**FATEF – Faculdade de Tecnologia de São Vicente**

**Bacharelado em Sistemas da Informação**

**RODRIGUES, Anderson Antonio Lopes**

**Aplicativo para coleta de informações automática quanto a dados de telemetria em karts utilizando Arduino**

***Application for automatic collection of information about the telemetry data in karts using Arduino***

Resumo

O objetivo desse artigo técnico é mostrar se é possível, através de equipamentos de baixo custo, e de fácil acesso, montar um equipamento de telemetria, para coleta de informações de veículos de corrida, para pilotos amadores. O autor desse projeto vê necessidade de um equipamento desse tipo, pois poderá ajudar na melhoria da pilotagem e também em uma oportunidade de negócios com os locais de prática (autódromos, kartódromos, etc.) para dar um diferencial para os seus clientes, pois com isso, esse cliente poderá visualizar dados como velocidade, traçado, rotações do motor em determinado ponto do traçado, além do tempo total e de voltas, que são colhidos pelo sistemas de cronometragem que normalmente já existem nesses locais. O artigo foi construído através de pesquisas em sites de equipes profissionais pois elas tem uma vasta documentação sobre telemetria e análise de dados gerados pelos seus equipamentos. Também foram feitas análises de equipamentos eletrônicos e micro controladores que pudessem suportar esse tipo de projeto, e com isso chegarmos a um escopo inicial para montagem do hardware que ficará embarcado no veículo. Para o software gerenciador, foi utilizado o conhecimento técnico do autor, já que o mesmo é profissional da área de TI no mundo corporativo, e esse conhecimento pode ser utilizado para construção de softwares para integração com hardware, direta ou indiretamente.

**Palavras-chaves**: Arduino; Telemetria; Aplicativo.

Abstract

The purpose of this technical paper is to show whether it is possible through low cost equipment and easy access, assemble a telemetry equipment to collect information of vehicles racing to amateur drivers. The author of this project see the need for such an appliance, as this may help in improving the steering and in a business opportunity with local practice (race tracks, kart tracks, etc.) To give an advantage to their customers , because with so that client can view data such as speed , stroke , engine revs at a given point on the line , plus the total time and turns, which are harvested by timing systems that typically already exist in these locations . The item has been built through research on professional teams sites because they have extensive documentation on telemetry and analysis of data generated by their equipment. Analysis of consumer electronics and micro controllers that could support such a project were also made, and thus reach an initial scope mounting hardware that will be embedded in the vehicle. For the manager software, technical knowledge of the author was used, since it is an IT professional in the corporate world, and this knowledge can be used to build software for integration with hardware, directly or indirectly.

**Keywords**: Arduino; Telemetry; Application.

INTRODUÇÃO

Com a invenção do automóvel, foi inevitável as pessoas começarem a idealizar corridas entre carros, com o objetivo de se determinar o conjunto carro-piloto mais rápido, com o melhor desempenho. Essas corridas, no início, foram idealizadas a partir das corridas de carroças, comum desde a época da Roma Antiga. O objetivo naquela época era, além de se determinar o conjunto mais rápido, era também ter um campo de prova que exigissem as máquinas ao extremo.

Primeiras corridas

  
Figura 1 - Pôster da primeira corrida realizada (fonte: <http://en.wikipedia.org>)

A primeira corrida de carros que se tem registro ocorreu em foi realizada na França, em 1894, ela foi realizada pela revista parisiense *Le Petit Journal*, com início em Paris e termino na cidade de Rouen. Essa corrida foi chamada de *Concours des Voitures sans Chevaux* (Competição de Carros sem Cavalos), e teve 69 conjuntos participantes. O primeiro vencedor dessa prova foi o conde Jules-Albert de Dion, chegando a Rouen no tempo de 6 horas e 48 minutos, numa velocidade média de 19 km/h, porém sua vitória não foi contabilizada já que o seu carro continha um acessório proibido, por essa razão o título da corrida foi dado a Georges Lemaître, que chegou na segunda colocação.

Primeiros autódromos

A partir dos primeiros eventos, a competição entre automóveis se tornou popular no mundo todo, mas num primeiro momento as corridas eram realizadas entre cidades ou entre distritos, e era difícil a organização, pois as corridas normalmente tinham grandes distâncias a serem cumpridas, além dos acidentes que ocorriam, normalmente com vítimas fatais. Com isso, ficou evidente que, para pratica do esporte, precisava-se de praças adequadas. Com isso, foram criados os autódromos, sendo o autódromo de *Brooklands*, na Inglaterra, o primeiro autódromo a ser construído. Ele foi aberto em 17 de junho de 1907, e foi idealizado e construído por Hugh Locke-King, famoso construtor do Reino Unido daquela época.



Figura 2 - Corrida em Brooklands, em 1932 (fonte: http://www.warbirdsnews.com/)

Mas para os americanos, o primeiro autódromo existente no mundo foi o Milwaukee Mile, inaugurado em 1903, segundo relatos. A construção de autódromo foi baseado no formato oval dos hipódromos. Localizado no estado de Wisconsin, nos Estados Unidos, abriga até hoje corridas das mais tradicionais categorias do automobilismo daquele país.



Figura 3 - Vista aérea do Milwaukee Mile (fonte: http://racingnews360.com/milwaukee-mile/)

TIPOS DE CARROS

Existem vários tipos de carros, abaixo, vamos falar brevemente sobre eles. Vale lembrar que todos os tipos citados tem um papel fundamental no desenvolvimento de carros de produção, que nada mais são os carros que você encontra na concessionária.

