Uma abordagem prática com sensores inerciais aplicado ao rastreamento de movimentos 3D

Regivaldo Costa (rcosta@rclabs.com.br)

ABSTRACT

A utilização de sensores inerciais no campo da Realidade Virtual (RV), Realidade Aumentada (RA) e de jogos interativos tem sido um realidade crescente junto a essas áreas. Assim, este trabalho tem como objetivo revisitar os conceitos dos sensores inerciais baseados em acelerômetros e giroscópios com tecnologia digital, os quais nos permite identificar com grande precisão e baixa latência a movimentação e rotação de um objeto/corpo no espaço tridimensional, isto é, nos eixos X, Y e Z. Também apresentaremos um simples caso de uso, onde usaremos um sensor inercial com 6 graus de liberdade (MPU-6050), representado fisicamente por um dado, cuja demonstração do seu movimento será visualizado na ferramenta Blender, esta largamente utilizada em aplicações 3D. O sensor foi acoplado a um Raspberry PI, onde foi possível explorarmos a comunicação do sensor com o Blender através de uma rede sem fios no padrão 802.11 usando a arquitetura cliente/servidor e o protocolo UDP, tendo como base a linguagem de programação Python e o módulo Blender Game Engine. Desta forma, esperamos agregar algum contribuição no sentido de estarmos desmistificando o uso desse tipo de sensor para aplicações associadas com RV/RA e Blender.

Keywords

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

1. INTRODUÇÃO

Investigação em Realidade Virtual (RV) e Realidade Aumentada (RA) tem se intensificado imensamente nos últimos anos e como resultado desse processo, novos produtos e dispositivos de alta tecnologia tem surgido para atender os segmentos de entretenimento (jogos e filmes), educação, engenharia, treinamento, medicina, esportes, etc [1, 2, 3].

Dentre esses dispositivos, temos os "Inertial Measurement Unit" (IMU) baseado em acelerômetro e giroscópio que nos possibilita identificar a rotação de um objeto no espaço 3D, isto é, em torno dos eixos X, Y e Z, e que geralmente são utilizados à produção de filmes e jogos interativos com o objetivo de efetuar o rastreamento do movimento do corpo humano, entretanto, também são utilizados na indústria da aviação e automobilística, entre outras áreas. Este trabalho tem como foco apresentar o uso acelerômetros e giroscópios digitais de três eixos, que nesse caso, nos fornecerá 6 graus de liberdade.

Em ambientes interativos de RV/RA como jogos e na produção filmes, as informações de posicionamento do objeto necessita ser enviado a uma unidade de processamento para que sejam tratados em tempo real. Dependendo da aplicação, o envio desses dados poderá ocorrer através da interconexão de fios, como também sem o uso deles. Considerando jogo interativos, onde o jogador interage com este através de sensores presos ao seu corpo, o uso de fios pode se tornar inconveniente ou até mesmo inviável. Assim, para garantir melhor usabilidade e desempenho, normalmente opta-se pelo uso de sensores sem fios, ou seja, usando redes "wifi".

Assim, este nosso trabalho também se concentrou em demonstrar o uso dos IMUs utilizando um segundo dispositivo que possibilitasse o envio dos dados usando uma rede de dados sem fios com o uso do padrão IEEE 802.11 através do Raspeberry PI.

O caso de uso a ser desenvolvido, o IMU foi moldado no formato físico de "dado"), onde a representação visual do uso do IMU, ou seja, a sua movimentação/orientação será visualizada na ferramenta Blender. O Blender é uma das ferramentas gráficas gratuita mais conhecidas à produção de animação gráficas em 3D. No Blender utilizamos o módulo Blender Game Engine (BGE), que possibilita a criação de aplicações interativas.

Para permitir a interação com os objetos do Blender, o BGE utiliza a linguagem de programação Python. Desse modo também exploramos o uso da linguagem Python, tanto para interagir com os objetos do Blender (neste caso o dado que será previamente construído), como à comunicação entre o Blender e o sensor usando a arquitetura TCP/IP, e mais especificamente usando o protocolo UDP.

