A practical approach using inertial sensors applied to 3D motion tracking

Regivaldo Costa (rcosta@rclabs.com.br)

ABSTRACT

The use of inertial sensors in the field of Virtual Reality (VR), Augmented Reality (AR) and interactive games has been growing along these areas. This work aims to revisit the concepts of inertial sensors based on accelerometers and gyroscopes with digital technology, which allows us to identify with great accuracy and low latency to move and rotate an object/body in three-dimensional space, i.e. , in X, Y and Z. We will also introduce a simple use case, which will use an inertial sensor with six degrees of freedom (MPU-6050) physically represented by a dice, whose demonstration of its motion will be displayed in the Blender that is an application widely used in 3D animations. The sensor is coupled to a Raspberry PI device, where was possible to explore the communication of the sensor with Blender through a wireless network in the IEEE 802.11 standard using the UDP client/server protocol. In both sides we used the Python programming language supported by Rasbian operating system and by Blender Game Engine module. In this way, we hope to add some contribution towards to demystifying the use of this type of sensor to applications associated with VR/AR and Blender.

Keywords

Blender, BGE, Inertial Sensor, IMU, Traking Motion Tracking MPU-6050, Virtual Reality

1. INTRODUCTION

Research on Virtual Reality (VR) and Augmented Reality (AR) has intensified greatly in recent years and as a result of this process, new products and high technology devices have emerged to meet the segments of entertainment (games and movies), education, engineering, training, medical, sport, virtual environments, etc [1, 2, 3].

Among these devices there's the "Inertial Measurement Unit" (IMU) based on accelerometer and gyroscope that allows us to identify the rotation of an object in 3D space, i.e., about the axes X, Y and Z, which usually are used for the production of films and interactive games which aims to do the motion tracking of the human body, however, they are also used in other areas as cited above. Thus, this work focuses on digital devices based in accelerometers and gyroscopes with three-axis, in this case, providing a total of six degrees of freedom.

In interactive environments of VR/AR as games and in the production of movies, the information of object position needs to be sent to a processing unit to be processed in real time. Depending on the application, sending of the data may occur through a wires connection, but can also without the use of them. Considering interactive games where the player interacts with the game through sensors attached to his body, the use of wires can become inconvenient or even infeasible. Thus, to ensure an improved usability and performance, usually chooses by using wireless sensors with wifi networks.

Thus, our work also is focused on demonstrating the use of IMus with a "credit-card sized" known as Raspberry PI[1] that will allow sending data using a wireless data network through of the IEEE 802.11 standard.

In the use case to be developed, the IMU (or sensor) was molded in the physical shape of a dice, where the visual representation of the use of the IMU, i.e., its movement and orientation is displayed through application called Blender. Blender is one of the most popular free graphical tools used in 3D animations and related areas of the Virtual Reality. In the Blender we use the module called Blender Game Engine (BGE), which enables the creation of interactive 3D applications.

To allow interaction with Blender objects, BGE uses the Python programming language. Thus, we explored the use of Python to interact with the Blender objects (in this case a dice), as to the communication between the Blender and the sensor using the UDP protocol.

The sensor type used is based on a MEMS[2] device, specifically, the MPU-6050 of InvenSense[3] company, which has 3-axis accelerometer and 3-axis gyroscope, and a Digital Motion Processor (DMP) incorporated. The communication of this sensor with the other devices is through of the I^2C standard. In general, implementations like this is performed using the Arduino architecture. But to innovate, we opted for make the interfacing of the sensor with the Raspberry PI architecture, and was a motivation object to use the Python language to programming the MPU-6050, as well as by the facility to use wireless network adapter.

This work is organized as follows: the section 2 revisits contextually and succinctly some concept involved in this proposal, including applied mathematics, IMUs concepts, rotational movements of the human body, among others; in section 3 we explore architectural and implementation aspects for the IMU (sensor) and Blender; in section 4 we discuss related work; and finally a short conclusion that includes some trends to future works.