Monoposto

Também conhecida como *openwheels*, ou carros de fórmula, é caracterizada pelos carros com rodas colocadas fora do corpo principal do carro, e também por ter lugar somente para uma pessoa, sendo que esse lugar ocupa o menor espaço possível no carro. As corridas disputadas com esse tipo de carro são disputadas normalmente em autódromos com piso de asfalto. Um exemplo de categoria que corre com esse tipo de construção de carro é a Formula 1.

Turismo

Os carros de competição do tipo Turismo são carros que vemos na rua, no dia a dia, com adaptações para competição, ou com chassis tubulares, mas com carroceria dos carros de rua. No cenário brasileiro, podemos citar a Stock Car como uma categoria que usa esse tipo de carro.

Protótipos e Grã-Turismo

São carros projetados, que podem ou não virar carros de rua, para compra e uso do consumidor comum. Normalmente os campeonatos que usam esse tipo de carro são corridas de longa duração, tendo sua duração limitada por tempo ou por distância. Por tempo, as provas são limitadas normalmente entre 12 e 24 horas, e por distância, são limitadas normalmente entre 500 e 1000 quilômetros. Por causa da longa duração, cada carro tem uma equipe de pilotos responsável pela pilotagem do mesmo, com normalmente três pilotos formando essa equipe. Esse tipo de carro carrega também o que há de mais moderno na indústria automobilística, com um adendo, que os carros e seus componentes são estressados por longos períodos de tempo. Esse cenário é extremamente importante para a indústria automobilística.

Kart

O Kart é um mini monoposto, criado para ser tornar o tipo de carro inicial de um piloto, mas a sua criação não teve esse propósito. Durante a II Guerra Mundial, para deslocamento das ferramentas e mecânicos nos campos de aviação, os militares americanos e ingleses construíram pequenos veículos motorizados, aproveitando peças sobressalentes de aviões e carros. Esse pequenos veículos também eram utilizados em corridas amigáveis entre esses militares, nos momentos de folga. Esses veículos criados pelos militares foram a base do primeiro kart criado para competição, em 1956, sendo que esses karts tiveram sua venda através de kit possibilitada em 1957, com a criação da primeira fábrica de kart, nos Estados Unidos.

Esse tipo de veículo é também muito utilizado por pilotos amadores, aqueles que não vivem do automobilismo, mas o usam como meio de diversão ou hobby. Normalmente existem lugares específicos, para essa pratica amadora, onde o piloto pode pagar para utilizar de forma organizada um kart, com direito a equipamentos de segurança e uma pista oficial. O veículo para esse fim é construído com uma estrutura mais reforçada, a fim de dar mais segurança ao piloto amador, esse tipo de kart é conhecido como “kart de aluguel”.



Figura 4 – Autor do artigo pilotando um kart de aluguel (fonte: http://www.luisfranca.net)

Objetivo

Como piloto amador de kart, sinto necessidade de algumas informações que ajudam na melhoria da pilotagem, como velocidade, aceleração lateral, entre outros. A partir daí, tive a ideia de criar alguma aplicação, em conjunto com um hardware, onde esse pudesse dar suporte para coleta e transmissão de dados, e num computador, esses dados fossem processados para análise posterior.

Justificativa

Não existe anda no mercado a nível amador, o que tem é caro, e necessita de importação**.**

Método

Ver depois

Telemetria

A palavra Telemetria é uma palavra de origem grega, onde “tele” significa remoto e “metron” significa medida. Essa tecnologia permite que informações de interesse sejam medidas, de maneira ativa, com comunicação em tempo real, ou de maneira passiva, através da gravação de dados e posterior leitura em aplicações preparadas para tal.

Hoje no mundo automobilístico, temos os dois cenários sendo amplamente aplicados, tanto no âmbito das competições como no âmbito de uso comum. Abaixo, alguns exemplos de utilização.

Telemetria no âmbito esportivo

Nas corridas de automóveis, a telemetria é amplamente utilizada. Um engenheiro de pista pode ter informações de vários itens do carro, como por exemplo a temperatura dos freios, temperatura do motor, posição do pedal do acelerador e do freio, número da marcha utilizada, ângulo do volante, todas em tempo real.

O posicionamento do carro no traçado também pode ser medido em tempo real, e essa informação é importante para que o engenheiro de pista deixe o piloto ciente da situação de pista.



Figura 5 - Exemplo de gráficos gerados pela telemetria (fonte: http://blog.ouseful.info)

Outros itens são medidos após a chegada do carro aos boxes, como temperatura dos pneus e desgaste de peças, como o assoalho do carro. Com todos os dados reunidos, os integrantes de uma equipe podem realizar as modificações necessárias para melhora da performance do carro, e também dar dicas de como melhorar a pilotagem.



Figura 6 - Medição de temperatura do pneu (fonte: http://www.formulastudent.de)

Sistema de navegação por satélite

O sistema de navegação por satélite é um sistema que estabelece o posicionamento geoespacial através do uso de satélites artificiais e com cobertura regional ou global. Esse tipo de sistema permite que aparelhos eletrônicos determine sua geoposicionamento (latitude, longitude e altitude), usando sinais horários transmitidos ao longo de uma linha de mira de rádio de satélites. Esses aparelhos eletrônicos possuem receptores que calculam com precisão o tempo e a posição, que pode ser utilizado como referência para as mais diversas aplicações no mundo comercial, militar e cientifico.

Nessa aplicação, usaremos o GPS (Global Positioning System), ou Sistema de Posicionamento Global, que é o sistema mais famoso, e o único utilizável atualmente para fins civis atualmente.

Sistema de Posicionamento Global

Esse sistema, também conhecido como GPS, é um sistema de navegação por satélite, criado na década de 1960 pelo exército americano, num projeto chamado “NAVSTAR”, mas o sistema só foi considerado completo em 1995, tendo o custo estimado de 10 bilhões de dólares americanos.