O sensor utilizado tem como base um dispositivo do tipo MEMS[1], mas especificamente, o MPU-6050 da empresa InvenSense[2], que possui acelerômetro e giroscópio de 3 eixos, além de um Digital Motion Processor (DMP) incorporado. A comunicação desse sensor com o mundo exterior se dá através de um barramento no padrão I^2C. Em geral, implementações similar a esta são executadas usando a arquitetura arduino. Mas para inovarmos, optamos em interfacear o sensor e processar seus sinais utilizando a arquitectura Raspberry PI[3] e que também foi objeto de motivação para uso da linguagem Python, como pela facilidade para interfacear adaptadores para redes sem fio.

Assim, este trabalho está organizado da seguinte forma: a seção 2 revisita contextualmente e de forma sucinta alguns conceito envolvidos nesta proposta, que inclui a matemática

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

aplicada, conceitos de IMUs, movimentos rotacionais do corpo humano, entre outros; a seção 3 que explora aspectos arquiteturais e de implementação relativo ao IMU e Blender; a seção 4 aborda os trabalhos relacionados; e por fim, uma breve conclusão que inclui exposição de melhorias e trabalhos futuros.

2. FUNDAMENTAÇÃO

Nesta seção iremos revisitar alguns tópicos de forma sucinta no contexto da nossa proposta, isto é, algumas questões envolvidas com o processo de rotação de objetos em 3D e particularidades do sensores ora utilizados nesta abordagem.

2.1 Rotação no espaço 3D

2.1.1 Matriz de rotação e ângulos de Euler

Os movimentos de translação e rotação são as duas principais ações de ordem física que denota o movimento de um corpo rigido (ou objeto) no espaço 3D. No entanto, como estamos a trabalhar com sensores inerciais para aplicações relacionadas com humanos virtuais, seja para o uso em jogos, como qualquer outra aplicação associado à realidade virtual, o movimento que mais nos interessa é o de rotação, pois o seu uso na principais articulações dos seres humanos estão associados ao movimento de rotação ao redor dos eixos X, Y e Z conforme poderá ser visto na Tabela 1 a seguir.

Assim, os movimentos de rotação ocorrem dentro do plano cartesiano, onde a rotação pode ocorrer através de um deslocamento e com determinada velocidade angular (em radianos ou graus) ao redor dos eixos X, Y e Z. A Figura 1 mostra esses movimentos com rotação negativa e positiva, também conhecido como regras da mão esquerda e direita, respectivamente.

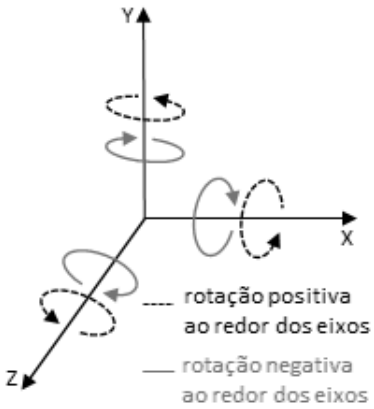

Figure 1: Rotação ao redor do eixos X, Y e Z.

O movimento de rotação em 3D pode ser representado matematicamente através desses três eixos, sendo que, normalmente se aplica os três ângulos de Euler (ϕ para o eixo X, θ para o eixo Y e ψ para o eixo Z) [4], entretanto, há outras abordagens, como por exemplo, o uso de quaterniões [5] que veremos mais adiante, e que também estaremos utilizando em nosso caso de uso de modo a evitar o problema conhecido como "guimbal lock" [4] passível de ocorrer na abordagem matemática de Euler. Na prática, usando Euler podemos obter as matrizes de rotação associada a cada eixo, as quais são apresentadas nas fórmulas 1 a 3, entretanto, o processo de rotação no plano cartesiano ocorre com o produto das três matrizes e que também será usado no caso de uso juntamente com o Blender num processo de conversão quaterniões/matriz.

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

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

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

Enfatizamos que os simbolos que denotam ângulos ora adotados neste trabalho estão relacionados com a nomenclatura utilizada no movimento de giroscópios e que é o foco deste trabalho, como também está relacionado com as nomenclaturas **roll**, **pitch** e **yaw** para ϕ, θ e ψ respectivamente.

2.1.2 Quarterniões

Quaterniões [5, 4] é mais um modelo matemático que nos permite rotacionar objetos no espaço 3D. Os quaterninões define um valor angular relativo aos eixos através de quatro vetores representadas por x, y, z e w, que quando aplicado a algumas transformações matemáticas em relação aos valores medidos em cada eixo, consegue se obter o angulo, velocidade e direção de rotação do objeto. O modelo de quaterniões não está sujeito ao problema do "guimbal lock".

