Plataforma de Monitoramento Esportivo

Arthur Simões Gonzaga e Victor Martins N. Luz

***Resumo*—Este projeto visa criar uma plataforma embarcada de monitoramento esportivo, que realize a aquisição de dados corporais e de movimentação do usuário em tempo real, de modo a efetuar uma análise qualitativa do desempenho do atleta alvo.**

***Keywords*—Sistemas Embarcados, Wearable, Antropometria, Raspberry Pi, Linux**

I. Introdução

Na última década, o dinamismo da evolução tecnológica pela qual a sociedade passa deixa de ser representado apenas por seus legados capitais, como o telefone, a internet, as impressoras, e ruma para nichos tidos até então como "indissociados"desse movimento de evolução, seguindo um processo contínuo de difusão. A busca pela otimização de processos e performances ganhou no desenvolvimento tecnológico um forte e versátil aliado, e o mundo esportivo, por exemplo, começa a experimentar o potencial da tecnologia como mais uma ferramenta para o aumento do rendimento dos atletas. Sem deixar de lado os tradicionais cuidados com o treino, condicionamento físico e alimentação, a tecnologia chega ao esporte para aprimorar o trabalho do atleta e equipe técnica, oferecendo novas possibilidades táticas e estratégicas capazes de agregar melhorias ao desempenho, seja qual for a modalidade.

No futebol não é diferente. A partir dos anos 60[1], a preocupação de tirar a sobrecarga de responsabilidades do técnico de futebol e melhorar o rendimento dos atletas resultou na formação das primeirtas comissões técnicas, que no decorrer das décadas viraram padrão para times de alto rendimento. Essas equipes, que de início focavam na figura do preparador físico, passaram a envolver profissionais de diversas áreas, da tática à psicologia, com o objetivo de atender a diversidade de demandas dos atletas e do esporte, contribuindo para o sucesso das equipes nas competições.

Atualmente, soma-se a todo esse trabalho especializado o uso de tecnologias para avaliações mais rápidas e eficazes do desempenho dos atletas e das táticas de equipe, contribuindo de forma mais ostensiva na prevenção de lesões, correção de posicionamentos táticos, avaliação das transições de jogos e revisão do comportamento geral do time. Vale salientar que o uso de meios tecnológicos não está restrito aos times, e hoje tem peso para alterar o resultado de um jogo. A arbitragem, além de dispor de um sistema de comunicação entre os componentes da equipe, tem agora a possibilidade de utilizar recurso de vídeo (e *software* que processa esse imageamento) para definir a ocorrência de um gol. Esse sistema foi utilizado em um jogo entre as equipes da França e da Espanha[2] , em que árbitro utilizou do recurso tecnológico para anular um gol da equipe francesa.

O uso do sistema de posicionamento global para fins de acompanhamento de movimentação localizada é tão recente quanto o uso geral do GPS. Apenas dois anos após a disponibilização da rede (em 1995), os primeiros estudos sobre a precisão do sistema para caracterizar a locomoção humana foram publicados.[3] O primeiro dispositivo comercial a utilizar a tecnologia foi lançado em 2003, e desde então vem sendo introduzido em esportes de campo aberto, como o futebol, o rugby, o hockey e o tênis.[4]

Atualmente, muitos times brasileiros de futebol já introduziram o uso de tecnologia GPS[5] para mapear a movimentação dos atletas durante os treinamentos e jogos, afim de melhor qualificar os treinamentos, proporcionando uma intervenção mais dinâmica e personalizada para o perfil e particularidades de cada jogador.

Figura 1. Utilização de GPS por jogadores de futebol

[1]"A Evolução das Comissões Técnicas no Futebol", Caderno de Campo, 2017. [Online]. Available: https://cadernodecampo.com/2008/09/11/a-evolucao-das-comissoes-tecnicas-no-futebol/. [Accessed: 03- Apr- 2017].

[2]"Vídeo-árbitro "trama"França por duas vezes na derrota frente a Espanha", O Jogo, 2017. [Online]. Available: http://www.ojogo.pt/internacional/noticias/interior/video-arbitro-anula-golo-a-griezmann-no-franca-espanha-5755641.html. [Accessed: 03- Apr- 2017].

[3]R. Aughey, "Applications of GPS Technologies to Field Sports", International Journal of Sports Physiology and Performance, vol. 6, no. 3, pp. 295-310, 2011.

[4]Julen Castellano; David Casamichana, "Deporte con dispositivos de posicionamiento global (GPS): Aplicaciones y limitaciones", Revista de Psicología del Deporte, vol. 23, no. 2, pp. 355-364, 2014.

[5]"Tecnologia em "top"de jogadores de futebol gera brincadeira, mas é coisa séria.", Canaltech, 2017. [Online]. Available: https://canaltech.com.br/noticia/gps/tecnologia-em-top-de-jogadores-de-futebol-gera-brincadeira-mas-e-coisa-seria-61281/. [Accessed: 03- Apr- 2017].

A. Revisão Bibliográfica

O monitoramento esportivo vem ganhando espaço e ainda não é uma tecnologia acessível a todos os atletas do time. Todavia, o tema vem sendo abordado em artigos científicos, sempre relacionados à utilização para medidas antropométricas.

Artigos de referência do assunto são: *Deporte con dispositivos de posicionamiento global (GPS): Aplicaciones y limitaciones*, desenvolvido por Julen Castellano e David Casamichana e; *Applications Of GPS Technologies to Field Sports*, de Robert J. Aughey. Porém, este artigos tratam somente da importância da utilização desta tecnologia no esporte e não na construção de um dispositivo/plataforma de aquisição de dados.

B. Objetivos e Requisitos

Este projeto almeja modelar e construir uma plataforma embarcada, que possa ser utilizada por um jogador de futebol ou qualquer outro esporte a céu aberto, com o intuito de mapear a sua posição, informando a uma terceira pessoa, dados qualitativos sobre o seu posicionamento no campo.

Os requisitos abrangem três etapas de funcionamento do protótipo: a coleta de dados do GPS que deve ser feita, afim de buscar dados sobre o posicionamento; o estabelecimento de uma comunicação com um computador externo, com o propósito de exibição de dados e avaliação em tempo real (a partir da geração de mapas de calor ou estatística de em qual local do campo o atleta mais se posicionou, por exemplo); e por fim, o dimensionamento do consumo energético da plataforma, tendo em vista a preocupação com a eficiência do consumo do dispositivo, que requer um tempo de utilização grande.

Além das estatísticas advindas diretamente do *tracking* GPS, é possível obter outros dados úteis para a avaliação de performance de um atleta, como um levantamento dos dados de velocidade ou aceleração desenvolvidas pelo jogador ao longo do treinamento.

O *hardware* a ser desenvolvido é caracterizado pela integração de sensores, incluídos neste o módulo GPS, e de um computador *Raspberry Pi* para o processamento de dados e gerenciamento da comunicação com o dispositivo em que serão exibidas as informações. Tanto o sensoriamento, quanto o *Rasberry* serão embarcados em um *wearable*, visando a portabilidade do sistema e o conforto do atleta sob monitoramento.

A difusão tecnológica, além de buscar novas áreas de atuação, também se expande atingindo novos públicos. Dispositivos que desempenham esse tipo de monitoramento, utilizados pelos clubes de futebol, envolvem tecnologia e fabricação estrangeiras, fato que agrega custos de tributação e transporte ao já elevado preço do equipamento. O desenvolvimento dessa plataforma visa obter um custo moderado de projeto, pelo qual clubes de pequeno porte ou até mesmo amadores possam pagar para dispor dessa tecnologia, tornando-a mais acessível.

Além disso, o time de futebol da Universidade de Brasília, tem um departamento de análise, o qual pode muito bem utilizar da tecnologia desenvolvida para qualificar suas análises e apontar as difuculdades a serem tratadas.