Ele é formado por uma constelação de 24 satélites, que foram lançados em orbita em duas datas, uma parte em fevereiro de 1978 e outra em novembro de 2004. Eles circundam o planeta Terra duas vezes ao dia, numa altitude de 20.200 quilômetros a uma velocidade de 11.265 quilômetros por hora. Esses satélites têm a bordo relógios atômicos, que são relógios que trabalham com a frequência máxima de liberação de energia dos átomos e frequência de oscilação dos mesmos, para chegar ao tempo horário exato do satélite. Com isso, basta ter um receptor estável, e que esse capture pelo menos quatro satélites para determinar seu posicionamento, velocidade e tempo.

Vale lembrar que essa tecnologia ainda é mantida pelo exército americano, que pode desativa-lo quando quiser caso, por exemplo, tenhamos uma guerra que o país esteja envolvido. Eles utilizam essa tecnologia num nível de precisão maior que o nível liberado para uso civil, em misseis, em aviões, navios, etc.

Geoposicionamento

Para uso de sistemas de posicionamento por satélite, seja qualquer um deles, é necessário o mínimo conhecimento de geoposicionamento. O geoposicionamento nada mais é que a posição geográfica no planeta Terra em relação a linha do Equador e ao meridiano de Greenwich. A informação dessa posição é constituída por 3 números: Latitude, Longitude e Altitude.

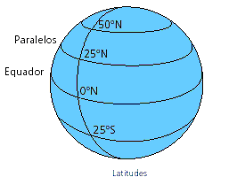


Figura 7 - Exemplificação da latitude

A latitude é a distância ao Equador medida ao longo do meridiano de Greenwich. Esta distância mede-se em graus, podendo variar entre 0º e 90º para Norte ou para Sul, sendo que para o Norte o número será positivo e para o Sul será Negativo.



Figura 8 - Exemplificação da longitude

A longitude é a distância ao meridiano de Greenwich medida ao longo do Equador. Esta distância mede-se em graus, podendo variar entre 0º e 180º para Leste ou para Oeste, sendo que para Leste o valor pode ser positivo e para Oeste o valor negativo.

A Terra é aproximadamente esférica, com um ligeiro achatamento nos polos. Para se definir a altitude de um ponto sobre a Terra define-se uma esfera, ou geoide, com um raio de 6378 km. A altitude num ponto da Terra é a distância na vertical à superfície desta geoide. Por exemplo, a altitude média do Aeroporto de Lisboa é de 114 m, mas a altitude média da Holanda é negativa.

Para melhor ilustração desses parâmetros, abaixo temos uma tabela com latitude, longitude e altitude média de algumas capitais de estados do Brasil.

Tabela 1 - Latitude, Longitude e Altitude de capitais brasileiras

|  |  |  |  |
| --- | --- | --- | --- |
| **Cidade** | **Latitude** | **Longitude** | **Altitude** |
| São Paulo/SP | -23° 32' 51'' | -46° 38' 10'' | 760 metros |
| Porto Alegre/RS | -30° 01' 59'' | -51° 13' 48'' | 3 metros |
| Cuiabá/MT | -15° 35' 46'' | -56° 05' 48'' | 176 metros |
| Manaus/AM | -03° 06' 07'' | -60° 01' 30'' | 92 metros |
| Recife/PE | -08° 03' 14'' | -34° 52' 52'' | 4 metros |
| Boa Vista/RR | 02° 49' 11'' | -60° 40' 24'' | 85 metros |

Padrão NMEA

Com o crescente uso de receptores GPS, para as mais diversas finalidades, no mundo todo, a comunicação entre esses dispositivos precisou ser normalizada. O orgão que se encarregou disso foi o NMEA (*National Marine Eletronics Association*), que inicialmente era responsável por regulamentar os dispositivos eletrônicos para uso marítimo nos Estados Unidos. Esse órgão criou em 1983 o protocolo NMEA 0183, que compreendia em uma série de regras para geração de dados provenientes de sistemas GPS. Ele é utilizado em todos os níveis de acesso (civil, militar e/ou governamental).

Esse protocolo é baseado em ASCII e transmitido serialmente, sendo que a comunicação é feita via mensagens. Essas mensagens devem seguir as normas estabelecidas por esse protocolo, abaixo veremos algumas dessas regras:

* O caractere inicial de cada mensagem deve ser um cifrão “$”;
* As sentenças da mensagem são separadas por virgulas “,”;
* Logo após o cifrão 2 letras identificam o dispositivo que está enviando a mensagem e 3 o tipo de mensagem;
* Após o último capo deve ter um asterisco “\*” seguido do *checksum* de 2 caracteres em hexadecimal;
* O *checksum* consiste em um XOR (exclusive OR – OU exclusivo) de toda a mensagem desde o “$” até o “\*”;

Para melhor entendimento dessas regras, veremos um exemplo abaixo:

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| $GPRMC | ,223735.700 | ,A | ,2357.5466,S | ,04622.5291,W | ,0.72 | ,204.26 | ,241113 | ,A\*68 |
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |

Campo 1: Tipo da sentença (RMC = Recommended minimum specific)

Campo 2: Horário de geração da sentença, no fuso horário UTC

Campo 3: Status do dispositivo (A = Ativo, V = Espera)

Campo 4: Latitude (Geoposionamento, Direção)

Campo 5: Longitude (Geoposionamento, Direção)

Campo 6: Velocidade na superfície, em nós

Campo 7: Ângulo de curso real

Campo 8: Data de geração da sentença

Campo 9: Dado de verificação

ARDUINO

É uma plataforma de prototipagem eletrônica de hardware e software livres, que simplifica a criação e prototipagem de projetos de eletrônica. Ele surgiu em 2003, na Itália, por Mássimo Banzi no Interaction Design Institute Ivrea, com fins educativos para pessoas com conhecimento básico de eletrônica.

