Position Estimation with a low-cost Inertial Measurement Unit

Gerard Llorach, Alun Evans, Javi Agenjo, Josep Blat
Interactive Technologies Group
Universitat Pompeu Fabra
Barcelona, Spain
{gerard.llorach, alun.evans, javi.agenjo, josep.blat}@upf.edu

Abstract— **In this paper the Inertial Measurement Unit (IMU) included inside the Oculus Rift virtual reality headset is considered for head position tracking. While the Oculus is capable of mapping rotational movement to a virtual scene, recovering translational movement is not possible by default. In this study, we extract position data using a different approach for real-time position tracking with double integration, as well as a new method for gravity compensation for accelerometers with different axis calibrations. The proposed tracking system is portable, simple and does not require a controlled environment.**

Keywords-component; Oculus Rift; IMU; INS

I. Introduction

The majority of existing solutions for free movement inside a virtual reality world, such as that presented to a user by the Oculus Rift headset [1], require the use of external devices such as game controllers. While the headset is capable of mapping rotational head movements into the displayed scene, translational movement of the user is not recorded, and the user usually remains seated. This partial immersion in a virtual world can result in some users suffering from disorientation and motion sickness, which occurs when movement inside the virtual environment is not correlated with the user's physical movement, and is due to the sensation of motion without actually moving [2]. Thus, there is a need to enable free movement in the virtual environment through position tracking, mapping the user's physical movement to the virtual domain.

There are two main approaches to the general task of position tracking: computer vision and inertial navigation systems with/without external sensors. Computer vision solutions (such as [3][4]) require a controlled environment or previous information about the environment, whereas Inertial Navigation Systems (INS) make use of accelerometers, gyroscopes and other sensors to track the position via dead reckoning. INS systems are usually integrated with other data sources (such as GPS or human input) to correct any errors [5], or are combined with gait analysis of the data if the motion follows a pattern (such as pedestrian walking) [6][7]. Dead reckoning with low-cost and small devices, such as microelectromechanical systems (MEMS), is prone to error due the drift caused by the inaccuracy of the measured data [8]. IMUs with high accuracy are more expensive and are too heavy for most human-oriented applications [9].

This study focuses on creating an INS using a portable, low-cost IMU, such as that found inside the Oculus Rift. To our knowledge, there are no reported experiments of position tracking that are successful with such a low-cost IMU. The end goal of work is to remove the disorientating effects caused by the lack of a link between physical and virtual movements. Thus, one key prior assumption of our study is that 100% accurate position tracking is not absolutely necessary, as previous studies have demonstrated [10] that blindfolded human subjects can only approximate their position after moving in open space.

The principal contribution of this paper is the development of a method of interaction and position estimation with a low-cost IMU, using gravity compensation, double integration methods, filtering and drift correction. Early testing on a development version of the Oculus Rift demonstrated that the measured gravity changes depending on the orientation of the on-board accelerometers. Our system compensates for this variable measurement using a cloud of samples which is spherically interpolated in real-time. We extract acceleration using double integration methods similar to Ribeiro et al. [11] and Slifka [12] and treat the signal with a series of filters to remove motion artifacts and compensate for drifting. All the data is obtained through Unity3D [13] plug-ins developed and provided by Oculus VR, and is processed with a laptop computer which also serves to power the Oculus unit. We present results showing that the system correctly tracks translational movement when the user makes short, marked movement, yet is prone to error when dealing with slow movements or motions with constant speed.

II. Overview

This study is based on the IMU inside the Oculus Rift Development Kit. It is a 9-axis tracker (gyroscope, accelerometer and magnetometer) with a 1kHz update rate and 2ms latency. The IMU, developed by Adjacent Reality, already carries out sensor fusion [5] which makes it capable of absolute head orientation tracking without any drift.

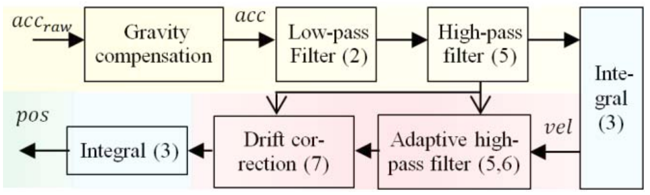


Figure 1. Overview of the system proposed for position tracking. Yellow sections represent acceleration measurement, blue is integration of signal, and red is velocity.