Em uma procura de como o mercado desta tecnologia se comporta, achou-se uma empresa brasileira, GPS Pro Soccer,

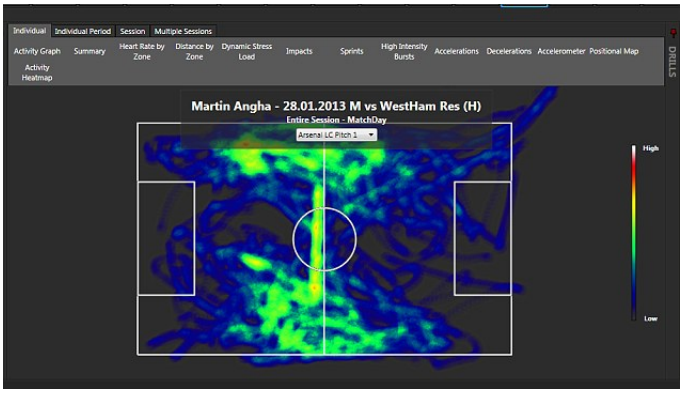

Figura 2. Exemplo de Mapa de Calor

que desenvolve produtos próximos ao qual irá se desenvolver, todavia, o custo dos equipamentos é elevado, ficando em torno de R$ 5000,00[6] para cada monitor, sendo de acesso local, somente. Algumas outras empresas internacionais também trabalham com plataformas parecidas. A SPT desenvolve a solução junto ao colete por um preço de 249,99 dólares[7], porém, o equipamento é importado (este preço não inclui taxas e impostos) e não gera os dados em tempo real, o que torna o nosso produto competitivo no mercado.

II. Modelo de Funcionamento

O *Raspberry Pi*, diferentemente de microcontroladores potencialmente utilizáveis para realizar esse projeto, permite o *multitasking* do *software* nele embarcado. Essa característica é importante para a coordenação de vários subsistemas necessários para solucionar o problema a que o projeto se propõe a lidar. Especificamente para este dispositivo de monitoramento, faz-se necessário gerenciar a comunicação serial com o módulo GPS, responsável por obter as informações de posicionamento, realizar a interpretação desses dados, lidar com conexões de *Internet* e enviar essas informações para um servidor, que, com o banco de dados de posicionamento, é capaz de disponibilizar remotamente as informações coletadas, gerando um ambiente de fácil entendimento para o usuário.

O usuário, através de um circuito de controle, poderá guiar o início e fim do registro de dados e, por consequência, do monitoramento do atleta. Essa ativação do *traking* se dará após a conexão serial *(UART)* entre o *Raspberry PI* e o módulo *GPS NEO-6M*, e a partir do momento que o módulo GPS fixar uma triangulação com os satélites. Os dados enviados pelo módulo são obtidos de *strings* de informações, enviadas a cada segundo para o *Rasberry*. As informações de latitude, longitude, velocidade e altitude são extraídas dessas *strings* e são transformadas em variáveis, tornando possível a realização

[6]P. Sistemas, "GPS PRO SOCCER-GPS Pro Soccer-Sistema de gerenciamento de atletas de futebol", Gpsprosoccer.com.br, 2017. [Online]. Available: http://gpsprosoccer.com.br/gps-pro-soccer/. [Accessed: 04- Apr- 2017].

[7]S. Vest, "SPT Pack (GPS + Vest) – SPT", Sports-performancetracking.com, 2017. [Online]. Available: https://www.sportsperformancetracking.com/product/spt-pack-gps-vest/#w3mIiem20adi2tVW.97. [Accessed: 04- Apr- 2017].

de operações matemáticas com esses valores. Após a discretização desses valores, eles são armazanados em um arquivo, que serve de base de dados para o servidor.

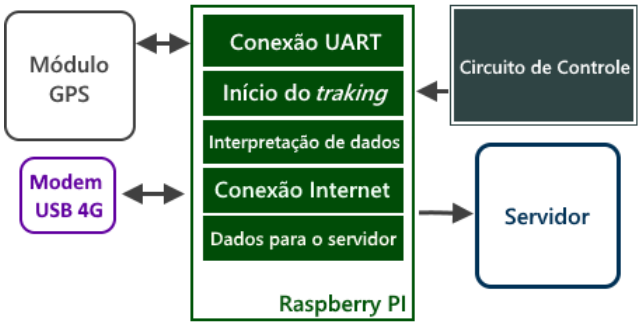

Figura 3. Fluxograma de funcionamento do sistema.

Ao utilizar uma solução já disponível para a interface de servidor, o *Apache*, pode-se tornar o *Raspberry* em um servidor, possibilitando a criação de *backlogs* e o uso de *scripts* em *html*, *css* ou *python* para desenvolver as interfaces gráficas. Um dos *scripts html* utlilizados tem seu código fonte disponibilizado gratuitamente pelo *Google* [8], sendo responsável por mapear, por incidência, latitudes e longitudes na plataforma do *Google Maps*, gerando mapas de calor de posicionamento, oferecendo uma interface de fácil compreensão e análise . De modo a garantir o acesso à *internet* para o carregamento das imagens e informações adicionais para o funcionamento da *API* do *Google Maps*, visto que a interface wifi do Raspberry fica dedicada ao *hotspot* para acesso remoto, buscou-se interfacear um dispositivo de *internet* móvel conectado serialmente ao *Raspberry*.

Inicialmente, a conexão *Point-to-point Protocol*[9] foi estabelecida com um módulo GSM/GPRS SIM800L, porém, a banda e a latência fornecidas pela rede GPRS foram insuficientes para carregar o volume de informações demandado. De forma análoga, foi possível estabelecer uma conexão com sucesso entre o Raspberry e o modem USB 4G/LTE *XSStick W100*, da fabricante *4G Systems*. Em ambos os casos foi utilizada a rede da operadora Vivo, e junto à seção de descrição de *software*(IV.D) será detalhado o processo de configuração e utilização de redes móveis integrado ao projeto.

Dispondo desse ferramental, foi possivel criar uma página *web* que utiliza dos pacotes *RPi.GPIO* para leitura de valores do *Raspberry*. De modo a testar essa interação entre servidor e *hardware*, Fez-se a leitura de valores nas GPIO do Raspberry com sucesso, sendo essas informações passadas para a interface *html*, que exibe o valor atual no *site*. Além disso, a partir de um banco de dados de latitudes e longitudes, gerou-se um mapa de calor de posicionamento.

Todo o processo é automatizado por completo, isto é, o usuário precisa apenas acessar o IP do raspberry na rede *Wifi* compartilhada para obter as informações. Isto se dá pelo fato do *script* desenvolvido fazer um *parse* do arquivo que constantemente recebe as informações tratadas pelo microprocessador (banco de dados), salvando-o essa interpretação em variáveis que alimentam as funções do *Google Maps* e também os campos de dados de distância e velocidades máxima e média apresaentados na interface.

Figura 4. Testes de interface com servidor.

III. Descrição de Hardware

Para tornar operacionais as etapas de funcionamento propostas para o sistema, o equipamento embarcado precisa atuar em pelo menos duas grandes frentes: Obtenção de dados de localização e integração com um servidor, para a disponibilização dos dados registrados.

A obtenção dos parâmetros essenciais para o sucesso da operação do dispositivo proposto é possível com a utilização de um módulo GPS integrado ao *Raspberry*. O módulo *NEO-6M*[10], da empresa suiça *U-blox Holding AG* fornece, via comunicação serial, *strings* no padrão *NMEA 0183 (National Marine Electronics Association)*[11], a partir das quais é possível discretizar as informações de latitude, longitude, velocidade, altitude - entre outras.