Existem várias versões de Arduino, pois esse projeto é open-source, então todos que queiram criar sua placa Arduino podem fazê-lo, desde que respeitem a construção mínima determinada pela empresa que controla o projeto, sendo que todas as informações necessárias você encontrará no site da comunidade, no endereço <http://arduino.cc/>.

Programando no Arduino

Para desenvolvimento dos sistemas embarcados, você utiliza linguagem C e/ou C++, o que facilita interações de entrada e saída. Os programas feitos para sistemas embarcados no Arduino, devem ter as duas funções principais, caso contrário, não será gerado o código compilado, e consequentemente não teremos o projeto funcionando no Arduino.

* setup() – Inserida no início, esse método conterá todo o código referente a inicialização do sistema embarcado;
* loop() – É o método que será executado em todo ciclo do controlador. É importante entender esse funcionamento, pois ele será fundamental para o funcionamento perfeito de todo sistema embarcado Arduino.

Abaixo, veremos um código exemplo para fazer o micro controlador acender e apagar um LED. Ele começa configurando o pino 13, da saída digital, no método setup. Após isso, no método loop, ele acenderá o LED por um segundo, apagar o LED por um segundo, esse ciclo ficará se repetindo, até que desliguem o micro controlador.

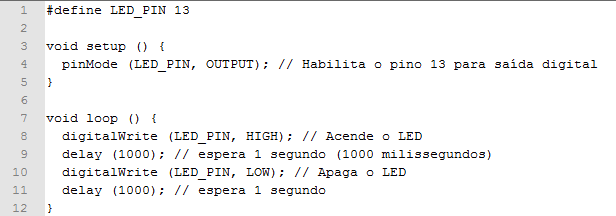


Figura 9 - Exemplo de código para Arduino

A partir desses métodos, o interessado em programar sistemas embarcados para o Arduino poderá desenvolver qualquer projeto, mas para auxiliar esse desenvolvimento, foi criado uma IDE, ou um ambiente integrado para desenvolvimento de software. O Arduino IDE foi produzido em Java, e foi baseada nas IDE para a linguagem de programação Processing e projetos Wiring. Esse aplicativo roda nos sistemas operacionais mais utilizados, como Windows, Linux e Mac OS X.

Shields

Uma peculiaridade dessa placa de prototipagem é o modo padrão que os conectores são expostos, permitindo que a sua CPU possa ser interligada a outros módulos de expansão, sendo esses módulos conhecidos como *Shields*. Existem várias *shields* disponíveis para aquisição, sendo que eles agregam funções como por exemplo: Comunicação via *bluetooth*, via rede celular GSM/GPRS, via rede ethernet, etc.

SOBRE O PROJETO

Neste capítulo, falaremos do projeto em si, mostrando uma visão geral de funcionamento e sobre o hardware e software utilizados para desenvolvimento do mesmo.

Visão Geral

O projeto funcionará da seguinte maneira:

* Um hardware ficará embarcado no veículo, coletando informações, e transmitindo via rádio e também mantendo um arquivo com essas informações no próprio hardware;
* Um software ficará encarregado de recepcionar as informações transmitidas e armazenar num banco de dados local para posterior análise.

Abaixo, temos uma ilustração do funcionamento do projeto. Nesse momento faremos somente com coleta de dados de geoposicionamento, mas posteriormente poderá ser coletada várias outras informações, como rotações do motor, nível de combustível, aceleração lateral, etc.



Figura 10 - Ilustração de funcionamento do projeto (fonte: própria)

Hardware

O hardware embarcado, construído para gerar as informações de tempo de volta e total, velocidade atual, e número de voltas, será baseado no Arduino Uno. Essa placa possui um micro controlador ATmega328, que é fabricado pela Atmel, possui 14 entradas/saídas digitais, 6 entradas analógicas, um ressonador cerâmico de 16 MHz, uma conexão USB, um conector de alimentação, e um botão reset. A tensão de funcionamento é de 5 Volts, sendo que a tensão de entrada recomendada é entre 7 e 12 Volts, podendo chegar a 20 volts. Para efeito de curiosidade, o termo “Uno” significa o numeral “um” em italiano, em referência ao número da versão, Arduino 1.0.