A matemática utilizada em quaterniões se utiliza do princípio de números complexos e interpolação linear e devido a sua complexidade, foge do foco desse trabalho. O sensor MPU-6050 e o Blender Game Engine suporta o uso de quaterniões, os quais serão utilizados em nosso caso de uso.

2.1.3 Rotações no corpo humano

Em Realidade Virtual e jogos interativos, o processo de track da movimentação das diversas partes do corpo humano no que tange as partes que possuem articulação/junta podem ser obtidas apenas pelo movimento de rotação.

Table 1: Limites rotacionais das juntas do corpo humano [6]

Membro	Junta	Movimento	X	Y	Z
Pé	Tornozelo	Rotacional	65	30	0
Canela	Joelho	Dobradiça	135	0	0
Coxa	Quadril	Bola/Soquete	120	20	10
Espinha	Espinha	Rotacional	15	10	0
Ombro	Ombro	Rotacional	20	20	0
Biceps	Ombro	Bola/Soquete	180	105	0
Antebraço	Cotovelo	Dobradiça	150	0	0
Mão	Pulso	Bola/Soquete	180	30	120

Em [6] e Chen [7] é apresentado um estudo que demonstra o tipo de movimento e o angulo de rotação dos principais membros do corpo humano possíveis de articulação, sendo portanto, uma ótima referência para o uso de sensores do tipo IMU. A Tabela 1 demonstra esses dados com valores angulares nos três eixos para cada tipo de junta.

2.2 Sensores de movimento

Há inúmeros sensores para aplicações em Realidade Virtual que possibilitam o rastreamento dos movimentos dos membros dos seres humanos, como também de qualquer objeto móvel, entretanto, os tipos mais utilizados atualmente, devido a sua acuracidade, facilidade de uso e baixo custo, são os baseados em câmeras de vídeo, como o kinect e os conhecidos como inerciais, como os giroscópios, acelerômetros, bussolas, etc. Sensores para expressões faciais também são baseados em câmeras associado a marcas visuais/fiduciais, no entanto, não será o foco deste trabalho.

Para o propósito deste trabalho estaremos abordando os sensores inerciais baseado em acelerômetro e giroscópio digitais de construção tipo MEMS (micro-electro-mechanical systems)[4], o qual possui ambos os sensores em um único circuito integrado e com dimensões inferiores a 0.5 cm^2 e com 6 graus de liberdade e que nos permite perceber movimentos tridimensionais com grande precisão.

Com esse tipo de sensor poderemos obter informações associadas a cinemática do movimento, nos possibilitando avaliar movimentos biomecânicos produzidos pela mudanças da velocidade e no movimento rotacional dentro do padrão dos movimentos do corpo humano com ótima precisão.

Sensores de tecnologia MEMS são robusto, resistente a altos estresses mecânicos, entretanto, são sensiveis a determinados patamares de temperatura, como também possuem problemas de ruído e *drift* quando associados aos acelerômetros e giroscópios, respectivamente.

2.3 Acelerômetros

Acelerômetros digitais são sensores baseado em transdutores que nos permite medir acelerações em um ou mais eixos do espaço 2D/3D, nos fornecendo o valor da aceleração ao redor destes quando em movimento, entretanto, quando parado, nos fornece apenas a direção do vetor de aceleração gravitacional.

Sem entrar nos pormenores, nos acelerômetros digitais, a aceleração é medida pela diferença de potencial fornecida por um transdutor decorrente do deslocamento de elementos móveis do conjunto eletro mecânico associado a uma componente gravitacional (em geral, trabalham de 2g a 16g) e que é, entre outros, um dos principais parâmetros determinantes da sensibilidade e precisão do sensor. Assim, com essa diferença de potencial e com algumas operações trigonométricas pode-se determinar sua posição angular e sua aceleração.

O ruído causado pelo acelerômetro, o qual já relatamos anteriormente é decorrente da rápida flutuação dos valores lidos quando este é movimentado ou até mesmo quando se encontra num estado inerte. A aplicação de filtros, como o de filtro de Kalman [8] e o complementar [9] são artifícios indispensáveis para se alcançar a precisão dos valores. Outra técnica usada e fazer a integração dos valores obtidos do acelerômetro com o do giroscópio, cuja técnica é conhecida como fusão e/ou integração.