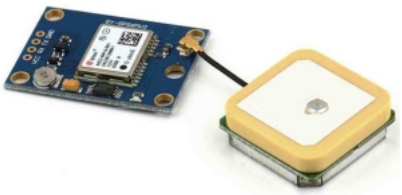

Figura 5. Módulo GPS NEO-6M.

Esse módulo exige entre 3.3V e 5V para a alimentação, e a sua comunicação UART tem nivel lógico alto baseado em 3.3V. Essa operação é ideal para sua integração com os principais microcontroladores e também com o *Raspberry Pi*, dispositivo utilizado como base para o projeto.

Com o objetivo de fornecer maior controle do usuário sobre o sistema, que operaria de forma autônoma a partir de sua

[8] "Google Maps API's - Heatmaps", Google, 2017. [Online]. Available: https://developers.google.com/maps/documentation/javascript/examples/layer-heatmap?hl=pt-br. [Accessed: 06- May- 2017]

[9] "O POINT-TO-POINT PROTOCOL (PPP)", UFPR. [Online]. Available: http://www.cricte2004.eletrica.ufpr.br/edu/anterior/cd00/trab/ppp/ [Accessed: 28- June- 2017]

[10] "NEO-6 U-blox 6 GPS Modules DataSheet", U-blox, 2011. [Online]. Available: https://www.u-blox.com/sites/default/files/products/documents/NEO-6-DataSheet-%28GPS.G6-HW-09005%29.pdf. [Accessed: 05- May- 2017]

[11] "NMEA data". [Online]. Available: http://www.gpsinformation.org/dale/nmea.htm. [Accessed: 01- Jun- 2017]

ativação, foi inserido um sistema de controle, por meio do qual o usuário operará botões de ativação/desligamento do *tracking*. Isso limita o funcionamento do sistema, que, enquanto ativo, alimenta continuamente um arquivo com as informações de rastreamento.

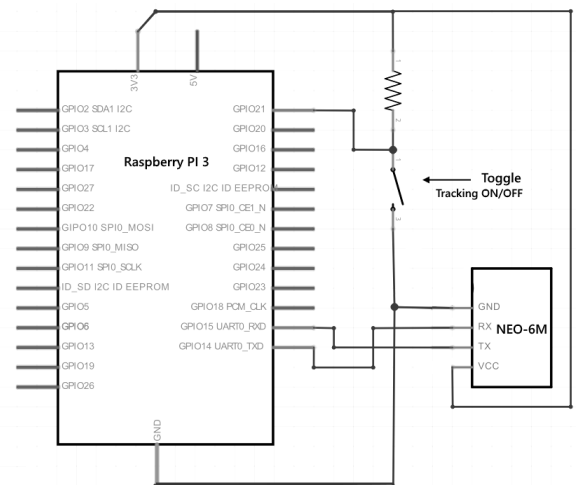

Figura 6. Esquemático inicial do sistema.

Além das conexões de GPIO, há outra integração de *hardware* no sistema, que garante a conexão do raspberry à internet ao mesmo tempo que utiliza sua interface *wifi* para compartilhar uma rede e possibilitar o acesso remoto do usuário ao sistema. O acesso à rede de internet por telefonia móvel (4G) é feita pelo modem USB 4G/LTE *XSStick W100*, fabricado pela *4G Systems* e distribuído pela operadora Vivo no Brasil.

Figura 7. Modem USB 4G XSStick W100, utilizado no projeto.

Esse dispositivo é *quad-band* e exige até 500mA da porta USB do Rasberry. Para utilizá-lo em conjunto com o microprocessador, foi preciso alimentar o sistema com uma fonte de saída 2A. O uso de uma fonte menor, de 850mA, não suportou a corrente exigida pelo modem no momento de conexão às redes e forçou o desligamento do *Raspberry* e também do modem, o que provoca sua desconfiguração.

IV. Descrição de Software

As soluções por *software* utilizadAs foram bem específicas. Os requisitos de programação levantados para viabilizar o projeto foram:

- Realizar a aquisição e *'parse'* dos dados do GPS para gerar os mapas;
- Criar um servidor *back-end*, para que computadores externos pudessem acessar o dispositivo;
- Gerar uma interface gráfica simples, de utilização intuitiva e de fácil análise;

A. Configurações de UART

A conexão UART exigiu procedimentos prévios de configuração do *Raspberry*. Como padrão, a transmissão serial é dedicada para operar o console, e é preciso desativar essa funcionalidade para fazer uso dos pinos seriais. Esse procedimento é feito removendo qualquer referência a "console"feita no arquivo */boot/cmdline.txt*. Faz-se necessário fazer o *reboot* do sistema para que a mudança surta efeito.

Vale salientar que, no *Raspberry Pi 3*, a adição do *Bluetooth* alterou o local para comunicação UART de */dev/ttyAMA0* para */dev/ttyS0* (utilizada nos códigos desse dispositivo de forma alternativa, como */dev/serial0*).

A conexão UART estabelecida entre o módulo GPS o *Raspberry* tem *baud rate* de 9600 *bits* por segundo. O módulo GPS envia, a cada segundo, um pacote de *strings* no padrão NMEA contendo informações sobre horário, posicionamento e satélites compondo a triangulação. Para os primeiros testes de comunicação e obtenção de dados, foi utilizado o *software gpsd/cgps*[12].

Com essa aplicação instalada, dispõe-se no terminal do sistema operacional o status da conexão com o módulo, e as principais informações obtidas, utilizando os comandos a seguir:

```
$ sudo gpsd /dev/serial0 -F
/var/run/gpsd.sock
$ cgps -s
```

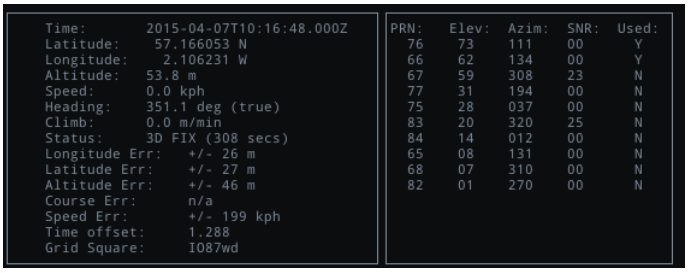

Figura 8. Exemplo de retorno da aplicação gpsd/cgps.

Obtendo retorno dessa aplicação, certifica-se que o módulo GPS e a comunicação estabelecida estão funcionando como previsto. A partir de então, o tratamento das *strings* NMEA

[12]"cgps - Linux Man Page". [Online]. Available: https://www.systutorials.com/docs/linux/man/1-cgps/. [Accessed: 02- Jun- 2017]

enviadas deve ser feito dentro do programa da plataforma de monitoramento, para possibilitar a manipulação desses valores e sua compilação em um banco de dados.

B. Padrão NMEA e 'parse' dos valores

O padrão NMEA é um protocolo estabelecido para a transmissão de dados para dispositivos de georreferenciamento. Esse protocolo consiste no envio de *strings* de estrutura padronizada, cada uma com um conjunto de infrmações específicas, e se diferenciam entre si de acordo com o ID de origem da mensagem (primeiros 6 *bits*). Abaixo, está um exemplo de *string GPGGA (GPS Fix information)*. Desta string especificamente, é feito o *parse* (discretização) dos valores de posicionamento, velocidade e altitude, a serem utilizados no projeto.

```
$GPGGA,092750.000,5321.6802,N,00630.3372,W,
1,8,1.03,61.7,M,55.2,M,,*76
```

Após os *bits* de identificação*($GPGGA)*, cada campo separado por vigula traz informações de GPS. Por exemplo, o primeiro campo informa o horário no fuso GMT, o segundo e terceiro campo trazem informações da latitude e o quarto e quinto campos, da longitude.