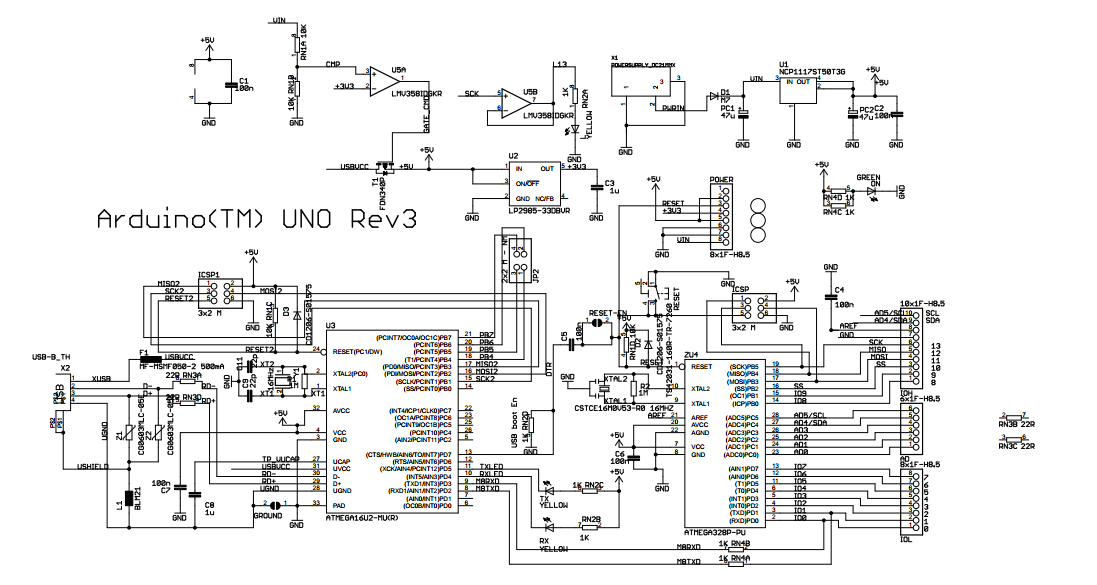


Figura 11 - Esquema eletrônico do Arduino Uno Revisão 3 (fonte: http://www.arduino.cc)



Figura 12 - Arduino Uno (fonte: própria)

Mas somente a placa *Arduino* não é suficiente para esse projeto de hardware, se faz necessário receber as informações de velocidade e coordenadas, para posterior leitura na aplicação Windows. Por isso, foi utilizado um *shield* chamado *GPS Shield Fastrax UP501*, fabricado pela empresa *Fastrax*, dos Estados Unidos. Esse *shield* disponibiliza também uma interface que aceita cartões de memória do tipo Micro SD, que facilita a geração de dados para posterior leitura.

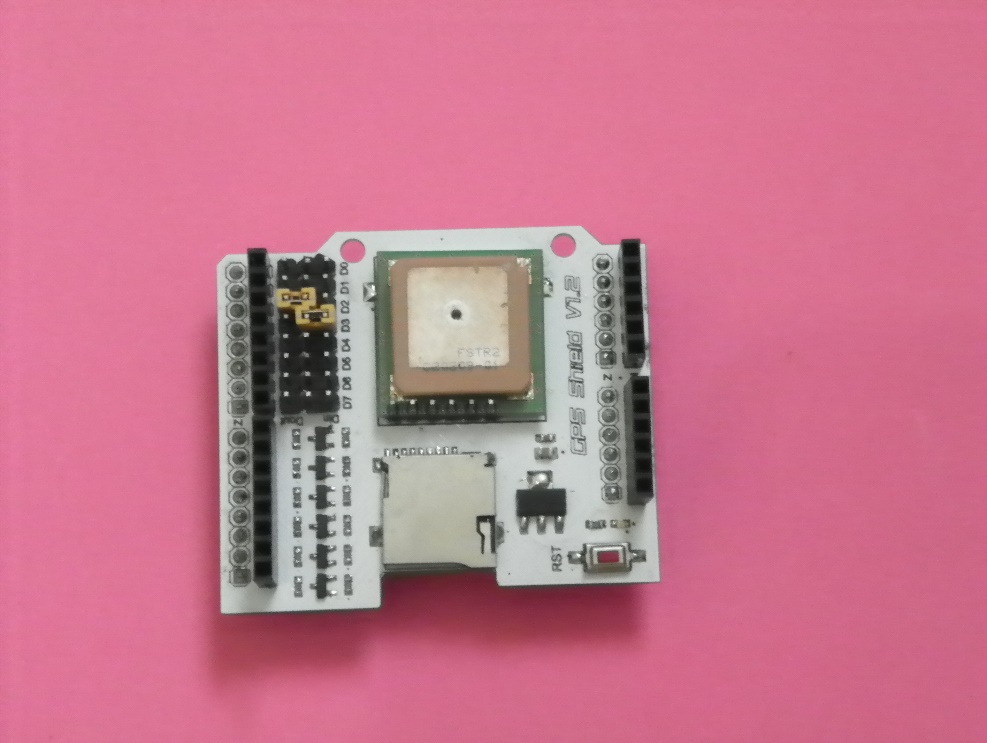


Figura 13 - GPS Shield Fastrax UP501 (fonte: própria)

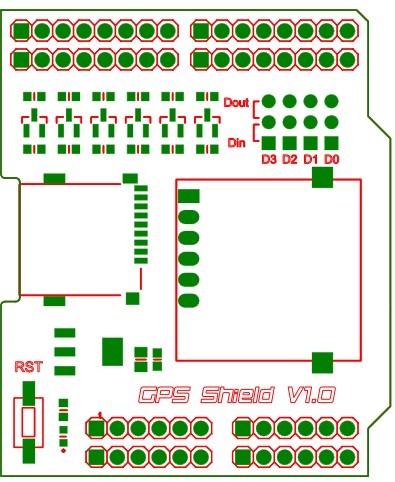


Figura 14 - Esquema da GPS Shield Fastrax UP501 (fonte: www.elecfreaks.com)

Ele tem como qualidade a facilidade de captura de satélites, mesmo com sinal em nível fraco e sua temperatura de operação está na faixa entre 40º Celsius negativos a 85º Celsius positivos. O protocolo de comunicação dessa placa é o UART, ou *Universal Asynchronous Receiver/Transmitter*, que é quando o módulo é composto geralmente por um único circuito integrado que contém, ao mesmo tempo, os circuitos de transmissão e recepção necessários para as comunicações seriais assíncronas.

Para realizar a transmissão de dados do módulo embarcado no veículo, serão utilizados módulos de rádio *XBee*, fabricados pela *Digi International*. O modelo selecionado é o PRO S2B, que tem alcance máximo de 3,2 km, mais do que necessário para cobrir um kartódromo, que normalmente contempla uma área de cento e cinquenta metros quadrados.