[1]https://www.raspberrypi.org
[2]http://www.memsindustrygroup.org
[3]http://www.invensense.com

2. BACKGROUND

In this section we will revisit some topics briefly in the context of our proposal, i.e., questions about 3D rotation process, as well as the utilized inertial sensor in this approach.

2.1 Rotation in 3D space

2.1.1 Rotation matrix and Euler angles

The translational and rotational movements are the two main actions of physical that denoting the motion of a rigid body (or object) in 3D space. However, as we are working with inertial sensors for applications related to virtual humans, either for use in games, like to any other application associated with the virtual reality, the movement that most interests us is the rotation, because is used in main joints of humans that are associated with rotational movement around the axes X, Y and Z, as can be seen in Table 1 below.

Thus, the rotation movements occur within the Cartesian plane, wherein the rotation can take place through a displacement and specific angular speed (in radians or degrees) around of the axes X, Y and Z. Figure 1 shows these movements considering negative and positive rotations, also known as rules of left and right hand, respectively.

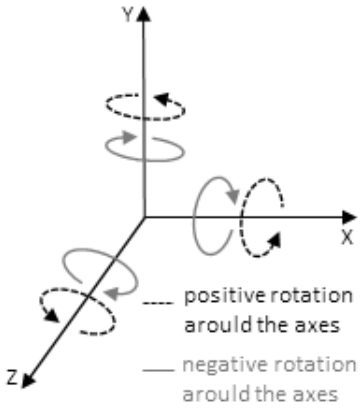

Figure 1: Rotation around of the axis X, Y e Z.

The 3D rotation movement may be mathematically represented by these three axes, and normally applies the three angels of Euler (ϕ for the X axis, θ for the Y axis and ψ for the Z axis)[4], however, there are other approaches, such as the use of quaternions [5] that we will see later, and that will also be using in our use case to avoid the problem known as "Gimbal lock" [4], that is a likely situation to occur in Euler. In practice, we can obtain the rotation matrix associated with each axis using Euler, which are shown in formulas 1 to 3.

However, the process of rotation occurs in the 3D plane with the product of these three matrices and that will also be used in our use case together with the Blender where we will use for a conversion from quaternions to a 3x3 rotation matrix.

$$Rot(\phi/roll) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & \sin\phi \\ 0 & -\sin\phi & \cos\phi \end{bmatrix} \quad (1)$$

$$Rot(\theta/pitch) = \begin{bmatrix} \cos\theta & 0 & -\sin\theta \\ 0 & 1 & 0 \\ \sin\theta & 0 & \cos\theta \end{bmatrix} \quad (2)$$

$$Rot(\psi/yaw) = \begin{bmatrix} \cos\psi & \sin\psi & 0 \\ -\sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad (3)$$

We emphasize that the symbols denoting angles adopted in this work are related to the nomenclature used in gyroscopes/accelerometers motion which is the focus of this work, as is also related to the nomenclatures **roll**, **pitch** and **yaw** to ϕ, θ and ψ, respectively.

2.1.2 Quarternions

Quaternions [5, 4] is more a mathematical model that allows us to rotate objects in 3D space. The quaternions defines a relative angular value to the axes through four vectors represented by x, y, z and w, which when applied to some mathematical transformations to the values measured on each axis, can obtain the angle, speed and direction of rotation of the object. In quaternions model not occur the "Gimbal lock" problem.

The mathematics used in quaternions is based on the principle of complex numbers and linear interpolation and due to its complexity, escapes the focus of this work. The sensor MPU-6050 and Blender Game Engine supports the use of quaternions, which will be used in our use case.

2.1.3 Rotations in the body human

In the field of Virtual Reality and interactive games, the motion tracking process of the various parts of the human body in relation the members that have joints can be obtained only by rotation.

