Relatório Final

Título do projeto de pesquisa:	Desenvolvimento de um monitor de cintilação ionosférica de baixo custo e seu uso em uma rede de larga escala para aprimoramento na modelagem do território brasileiro
Bolsista:	Igor Bragaia
Orientador(a):	Dr. Alison de Oliveira Moraes
Período a que se refere o relatório:	Agosto de 2016 a Julho de 2017

Resumo

O monitor de cintilação é uma ferramenta fundamental para estudo da ionosfera, bem como para entender seus efeitos para os usuários de sistemas de navegação via satélite. Os monitores de cintilação profissionais disponíveis no mercado custam entre 3.000 e 20.000 dólares, sem considerar os impostos. O objetivo desta linha de ação é validar um protótipo de um monitor de cintilação ionosférico de baixo custo (menos de 100 dólares) e resolução adequada, capaz de captar dados que possam auxiliar nas pesquisas relacionadas ao tema. Através desse protótipo pretende-se obter o índice de cintilação S4, o parâmetro tempo de descorrelação τ (Moraes et al., 2011) e os coeficientes de fading $\alpha - \mu$ (Yacoub 2007). Para cumprir estes requisitos básicos, este projeto utilizou a plataforma Arduino, dispensando assim o uso de procedimentos de operação complexos. Após uma série de testes com dados de campo, decidiu-se adotar a taxa de amostragem 10 Hz para cada canal GPS e a aquisição e gravação dos dados GPS será realizada das 20:00 GMT às 05:00 GMT, diariamente, de maneira automática, de modo a alimentar um banco de dados que posteriormente tem suas informações processadas no front-end de um website. Por fim, os dados obtidos foram analisados e o protótipo foi validado com base em comparações dos resultados com monitores profissionais já consolidados.

1. Introdução

Um monitor ionosférico é um dispositivo físico de medição que tem como finalidade obter dados sobre o estado da ionosfera ao longo de um prolongado período de tempo a uma alta

taxa de amostragem, para posterior determinação de índices ionosféricos. A explicação da importância e usos de um monitor ionosférico se encontra fora do escopo deste documento.

O monitor ionosférico desenvolvido neste projeto, denominado de IONIK, é baseado na plataforma Arduino e realiza medições com o uso de um sensor GPS a uma taxa de amostragem de 10 Hz. Estas medições são no formato NMEA (é um conjunto de especificações de dados e elétricas para comunicação de dispositivos eletrônicos de navegação) armazenadas como arquivo de texto em um cartão SD e em seguidas enviadas para um banco de dados para posterior processamento e determinação de índices por meio da execução de algoritmos.

Em relação à taxa de amostragem, um monitor ionosférico comercial tipicamente atua a uma taxa de amostragem em média de 50 Hz. No entanto, de acordo com testes preliminares com dados obtidos em monitores em operação do Instituto Nacional de Pesquisas Espaciais (INPE), verificou-se que uma taxa de amostragem de 10 Hz apresenta desempenho satisfatório na determinação de índices que descrevem a condições ionosféricas.

No contexto do sensor GPS, verificou-se a viabilidade do uso do sensor Adafruit (Companhia de Hardware Open-Source) Ultimate GPS Breakout. A escolha deste sensor foi devida ao seu relativo baixo custo, com o objetivo de manter cada sensor IONIK barato; à sua taxa de 10 Hz, que cumpre o pré-requisito de determinação de índices ionosféricos; à possibilidade de configuração através de comunicação serial, visando obtenção de dados necessários para determinação de índices; e fácil integração à plataforma Arduino, com bibliotecas oferecidas pelo fabricante do sensor. A escolha da plataforma Arduino foi devido ao seu baixo custo, ampla disponibilidade e facilidade de desenvolvimento.