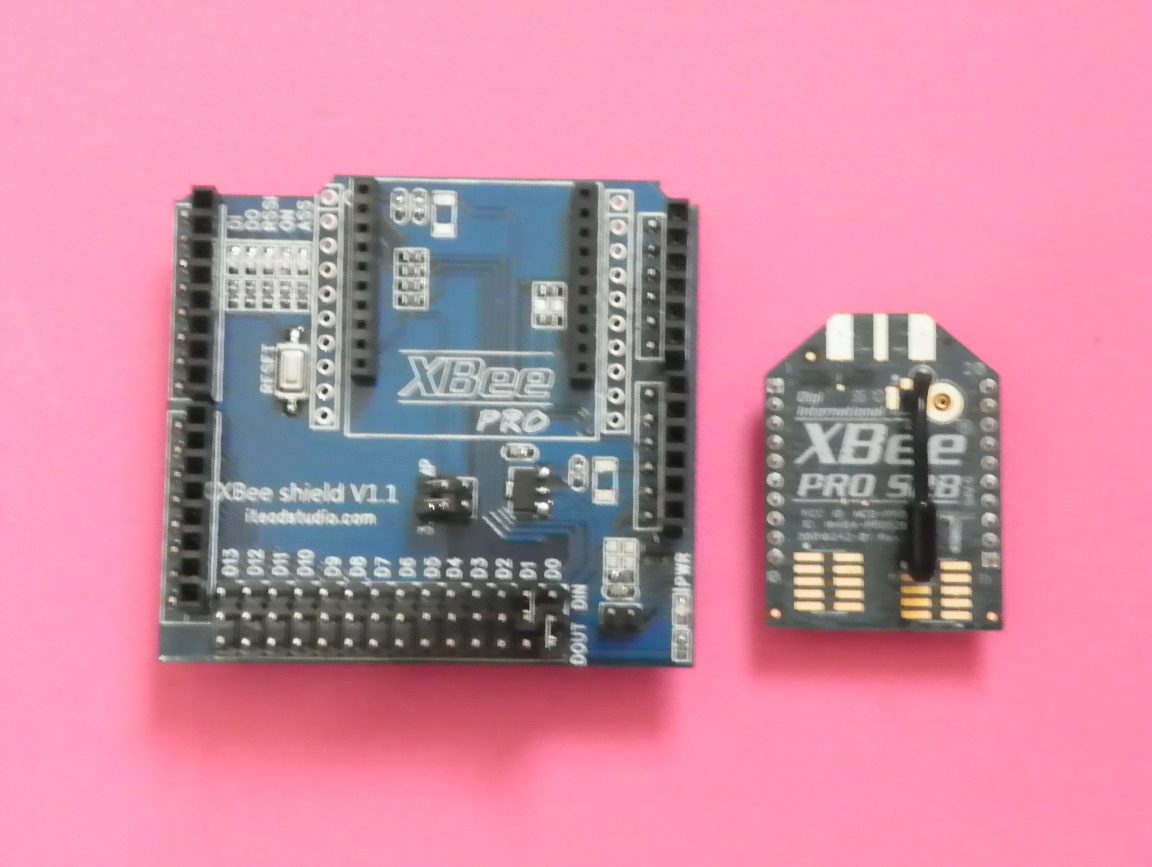


Figura 15 - XBee PRO S2B (fonte: própria)

Software

O software coletor de informação, foi desenvolvido na plataforma Microsoft .NET Framework, utilizando a API Windows Forms, fornecida por essa plataforma, na versão 4.0.



Figura 16 - Diagramas de casos de uso do gerenciador

Aqui, uma breve explicação do diagrama de casos de uso:

* Manter locais: Mantém os locais de competição automobilística (autódromos, kartódromos, etc.). Serve como um indexador para as sessões realizadas, para facilidade de localização e comparação de dados;
* Manter marcações: Mantém as marcações pertinentes a um local de competição automobilística, os tipos são: Linha de largada/chegada, linha para captura de velocidade máxima, e linha para marcação de tempo parcial de volta. Essas marcações são utilizadas no momento da análise;
* Importar sessão *off-line*: O hardware que ficará embarcado no veículo, irá gerar um log das informações localmente, para caso falhe a transmissão via rádio. Caso o usuário deseje importar esse log, ele o fará por esse caso de uso;
* Capturar sessão *on-line*: Nesse caso de uso, o gerenciador estará online com o hardware embarcado, e fará o armazenamento dos logs enviados;
* Manter sessões: Nesse caso de uso, o usuário poderá manipular todas as sessões capturadas, tanto pelo método *on-line* como também no método *off-line*;
* Analisar dados da sessão: Nesse caso de uso, ficarão disponíveis os dados colhidos para análise, de uma maneira de fácil leitura.

Para integração com o Google Maps, foi utilizada uma solução customizada do desenvolvedor, pois foi implementada páginas HTML, que são lidas dentro do sistema por um plug-in open-source chamado Awesomium. Ele é open-source para sistemas desenvolvidos para fins acadêmicos ou que não gerem lucro acima de cem mil dólares americanos.

