COBEM UBERLÂNDIA 2019

ABCM Associação Brasileira de Engenharia e Ciências Mecânicas

25th ABCM International Congress of Mechanical Engineering
October 20-25, 2019, Uberlândia, MG, Brazil

COB-2019-0606

# MOTION CAPTURE OF A LOWER LIMBS EXOSKELETON FOR VIRTUAL REALITY APPLICATION WITH A TREADMILL

**Ícaro Ostan**
**Adriano Almeida Gonçalves Siqueira**
University of São Paulo, Av. Trabalhador São-carlense, 400, São Carlos, São Paulo, Brazil.
icaro.ostan@usp.br, siqueira@sc.usp.br

***Abstract.*** *Worldwide, the average life expectancy has been increasing. Thus, the number of people affected by disabilities, in need of physical therapy, also increases. In order to turn therapy more efficient, games that can create an immersive experience for the patient through virtual reality environments have been developed. The advantages of immersive gaming experiences include not only more engagement during the practice, but also an increase in pain tolerance. Likewise, robotic exoskeletons capable of actively enhancing the mobility of a patient and tracking his movements have contributed to improve therapy. Hence, given the recent progress noticed in therapy allied with virtual reality and robotic exoskeletons, this work describes the development of a virtual reality application, in which the legs movement of a person wearing a lower limbs robotic exoskeleton can be tracked and reproduced virtually, while the person walks on a treadmill. It is known that the higher the frame rate of the virtual reality application, the smaller will be the chances of the patient being affected by motion sickness, and the greater will be his engagement. Thus, the high frame rate will be treated as a requirement.*

***Keywords:*** *Rehabilitation Robotics, Exoskeleton, Virtual Reality, Gaming, Unity.*

## 1. INTRODUCTION

According to the latest World Health Organization (WHO) report on disabilities, a worldwide increase on life expectancy results on a greater number of people affected by chronicle diseases and disabilities. Furthermore, the same report also states that, at some point of life, it is very likely that one may be affected by some sort of temporary disability. Therefore, both population growth and life expectancy are critical factors that affect health institutions, particularly the ones aimed to physical rehabilitation, due to the increasing demand. The report concludes that new and more-efficient methods for therapy should be investigated in order to deal with the upcoming scenario (World Health Organization, 2011).

With that in mind, two kinds of technologies explore ways to enhance the practice of physical therapy. On one hand, there are robotic exoskeletons, which are capable of not only supporting the body parts of a patient, but also capable of increasing his range of motion and strength (Pons, 2008). Rehabilitation robotics, in this case, is the practice of physical therapy with the aid of robotic exoskeletons for upper or lower limbs.

Another technology that intends to make the therapy practice more engaging and efficient is virtual reality applications. Virtual reality is a three-dimensional computer-generated simulation, which is capable of providing a strong feeling of immersion, with the aid of proper software and hardware (Linowes, 2015). With virtual reality, pain complaints are less often. In addition, patients consider the therapy with the aid of virtual reality applications more engaging, whereas most consider the usual practice a boring and demotivating activity (Crocetta *et al.*, 2018).

This paper shows the development of a simple virtual reality application designed with Unity, which tracks a lower limbs exoskeleton hip movement and reproduces it in a virtual environment, while the user walks on a treadmill. In that way, whenever the user wearing the lower limbs exoskeleton moves, the avatar will try to perform the same leg movement. This work was done not only to create an application in which the patient feels engaged to practice movement, but also to explore the lower limbs exoskeleton integration with the virtual environment.

## 2. MATERIALS AND METHODS

### 2.1 Lower limbs exoskeleton

An exoskeleton developed by the Rehabilitation Robotics Laboratory from EESC-USP was used in this work (Santos, 2017). Essentially, it comprises six rotary one-degree of freedom joints (three for each leg) connected by steel bars. Each rotary joint contains one AksIM rotary absolute encoder module. The joints match the human hip, knee, and ankle joints.

This work only uses the hip joints. Each hip joint is located on the sagital plane so they rotate along the z-axis (see Figure 7).