2. Materiais e métodos

O monitor ionosférico IONIK possui os componentes:

- Placa Arduino UNO R3 ($15)
- Sensor GPS Adafruit Ultimate GPS Breakout ($40)
- Shield adaptador para Ethernet e Cartão SD W5100 ($10)
- Cartão SD Kingston de 4 GB ($5)
- Placa de prototipagem de 400 furos ($4)
- Jumpers
- 1 Botão switch
- 3 Resistores 1 k ohm
- 1 LED vermelho
- 1 LED verde
- 1 cabo USB Tipo A macho para Tipo B macho

- 1 Carregador de celular USB 5V Tipo A fêmea
- 1 cabo Ethernet

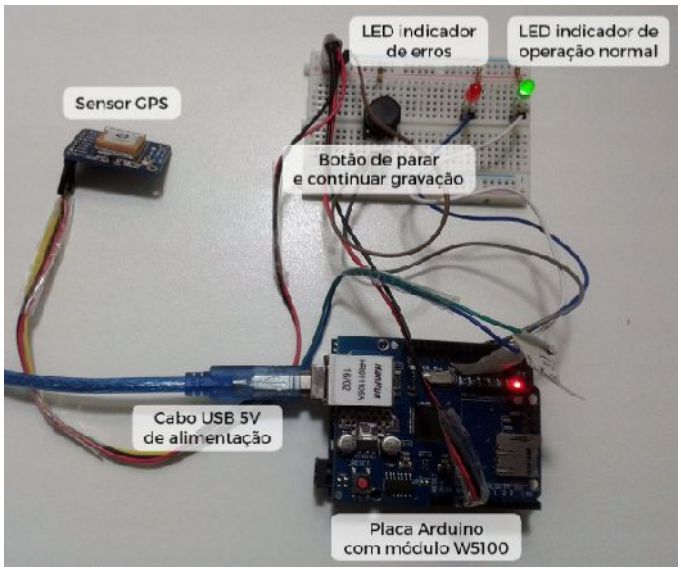

Figura 1 Disposição dos componentes em seu estado final montado.

Diante dos componentes listados acima e seus preços, nota-se que o protótipo pode ser construído com menos de 100 dólares, o que justifica seu caráter de baixo custo e permite ser obtido em larga escala.

A placa Arduino é alimentada através de sua entrada USB Tipo B fêmea de 5V. Para tal, faz-se uso de cabo USB Tipo A macho para USB Tipo B macho, conectado a um carregador de celular com entrada USB Tipo A fêmea, que atua como fonte de 5V para o circuito. A placa Arduino é responsável por alimentar os demais componentes do circuito, o sensor GPS, o shield W5100 e os LEDs. Devido aos baixos requerimentos de corrente e energia, tal configuração se mostrou satisfatória.

Além disso, a programação da placa Arduino é realizada com o software disponível no site da plataforma Arduino (o código embarcado no Arduino é escrito em C) e é possível a necessidade de instalação de drivers adicionais para placas não-genuínas. Também se tem dispostos LEDs verde e vermelho para indicar o correto funcionamento do dispositivo. Por fim, o cartão SD a ser utilizado deve estar necessariamente formatado no padrão FAT32 e preferencialmente vazio.

2.1 Visão geral e Requerimentos do sistema

O funcionamento do IONIK se dá da seguinte forma: à medida que o sensor GPS recebe dados dos satélites disponíveis naquele instante, os dados são capturados em um buffer

UART e consumidos, sendo escritos em um cartão SD para evitar descontinuidade na captação. Após um longo período de exposição programado, inicia-se o envio dos dados do cartão SD à um banco de dados. A posteriori, um website realiza consultas nesse banco de dados, capturando o dado bruto em NMEA e processando conforme interesse do pesquisador.

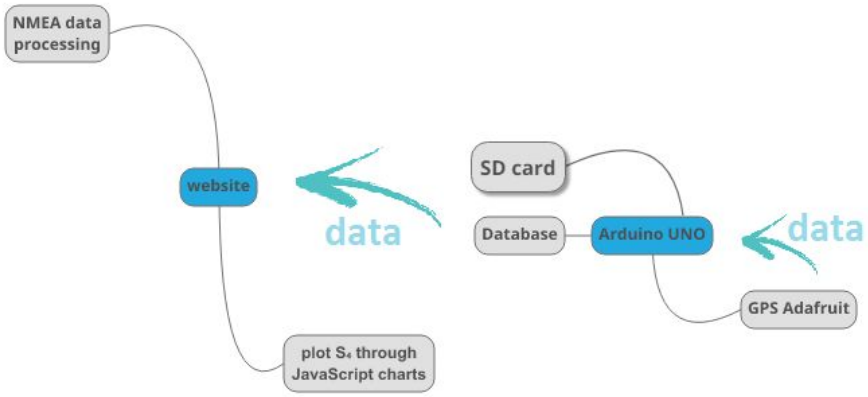

Figura 2 Visão geral do funcionamento do IONIK.