2.4 Giroscópios

Giroscópios digitais são sensores com feedback capacitivo proveniente de um sistemas de vibração e ressonância, os quais nos permite medir a velocidade angular (ou a taxa da alteração no ângulo de orientação) em torno do eixos.

Os giroscópios possuem o problema de *drift* [10], onde se fizermos sua orientação a partir de um angulo de referência para um novo anglo, quando retornarmos ao anglo de referência, o valor informado não será o mesmo.

Este problema pode ser corrigido facilmente através de um processo de ajuste, onde se mede o valor do desvio causado pelo giroscópio para cada volta com um ΔT associado, obtendo assim, um offset de correção

3. ARQUITECTURA E ASPECTOS DE IMPLEMENTAÇÃO

O objectivo desta seção é apresentar aspectos práticos do sensor propriamente, MPU-6050 em conjunto com o Raspeberry PI e Blender, entre outros artefatos que serão visto a seguir.

A escolha do Blender[5] se deu devido este ser usado vastamente na área de animação 3D (principalmente), Realidade Virtual e jogos. Assim, o Blender será a entidade que receberá os dados do sensor e os processará apresentando visualmente a ação de movimento do objeto associado ao sensor ou IMU.

Também exploramos a comunicação TCP/IP entre as entidades usando a linguagem Python, tanto do lado do sensor, como do Blender usando o BGE (Blender Game Engine), módulo este, que possibilita através do Python a manipulação dinâmica ou em tempo real dos objetos presentes na "cena" do Blender.

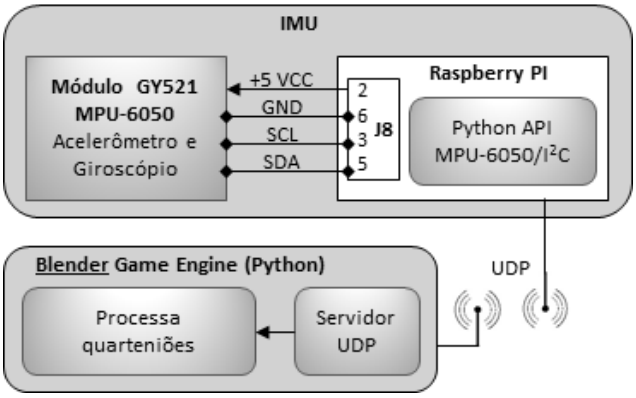

Figure 2: Arquitectura IMU/Blender

A Figura 2 demonstra de forma simplificada a arquitetura desenvolvida. Utilizamos como sensor o MEMS MPU-6050 da empresa InvenSense[6], entretanto, como este sensor se comunica através de um canal de comunicação no padrão I^2C tivemos que interfacear o mesmo com uma unidade de processamento que tivesse a pilha TCP/IP e suporte a comunicação sem fio, onde foi usado o já conhecido "credit-card sized computer", o Raspberry PI[7].

Na arquitetura da Figura 2, quando o Raspberry é ligado, este também alimenta o módulo GY-521 através de seu barramento de I/O (pinos 2 e 6). Após a carga do sistema operativo do Raspberry, este inicializa os drives do barramento I^2C e em sequência executa o aplicativo que passa a ler os dados do MPU-6050 e os envia através da rede sem

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

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

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

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

fios usando o protocolo UDP (User Datagram Protocol)[8] ao recurso computacional na qual o Blender encontra-se em execução.

Os dados enviados pelo sensor através da conexão UDP compreende as informações de quaterniões (X, Y, Z e W), as quais são processadas pelo Digital Motion Processor a partir dos dados do acelerômetro/giroscópio. Após inicialização, o MPU-6050 efetua um processo de calibração que dura em média 20s, onde no decorrer deste tempo, o valores de quaterniões informados não correspondem de forma fidedigna, a posição do sensor.

O Blender, quando ativo, recebe os dados de quaterniões, processa-os e os representa através de um dado.

Nos subitens a seguir será efetuado uma exposição individual dos aspectos técnicos e de integração entre as partes envolvidas, isto é, do conjunto "sensor" e Blender.