The encoders used have a reading resolution up to 20-bits and are compatible with different communication interfaces, such as Pulse Width Modulation (PWM), Synchronous Serial Interface (SSI), Serial Peripheral Interface on slave mode (SPI), and Inter-Integrated Circuit interface ($I^2C$).

## 2.2 Virtual reality

The virtual reality environment was modeled with the game engine Unity (personal license), which has also a physics engine embedded. We designed the whole application with free 3D models from the Unity's assets store. All programming was implemented in C-sharp language, besides the Arduino communication between encoders and Unity, which consists of a script in Arduino programming language. The hardware used to run the virtual reality application was the HTC Vive, without any additional sensors, except the exoskeleton encoders (see Figure 7). The ideal frames per second (fps) rate for desktop virtual reality applications should vary from 90 to 120fps, while a 60fps rate is also acceptable in cases where the processing is limited, such as for mobile virtual reality applications.

## 2.3 Communication

An Arduino Mega 2560 microcontroller receives data coming from the encoders. The protocol used was the Serial Peripheral Interface (SPI) on slave mode, since all components were located nearby. Using SPI, the angle displacement of each joint consisted of a 16-bit data package. The Arduino's baud rate was set to 9600.

Once the microcontroller receives the data from the encoders, it converts the data from bits to degrees. Then, the microcontroller sends the data to the computer running the Unity application via universal serial bus (USB) as a string.

The application, running a C-sharp script, receives the data and parses the string into float values. Afterwards, the C-sharp script assigns the values to the transform component of game objects inside the virtual environment. Each one of these game objects represents a joint. This way, whenever a joint of the exoskeleton rotates along the axis perpendicular to the sagital plane, its virtual representation performs the same rotation.

The linear velocity of the avatar during the walking movement is based on the angular velocity observed on the hip joints.

## 2.4 Walking

In order to simplify the modeling of the walking process, we considered only the hip joints. Therefore, we modeled the walking as a one-degree-of-freedom inverted-pendulum, subjected to an angular displacement $\theta$. During a step, one foot always stays on the ground (stance leg) while the other moves forwards (swing leg). The inverted pendulum represents the stance leg.

The angular velocity of each joint $\omega$ is calculated through three different approaches. The first method consisted of backward finite differences of accuracy $\mathcal{O}(h)$ as in Equation 1. It estimates the angular velocity as the difference between the angular displacements of two consecutive readings.

$$\omega = \frac{d\theta}{dt} \approx \frac{\theta_{t_i} - \theta_{t_{i-1}}}{t_i - t_{i-1}} \tag{1}$$

The second method consisted of backward finite differences of accuracy $\mathcal{O}(h^2)$ as in Equation 2. It estimates the angular velocity considering three consecutive readings.

$$\omega = \frac{d\theta}{dt} \approx \frac{3\theta_{t_i} - 4\theta_{t_{i-1}} + \theta_{t_{i-2}}}{(t_i - t_{i-2})} \tag{2}$$

The third method consisted of a Kalman filter, which estimates the angular velocity values based on the angular displacement measurements. The Kalman filter structure is further explained on section 2.5

The linear velocity $v$ is calculated as in Equation 3, where $R$ in this case is the approximate leg length of the user.

The game engine processes the velocity value as a physical attribute. In that way, the final movement performed by the avatar depends not only on the velocity value itself, but also on the mass of the avatar, and the drag present in the scene. Since these specific attributes were not modeled, the linear velocity itself had to be adjusted by a parameter $\alpha$, intended to turn the final movement more realistic and comfortable to the user, thus avoiding motion sickness. This parameter is strictly positive, otherwise it would interfere in the movement direction, while it should only adjust the magnitude of the velocity.

$$v = \alpha R \omega \tag{3}$$

## 2.5 Kalman Filter

A Kalman filter is a mathematical tool, suited to digital computer implementation, which estimates the instantaneous state of a linear dynamic system perturbed by white noise. It is particularly advantageous with respect to applications in which it is not possible (or desirable) to measure every variable. This way, a Kalman filter provides a estimation of what is missing from indirect, noisy measurements (Grewal and Andrews, 2001).