2.1 Dados NMEA

O monitor ionosférico IONIK grava em seu cartão SD arquivos de texto com a saída bruta do sensor GPS no padrão NMEA. Posteriormente, os dados são enviados para um banco de dados por meio de "GET method" no software do Arduino. Assim, garante-se que todo conteúdo recebido é persistido. Além disso, verificou-se que para 8 horas de gravação os dados ocupam 100 MB para 8 horas de gravação.

O quadro abaixo é um típico exemplo obtido com o protótipo desenvolvido. O sensor gera 10 quadros como esse por segundo.

```
$GPGGA,220020.300,2312.1863,S,04552.4492,W,1,06,1.49,574.2,M,-4.2,M,,*74
$GPGSA,A,3,31,22,23,11,01,03,,,,,,,1.77,1.49,0.95*02
$GPGSV,3,1,12,01,67,311,38,22,62,126,15,03,50,196,26,33,46,056,28*72
$GPGSV,3,2,12,23,43,249,30,11,42,326,22,31,29,138,15,26,20,077,18*78
$GPGSV,3,3,12,09,18,273,,16,16,048,,17,04,252,,08,02,354,*7F
$GPZDA,220020.300,10,04,2016,,*57
```

Figura 3 Dado bruto em padrão NMEA.

O quadro pode ser interpretado da seguinte maneira:

$GPGGA,220020.300,2312.1863,S,04552.4492,W,1,06,1.49,574.2,M,-4.2,M,,*74
Dados da localização obtida via satélite (*Fix*) Tempo: GMT: 22 horas 0 minutos 20 segundos 300 microssegundos Latitude: 23° 12,1863' S Longitude: 45º 52,4492' O 6 satélites em vista 574.2 metros acima do nível do mar
$GPGSA,A,3,31,22,23,11,01,03,,,,,,,1.77,1.49,0.95*02
Dados de erro do satélite (*Dilution of Precision*) e identificadores dos satélites visíveis
$GPGSV,3,1,12,01,67,311,38,22,62,126,15,03,50,196,26,33,46,056,28*72 $GPGSV,3,2,12,23,43,249,30,11,42,326,22,31,29,138,15,26,20,077,18*78 $GPGSV,3,3,12,09,18,273,,16,16,048,,17,04,252,,08,02,354,*7F
Dados dos satélites visíveis: elevação, azimute e intensidade do sinal para cada satélite
$GPZDA,220020.300,10,04,2016,,*57
Dados de tempo de aquisição. Tempo GMT: 22 horas 0 minutos 20 segundos 300 microssegundos Data: 10 de abril de 2016

Figura 4 Interpretação do dado bruto em padrão NMEA.

Por fim, o dado precisa ser organizado de forma apropriada para que assim coeficientes de interesse possam ser obtidos. Assim, após a execução dos códigos em JavaScript abaixo, obtém-se um novo conjunto de dados organizados em:

"HrMinSec(UTC);Day;Month;Year;Satellite PRN number;Elevation;Azimuth;SNR".

```
function getTime(x, dd, mm, yy) {
    hh = parseInt(x.substring(0, 2)) - 1;
    min = parseInt(x.substring(2, 4)) - 1;
    yy = parseInt(yy);
    mm = parseInt(mm) - 1;
    hh = parseInt(hh) - 1;
    return new Date(yy, mm, dd, hh, min, 0, 0).getTime();
}
```