3.1 O sensor inercial MPU-6050

O MPU-6050 é um dispositivo (ou circuito integrado) caracterizado como MEMS que integra em seu invólucro um acelerômetro e um giroscópio, cada um com 3 eixos, além de um DMP (Digital Motion Processor) e um sensor de temperatura. Sua comunicação com o mundo exterior se dá através do padrão I^2C em até 400Khz. Para dar vazão e evitar perdas na leitura dos valores dos sensores, um buffer FIFO de 1024 está disponível. Este IMU também possibilita interfacear um compasso externo de três eixos. A figura 3 demostra o diagrama de blocos macro do MPU-6050

Figure 3: Diagrama de blocos macro do MPU-6050[9]

Para cada eixo, há um conversor analógico-digital (ADC)[10] permitindo leituras com valores precisos quanto ao rastreio do movimento. O acelerômetro permite ser configurado em escalas de gravidade negativa/positiva de 2g, 4g, 8g e 16g. Já o giroscópio pode ser programado em escalas de positiva/negativa de 25, 500, 1000 e 2000°/segundos. A figura 4 demostra a orientação dos eixos e a polaridade de rotação do MPU-6050, que segue a regra da mão direita conforme já demonstrado na Figura 1.

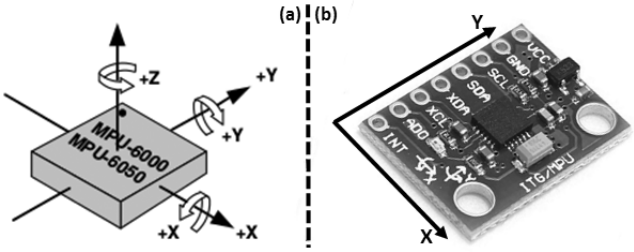

Figure 4: Em (a), orientação/polaridade dos eixos do chip MPU-6050[9] **e em (b) o módulo GY-521 refletindo o mesmo posicionamento**

Em nosso caso de uso, foi utilizado o módulo GY-521 de fabricação chinesa. Este módulo possui o MPU-6050 soldado numa placa de circuito impresso, o que facilita o seu manuseio e ainda permite alimentação entre 3 e 5 volts. A Tabela 2 demonstra a ligações do sinais do módulo GY-521 com o conector J8 do Raspberry PI.

Table 2: Interfaceamento GY-521/Raspberry PI

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

3.2 Artefatos de software

O software envolvido é bastante simples e compreende duas partes distintas, o do sensor o qual é interfaceado com o Raspberry e a ferramenta gráfica Blender que apresenta os resultados do sensor.

Ainda, todos os artefatos envolvidos seguem a filosofia "open source", além de que, métodos e funções utilizadas são devidamente documentadas no próprio código e encontram-se disponíveis no repositório **github**[11] e o download pode ser efetuado conforme o comando abaixo, considerando um ambiente unxi-like e que se tenha o pacote "git" instalado:

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

As subseções a seguir aborda maiores detalhes do software envolvido no Raspberry (sensor propriamente) e Blender (ferramenta gráfica).

3.2.1 Raspberry/sensor

O módulo sensor foi desenvolvido tendo como base o módulo GY-521 que contém um MEMS baseado no MPU-6050 que contempla em seu invólucro um acelerômetro e um giroscópio conforme detalhes já citados anteriormente.

A comunicação do GY-521 com o mundo exterior para o fornecimento dos dados do acelerômetro/giroscópio se dá pelo padrão de comunicação I^2C. Assim para processarmos dados provenientes do GY-521 foi utilizado a plataforma Raspberry PI que também contém uma interface no mesmo padrão. A Figura 5 mostra o conjunto, módulo GY-521 interfaceado com o Raspberry PI.

O sistema operacional usado no Raspberry foi o Rasbian[12] que é versão do Debian para Raspberry. Quaisquer referências à instalação de pacotes, se dará a partir do repositório padrão do Rasbian, salvo se houver indicações especificas.

Para que a comunicação com o sensor seja possível, como também a comunicação com o barramento de I/O do Raspberry, dois pacotes devem ser instalados conforme abaixo:

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

Há inúmeras APIs e "wrappers" escritos para acesso ao MEMS MPU-6050 e para não ficarmos "reinventado a roda" foi utilizado a API para Python de Stefan Kolla[13]. Na estrutura do código fonte indicado acima, estes artefatos

[8] http://goo.gl/voOc0X
[9] Imagem retirada de http://www.invensense.com
[10] http://goo.gl/zsNGHc
[11] https://github.com
[12] http://www.raspbian.org
[13] https://github.com/cTn-dev