In this work, a simple Kalman filter is used to estimate the angular velocity value from the angular displacement measurements.

## 2.6 Threading

For each frame, Unity executes in a predetermined order a sequence of functions called event functions (Unity Technologies, 2019a). The avatar's position in the virtual environment, for instance, must be updated every frame. Thus, one must guarantee that at least once every frame the function that updates the avatar's position is called. In a similar way, many other operations are performed at the same time, in order to update the virtual environment and its game objects.

Too many calculations to be performed at least once per frame may cause the frame rate to drop. Specially for virtual reality applications, besides undermining the game experience, which affects the player's engagement, low frame rates are also responsible for motion sickness (Jerald, 2016).

In view of the need to maintain a high frame rate at the same time the script performs a series of calculations, the threading resource was employed. Threading is a way to take advantage of a CPU's capability to perform many operations across multiple cores. This way, instead of instructions that run one after another, with threading the instructions can run simultaneously (Unity Technologies, 2019b).

C-sharp has already a threading library. Some minor particularities had to be considered when implementing threads in Unity.

## 3. RESULTS AND DISCUSSION

The encoders tracked the angular displacement of the joints as indicated by the blue line in figure below, where the movement of one hip joint was emulated.

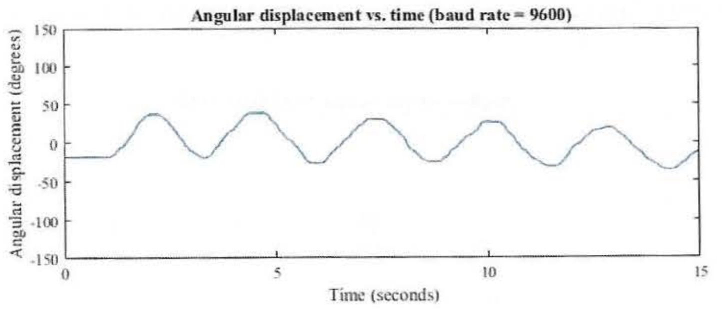

Figure 1. Angular displacement obtained through the rotary absolute encoder readings.

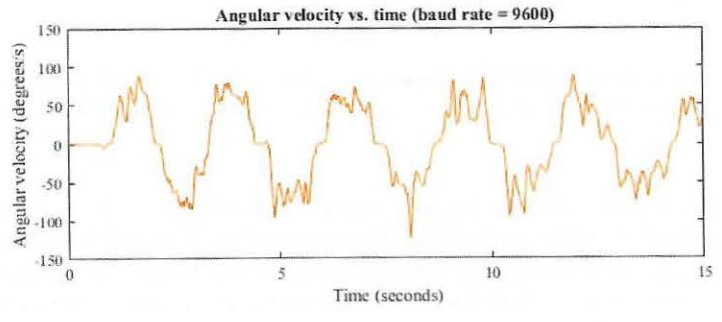

Figure 2. Angular velocity estimated by backward finite differences according to Equation 1.

Figure 2 depicts the angular velocity values according to Equation 1. It is clear that the velocity values are not as seamless as the displacement ones. This is not related to the variations on the frame duration or the time between two readings, which could affect the denominator of Equation 1 abruptly. The noisy waveform is due to limitations in the way the finite difference method was implemented for this application. In order to reduce the noise, three approaches were evaluated.

First, the baud rate, which is related to the frequency of data acquisition done by the microcontroller, was increased from 9600 to 115200. By doing this, we expected to enhance our numeric differentiation precision through finite differences by approaching the denominator value to zero, that is, the time between two readings. However, by doing this the noise increased, as seen in Figure 3. Then, we set our baud rate value back to 9600.