A biblioteca *gps.h*, a partir do arquivo nmea.c, recolhe esses valores de cada envio da string e os disponibiliza em uma struct, da qual e possível resgatar os valores no programa principal já como variáveis, do tipo *double*.

C. Servidor

Na criação do servidor foi utilizado a solução Apache, pois a configuração da mesma pareceu mais simplória e fácil de realizar-se. O Apache é um aplicativo que nos permite habilitar o acesso de páginas web pela porta 8080. Em seu renderiza as páginas HTML, além de permitir as chamadas de scripts em Javascript, CSS e outros tipos de extensão. Outra facilidade que o mesmo permite é a manutenção em tempo real de todo o *website*. Caso seja necessário, basta apenas editar o arquivo que deseja e reiniciar o Apache, que atualiza a página web. Para a interface foram usados scripts com *APIs* para fazer o *HeatMap*. Toda a interface foi configurada em HTML.

D. Configuração de conexão do Raspberry a redes móveis

Ao plugar o modem USB 4G na porta *usb0* do *Raspberry*, o sistema identifica o aparelho apenas como um dispositivo de armazenamento de dados, onde estão armazenados os dados para a instalação da interface para sistemas *Windows*. É necessário, portanto, realizar um *bypass* e identificar a função de modem, já que os *drivers* de funcionamento serão incluídos ao *kernel* do *raspian*. Para realizar essa troca se utiliza a aplicação *usb mode-switch*[13], própria para sistemas *linux/debian*.

Por meio desse aplicativo, e também utilizando como auxiliares as funções *lsusb e usb-devices* (verificação de *status* de conexão e funcionamento de dispositivos ligados via USB), faz-se a troca de interface de funcionamento do modem. Com o comando abaixo, foi possível habilitar o reconhecimento do dispositivo utilizado como modem 4G. Vale salientar que os parâmetros passados para a função usb mode-switch variam, dependendo de cada modelo e fabricante do *stick* USB.

```
usb_modeswitch -v 1c9e -p 9bfe -S
```

E faz-se necessária a adição das informações de troca, obtidas atraves de lsusb, nos arquivos de configuração do *usb-modeswitch (/etc/usb_modeswitch.conf)*

```
DefaultVendor:0x1c9e
DefaultProduct:0x9b01
MessageEndPoint:"0x01"
```

Com o modem configurado, a conexão é estabelecida e mantida utilizando as aplicações pppd (para reger o protocolo *Point-to-Point*), *Sakis3G*[14] (que realiza a conexão com a rede, a partir da entrada das informações de APN, usuário e senha da Vivo[15]) e *UMTSkeeper*, responsável pela reconexão da rede (é um acessório da aplicação *Sakis3G*).

A instalação dessas aplicações foi feita da seguinte forma, via terminal:

```
$ sudo apt-get install ppp

$ sudo mkdir umtskeeper
$ cd umtskeeper
$ sudo wget "http://zool33.uni-
graz.at/petz/umtskeeper/src
/umtskeeper.tar.gz"
$ sudo tar -xzvf umtskeeper.gz
$ sudo chmod +x umtskeeper

$ sudo wget "http://downloads.sourceforge
.net/project/vim-n4n0/sakis3g.tar.gz?
r=http%3A%2F%2Fsourceforge.net%2Fprojects%2
Fvim-n4n0%2Ffiles%2F&ts=1363537696&
use_mirror=tene~t"
-O sakis3g.tar.gz
$ sudo tar -xzvf sakis3g.tar.gz
$ sudo chmod +x sakis3g
```

Após isso, apenas a conexão via *Sakis3G* é necessária para por em funcionamento o modem USB. Para inicializar a interface via terminal, foi preciso executar o comando no diretório de instalação:

```
./sakis3g
```

A configuração é feita selecionando o dispositivo USB e nserindo os dados de registro em redes móveis da operadora (APN, usuário e senha).

[13]"Package: usb-modeswitch", Debian, 2017. [Online] Available: https://packages.debian.org/pt-br/jessie/usb-modeswitch [Acessed: 01- July- 2017]

[14]"Sakis3G source- GitHub". Available: https://github.com/Trixarian/sakis3g-source

[15]"Configuração Vivo Internet", Vivo. [Online] Available:http://www.vivo.com.br/configurar-aparelhos/?pagina=device/motorola/moto-g-4g/topic/internet/como-configurar-seu-celular-para-navegar-atraves-de-vivo-internet/1 [Acessed: 01- July- 2017]

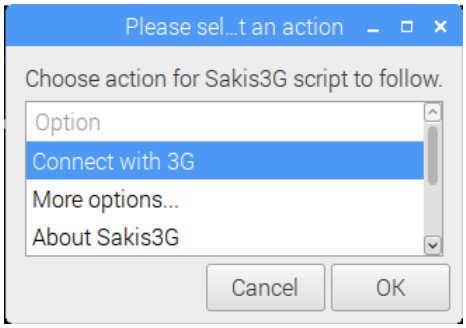

Figura 9. Interface de configuração de conexão *Sakis3G*.

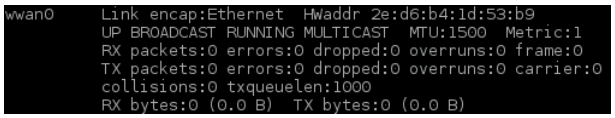

Figura 10. Rede *wwan* de dados móveis sianlizada na aplicação *ifconfig*.

E. Interface de controle

Utilizaremos dos GPIOs para fazer o controle do aparelho final. Haverão botões para inicialização das gravações e de liga e desliga do sistema. Esse botões acionam sinais, que acionam funções. O botão ON/OFF, por exemplo, a ser acionado, despertará uma função *toggle*, que fará a interrupção de um laço de aquisição e armazenamento dos dados.

F. Maquina de Estados

Foi feita uma máquina de estados afim de deixar o usuário livre para registrar novos dados ou para de gravar os dados em determinados momentos (em intervalos de jogos, por exemplo). Para tal foi pensada em três estados: o estado de espera, no qual o GPS estava coletando os dados, mas ainda não os gravava, este estado era demonstrado com um LED piscando a uma frequência de 1 hertz, esperando que um botão de gravação fosse apertado. Neste momento, o processo pai envia ao processo filho um sinal, que aciona a gravação em um arquivo no formato CSV, acendendo também um novo LED, que ficava intermitente na frequência de 2 hertz. Assim que o botão fosse clicado novamente, o processo filho deixava de gravar os dados no arquivo CSV.

Havia ainda um outro estado, paralelo, que, quando o programa estiver no estado de espera, permite que o usuário apague o arquivo gravado, podendo gerar um novo arquivo do zero.

V. Resultados

A. Resultados preliminares

O servidor funciona como esperado. Após uma requisição, é gerado um *log* pelo *Raspberry* que pode retornar o IP da máquina que está requisitando acesso ao monitoramento. O acesso até então foi realizado em rede cabeada, e o acesso feito via IP do microprocessador diretamente do *browser* do computador.

A interface gráfica com a API do *Google* mostra-se como uma ferramenta bastante adequada para o projeto, exibindo os dados de forma intuitiva e simplificada, localizando as coordenadas no terreno do mapa. Em um primeiro momento, a estrutura *html* do servidor estava limitada a permitir a inclusão do arquivo de dados manualmente, também demandando intervenção direta do usuário para gerar a atualização. A completa automatização do projeto, característica buscada para garantir uma operação de fato autosuficiente, passa a ser garantida após uma revisão de código de interface *html*, em que o arquivo gerado é continuamente lido, não mais precisando indicar o seu caminho no sistema de arquivos, e a página é recarregada de forma autônoma, e já com dados referentes ao segundo anterior ao da atualização.