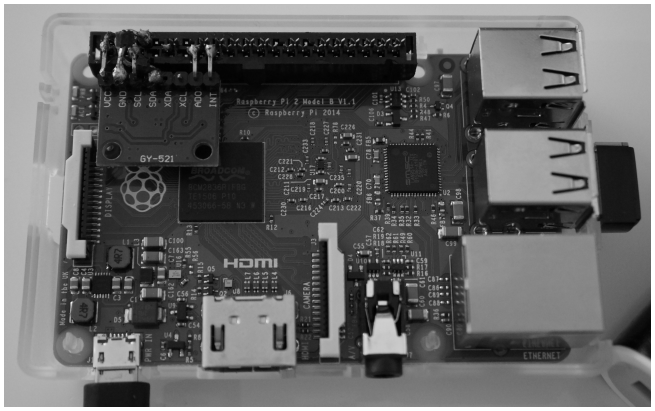

Figure 5: Módulo GY-521 interfaceado com o Raspberry PI

encontram-se na pasta MPU6050, que consiste do ficheiro **mpu6050.py**, que é a classe que contém mapeamento do registradores do MPU-6050 com funções de leitura/escrita para os mesmos; e o **pycomms.py**, que é o wrapper para acesso ao "smbus" do Raspberry de modo prover a comunicação I^2C.

Para isso, foi desenvolvido um aplicativo em Python que utiliza a API citada, o qual lê os dados do sensor e os envia para o Blender através do protocolo UDP. A comunicação entre o Raspberry e Blender pode ocorrer através de qualquer conectividade de rede disponível (cabo ou wireless) que exista entre ambos.

O aplicativo foi nominado como **imu-blender.py** e encontra no raiz do repositório. Há também um ficheiro de configuração que é utilizado pelo **imu-blender.py**, que é o **imu-blender.cfg**.

Para utilizar o aplicativo é preciso editar o ficheiro **imu-blender.cfg** e efetuar alterações de dois parâmetros obrigatórios e um opcional, conforme abaixo:

- **REMOTE_IP (obrigatório):** Endereço IP ou FQDN (nome associado ao domínio de nomes da Internet) do recurso onde se estar executando o Blender (exemplo, 192.168.0.1 ou blender.xpto.com);
- **DST_PORT (obrigatório):** Número da porta UDP conforme configurado no Blender (o valor padrão é 9999, mas pode ser qualquer outra porta que não esteja em uso);
- **DEBUG (opcional):** Deve conter 1, caso se deseje visualizar na tela do Raspberry os valores de X, Y, Z e W que corresponde ao valores de quaterniões obtidos do sensor. Caso não deseje, deve conter o valor 0 (é indicado habilitar o "debug" nos testes iniciais e desabilitar posteriormente após os mesmos).

Efetuado as configurações acima o aplicativo pode ser executado, bastando usar o comando abaixo a partir da pasta onde se encontra o aplicativo e de qualquer Linux/shell. O Blender não precisa estar ativo para que faça um teste inicial de modo a verificar se o sensor está lendo as informações corretamente.

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

Se tudo estiver correto, iniciará a correr no ecrã os valores referente ao quaterniões, e ao se movimentar o sensor, a variação dos valores ocorrerá numa faixa maior, uma vez que nas últimas casas decimais há uma variação mesmo sem que ocorra a movimentação do sensor.

3.2.2 *Blender/Blender Game Engine*

O Blender é a ferramenta gráfica que usamos para representar em tempo real o movimento do sensor. O Blender possui o módulo denominado Blender Game Engine (BGE), que é uma API que tem o Python como linguagem de programação e que nos permite manipular quaisquer objetos da cena e em tempo real quando em renderização.

Assim, o que fizemos foi modelar um objeto no formato de um dado e no BGE implementamos um servidor UDP bloqueante, que ao receber os dados de quaterniões e com o uso de das bibliotecas "bge.logic" e "math.utils" processa algumas operações, cujo resultado é aplicado ao objeto da cena (o dado).

Para iniciar a visualização através do Blender, devemos carregar o código **imu-blender.blend** que se encontra na pasta **blender** do repositório. A renderização efetivamente pode ser iniciada pressionando a tecla "P" com o cursor posicionado na janela de visualização 3D ou iniciando o "standalone player" a partir da janela de visualização das propriedades (ou "properties").

Entretanto, para se obtenha a representação visual do dado (ou qualquer outro objeto na qual o sensor está associado) no ecrã onde Blender está sendo executado de modo a corresponder exatamente à mesma posição física do dado, provavelmente se fará necessário um ajuste de "offset".