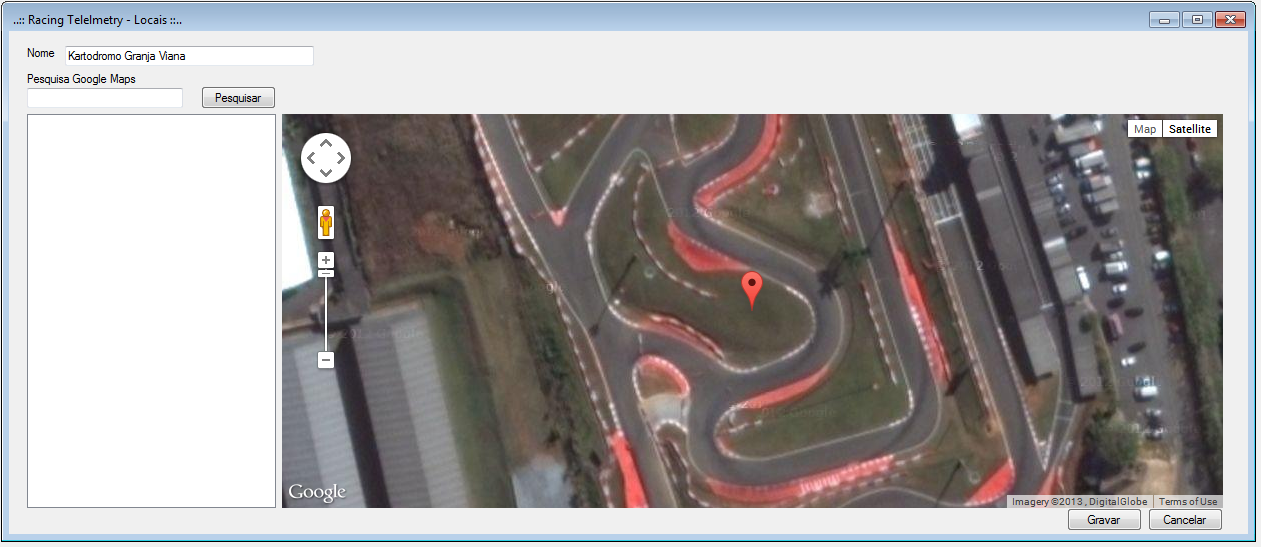


Figura 17 - Sistema utilizando páginas HTML

Para armazenamento de dados, foi utilizada uma solução NoSQL, o MongoDB, uma solução open-source, que para soluções que rodam local, é uma boa alternativa, pois o desenvolvedor pode incluir arquivos dessa solução junto aos arquivos do seu sistema, no pacote de instalação. Com isso, ele ganhar em facilidade de uso do seu problema, e desempenho, já que o uso de hardware de solução de armazenamento de dados do tipo NoSQL são menores que dos tradicionais bancos relacionais (Ex: SQL Server, MySQL, SQLite, etc.).

Com a utilização de tecnologia NoSQL, para armazenamento de dados, também temos a vantagem de não ter problema de conversão de dados, já que a tecnologia escolhida armazena os objetos como documentos (JSON, XML, etc).

Abaixo, veremos as classes construídas para manipulação das informações dentro do sistema.



Figura 18 - Diagrama de Classes

* Location: Classe que representa os locais de competição;
* SpltMarker: Marcações na pista, que podem ser de qualquer tipo, conforme o enumerador SpliMarkerType;
* LogData: Classe que representa as sessões coletadas;
* LogData::NMEA\_GPRMC: Classe interna da classe LogData, para facilitar a leitura das sentenças NMEA, geradas pelo GPS, contido no hardware embarcado.

Para armazenamento das informações, foi criado uma camada de repositório genérica, aproveitando os recursos da linguagem C#, que dá ao desenvolvedor essa possibilidade, mas requer dele conhecimentos fortes de programação orientada a objeto, pois muitos desses conceitos devem ser aplicados para se ter sucesso nesse cenário.



Figura 19 - Diagrama de Classes, repositório de dados

RESULTADOS

Foram realizados testes de prova de conceito, para saber se é o projeto está no caminho certo, o que precisa ser melhorado, entre outras coisas.

O teste off-line foi realizado com sucesso. Nesse teste, utilizamos um carro de rua, e fizemos o trajeto num circuito fictício. Isso foge um pouco do escopo, porém como nesse momento, o hardware embarcado irá capturar somente dados do GPS, ele se tornou uma alternativa para aplicação do teste.

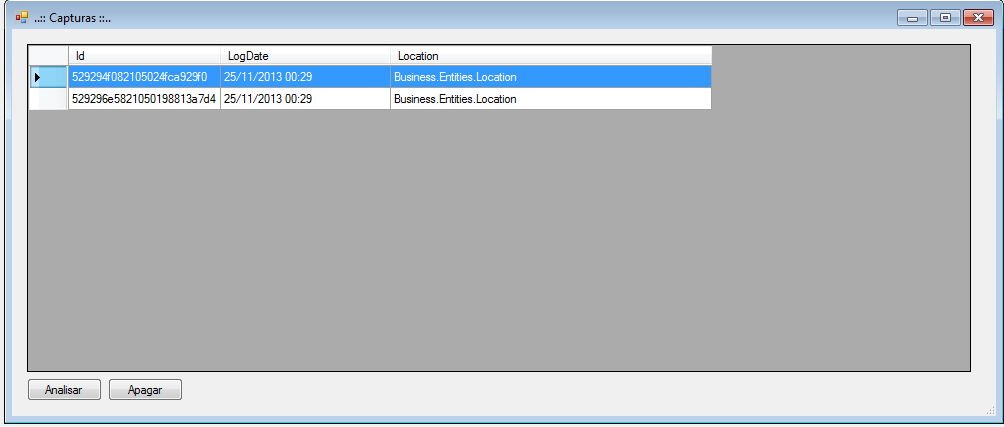


Figura 20 - Capturas realizadas, no modo off-line (fonte: registro pessoal)