Fig. 1 shows an overview of our approach. Firstly we compensate for gravity measured via the IMU. Then we apply a series of filters to compensate for various motion artifacts. After applying the first integration, we filter again and correct for drifting. Finally we integrate a second time to retrieve the final position.

III. Gravity Compensation

The first step in our method is to isolate and remove the gravity component. The earth's gravity produces a constant groundward acceleration, which needs to be subtracted because, in a virtual world, it simulates a constant free fall of the body. The acceleration is given to us by body coordinates. It needs to be translated to world coordinates in order to remove the gravity from the Y axis. The orientation of the Oculus Rift is given by a quaternion inside Unity3D. The data is translated from body coordinates to world coordinates using a rotation matrix Gravity can be removed now by subtracting the value from the Y axis. Much of the previous research on this subject chooses to subtract a constant gravity value [14]. Early tests with an Oculus Rift development kit showed the acceleration due to gravity varied depending on the orientation, from 9.59 to 10.08 m/s^2, due a lack of calibration of the accelerometer axis of the Adjacent Reality Tracker.

A. Calibration

If the gravity component is not properly removed it creates an offset in the body acceleration. Offsets in acceleration are problematic as they cause a constant increase in the velocity and a huge drift in position [15]. To calibrate the gravity a set of points is stored and then interpolated. There are three steps to the calibration process. The first is to measure all the different possible orientations of the Oculus, and represent them as a cloud of points on the surface of a sphere. The second step is to interpolate and map this point cloud to a regular set of points represented as a sphere mesh. This enables the final step, which is to use the sphere as an input for the final tracking algorithm, using GPU triangle interpolation to find any value on the surface of the sphere suggested by [16]. We use this approach because reading from the surface of a sphere is simple, fast and computationally cheap, rather than constantly interpolating between samples. A set of 820 points containing the gravity value and correspondent orientation is stored for further interpolation. Gravity values are obtained by manually rotating the device, and storing a value when no other values are detected within five degrees. As the raw signal from the device is noisy, before measuring a value the signal is filtered with an exponential moving average (EMA) filter with a 0.1 coefficient. The filter can be expressed like this:

$$y_n = y_{n-1}(1-\alpha) + x_n\alpha \qquad (2)$$

where y_n is the filtered sample, y_{n-1} is the previous filtered sample, x_n is the new sample and α is the filter coefficient. Fig. 2 shows four viewpoints of the set of points obtained during this step.

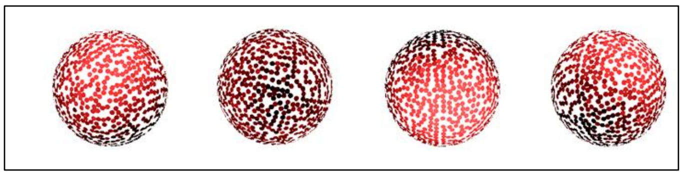


Figure 2. Different points of view of the cloud of 820 points representing the measured gravity according to the orientation of the Oculus. Darker points indicate a higher gravity value.

B. Interpolation

The measured points are now interpolated onto the vertices of a sphere of 60 horizontal lines and 30 vertical rings. The interpolation method used is the Inverse Distance Weighting (IDW) with 16 neighbours (n) and a power parameter (p) of 1. Radial Basis Functions should be considered for better interpolation results.

Now that each vertex of the sphere is assigned a value, we can obtain a value for any orientation of the Oculus Rift unit, by interpolating between the points of the sphere. As the values are stored as the vertices of a regular triangular mesh, interpolated values for any point can be obtained quickly via the GPU. These values are the gravitational force that we subtract from the Y axis when carrying out the steps in Sections IV and V.

IV. Signal Processing

In an ideal world, the position of the unit could be obtained directly via the double integration of the acceleration measured via the sensor (once gravity is successfully removed). However, low-cost accelerometers do not have the required precision, and so to estimate the position the signal needs to be treated. Unfortunately, for each filtering step applied, the system will lose the ability to track a certain kind of motion. These particular cases are explained in Section IV.C below. Much of the related work in this field processes the signal in the frequency domain [11][12]. However, working in the frequency domain is problematic in this case, because the Adjacent Reality Tracker signal rate is too low for real-time processing – typically, real time signals which are treated in the frequency domain have a much higher rate, such as the 44.1 kHz rate of an audio signal.