Table 1: Rotational limits of human joints [6]

Segment	Joint	Type	X	Y	Z
Foot	Ankle	Rotational	65	30	0
Shin	Knee	Hinge	135	0	0
Thigh	Hip	Ball/Socket	120	20	10
Spine	Hip/Spine	Rotational	15	10	0
Shoulder	Spine	Rotational	20	20	0
Bicep	Shoulder	Ball/Socket	180	105	0
Forearm	Elbow	Hinge	150	0	0
Hand	Wrist	Ball/Socket	180	30	120

In Maestri [6] and Chen [7] are presented a study that shows the type of movement and the rotation angle about the main members of the humans body. Therefore, is a great reference to the use of IMU sensors. Table 1 shows these data in the three axis angular values for each type of joint.

2.2 Motion sensors

There are numerous sensors for applications to Virtual Reality that enable the tracking motion of human body, as well as any other object, however, the types most widely used due to its accuracy, ease of use and low cost, are based in cameras like the kinect and also the known as inertial, such as gyroscopes, accelerometers, compasses, etc. Sensors for facial expressions are also based on cameras associated with visual/fiducial marks, but will not be the focus of this work.

For the purpose of this work we will be addressing the inertial sensors based on accelerometer and digital gyroscope, where its construction is a MEMS (micro-electro-mechanical systems)[4], which has both sensors into a single integrated circuit and with dimensions of less than 0.5 cm^2 and 6 degrees of freedom and allows us to perceive three-dimensional movements with great precision.

With this type of sensor we can obtain information associated with kinematics of movement, enabling us to assess biomechanical movements produced by changes of speed and rotational movement within the pattern of movements of the human body with great accuracy.

MEMS sensors are robust, resistant to high mechanical stresses, however, they are sensitive to certain temperature thresholds, but also have noise problems and *drift* when associated with accelerometers and gyroscopic respectively.

2.3 Accelerometers

Digital Accelerometers are sensors based in transducers that allows us to measure accelerations in one or more axes of 2D/3D space providing the value of acceleration around these when are moving, however, when is stopped, it gives us only the direction of the gravitational acceleration vector.

Without going into details, in the digital accelerometer, the acceleration is measured by difference of potential provided by a transducer due the displacement of movable elements in the electro-mechanical set associated with a gravitational component (generally between 2g and 16g) and which, among others, a major determinant parameters of sensitivity and accuracy of the sensor. Thus, with this difference of potential some trigonometric operations can determine the acceleration and its angular position.

The noise caused by the accelerometer as previously reported is the result from the quick fluctuation of values read when the accelerometer is moving or even when it is in an inert state. The application of filters, such as Kalman [8] filter and the complementary [9] filter are necessary to achieve the accuracy of the values. Another similar technique (some people consider the same) used is the integrate the values of the accelerometer with the gyroscope, whose technique is known as fusion or integration.

2.4 Gyroscopes

Digital gyroscopes are sensors with capacitive feedback from a vibration and resonance system, which allows us to measure angular velocity (or rate of change in the orientation angle) around the axes.

The gyroscope have the *drift* [10] problem, where us will moving it from a point of reference to a new point, and when us return it to the reference angle, the value read will not be the same.

This problem can be fixed easily through an adjustment process, where is need to measures the deviation value caused by the gyroscope for each lap with associated a ΔT, in order to obtain an offset value for correction.

3. ARCHITECTURE AND IMPLEMENTATION ASPECTS

The purpose of this section is to present the practical aspects of sensor itself, i.e., the MPU-6050 in conjunction with the Raspeberry PI and Blender, among other artifacts which will be seen below.

The choice of Blender[5] came due it be used widely in the area of 3D animation (mostly), Virtual Reality and games. Thus, the Blender will be the entity that will receive the data from the sensor and process it to present graphically in 3D the object movement associated with the IMU (in this case represented as a dice).