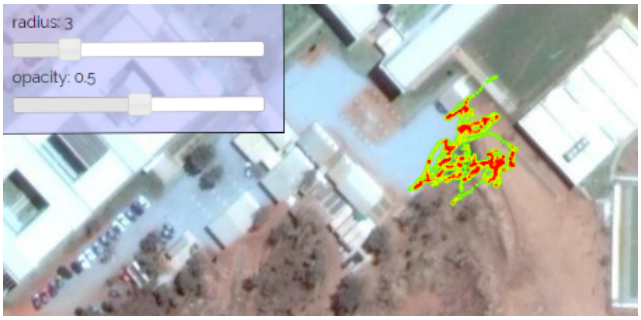

Figura 11. Interface de servidor exibindo dados armazenados.

O uso do GPS demanda locais abertos. Em lugares fechados, geralmente, há dificuldade na obtenção de fixos (triangulação com satélites), custando para retornar valores válidos de *tracking*. Foi possível, após uma quantidade de tempo maior que a observada em acionamentos anteriores, obter fixos dentro da sala de aula, mas os valores obtidos não se mostravam precisos o suficiente para a geração dos mapas de calor, resultando em discrepâncias de deslocamento no mapa.

B. Teste de Movimento em espaço aberto e certificação da interface

De modo a verificar experimentalmente a capacidade do módulo GPS de obter dados suficientemente precisos para a aplicação desejada, foi realizado um teste de deslocamento a bordo de um veículo. O teste foi idealizado para expor o sistema a um longo período de funcionamento, o que acaba por exigir a geração e o processamento de um banco de dados consideravelmente maior, de aproximadamente 2410 *strings* de dados (Quantidade de informações correspondente a aproximadamente 40 minutos - duração similar ao de 1 tempo de jogo no futebol). Além disso, esse teste comprovou a diferença de comportamento do sistema em áreas fechadas ou "semiabertas"para áreas abertas de fato e a capacidade de dispor essa informação dentro de uma escala de tamanho adequada para espaços esportivos. A precisão nos fixos se mostrou muito mais acertada que nos demas acionamentos,

feitos em condições restritivas de ambiente, como as descritas anteriormente.

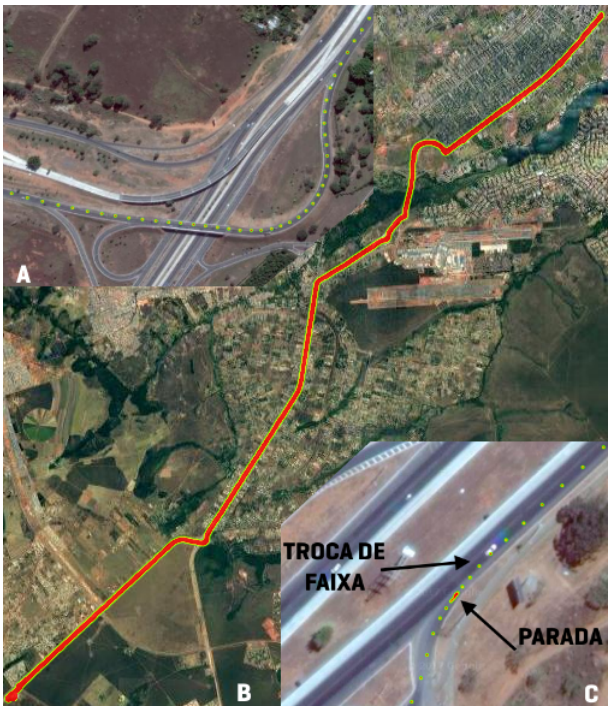

Figura 12. A - Periodicidade e presição de dados obtidos; B- Trajeto de 40 minutos desempenhado; C - Comportamento de dados em relação à movimentação.

Muitos dos movimentos executados pelo veículo, especialmente em baixa velocidade, são análogos aos de um atleta em campo. A cada segundo, o sistema de GPS localiza o módulo, e a sua movimentação fica registrada no banco de dados. Em campo aberto, neste experimento, foi possível detectar as paradas do veículo, onde há concentração de fixos de latitude e longitude, e as mudanças de curso realizadas, como a troca de faixas de rolamento. A API do *Google*, utilizada como interface de apresentação gráfica de informações, se mostrou adequada para o propósito do projeto. A interseção mostrada na figura 9A corresponde à medida de 3 campos de futebol oficiais somados, e a disposição de dados, mesmo considerando a velocidade do veículo (que provocou grandes espaçamentos entre as amostras de 1 segundo), é suficiente para descrever a movimentação realizada.

C. Teste de Movimento em espaço aberto conduzido por pessoa

Além do teste conduzido por movimentação de veículo, foi realizado um teste de movimentação pedestre, mais adequada as fins aos quais o projeto se propõe. Apesar desta diferença, esse teste confirma a efetividade do teste anterior, pois o movimento realizado foi descrito com sucesso, assim como o teste anterior, mesmo considerando as diferenças de velocidades desenvolvidas por um veículo e uma pessoa.

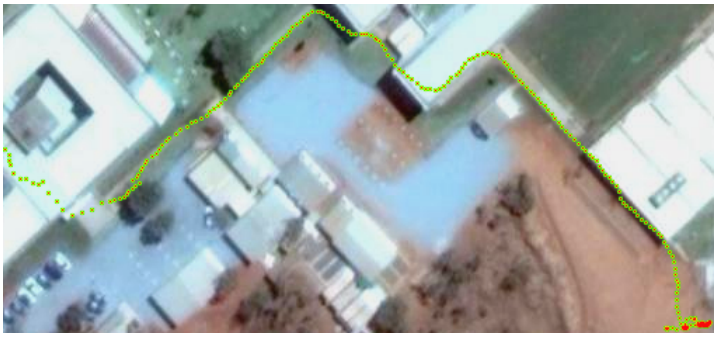

Figura 13. Teste de movimentação com pedestre

Comparando o resultado gráfico dos dois testes realizados, nota-se a diferença de espaçamento de amostras, que se dão a cada segundo, nos dois casos. Esse é claramente um dos efeitos da diferença de velocidade em cada experimento.

D. Interface gráfica final

A interface gráfica final continua priorizando a exibição dos dados gráficos do mapa de calor, mas traz ao usuário um visual mais intuitivo e uma navegação em abas para a obtenção das demais informações de rastreamento (Velocidades média e máxima e distância) e de descrição do projeto (Aba *"Sobre"*). A taxa de atualização das informações na página se dá a cada 30 segundos, para evitar recarregamentos sucessivos em um curto espaço de tempo dos mapas e das informações.