Figura 5 Função que padroniza o formato da data.

```
function processData(array){
    var newArray=[],currentPackage=[];
    newArray.push("HrMinSec(UTC);Day;Month;Year;Satellite PRN number;Elevation;Azimuth;SNR;");
    var gpzda,gpgsv,length;
    var y=0,n=0;
    for(var row of array){
      try {
          if(row[1]=='G' && row[2]=='P' && row[3]=='G' && row[4]=='S' && row[5]=='V')
          {
            /// example $GPGSV,2,1,08,13,55,228,35,19,54,009,34,30,44,110,31,28,43,159,31*7A
            gpgsv=row.split(',');
            gpgsv=gpgsv.slice(4,gpgsv.length);
            gpgsv[gpgsv.length-1]=gpgsv[gpgsv.length-1].split('*')[0];
            for(var i=0;i<=gpgsv.length-4;i+=4)
            {
              //currentPackage.push(gpgsv[i]+";"+gpgsv[i+4]+";");
              currentPackage.push(gpgsv.slice(i,i+4).join(';')+";");
            }
          }
          else if(row[1]=='G' && row[2]=='P' && row[3]=='Z' && row[4]=='D' && row[5]=='A')
          {
            /// example $GPZDA,201530.00,04,07,2002,00,00*60
            gpzda=row.split(',').slice(1,5).join(";")+";";
            for(var newRow of currentPackage)
            {
              newArray.push(gpzda+newRow);
            }
            currentPackage=[];
          }
      } catch (e) {
          continue;
      }
    }
    return newArray;
}
```

Figura 6 Função que organiza o dado bruto do NMEA por satélites.

Em seguida, a partir desse conjunto de dados é obtido um array de dados na forma "*Date Object; S_4 Index*" para um *Satellite PRN* especificado. Esse procedimento está descrito abaixo (Para o caso em que PRN=14). Nota-se que a função *getSummary(array)* abaixo recebe como parâmetro um *array* de NMEA processado acima. Além disso, nesse trabalho, o índice S_4 por minuto utilizando-se, em média, 600 amostras de SNR para dado satélite por meio da expressão

$$S_4 = \frac{std(I)}{mean(I)}, \text{ onde } I = 10^{\frac{SNR}{10}}$$

```
function getSummary(array) {
    var lista = [], returnArray = [], summary = [], summarize = [], snr, time;
    var minute = array[1].split(';')[0].substring(2, 4);
    for (var i = 1; i < array.length; i++) {
        try {
            snr = array[i].split(';')[7];
            var id = array[i].split(';')[4];
            if (id=="26") {
                if (i == array.length - 1 || minute != array[i + 1].split(';')[0].substring(2, 4)) {
                    if (i != array.length - 1)
                        minute = array[i + 1].split(';')[0].substring(2, 4);
                    time = array[i].split(';');
                    if (summarize.length > 0)
                        summary.push(getTime(time[0], time[1], time[2], time[3]) + ";" + s4index(summarize) + ";");
                    summarize = [];
                }

                if (snr != "") {
                    summarize.push(snr);
                }
            }
        } catch (e) {
            continue;
        }
    }
    return summary;
}
```

Figura 7 Função que recebe o dado NMEA organizado e prepara de forma mais fina para obtenção do índice S_4

```
function s4index(array) {
    var intensity = [];
    for (var snr of array) {
        intensity.push(Math.pow(10, parseInt(snr) / 10.0));
    }
    return (100 * math.std(intensity) / math.mean(intensity)).toFixed(1);
}
```

Figura 8 Função que extrai o índice S_4

Também é realizado um procedimento semelhante a partir do NMEA processado para se obter um array de dados no formato "*Date Object; Ionik's satellite elevation"*;

2.1 Web

Para a arquitetura do back-end, optou-se pela linguagem de programação PHP pela facilidade de se executar um servidor local e ter-se a disposição um banco de dados mySQL, bem como pela ampla comunidade existente documentando a linguagem de forma completa e objetiva. Além disso, para a execução de um protótipo inicial e sua posterior validação, utilizou-se um servidor XAMPP, o qual é um servidor independente de plataforma baseado na base de dados mySQL e que permite criar um servidor local de fins de teste.

```php
<?php
  $usuario="root";  $senha="";  $host="localhost";

  $conexao=mysql_connect($host,$usuario,$senha);
  $selecionadb=mysql_select_db('ionik',$conexao);

  $status=$_GET["status"];
  $data=$_GET["data"];

  $sql_insert = "insert into test (status,data) values ('$status','$data')";

  mysql_query($sql_insert);
?>
```

Figura 9 Código em php para abrir conexão com banco de dados e salvar dados

Para a linguagem do front-end, programou-se em HTML, CSS e fez-se uso de Javascript, Bootstrap e Chart.js para a representação gráfica dos dados obtidos.