We Also explore the TCP/IP communication between entities using Python, in both on the sides, i.e., in the Raspberry PI, as in the Blender using the BGE (Blender Game Engine). This module enables through Python the dynamic manipulation (or in real time) of objects present in Blender scene.

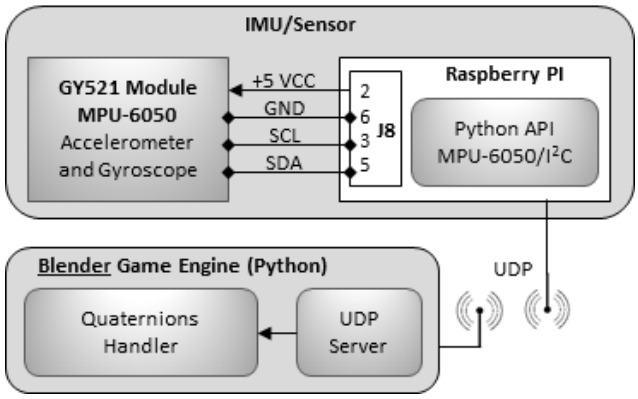

Figure 2: IMU/Sensor architecture

Figure 2 shows in simplified form the architecture developed. We used as the sensor the MEMS MPU-6050 of InvenSense[6] company, however, as this sensor communicates through a channel I^2C standard we had to interfacing it with a processing unit that has the TCP/IP stack and supports communication wireless, where was used the known "credit-card sized computer", the Raspberry PI[7].

In the architecture of Figure 2, when the Raspberry is on, it also provides 5 volts to the GY-521 module (MPU-6050) through its I/O bus (pins 2 and 6). After loading the operating system on Raspberry, the drives initializes the I^2C bus and sequentially executes the application that goes to read the MPU-6050 data and sends it over the wireless network using UDP (User Datagram Protocol) to computational resource in which Blender is running.

Data sent by the sensor through the UDP connection comprises the quaternion information (X, Y, Z and W), which are processed by Digital Motion Processor from the accelerometer/gyroscope. After initialization, the MPU-6050 performs a calibration process that lasts an average of 20s, where during this time, informed quaternion values do not match with exactness the position of the sensor.

The Blender when active, receives data quaternion, processes them and represents them through a dice.

In the following sub-items will be made an exhibition of the technical aspects and integration among the involved parties.

[4]http://www.memsindustrygroup.org/

[5]http://www.blender.org

[6]http://www.invensense.com

[7]http://www.raspberrypi.org

3.1 The inercial sensor MPU-6050

The MPU-6050 is a device (or chip) characterized like MEMS in that integrates into its casing one accelerometer and one gyroscope, both having three axes, one DMP (Digital Motion Processor) and one temperature sensor. Their communication with the outside world is through the I2C standard up to 400kHz. To avoid bottleneck and prevent losses on reading the sensor values, a FIFO buffer of 1024 bytes is available. This IMU also allows interfacing an external compass of three axes. Figure 3 demonstrates the macro block diagram of the MPU-6050.

Figure 3: Macro block diagram of the MPU-6050[8]

For each axis there is a analog to digital converter (ADC)[9] allowing accurate readings with values relating by tracking motion. The accelerometer allows it to be configured in negative/positive of gravity as 2g, 4g, 8g and 16g. The gyrocope can also be set in ranges of negative/positive as 25, 500, 1000 and 2000°/sec. Figure 4 demonstrates the orientation of the axes and the rotation polarity of the MPU-6050, following the right hand rule as already shown in Figure 1.