Este ajuste poderá ser efetuado usando as teclas simbolizadas pelas quatro setas e pelas teclas "z" e "x". Esse ajuste está associado a rotação câmera ao redor dos três eixos, que tem o dado ponto de visão. Para isso, foi adicionado um objeto "empty" ligado a câmara. Assim, foi possivel rotacionar a câmera em volta do dado usando sensores do tipo "keyboard", controladores to tipo "and" e atuadores do tipo "motion".

4. AVALIAÇÃO

Para este trabalho resumimos nossa avaliação apenas no requisito de maior importância para este tipo de sensor que é a latência. A latência corresponde ao tempo decorrido entre um movimento do sensor, onde novos valores dos quaterniões são enviados ao Blender, e o efetivo recebimento desses dados pelo Blender e seu processamento visual.

Para que pudéssemos efetuar esta avaliação, sincronizamos os relógios do sensor (Raspberry) e do computador onde o Blender estava sendo executado através de um serviço de hora usando o NTP (Network Time Protocol)[14].

Também fizemos com que o sensor enviasse um número sequecial, mais um "timestamp" junto com os dados de quaterniões. Assim, **coletamos 500 valores e obtemos uma latência de em média 8 milissegundos** entre o movimento do sensor e a sua percepção pelo Blender.

Com 8 ms de latência, teremos uma frequência de em média 125 quadros por segundo, considerando as leis da física para o movimento circular uniforme (Freq = $1/\Delta T$). Normalmente, com apenas 30 Hz de taxa de "refresh", os seres humanos já conseguem visualizar movimentos com bastante suavidade, que o caso da transmissões de TV que ocorrem a partir dos 25 Hz ou 25 quadros por segundo.

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

Devido os sistemas estarem distribuídos, acreditamos também haver uma pequena margem de erro devido alguma imprecisão no processo de sincronismo através do NTP. Há outros fatores que podem contribuir, mas não alargaremos essa discussão aqui.

Devido a transferência dos dados ocorrer usando o protocolo UDP, onde não há nenhum mecanismo de confirmação de recebimento (como o TCP), esse fato contribui para a baixa latência. Também verificamos através da sequenciação que não houve perda de pacotes.

Como MPU-6050 foi interfaceado com o Raspberry PI e este poderia ser usado para centralizar os dados de um conjunto de sensores, por exemplo, distribuídos nas articulações do corpo humano (conforme a Tabela 1) para que o rastreio do movimento viesse a ser completo, **alimentamos o Raspberry com uma bateria de lítio de 2200 mAh e a autonomia com o sensor em estado inerte foi de aproximadamente 5 hs**, que é um bom resultado para a maioria das aplicações envolvendo Realidade Virtual ou jogos interativos.

5. TRABALHOS RELACIONADOS

Apresentamos abaixo alguns trabalhos relacionados que aborda pesquisas no contexto do rastreamento usando sensores inerciais e correlações com técnicas de rotação e suas aplicações. Como neste trabalho apenas revisitamos alguns conceitos apresentando dispositivos no estado da arte, acompanhado de técnicas recentes e eficientes para o rastreio de objetos e humanos, não vamos apresentar nenhuma colocação de cunho crítico.

Rastreamento de movimento a partir de câmeras tem também sido alvo de diversas pesquisas, porém, alguns problemas inerentes a esta tecnologia, principalmente quanto a obstrução do objeto da cena faz com que se perda partes do movimento e por isso, a opção parcial ou total por sensores inerciais tem sido usado com certa frequência, como poderemos ver abaixo, abordagens utilizando ambos, isto é, sensores ópticos e sensores inerciais para rastrear o movimento num mesmo objeto.

No trabalho "Sensor Fusion for Augmented Reality" [11], identificou problemas de perda do rastreamento quando da ocorrência de uma interrupção do campo de visão do dispositivo óptico, nesse caso uma câmera. Assim, os autores propuseram uma fusão dos dados de rastreamento usando sensores inerciais baseado em acelerômetro/giroscópio (IMU), onde a fusão ou integração dos dados de rastreio obtido pela câmera e pelos sensores inerciais foi possível obter em tempo real valores estáveis, precisos e sem perda do rastreamento.