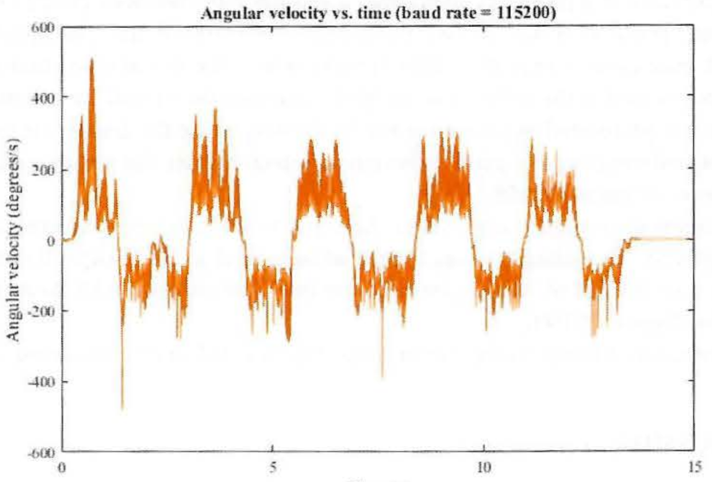

Figure 3. Angular velocity calculate according to Equation 1. In this case, the baud rate was set to 115200, and more noise was observed.

Second, the backward finite difference, which consisted of only the values between the current and previous frame, was adjusted to consider also the value of the reading before the previous frame, as in Equation 2.

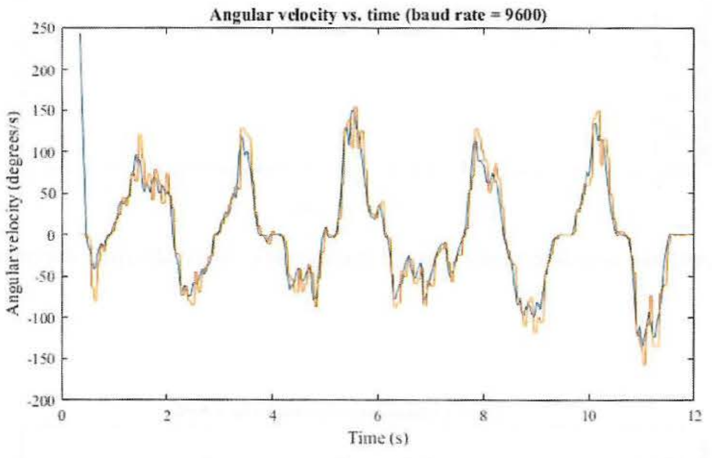

Figure 4. Angular velocity calculated according to Equation 1 (blue) and according to Equation 2 (red). The baud rate was set again to 9600. The results with this method are not very satisfactory. The noise remains.

With this, the peak values became flat lines, but they remained around the same magnitude and presented the same abrupt variations, as seen in Figure 5. Thus, the addition of terms in the finite backward difference was not effective.

Finally, we implemented a simple Kalman filter, which estimates the angular velocity based on the angular displacement readings. At last, the results were satisfactory. The filtered signal behaves accordingly, even when the angular displacement variation changes abruptly (see Figure 6).

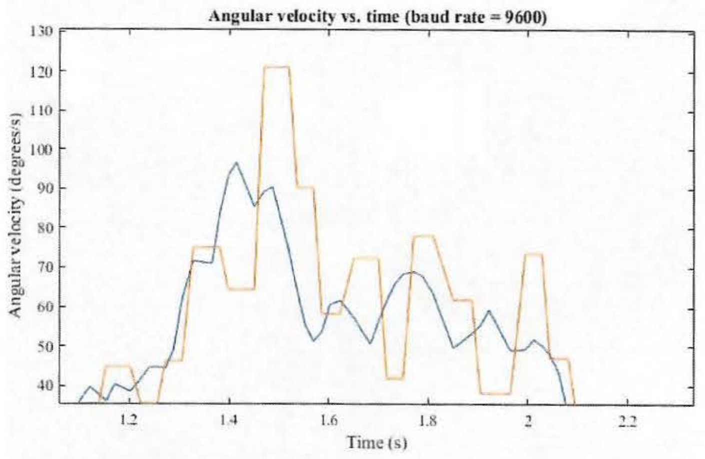

Figure 5. Detail of Figure 4. The flattened peak values are visible. The noise remains.