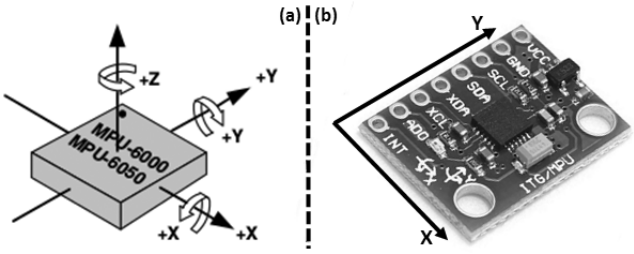

Figure 4: In (a), orientation/polarity of the axes on MPU-6050[9] **and in (b) the GY-521 module in the same position**

In our use case, we used the GY-521 module of Chinese manufacturing. This module has the MPU-6050 chip welded to a printed circuit board, which facilitates its handling and allows power supply between 3 and 5 volts. Table 2 shows the connections of the GY-521 module with the J8 connector on Raspberry PI.

Table 2: Interfacing of GY-521 with the Raspberry PI

GY-521 signals	J8 Conector on Raspberry
+5 VCC	2
GND/Ground	6
SCL	3
SDA	5

3.2 Software arterfacts

The software involved is very simple and comprises two separate parts, the sensor which is interfaced Raspberry PI and the Blender graphical tool that showing the motion of the sensor.

Still, all the artifacts involved follow the philosophy "open source". In addition to that, methods used and functions are documented in the code itself and are available on **github**[10] repository and the download can be performed as the following command (considering a unix-like environment), but is necessary to have package "git" installed:

```
$ git clone https://github.com/rclabs-dev/
imu-blender
```

The following subsections address details of the software involved in Raspberry (sensor) and Blender (graphical tool).

3.2.1 Raspberry/MPU-6050 integration

The sensor module was developed based on the GY-521 module that contains a MEMS MPU-6050 that has an accelerometer and an gisroscópio as previously mentioned.

The communication of the GY-521 with the outside world for the provide of accelerometer/gyroscope data is through standard I^2C communication. Thus to process data from the GY-521 was used at Raspberry PI platform also contains an interface to the same standard. The Figure 5 shows the Raspberry PI with the GY-521 module connected in the J8 connector.

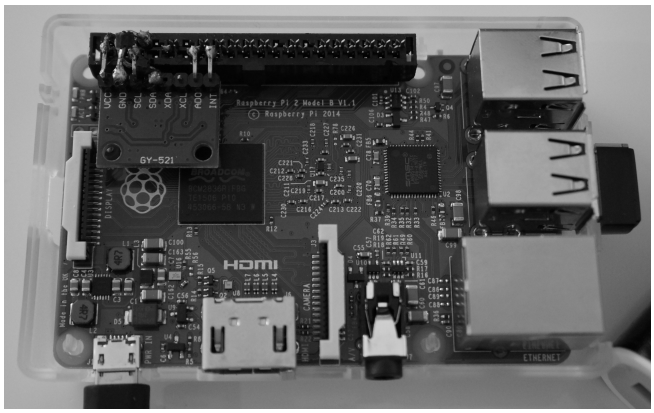

Figure 5: Raspberry PI with the GY-521 module connected in the J8 connector

The operating system used in the Raspberry was the Rasbian which is Debian release to Raspberry PI. Any references to the installation of packages, will take from standard Rasbian repository, unless there are specific indications.

For possible communication with the sensor, as well as communicating with the bus of the Raspberry two packs must be installed as follows:

```
$ sudo apt-get install python-smbus i2c-tools
```

There are numerous APIs and "wrappers" written for access to MPU-6050 and not to be "reinvented the wheel" was used the API in Python of the Stefan Kolla[11]. In the repository of the source code listed above, these artifacts are in "MPU6050" directory, which are : **mpu6050.py** file, which is the class that contains registers mapping of the MPU-6050

[8]Image from http://www.invensense.com
[9]http://goo.gl/zsNGHc
[10]https://github.com
[11]https://github.com/cTn-dev

functions to read/write; and **pycomms.py**, which is the wrapper for access to the SMBus of Raspberry PI to provide the I^2C communication.