A. Integration

The proposed integration methods are the trapezoidal integral and the Simpson's rule method mentioned in [11][12]. Although these studies suggest better results if the integration is performed in the frequency domain, the real-time requirement of our situation means that this is not possible as it would introduce delay. Our tests show that there is very little difference between the results obtained using the two integration methods mentioned above. However, given that the trapezoidal method does not make use of a future sample, it is faster than Simpson's rule and therefore was chosen for this work.

B. Filters

The signal is itself very jittery and contains low frequency oscillations. Finite Impulse Response (FIR) filters are discarded as they introduce a delay in the signal. As mentioned above, working in the frequency domain is also not feasible because of the sampling rate and real-time constraints. The first filter implemented is the EMA, a low-pass filter. The coefficient has a range from 0 to 1, both excluded. The closest to zero it is, more delay and averaging will be introduced. The equation for the filter (1) can be found in Section III.A above. The high-pass filter used has the inverse response of the EMA. The bigger the cutoff frequency of the filter is, the fastest the signal will decay to zero.

$$y_n = y_{n-1}\alpha + (x_n - x_{n-1})\alpha \qquad (5)$$

where $\alpha = \frac{RC}{RC+\Delta t}$, $RC = \frac{1}{2\pi f_c}$, Δt is the time step and f_c is the cutoff frequency of the filter.

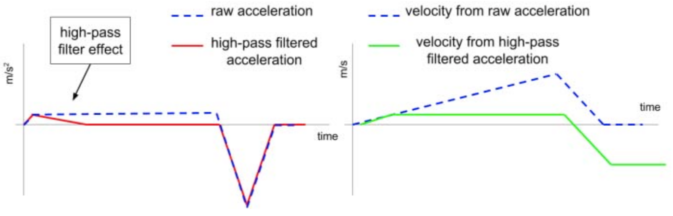
In this case, the adaptive high-pass filter changes its cutoff frequency according to the magnitude of the body's acceleration. The formula is the same as the high-pass filter (5) and the parameter that changes is the cutoff frequency:

$$f_c = \begin{cases} \frac{1}{|acc|\,p} & for\ |acc| > \frac{1}{p} \\ 2 & otherwise \end{cases} \qquad (6)$$

where $|acc|$ is the absolute value of the filtered acceleration and p is the adaptive high-pass filter factor.

C. *Signal Processing Workflow*

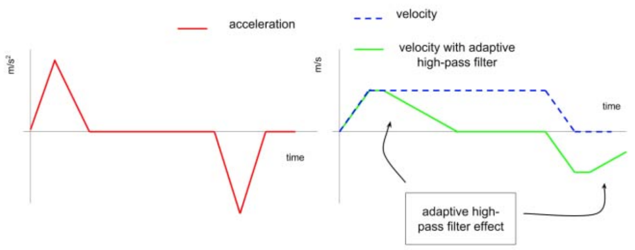
Our first filtering step is to use the EMA to smooth the signal. Choosing the right coefficient is crucial, as it will affect all the rest of the process. This filter, used with the right coefficient, does not adversely affect the position tracking. Slifka [12] suggests a series of high-pass filters for acceleration, velocity and position. These filters are proposed to remove any unknown initial conditions in the signal. We tested each of these filters and chose only to use the acceleration filter and a variation of the velocity filter. Using a high-pass filter for the position is discarded as it would make the signal decay to zero. This first high-pass filter is used to remove a possible offset that may be present in the acceleration signal due to gravity compensation or residual averaged acceleration. The result of this step is that motions with small accelerations will be not considered, and motions with a smooth start and a sharp end will produce velocity in the opposite direction (Fig. 3).



high-pass filter effect
raw acceleration
high-pass filtered acceleration
velocity from raw acceleration
velocity from high-pass filtered acceleration
m/s²
m/s
time
time

Figure 2. Effect of the high-pass filter on the acceleration and velocity. The graph on the left shows the acceleration and the graph on the right its integral. A negative constant velocity is generated.

Occasionally, a small offset remains in the velocity, causing constant movement. This problem is resolved through the adaptive high-pass filter applied to the velocity. This filter helps to remove some of the offsets created by bad measurements or the previous filters. A clear example would be when the device is tapped or hit by something: the previous filters won't cancel this high and fast acceleration changes and will produce undesirable velocity offsets. The drawback of this filter is that motions that have a constant velocity will not be properly estimated (Fig. 4).



acceleration
velocity
velocity with adaptive high-pass filter
adaptive high-pass filter effect
m/s²
m/s
time
time

Figure 3. Effect of the adaptive high-pass filter on the velocity. The graph on the left shows the acceleration and the graph on the right its integral.