Figura 14. Visão geral da interface

APÊNDICE A
CÓDIGOS PRINCIPAIS DO SISTEMA

A. *Código 1 - Código do sistema de tracking ISoccer - V2.0*

```
//compile usando: gcc -o
//tracking tracking.c -lgps -lm -lwiringpi

#include <unistd.h>
#include <fcntl.h>
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <signal.h>
#include <time.h>
#include <sys/types.h>
#include <wiringPi.h>
#include <gps.h>
#include <math.h>

//int fd;
FILE *fp;
unsigned char status = 0;

void fechar(){
        printf("Fechando programa\n");
        fclose(fp);
        sleep(1);
        exit(0);
}
//funcao para calculo de distancias a partir
    de latitude e longitude.
double haversine(double lat_inicial, double
    lon_inicial, double lat_final, double
    lon_final){

    double d2r = 0.017453292519943295769236;
        //conversao graus -> radianos
    double a,c,dlong, dlat, temp_sin,
        temp_sin2, temp_cos, distancia;

         dlong = (lon_final - lon_inicial) *
             d2r;
         dlat = (lat_final - lat_inicial) *
             d2r;

         temp_sin = sin(dlat/2.0);
         temp_cos = cos(lat_inicial * d2r);
         temp_sin2 = sin(dlong/2.0);

          a = (temp_sin * temp_sin) + (
              temp_cos * temp_cos) * (temp_sin2
               * temp_sin2);
         c = 2.0 * atan2(sqrt(a), sqrt(1.0 - a
             ));

         distancia=6368.1*1000.0*c; //
             conversao para metros (6368.1:
             raio aproximado da Terra, em Km).

        if(distancia<1.5){
                return 0;
        }
        else{
                return distancia;
        }
}

void leitura(){
        if(status == 0){
                puts("Recording...");
                status=1;
                return;
        }
        if(status == 1){
                puts("Stop Recording...");
                status=0;
                return;
        }
}

int main(void){

        int i=0,u=0;
        long int count=0;
        double last_lat, last_lon, speedsum
            =0.0,speedmed=0.0, speedmax=0.0,
            dist=0.0;
    gps_init();
    loc_t data;

    pid_t pid_id;

    wiringPiSetup();
    pid_id = fork();

    signal(SIGINT,fechar);

    if(pid_id == 0){
      signal(SIGUSR1,leitura);
      signal(SIGUSR2,fechar);
      pinMode(9,OUTPUT);
        while(1){
        while(status){

        fp = fopen("/var/www/html/data.csv", "
            a+");
        if(u==0){
                fprintf(fp,"\"speedmed\",\"
                    speedmax\",\"lat\",\"lon
                    \",\"dist\"\n");
                u=1;}

        //Aquisicao de dados
        gps_location(&data);
        count++;

        //Processamento
                if(i==1){
                        speedsum=data.speed;
                }
                if(i>2){
                        dist=dist+haversine(
                            last_lat,last_lon,
                            data.latitude,data.
                            longitude);
```

```
                        speedsum=speedsum+data
                            .speed;
                        speedmed=speedsum/(
                            count * 1.0);
                }
                if(data.speed>speedmax){
                        speedmax=data.speed;
                }
                        last_lat=data.latitude
                            ;
                        last_lon=data.
                            longitude;


        fprintf(fp,"%lf,%lf,%lf,%lf,%lf\n",
            speedmed,speedmax,data.latitude,
            data.longitude, dist);

        fclose(fp);
                digitalWrite(9,HIGH);
                usleep(500000);
                digitalWrite(9,LOW);
                usleep(500000);
        }
        }
}

else{
        int btn = 0,btn1=0;

        //Botao 0 - Controle do estado de
            espera
        pinMode(28,INPUT);
        pullUpDnControl(28,PUD_DOWN); //Botao
            de Pull Down
        //Botao 1 - Clean
        pinMode(29,INPUT);
        pullUpDnControl(29,PUD_DOWN); //Botao
            de Pull Down
        //Led 1 - Estado de espera
        pinMode(8,OUTPUT);
        //Led 2 - Lendo dados GPS
        pinMode(9,OUTPUT);
        //Led 3 - Limpando Arquivo
        pinMode(7,OUTPUT);

        while(1){
           btn = digitalRead(28); //Leitura do
                Botao
           btn1 = digitalRead(29); //Leitura
               do Botao1
           if(btn == 1){
                digitalWrite(8,HIGH); //Envia
                    sinal para salvar dados
                kill(pid_id,SIGUSR1);
           }
           else{
                digitalWrite(8,LOW);
           }
           if(btn1 == 1){
                digitalWrite(7,HIGH);
                usleep(200000);
                digitalWrite(7,LOW);
                usleep(200000);
                digitalWrite(7,HIGH);
                usleep(200000);
                digitalWrite(7,LOW);
                usleep(200000);
                digitalWrite(7,HIGH);
                usleep(200000);
                digitalWrite(7,LOW);
                usleep(200000);
                system("sudo rm -rf /home/pi/
                    sistemas_embarcados/PFinal/
                    maquina_estados/teste.txt")
                    ;
                kill(pid_id,SIGUSR2);
                fechar();
           }
                sleep(1);
        }

    }
    return 0;
}
```

B. *Código 2 - Instalação do servidor Apache em Bash*

```
sudo apt-get install apache2
```

C. *Códigos para Web*

Código - Web Page Main

```
<!DOCTYPE html>
<html>
<title>iSoccer</title>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-
    width, initial-scale=1">
<link rel="stylesheet" href="https://www.
    w3schools.com/w3css/4/w3.css">
<link rel="stylesheet" href="https://fonts.
    googleapis.com/css?family=Raleway">
<link rel="stylesheet" href="https://cdnjs.
    cloudflare.com/ajax/libs/font-awesome
    /4.7.0/css/font-awesome.min.css">
<style>
    html,
    body,
    h1,
    h2,
    h3,
    h4,
    h5 {
        font-family: "Raleway", sans-serif
    }
</style>

<head>
```

```
    <script src="https://d3js.org/d3.v4.js"></
        script>
    <script src="./jquery-2.1.1.min.js"></
        script>
    <script src="https://code.jquery.com/ui
        /1.11.0/jquery-ui.js"></script>
    <script src="https://maps.googleapis.com/
        maps/api/js?key=
        AIzaSyDVCnJEnsmaQiJUqOmTfQlGn-eHBzYP6Ps
        &libraries=visualization"></script>

    <script>
        var csv = [];
        var speedmax = [];
        var speedmed = [];
        var dist = [];
        var tamanho;


        function handleFileSelect() {
            d3.csv("data.csv", function(
                results) {
                for (idx in results) {
                    var row = results[idx];
                }
                tamanho = results.length - 2;
                row = results[tamanho]; //
                speedmax.push(row["speedmax"])
                    ;
                speedmed.push(row["speedmed"])
                    ;
                dist.push(row["dist"]);
            });
        }

        console.log(speedmax);
        console.log(speedmed);
        console.log(dist);
    </script>

</head>

<body class="w3-light-grey">

    <script>
        handleFileSelect();
        window.alert("Dados Atualizados!");
    </script>

    <!-- Top container -->
    <div class="w3-bar w3-top w3-black w3-
        large" style="z-index:4">
        <button class="w3-bar-item w3-button
            w3-hide-large w3-hover-none w3-
            hover-text-light-grey" onclick="
            w3_open();"><i class="fa fa-bars"
            ></i> Menu</button>
        <span class="w3-bar-item w3-right">
            iSoccer</span>
    </div>

    <!-- Sidebar/menu -->
    <nav class="w3-sidebar w3-collapse w3-
        white w3-animate-left" style="z-index
        :3;width:300px;" id="mySidebar"><br>
        <div class="w3-container w3-row">
            <div class="w3-col s4">
                <img src="unb.png" class="w3-
                    circle w3-margin-right"
                    style="width:56px">
            </div>
            <div class="w3-col s8 w3-bar">
                <span>Ola, <strong>UnB</strong
                    ></span><br>
            </div>
        </div>
        <hr>
        <div class="w3-container">
            <h5>Dashboard</h5>
        </div>
        <div class="w3-bar-block">
            <a href="#" class="w3-bar-item w3-
                button w3-padding-16 w3-hide-
                large w3-dark-grey w3-hover-
                black" onclick="w3_close()"
                title="close menu"><i class="fa
                 fa-remove fa-fw"></i>Fechar
                Menu</a>
            <a href="index.html" class="w3-bar
                -item w3-button w3-padding w3-
                blue"><i class="fa fa-eye fa-fw
                "></i> Visao Geral</a>
            <a href="heatmap.html" class="w3-
                bar-item w3-button w3-padding"
                ><i class="fa fa-bullseye fa-fw
                "></i>Mapa de Calor</a>
            <a href="about.html" class="w3-bar
                -item w3-button w3-padding"><i
                class="fa fa-users fa-fw"></i>
                Sobre</a>
        </div>
    </nav>


    <!-- Overlay effect when opening sidebar
        on small screens -->
    <div class="w3-overlay w3-hide-large w3-
        animate-opacity" onclick="w3_close()"
        style="cursor:pointer" title="close
        side menu" id="myOverlay"></div>

    <!-- !PAGE CONTENT! -->
    <div class="w3-main" style="margin-left
        :300px;margin-top:43px;">

        <!-- Header -->
        <header class="w3-container" style="
            padding-top:22px">
            <h5><b><i class="fa fa-eye fa-fw"
                ></i> Visao Geral</b></h5>
        </header>


        <div class="w3-container">
            <h5>Dados</h5>
```

```
            <table class="w3-table w3-striped
                w3-bordered w3-border w3-
                hoverable w3-white">
                <tr>
                    <td>Velocidade Media</td>
                    <td>
                        <script>
                            document.write(
                                speedmed);
                        </script> m/s
                    </td>
                </tr>
                <tr>
                    <td>Velocidade Maxima</td>
                    <td>
                        <script>
                            document.write(
                                speedmax);
                        </script> m/s</td>
                </tr>
                <tr>
                    <td>Distancia</td>
                    <td>
                        <script>
                            document.write(
                                dist);
                        </script> m</td>
                </tr>
            </table><br>
        </div>

        <!-- End page content -->
    </div>

    <script>
        // Get the Sidebar
        var mySidebar = document.
            getElementById("mySidebar");

        // Get the DIV with overlay effect
        var overlayBg = document.
            getElementById("myOverlay");

        // Toggle between showing and hiding
            the sidebar, and add overlay effect
        function w3_open() {
            if (mySidebar.style.display === '
                block') {
                mySidebar.style.display = '
                    none';
                overlayBg.style.display = "
                    none";
            } else {
                mySidebar.style.display = '
                    block';
                overlayBg.style.display = "
                    block";
            }
        }

        // Close the sidebar with the close
            button
        function w3_close() {
            mySidebar.style.display = "none";
            overlayBg.style.display = "none";
        }
    </script>
    <meta http-equiv="refresh" content="30" />
</body>

</html>
```