For this, an application in Python that uses the API mentioned above, which reads the data from the sensor and sends them to the Blender through the UDP protocol was developed. The communication between the Raspberry and Blender can occur through any network connectivity available (cable or wireless) that exists between them, in our case was wireless.

The application was nominated as imu-blender.py and find the root of the repository. There is also a configuration file that is used by **imu-blender.py**, which is the **imu-blender.cfg**.

To use the application you need to edit the **imu-blender.cfg** file and make changes to two mandatory and one optional parameters, as follows:

- **REMOTE_IP (mandatory)**: is the IP address or FQDN (name associated with Internet domain names) of the computer where be running Blender (e.g., 192.168.0.1 or blender.xpto.com);
- **DST_PORT (mandatory)**: is the UDP port number as configured in Blender (the default value is 9999, but can be any other port that is not in use);
- **DEBUG (optional)**: this option can be 1 whether desired to view in Raspberry the values of X, Y, Z and W which corresponds to quaternions values obtained from the sensor. If you do not want, it must contain the value 0 (but is indicated to enable the debug in the initial tests and then disable after the tests).

Made the settings above, the application can run, simply use the following command from the folder where the application and any Linux/Shell. Blender does not need to be active to do an initial test in order to verify that the sensor is reading the information correctly.

```
$ sudo ./imu-sensor.py
```

If everything is correct, will start to run on the screen the values relating to the quaternions and if we move the sensor, the variation of the values occur in a higher range, since that in the number of decimal places occurs a big variation even without movement of sensor.

3.2.2 Blender/Blender Game Engine

Blender is a graphical tool that we use to represent real time sensor motion. Blender has a module called Blender Game Engine (BGE), which is an API that have Python as the programming language and allows us to handle any objects of the scene in real time.

So what we did was to model an object in the format of a dice where in the BGE was implemented a blocking UDP server to receive data from quaternions and using libraries "bge.logic" and "math.utils" that handles some operations, where the results are applied to the object of the scene (the dice).

To start viewing through Blender, we must bear the **imu-blender.blend** code can found in the **blender** folder at the repository. After load the code, the rendering can effectively be started by pressing the "P" key with the cursor positioned in the 3D visualization window or starting the "standalone player" (is the better option) from the properties of the preview window (or "properties").

However, to get a visual representation of the dice (or any other object in which the sensor is associated) in the computer where Blender is running to match exactly the same physical position of the dice, probably an adjust of offset will need.

This adjustment may be effected using the four arrow keys and the keys " z " and " x ". This adjustment is associated with the rotation of camera around three axes, which has the dice as point of view. For this, an "empty" object on the camera was added. Thus, it was possible to rotate the camera around the data using in the logic window, a "**keyboard** sensor", "**and** controller", and "**motion** actuators".

4. EVALUATION

In this work we summarize our evaluation in the most important requirement for this type of sensor that is the latency. Latency is the time elapsed between a motion sensor, where new quaternions values are sent to Blender and the receiver such data by Blender and their visual processing.

So we could make this evaluation, we synchronize the clock of sensor (Raspberry PI) and the computer where the Blender was running using a time service known as NTP (Network Time Protocol)[12].

We also sent a sequential number more a timestamp along with quaternions data. Thus, we collected 500 values and get a average latency of 8 milliseconds between the sensor movement and their perception by Blender.

With 8 ms of latency, we will have a average frequency of 125 frames per second, considering the laws of physics to the uniform circular movement (Freq = $1/\Delta T$). Normally, with only 30 Hz of refresh rate, humans can already see quite smoothly any movements, which is the case of TV transmissions that occur from 25 Hz or 25 frames per second.

Because the systems are distributed, we also believe there is a small margin of error due some imprecision in the time measure process by the NTP. There are other factors that can contribute, but we will not expand this discussion here.

Due to data transfer occur using the UDP protocol, where there is no acknowledgment mechanism (such as TCP), this fact contributes to the low latency. We also verified by sequence number that there was no packet loss.