V. Drifting

Previous research has presented several interesting methods for correction of drifting, such as using a high-order polynomial function to approximate the displacement drift [15]. Unfortunately this is all done in post-processing and is not suitable for our real-time application. The majority of the drifting is caused by orientation changes as there is an acceleration component thus the adaptive high-pass filter does not stop velocity offsets. Static situations and movements that do not imply an orientation change do not cause too much drifting and are easy to estimate within this system. Setting the optimum variables for the system involves an inherent compromise. The drifting caused by fast, continuous, changes in orientation can be large and needs to be corrected. The adaptive high-pass filter solves this problem, but if the orientation changes during too long a time period, then the offset is too big to be cancelled and the position cannot be estimated properly (Fig. 5).

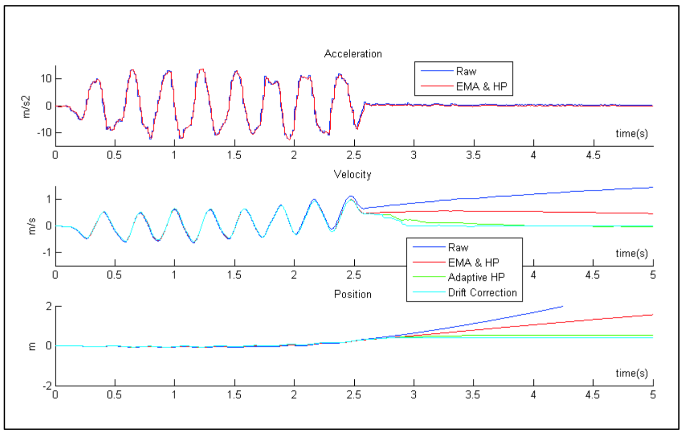


Figure 4. Recorded signal when orientation changes fast. The first plot represents the acceleration, the second the velocity and the third the position from the same axis. The plot shows how each step affects the signal.

In our system we decided to decay the velocity to zero when the acceleration is lower than a threshold. The formula (7) of the threshold function is:

$$v = v \cdot f \qquad (7)$$

$$f = \begin{cases} r^{t-|acc|} & for\ \ |acc| < t \\ 1 & otherwise \end{cases} \qquad (8)$$

where v is the velocity, f is the stop factor, r is the range of the factor, t is the threshold and $|acc|$ is the magnitude of the filtered acceleration. The range of the factor can have values from 0 to 1, both excluded. If the acceleration generated is smaller than the threshold, the velocity will decay rapidly to zero. It has a similar effect as the adaptive high-pass filter (Fig. 6).

VI. Results

The initial results concerning the position tracking are better than expected, considering the constraints of real-time processing and a low-cost IMU. The user is able to move around the virtual environment, with some restrictions. When the user takes separate, marked steps, the system works as planned and the position is tracked. Drifting caused by orientation changes is eliminated effectively, and only a small amount of displacement can be noticed when the user finishes the movement. However, long movements where velocity is constant do not work very well, as the system counts it as a drifting movement and compensates for it. Likewise, slow movements that generate little acceleration are not properly tracked as they are considered an offset of the measurement. The system is computationally light and does not cause delays or latencies while being used in a 3D scene.

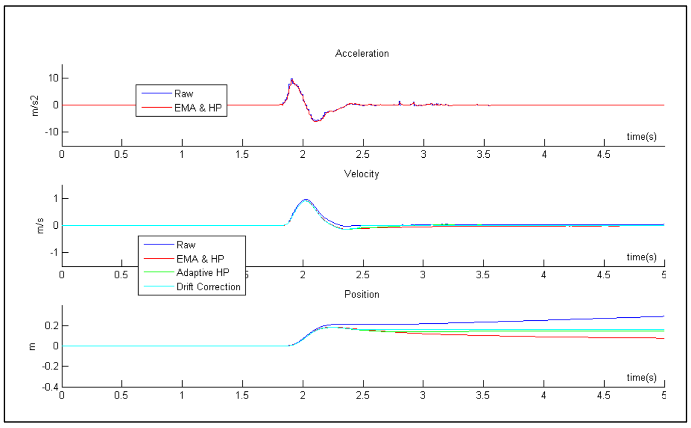


Figure 5. Properly tracked motion. The result of the system (cyan) avoids drifting and gets the more approximate result. The values chosen for the filters in this case are 0.2 coefficient for the EMA, 0.1 Hz for the high-pass filter, 50 for the adaptive high-pass filter factor, 0.4 for the range and 0.4 for the threshold of the drift correction.