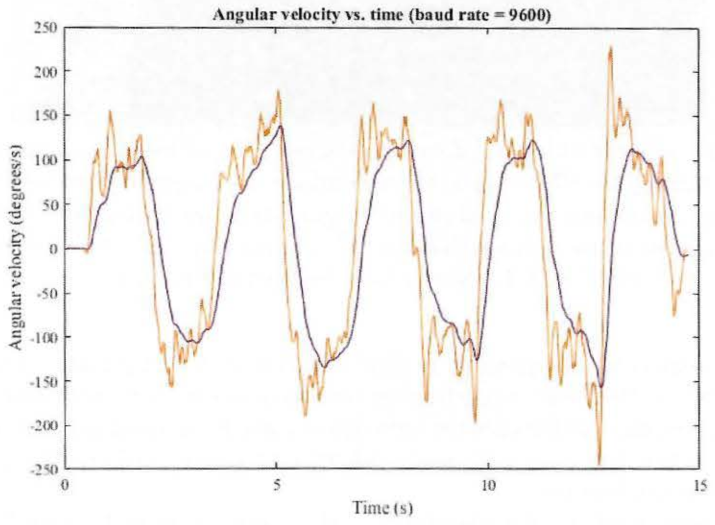

Figure 6. Angular velocity through backward finite difference method acccording to Equation 1 (red) and angular velocity estimated by the Kalman filter (purple).

Without threading, the frame rate of the whole application was stable around 60fps. In some moments, though, the frame rate dropped to values close to 30fps, while the recommended frame rate was 90fps. The most demanding process, according to Unity's Profiler, was the script responsible for the communication between the microcontroller and the game engine. This script was not only constantly reading values sent through the serial port, but also performing the Kalman filter operations, every frame. These demanding operations being executed every frame caused the frame rate to drop.

In order to maintain a high frame rate, all new readings from the serial port and Kalman filter operations were performed in a thread which was separate from Unity's event functions.

By doing this, the frame rate came back to its recommended value, around 120fps.

The author performed tests over a treadmill, in order to evaluate the overall system performance from a user point-of-view, as shown in Figure 7.

## 4. CONCLUSIONS AND FUTURE WORKS

The tracking of the angle displacement regarding the two hip joints proved to be accurate, even though the SPI protocol used was not the fastest nor the one that could provide the most accurate angle reading among the protocols available. It was possible, nevertheless, to maintain a high frame rate and a smooth angular displacement curve.

However, when it comes to the angular velocity estimation, a simple backward finite difference could not provide the same results. Even though the avatar in the simulation rarely showed abrupt movement due to velocity peaks and valleys observed in the curve, better ways of estimating the velocity from the displacement were implemented. In that way, a Kalman filter seemed feasible and could provide better angular velocity values.