Código - Web Page Heatmap

```
<html>

<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=
        device-width, initial-scale=1">
    <link rel="stylesheet" href="https://www.
        w3schools.com/w3css/4/w3.css">
    <link rel="stylesheet" href="https://fonts
        .googleapis.com/css?family=Raleway">
    <link rel="stylesheet" href="https://cdnjs
        .cloudflare.com/ajax/libs/font-awesome
        /4.7.0/css/font-awesome.min.css">

    <style>
        #map-canvas {
            position: absolute;
            top: 110px;
            right: 1%;
            width: 76%;
            height: 83%;
            font-family: 'Raleway', sans-serif
                ;
            z-index: 1;
        }

        html,
        body,
        h1,
        h2,
        h3,
        h4,
        h5 {
            font-family: "Raleway", sans-serif
        }
    </style>

    <script src="https://d3js.org/d3.v4.js"></
        script>
    <script src="./jquery-2.1.1.min.js"></
        script>
    <script src="https://code.jquery.com/ui
        /1.11.0/jquery-ui.js"></script>

    <link rel="stylesheet" href="https://code.
        jquery.com/ui/1.11.0/themes/smoothness/
        jquery-ui.css">
    <script src="https://maps.googleapis.com/
        maps/api/js?key=
        AIzaSyDVCnJEnsmaQiJUqOmTfQlGn-eHBzYP6Ps
        &libraries=visualization"></script>
    <script>
```

```
        var csv = [];
        var heatmap, pointArray, map;

        function handleFileSelect() {
            d3.csv("sample.csv", function(
                results) {
                for (idx in results) {
                    var row = results[idx];
                    csv.push(new google.maps.
                        LatLng(row["lat"], row[
                        "lon"]));
                }
            });
            loadHeatmap(csv);
        }

        function initialize() {
            var mapOptions = {
                zoom: 20,
                center: new google.maps.LatLng
                    (-15.989520, -48.044480),
                mapTypeId: google.maps.
                    MapTypeId.SATELLITE
            };
            map = new google.maps.Map(document
                .getElementById('map-canvas'),
                mapOptions);
        }

        function loadHeatmap(csv) {
            var pointArray = new google.maps.
                MVCArray(csv);

            heatmap = new google.maps.
                visualization.HeatmapLayer({
                data: pointArray,
                radius: 2,
                opacity: 0.8,
                map: map
            });
            heatmap.setMap(map);
        }

        $(document).ready(function() {
            initialize();
            handleFileSelect();
        });
    </script>
    <meta http-equiv="refresh" content="30" />
</head>

<body class="w3-light-grey">
    <!-- Top container -->
    <div class="w3-bar w3-top w3-black w3-
        large" style="z-index:4">
        <button class="w3-bar-item w3-button
            w3-hide-large w3-hover-none w3-
            hover-text-light-grey" onclick="
            w3_open();"><i class="fa fa-bars"
            ></i> Menu</button>
        <span class="w3-bar-item w3-right">
            iSoccer</span>
    </div>

    <!-- Sidebar/menu -->
    <nav class="w3-sidebar w3-collapse w3-
        white w3-animate-left" style="z-index
        :3;width:300px;" id="mySidebar"><br>
        <div class="w3-container w3-row">
            <div class="w3-col s4">
                <img src="unb.png" class="w3-
                    circle w3-margin-right"
                    style="width:56px">
            </div>
            <div class="w3-col s8 w3-bar">
                <span>Ola, <strong>UnB</strong
                    ></span><br>
            </div>
        </div>
        <hr>
        <div class="w3-container">
            <h5>Dashboard</h5>
        </div>
        <div class="w3-bar-block">
            <a href="#" class="w3-bar-item w3-
                button w3-padding-16 w3-hide-
                large w3-dark-grey w3-hover-
                black" onclick="w3_close()"
                title="close menu"><i class="fa
                 fa-remove fa-fw"></i>Fechar
                Menu</a>
            <a href="index.html" class="w3-bar
                -item w3-button w3-padding w3-
                blue"><i class="fa fa-eye fa-fw
                "></i> Visao Geral</a>
            <a href="heatmap.html" class="w3-
                bar-item w3-button w3-padding"
                ><i class="fa fa-bullseye fa-fw
                "></i>Mapa de Calor</a>
            <a href="about.html" class="w3-bar
                -item w3-button w3-padding"><i
                class="fa fa-users fa-fw"></i>
                Sobre</a>
        </div>
    </nav>

    <!-- Overlay effect when opening sidebar
        on small screens -->
    <div class="w3-overlay w3-hide-large w3-
        animate-opacity" onclick="w3_close()"
        style="cursor:pointer" title="close
        side menu" id="myOverlay"></div>

    <div id="map-canvas"> </div>

    <div class="w3-main" style="margin-left
        :300px;margin-top:43px;">

        <!-- Header -->
        <header class="w3-container" style="
            padding-top:22px">
            <h5><b><i class="fa fa-bullseye fa
                -fw"></i> Mapa de Calor</b></h5
                >
        </header>
```

```
        <!-- End page content -->
    </div>


    <script>
        // Get the Sidebar
        var mySidebar = document.
            getElementById("mySidebar");

        // Get the DIV with overlay effect
        var overlayBg = document.
            getElementById("myOverlay");

        // Toggle between showing and hiding
            the sidebar, and add overlay effect
        function w3_open() {
            if (mySidebar.style.display === '
                block') {
                mySidebar.style.display = '
                    none';
                overlayBg.style.display = "
                    none";
            } else {
                mySidebar.style.display = '
                    block';
                overlayBg.style.display = "
                    block";
            }
        }

        // Close the sidebar with the close
            button
        function w3_close() {
            mySidebar.style.display = "none";
            overlayBg.style.display = "none";
        }
    </script>

</body>

</html>
```