VII. Conclusions and Future Work

The main goal of this study is to allow the user to move in the virtual environment. While the implemented system does not track the exact user's position, and does not allow a perfectly natural interaction in the virtual world, it does allow step by step movement inside a virtual environment using only an Oculus Rift and connected laptop. No power cable is required as the Oculus can be powered via USB connection to the laptop.

The Oculus unit needs to undergo a calibration step prior to be using with our developed system. One of our future tasks is to determine whether the differing gravity values (which necessitate the calibration) are unique to each individual unit, or whether calibration carried out for one Oculus unit is also valid for another. Furthermore, in this study the Allan variance [17] is not tested, meaning that changes in temperature may affect the recorded values from the sensor, and thus gravity is not properly subtracted. Other IMUs could be used, but some steps should be added, like sensor fusion, and others should be removed, like gravity compensation if the accelerometer is properly calibrated.

The next step for this project is testing the system to measure user reactions, to test whether the sensation of motion sickness is eliminated.

Acknowledgment

This work has been partially funded by the Spanish Ministry of Science and Innovation (TIN2011-28308-C03-03), and the IMPART FP7 European Research project (http://impart.upf.edu)

References

[1] Oculus VR, Oculus Rift Development Kit, Retrieved December 2013, “http://www.oculusvr.com”

[2] JT Reason, JJ Brand, “Motion Sickness,” Academic Press, January 1975.

[3] Lin Chai, William A. Hoff and Tyrone Vincent, “3D Motion and structure estimation using inertial sensors and computer vision for Augmented Reality,” Presence vol. 11 pp. 474–492, 2000.

[4] Hanna Nyqvist and Fredrik Gustafsson, “A high–performance tracking system based on a camera and IMU,” 16th International Conference on Information Fusion, Turkey, July 2013.

[5] S. Sukkarieh, E.M. Nebot and H.F. Durrant-Whyte, “A high integrity IMU/GPS navigation loop for autonomous land vehicle applications,” Robotics and Automation, IEEE Transactions vol.15 I. 3, Jun 1999.

[6] Oliver J. Woodman, “Pedestrian localisation for indoor environments,” University of Cambridge, September 2010.

[7] E. Foxlin, “Pedestrian tracking with shoe-mounted inertial sensors,” Computer Graphics and Applications, IEEE vol. 25 I. 6 Nov.-Dec. 2005

[8] D. Vissière, A. Martin and N. Petit, “Using distributed magnetometers to increase IMU-based velocity estimation into perturbed area,” 46th IEEE Conference on Decision and Control, USA, December 2007.

[9] Paul D. Groves, Principles of GNSS, Inertial, and multisensor integrated navigation systems, Artech House, 2013

[10] I. Israel, N. Chapuis, S.Glasauer, O. Charade, and A. Berthoz, “Estimation of passive horizontal linear whole-body displacements in humans,” AJP-JN Physiol vol. 70 no. 3 pp. 1270-1273, September 1993.

[11] J.G.T. Ribeiro, J.T.P. Castro and J.L.F. Freire, “New improvements in the digital double integration filtering method to measure displacements using accelerometers,” 19th International Modal Analysis Conference, IMAC XIX 2001.

[12] Lance D. Slifka, “An accelerometer based approach to measuring displacements of a vehicle body,” Horace Rackham School Of Graduate Studies of the University of Michigan, 2004.

[13] Unity Technologies, Unity3D, Retrieved December 2013, “http://unity3d.com”

[14] S. Reuveny and M. Zadik, “3D motion tracking with gyroscope and accelerometer,” The Rachel and Selim Benin School of Computer Science and Engineering, Hebrew University of Jerusalem, Israel.

[15] M. Arraigada and M. Partl, “Calculation of displacements of measured accelerations, analysis of two accelerometers and applications in road engineering,” 6th Swiss Transport Research Conference, Switzerland, March 2006.

[16] F. Ye, X. Shi, S. Wang, Y. Liu, S. Y. Han, “Spherical interpolation over graphic processing units,” Proceedings of the ACM SIGSPATIAL Second International Workshop on High Performance and Distributed Geographic Information Systems, pp. 38-41, USA, 2011

[17] V. Choudhary and K. Iniewski, MEMS: Fundamental Technology and Applications, CRC Press, 2013, pp. 86, pp.403.