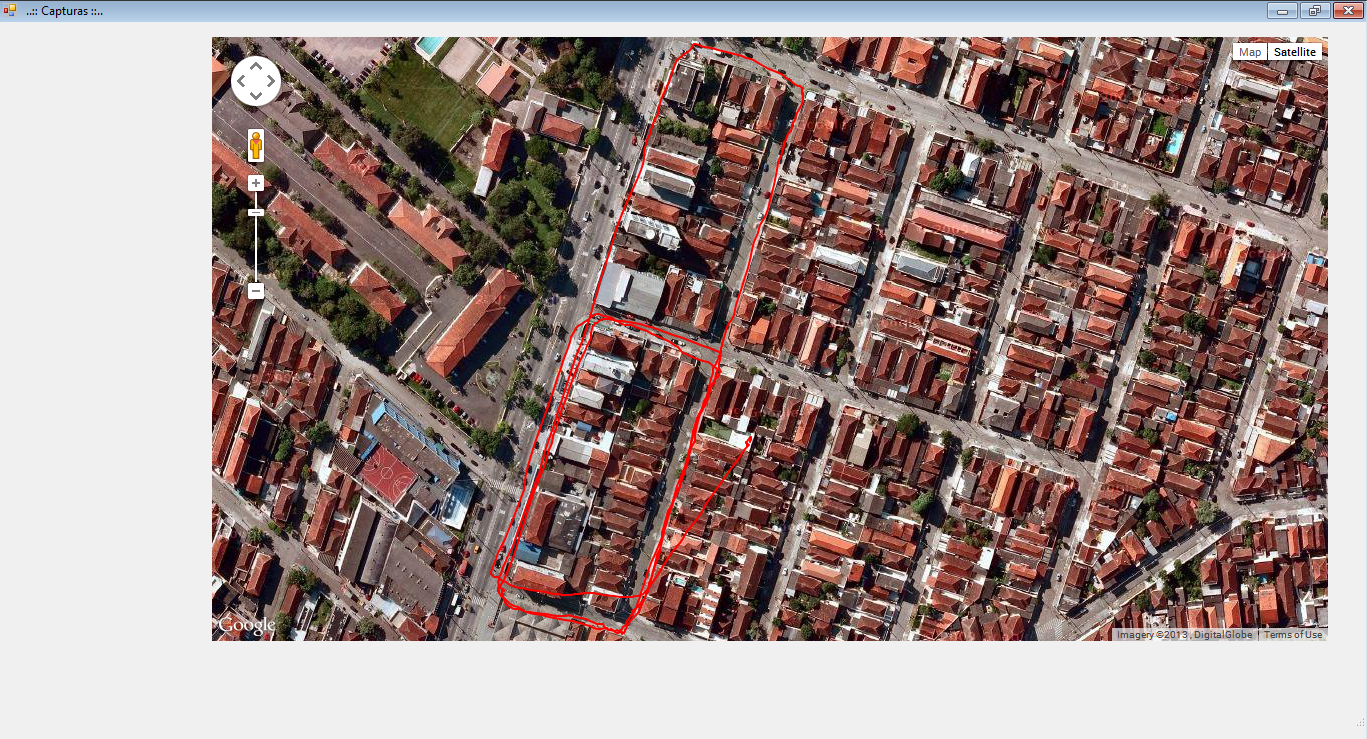


Figura 21 - Trajeto realizado em um dos testes off-line (fonte: registro pessoal)

Para esse método de coleta, podemos considerar um sucesso o resultado, já que ele gravou o arquivo corretamente no hardware embarcado, e o mesmo foi lido sem problemas na importação de dados. E como vemos acima, os dados são fiéis ao teste realizado.

No método online, o autor desse projeto não teve o sucesso esperado, já que não conseguimos nem realizar o teste no campo, pois nos testes de bancada foram identificados alguns possíveis problemas, que estão listados abaixo:

* Possível problema com a voltagem necessária para o conjunto Arduino + GPS *Shield* e XBee. O Arduino trabalha com voltagem máxima de 12 Volts, o GPS Shield trabalha entre 3 e 4.2 Volts, e o XBee PRO S2B trabalha entre 2.7 e 3.6 Volts. Isso não deveria ser um problema, porém estamos levando em consideração;
* Possível problema com interferência entre as frequências de trabalho, pois pela posição física, podemos ter os dois aparelhos anulando o sinal um do outro. Lendo pessoas com projetos parecidos, algumas tiveram o mesmo problema, e não tiveram solução. Uma solução nesse caso é sair da estrutura “*plug and play*” do projeto, e partir para o modo tradicional, soldando as placas, e deixando-as numa disposição que o GPS não interfira no funcionamento do módulo XBee e vice versa.

CONSIDERAÇÕES FINAIS

No princípio, tive a impressão que seria bem mais fácil, ledo engano. Por eu ser um apaixonado pelos todos os assuntos envolvidos nesse artigo (Eletrônica, desenvolvimento de sistemas e automobilismo) ajudou muito na evolução do projeto. Muitos assuntos abordados nesse artigo vieram de conhecimento próprio, já que muita coisa está inerente ao meu dia a dia.

O hardware baseado no micro controlador *Arduino* se mostra boa para soluções off-line, já que o poder de manipulação de periféricos é limitado, devida a baixa memória ROM dele, e por consequência, o código deve ser bem enxuto. Como exemplo dessa limitação, o projeto inicial seria fazer o hardware com um monitor LCD 16x2, para visualização do tempo total, tempo de volta, velocidade e aceleração lateral, porém devido a esse problema citado, em opinião conjunta com a orientadora do artigo Profª Soraya, tomamos a decisão de fazer algo transmitindo.

Baseado nessa limitação, podemos dizer que soluções embarcadas desse tipo devem usar micro controladores mais poderosos, como o RaspBerry Pi, que é uma solução barata e de fácil integração com hardwares periféricos, além de rodar um sistema operacional, onde você não ficará preso somente a uma plataforma de desenvolvimento, já que ele possui um distribuição Linux.

BIBLIOGRAFIA