As the MPU-6050 was interfaced with Raspberry PI and this could be used to centralize data to a set of sensors, for example, distributed in the joints of human body (as listed in Table 1) to allow full tracking.

For this proposal, we plug the Raspberry in a lithium battery with 2200 mAh and its autonomy of the sensor in an inert state (without movement) was approximately 5 hours continuous, which is a good reference for most applications involving virtual reality or interactive games.

5. RELATED PAPER

Below are some related work that addresses research in the context of motion tracking using inertial sensors and their correlations with techniques of rotation and its applications. As in this paper we only revisited some concepts of inertial sensors, we will not weave any criticism concerning of approaches below.

[12]http://goo.gl/8t6fYL

Motion tracking using video cameras has also been the subject of several studies, however, there some problems inherent in this technology, especially in relation the obstruction of object in the scene that can causes loss of the tracking. The partial or total option by inertial sensors has been used with some frequency, as we will see below some work using both approaches, i.e., optical sensors and inertial sensors to track the movement in the same object.

In the work "Sensor Fusion for Augmented Reality" [11] was identified problems in tracking of the object when a interruption occurs in the camera's field of view. Thus, the authors proposed a fusion data of the tracking using inertial sensors based on accelerometer/gyroscope, where the fusion or integration of data tracking obtained by the camera and the inertial sensors was possible to obtain stable values, accurate and without loss of tracking.

In the fusion technique, the authors used an algorithm based on observation model associated with "computer vision" [12], that with the fusion of the data of camera and IMU they were able to get 2D/3D coordinates of the tracked object based on data comparison and a video previously generated in the environment without the presence of the object to be tracking and then with him.

The work "Position Estimation With the low-cost Inertial Measurement Unit" [13] proposes the use and inertial sensors with 9 degrees of freedom (accelerometer, gyroscope and magnetometer) fixed in a glasses known as "Rift", which also is used for "computer vision" to eliminate the disorientation effect caused by the lack of connection between the physical movement of the tracked object (in this case a person) in relation to the virtual environment achieved by the "Rift".

The authors also used the sensor data fusion techniques associated with filters for removing noise and drift problems already reported in this work. Thus, they reached a satisfactory result which was graphically demonstrated by Unit3D, where the rotation data were also processed using quaternions.

This other work, the "Real Time Human Motion Capture Driven By Wireless Sensor Network" [7], the authors propose a method based on the "surface model theory" [14] to build 3D models of human skeletons, where the representation of movements occurs using inertial sensors based on MEMS MPU-9150 and wireless communication.

The work focuses on the technical aspects of the sensors and filters using techniques of integration/fusion, as well as a study about the angular movement of the joints of various parts of the human body similar to that shown in Table 1.

To validate the work the authors compare real movements of arms from video images generated by the sensors in order to demonstrate the accuracy of angular movements. Also evaluated the latency in data transfer between the sensor and the graphic visualization system that was 20ms, which was slightly up on the results we present in our approach, however, they have used Bluetooth technology which notably has lower performance networks than the IEEE 802.11 standard.

6. CONCLUSION

Inertial sensors is a reality in many areas, such as mobile phones, tablets and industries of the entertainment, robotics, aviation, automotive, among others.

In the areas of Virtual Reality (VR) and interactive games the inertial sensors are used to perform the motion tracking of human body, which are usually positioned near the joints in order to identify the angular movement of these parts (see Table 1) and represent it in a graphics application with specific objectives or through an interactive game.

Aiming to explore the use of these devices, also known as IMU (Inertial Measurement Unit) based in MEMS (Micro-Electro-Mechanical Systems) technology we present a practical approach, where was used data from one accelerometer and one gyroscope to identify the motion of a body/object in 3D space.

For the representation of movement, the graphical tool used was Blender, which is widely used for 3D animation, where the sensor was represented as a dice and allowed us to observe its movement in 3D.