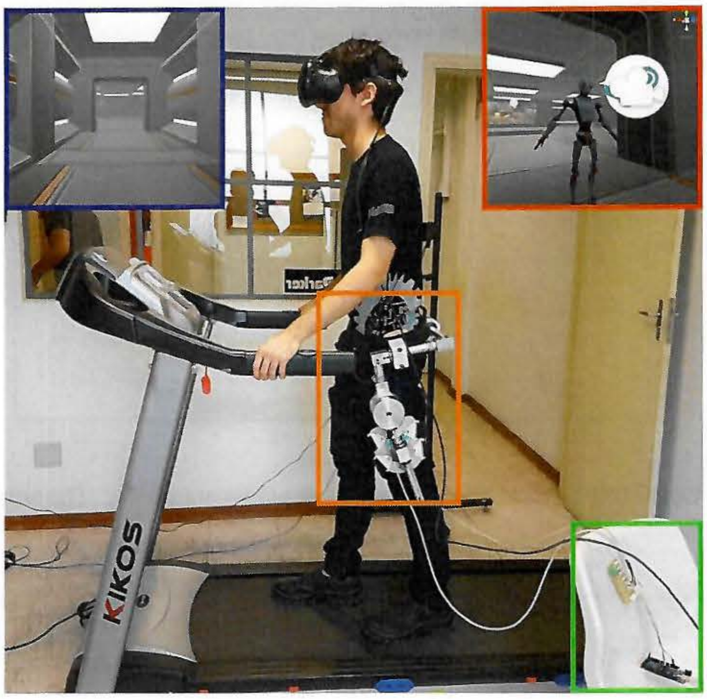

Figure 7. Author running the VR application while wearing the hip joints of the lower limbs exoskeleton. The blue box outlines the person's view wearing the HMD, while the red outlines a third-person view of the virtual environment and the user's avatar. The orange box outlines the exoskeleton; the green box outlines the microcontroller (and the supporting board for cable connections). The avatar is Space Robot Kyle, developed by Unity Technologies, whilst the scene is part of 3D Scifi Kit Starter Kit developed by Creepy Cat.

The virtual reality environment itself, regarding its attributes such as level of presence, interaction and autonomy, as defined by David Zeltzer (Zeltzer, 1992) can always be improved. However, it seems more important, by now, to guarantee an efficient and seamless communication between the virtual joints and the physical joints, maintaining a high frame rate at the same time. Therefore, when it becomes necessary to add more interactive tasks for the user, the high frame rate will be able to accommodate these new features.

Tracking the knee and ankle joints is also interesting, as the avatar would replicate their movement and, thus, enhance the user experience. The scripts also could consider the angular displacement of knee and ankle joints in order to implement a more accurate algorithm for the walking process. However, when it comes to immersion, not always it is advantageous to program the avatar to perform the very same movement performed in the real world. A walking animation may seem more immersive and engaging for the user. This has to be studied as well.

This work was the first step towards tracking the user's movement in the physical-world in order to represent it in a virtual environment using only the sensors already found in the exoskeleton. More sensors, though, can be added in order to enhance the tracking, such as inertia measurements units (IMU).

## 5. ACKNOWLEDGEMENTS

This work is supported by Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Finance Code 001, Support Program for Postgraduate and Scientific and Technological Research for Assistive Technology in Brazil (PGPTA), under grant 3457/2014, and CNPq.

## 6. REFERENCES

Crocetta, T.B., de Araújo, L.V., Guarnieri, R., Massetti, T., Ferreira, F.H.I.B., de Abreu, L.C. and de M Monteiro, C.B., 2018. "Virtual reality software package for implementing motor learning and rehabilitation experiments". *Virtual Reality*, Vol. 22, No. 3, pp. 199–209.

Grewal, M.S. and Andrews, A.P., 2001. *Kalman Filtering: Theory and Practice*. John Wiley & Sons, New York, 2nd edition.

Jerald, J., 2016. *The VR Book: Human-Centered Design for Virtual Reality*. Association for Computing Machinery and Morgan Claypool, New York, NY, USA, 1st edition.

Linowes, J., 2015. *Unity Virtual Reality Projects*. Packt Publishing, Livery Place, 1st edition.

Pons, J.L., 2008. *Wearable Robots: Biomechatronic Exoskeletons*. John Wiley & Sons, West Sussex, 1st edition.

Santos, W.M., 2017. "Design and evaluation of a modular lower limb exoskeleton for rehabilitation". In *Proceedings of the International Conference on Rehabilitation Robotics - ICORR 2017*. London, UK.

Unity Technologies, 2019a. "Unity user manual: Order of execution for event functions". Unity Documentation <https://docs.unity3d.com/Manual/ExecutionOrder.html>.

Unity Technologies, 2019b. "Unity user manual: What is multithreading?" Unity Documentation <https://docs.unity3d.com/Manual/JobSystemMultithreading.html>.

World Health Organization, 2011. "World report on disability". World Health Organization Website <https://www.who.int/disabilities/world_report/2011/en/>.

Zeltzer, D., 1992. "Autonomy, interaction, and presence". *Presence: Teleoperators and Virtual Environments*, Vol. 1, No. 1, pp. 127–132.

## 7. RESPONSIBILITY NOTICE

The authors are the only responsible for the printed material included in this paper.