Na técnica de fusão, utilizam um modelo de observação baseado em algoritmo associado a "computer vision" [12] que com a integração ou fusão dos dados da câmera e IMU conseguiram obter coordenadas 2D/3D do objeto rastreado (objetivo da pesquisa) com base na comparação de dados e video gerados anteriormente do ambiente sem a presença do objeto a ser rastreado e depois com ele.

O trabalho "Position Estimation With a Low-cost Inertial Measurement Unit" [13] propõe o uso e sensores inerciais com 9 graus de liberdade (acelerômêtro, giroscópio e magnetometer) fixados na estrutura de um óculos conhecidos como "Rift", estes usados também para "computer vision" de modo e eliminar o efeito de desorientação causado pela falta da ligação entre o movimento físico do objeto rastreado (nesse caso uma pessoa) em relação ao ambiente virtual obtido pelo Rift.

Os autores também usaram técnicas de fusão de dados dos sensores associado a filtros para eliminação dos problemas de ruído e drift já relatados neste trabalho e assim, chegaram a um satisfatório resultado o qual foi demonstrado graficamente através do Unit3D[15], onde os dados de rotação também foram processados com o uso de quaterniões.

Este outro recente trabalho, o "Real Time Human Motion Capture Driven By a Wireless Sensor Network" [7], os autores propõem um método baseado no "surface model theory" [14] para construir modelos 3D de esqueletos humanos, onde a representação dos movimentos se dá com o uso de sensores inerciais e comunicação sem fios baseado no MEMS MPU-9150.

O trabalho foca aspectos técnicos dos sensores e nos filtros usando técnicas de integração/fusão, como também um estudo a respeito do movimento angular da juntas articulares das várias partes do corpo humano similar ao apresentado na Tabela 1. Para validar o trabalho os autores comparam movimentos reais dos braços a partir de imagens de vídeo com os gerados através dos sensores de modo a demonstrar a precisão angular dos movimentos. Também avaliou-se a latência na transferência dos dados entre o sensor e o sistema gráfico de visualização que foi de 20ms, que ficou um pouco acima em os resultados que apresentamos em nossa abordagem, entretanto, usaram tecnologia Bluetooth que notadamente possui menor desempenho que redes baseadas no padrão IEEE 802.11.

6. CONCLUSÃO

Sensores inerciais é uma realidade em inúmeras áreas, como nos telemóveis, tablets e indústrias do entretenimento, robótica, aviação, automobilística e espacial, entre outros. Nas áreas de de Realidade Virtual (RV) e jogos interativos (JI) esses dispositivos são utilizados para efetuar o rastreamento do movimento do corpo humano, onde normalmente são posicionados próximos as articulações de modo a identificar o movimento angular dessas partes (ver Tabela 1) e representá-los graficamente, seja num aplicativo gráfico com objetivos específicos ou através de um jogo interativo.

Procurando explorar o uso desses dispositivos, também conhecidos como IMU (Inertial Measurement Unit) de tecnologia MEMS (Micro-Electro-Mechanical Systems) apresentamos uma abordagem de ordem prática onde foi utilizado as informações de um acelerômetro e giroscópio para identificar o movimento de um corpo no espaço 3D. Para representação do movimento, foi utilizada a ferramenta gráfica Blender, esta usada para largamente em animações em 3D, onde o sensor foi representamos através de um dado e que nos permitiu observar o movimento do dado em em 3D.

Com isso, conseguimos avaliar que os atuais dispositivos MEMS são bastantes adequados para uso no rastreio dos movimentos humanos, pois devido o seu diminuto tamanho (menor de 0.25 cm^2) e baixa latência (média de 8 ms) que nos permite uma taxa de 125 quadros por segundo, sendo portanto, ideal para o uso em RV e JI.

IMUs, como o utilizado nesta abordagem é um dispositivo de baixo custo (em média 3 euros), sendo que o Raspberry PI não passa dos 40 euros e pode ser usado como um ponto de concentração dos diversos IMUs posicionados pelo corpo ou qualquer outro objeto que se pretenda detectar o seu movimento em 3D, portanto, permite viabilizar qualquer projeto

[15]http://unity3d.com

com facilidade e abaixo custo.

Imagens do protótipo usado, como o sensor (Raspberry + GY-521 + bateria) e a caixa que acomodou o sensor em forma de dado, como também um vídeo demostrativos poderá ser visualizado a partir do repositório já citado anteriormente.

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.