Thus, we assess that the current MEMS devices are quite suitable for use in the tracking of human movements, because due to their small size (less than 0.25 cm^2) and low latency (average of 8 ms) that allows us a rate of 125 frames per second and is therefore ideal for use in VR and interactive games.

Inertial sensor as used in this approach is a low-cost device (more or less three euros), wherein the Raspberry PI is no more than 40 euros and can be used as a point of concentration of the various sensors positioned by the body or any other object that to be detected its motion in 3D, therefore, allows working easily in any project.

Images of the prototype used as the sensor (Raspberry + GY-521 + battery) and the box that settled the sensor in the form of dice, as well as a demo video can be viewed starting repository previously mentioned.

7. REFERENCES

[1] J. Marco. List of haptic controllers under development for virtual reality. `http://goo.gl/AKdPfW`. [Online; last access May/2015].

[2] Best Performance Group. Future inertial sensor sports science. `http://goo.gl/K3xpzX`. [Online; last access May/2015].

[3] L. Kozlowski. The future is in motion: Virtual reality gloves puts control in your hands. `http://goo.gl/y9k3gC`. [Online; last access May/2015].

[4] Vincent Lepetit and Pascal Fua. Monocular model-based 3d tracking of rigid objects: A survey. *Foundations and trends in computer graphics and vision*, 1(CVLAB-ARTICLE-2005-002):1–89, 2005.

[5] R. Hartley, J. Trumpf, Y. Dai, and H. Li. Rotation averaging. *International Journal of Computer Vison*, 10(2):41, 2013.

[6] G. Maestri. *Digital Character Animation*. New Riders Publishing, 1996.

[7] Man Luo Peng-zhan Chen, Jie Li and Nian hua Zhu. Real-time human motion capture driven by a wireless sensor network. *International Journal of Computer Games Technology*, page 14, 2015.

[8] H.J. Luinge, P.H. Veltink, and C.T.M. Baten. Estimation of orientation with gyroscopes and accelerometers. In *[Engineering in Medicine and Biology, 1999. 21st Annual Conference and the 1999 Annual Fall Meetring of the Biomedical Engineering Society] BMES/EMBS Conference, 1999. Proceedings of the First Joint*, volume 2, pages 844 vol.2–, Oct

1999.

[9] M. Euston, P. Coote, R. Mahony, Jonghyuk Kim, and T. Hamel. A complementary filter for attitude estimation of a fixed-wing uav. In *Intelligent Robots and Systems, 2008. IROS 2008. IEEE/RSJ International Conference on*, pages 340–345, Sept 2008.

[10] Daniel Roetenberg, Henk J. Luinge, Chris T. M. Baten, and Peter H. Veltink. Compensation of magnetic disturbances improves inertial and magnetic sensing of human body segment orientation. *IEEE Transactions on Neural Systems and Rehabilitation Engineering, [seealsoIEEE Transactions on Rehabilitation Engineering*, 13:395–405, 2005.

[11] J.D. Hol, T.B. Schon, F. Gustafsson, and P.J. Slycke. Sensor fusion for augmented reality. In *Information Fusion, 2006 9th International Conference on*, pages 1–6, July 2006.

[12] Lin Chai, William A. Hoff, and William A. Hoff. 3-d motion and structure estimation using inertial sensors and computer vision for augmented reality. *Presence*, 11:474–492, 2000.

[13] G. Llorach, A. Evans, J. Agenjo, and J. Blat. Position estimation with a low-cost inertial measurement unit. In *Information Systems and Technologies (CISTI), 2014 9th Iberian Conference on*, pages 1–4, June 2014.

[14] Xiaoli Meng Gang Li, Zheng Wu and Jiankang Wu. Modeling of human body for animation by micro-sensor motion capture. In *Knowledge Acquisition and Modeling, 2009. KAM '09. Second International Symposium on*, volume 3, pages 98–101, Nov 2009.