. Brandão, L. Brandão, G. T. Nascimento, B. Moreira, P. T. M. ao, S. R. Brandão, C. N. Vasconcelos, and E. W. G. Clua, "Jecripe: stimulating cognitive abilities of children with down syndrome in pre-scholar age using a game approach." in *7th International Conference on Advances in Computer Entertainment Technology*. ACM, 2010.

[28] A. Brandão, D. Trevisan, L. Brandão, B. Moreira, G. Nascimento, P. Mourão, C. N. Vasconcelos, and E. Clua, "Semiotic inspection of a game for children with down syndrome," in *Proceedings of the IX Brazilian Symposium on Games and Digital Entertainment*, ser. SBGAMES '10, 2010, pp. 199–210.

[29] A. L. Brandão, L. A. F. Fernandes, D. Trevisan, E. Clua, and D. Stricker, "Jecripe: how a serious game project encouraged studies in different computer science areas," in *3rd International Conference on Serious Games and Applications for Health (SeGAH)*. IEEE, 2014, pp. 1–8.