APÊNDICE B
CÓDIGOS QUE COMPÕEM A BIBLIOTECA *gps.h*

A. *Código 4 - Comunicação via UART*

```
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <fcntl.h>
#include <termios.h>
#include <inttypes.h>
#include <string.h>

#include "serial.h"
#define PORTNAME "/dev/serial0"
// porta /dev/ttySO do Raspberry PI 3

int uart0_filestream = -1;

void serial_init(void)
{
    uart0_filestream = open(PORTNAME, O_RDWR |
         O_NOCTTY | O_NDELAY);

    if (uart0_filestream == -1)
    {
        //TODO error handling...
    }
}

void serial_config(void)
{
    struct termios options;
    tcgetattr(uart0_filestream, &options);
    options.c_cflag = B9600 | CS8 | CLOCAL |
        CREAD;
    options.c_iflag = IGNPAR;
    options.c_oflag = 0;
    options.c_lflag = 0;
    tcflush(uart0_filestream, TCIFLUSH);
    tcsetattr(uart0_filestream, TCSANOW, &
        options);
}

void serial_println(const char *line, int len)
{
    if (uart0_filestream != -1) {
        char *cpstr = (char *)malloc((len+1) *
             sizeof(char));
        strcpy(cpstr, line);
        cpstr[len-1] = '\r';
        cpstr[len] = '\n';

        int count = write(uart0_filestream,
            cpstr, len+1);
        if (count < 0) {
            //TODO: handle errors...
        }
        free(cpstr);
    }
}

// Lendo uma linha recebida pela UART
void serial_readln(char *buffer, int len)
{
    char c;
    char *b = buffer;
    int rx_length = -1;
    while(1) {
        rx_length = read(uart0_filestream, (
            void*)(&c), 1);

        if (rx_length <= 0) {
            //espera por mensagens
            sleep(1);
        } else {
            if (c == '\n') {
                *b++ = '\0';
                break;
            }
            *b++ = c;
        }
    }
```

```
}

void serial_close(void)
{
    close(uart0_filestream);
}
```

B. Código 5 - Conversão do padrão NMEA para variáveis double

```
//Tratamento dos dados das strings NMEA
#include <stdio.h>
#include <stdlib.h>
#include <inttypes.h>
#include <string.h>
#include <math.h>

#include "nmea.h"

void nmea_parse_gpgga(char *nmea, gpgga_t *loc
    )
{
    char *p = nmea;

    p = strchr(p, ',')+1;

    p = strchr(p, ',')+1;
    loc->latitude = atof(p);

    p = strchr(p, ',')+1;
    switch (p[0]) {
        case 'N':
            loc->lat = 'N';
            break;
        case 'S':
            loc->lat = 'S';
            break;
        case ',':
            loc->lat = '\0';
            break;
            //Tratamento de hemisferios de
                latitude (Norte e Sul)
    }

    p = strchr(p, ',')+1;
    loc->longitude = atof(p);

    p = strchr(p, ',')+1;
    switch (p[0]) {
        case 'W':
            loc->lon = 'W';
            break;
        case 'E':
            loc->lon = 'E';
            break;
        case ',':
            loc->lon = '\0';
            break;
            //Tratamento de hemisferios de
                latitude (Leste e Oeste)
    }

    p = strchr(p, ',')+1;
    loc->quality = (uint8_t)atoi(p); //
        Qualidade da recepcao de dados

    p = strchr(p, ',')+1;
    loc->satellites = (uint8_t)atoi(p);

    p = strchr(p, ',')+1;

    p = strchr(p, ',')+1;
    loc->altitude = atof(p);
}

void nmea_parse_gprmc(char *nmea, gprmc_t *loc
    )
{
    char *p = nmea;

    p = strchr(p, ',')+1;
    p = strchr(p, ',')+1;

    p = strchr(p, ',')+1;
    loc->latitude = atof(p);

    p = strchr(p, ',')+1;
    switch (p[0]) {
        case 'N':
            loc->lat = 'N';
            break;
        case 'S':
            loc->lat = 'S';
            break;
        case ',':
            loc->lat = '\0';
            break;
    }

    p = strchr(p, ',')+1;
    loc->longitude = atof(p);

    p = strchr(p, ',')+1;
    switch (p[0]) {
        case 'W':
            loc->lon = 'W';
            break;
        case 'E':
            loc->lon = 'E';
            break;
        case ',':
            loc->lon = '\0';
            break;
    }

    p = strchr(p, ',')+1;
    loc->speed = atof(p);

    p = strchr(p, ',')+1;
    loc->course = atof(p);
}

uint8_t nmea_get_message_type(const char *
    message)
{
    uint8_t checksum = 0;
```

```
    if ((checksum = nmea_valid_checksum(
        message)) != _EMPTY) {
        return checksum;
    }

    if (strstr(message, NMEA_GPGGA_STR) !=
        NULL) {
        return NMEA_GPGGA;
    }

    if (strstr(message, NMEA_GPRMC_STR) !=
        NULL) {
        return NMEA_GPRMC;
    }

    return NMEA_UNKNOWN;
}

uint8_t nmea_valid_checksum(const char *
    message) {
    uint8_t checksum= (uint8_t)strtol(strchr(
        message, '*')+1, NULL, 16);

    char p;
    uint8_t sum = 0;
    ++message;
    while ((p = *message++) != '*') {
        sum ^= p;
    }

    if (sum != checksum) {
        return NMEA_CHECKSUM_ERR;
    }

    return _EMPTY;
}
```

C. Código 6 - Funções para aquisição dos dados do GPS no programa principal

```
#include "gps.h"
#include <stdio.h>
#include <stdlib.h>

#include <math.h>

#include "nmea.h"
#include "serial.h"

extern void gps_init(void) {
    serial_init();
    serial_config();

}

extern void gps_on(void) {
}

extern void gps_location(loc_t *coord) {
    uint8_t status = _EMPTY;
    while(status != _COMPLETED) {
        gpgga_t gpgga;
        gprmc_t gprmc;
        char buffer[256];

        serial_readln(buffer, 256);
        switch (nmea_get_message_type(buffer))
            {
            case NMEA_GPGGA:
                nmea_parse_gpgga(buffer, &
                    gpgga);

                gps_convert_deg_to_dec(&(gpgga
                    .latitude), gpgga.lat, &(
                    gpgga.longitude), gpgga.lon
                    );

                coord->latitude = gpgga.
                    latitude;
                coord->longitude = gpgga.
                    longitude;
                coord->altitude = gpgga.
                    altitude;

                status |= NMEA_GPGGA;
                break;
            case NMEA_GPRMC:
                nmea_parse_gprmc(buffer, &
                    gprmc);

                coord->speed = gprmc.speed;
                coord->course = gprmc.course;

                status |= NMEA_GPRMC;
                break;
        }
    }
}

extern void gps_off(void) {
    serial_close();
}

void gps_convert_deg_to_dec(double *latitude,
    char ns,  double *longitude, char we)
{
    double lat = (ns == 'N') ? *latitude : -1
        * (*latitude);
    double lon = (we == 'E') ? *longitude : -1
         * (*longitude);

    *latitude = gps_deg_dec(lat);
    *longitude = gps_deg_dec(lon);
}

double gps_deg_dec(double deg_point)
{
    double ddeg;
    double sec = modf(deg_point, &ddeg)*60;
    int deg = (int)(ddeg/100);
    int min = (int)(deg_point-(deg*100));

    double absdlat = round(deg * 1000000.);
    double absmlat = round(min * 1000000.);
    double absslat = round(sec * 1000000.);
```

```
    return round(absdlat + (absmlat/60) + (
        absslat/3600)) /1000000;
}
```