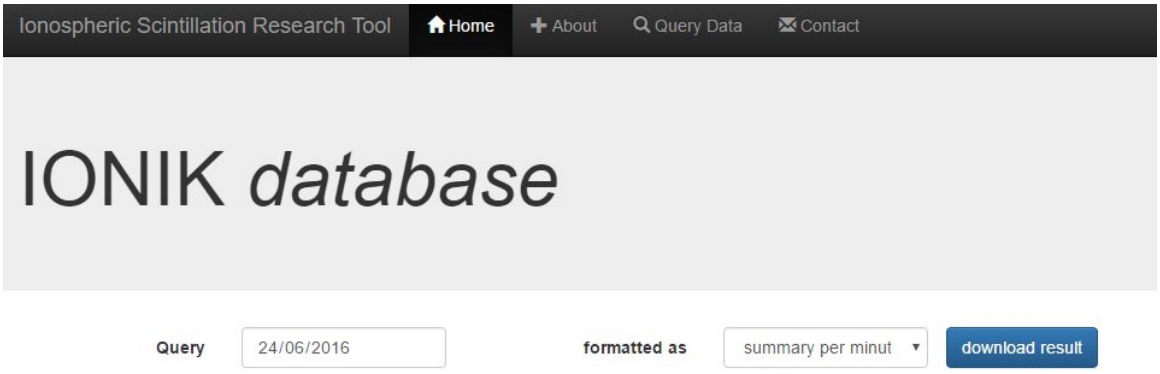

Figura 10 Website em desenvolvimento.

Na arquitetura desse projeto de web development, o usuário seleciona um dia a se fazer a busca, bem como a forma como deseja receber os dados, e em seguida seus inputs são enviados à uma página PHP, uma busca no banco de dados é feita e o resultado é retornado. Nesse sentido, a comunicação entre o PHP e o JavaScript é feita via JSON, entre o HTML e o PHP é feita via "POST method" e entre o HTML e JavaScript é feita diretamente via edição do DOM da página.

Finalmente, após o JavaScript receber um conjunto de dados bruto no padrão NMEA, os algoritmos para processar o dado e obter o índice S_4, descritos na seção anterior, são aplicados no client-side e os resultados são exibidos ao usuário.

3. Resultados

Inicialmente, após finalizar um protótipo que apenas grava os dados em um cartão SD, submeteu-se diretamente os dados brutos do NMEA em um script de Matlab a fim de validação do projeto. Os resultados obtidos estão descritos abaixo.

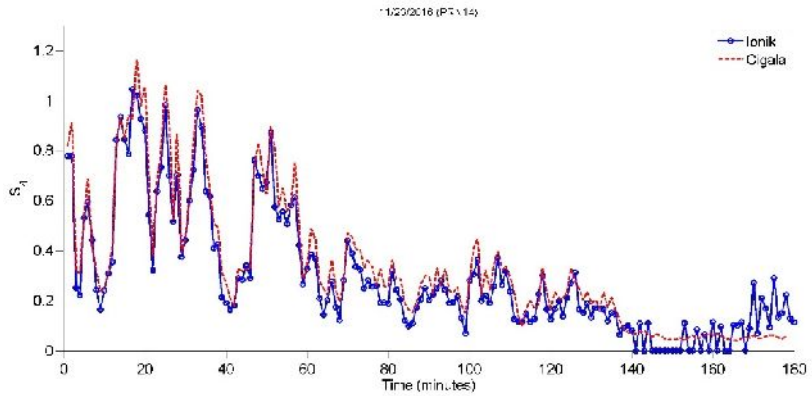

Figura 11 Validação em Matlab para Satellite PRN = 14.

Em seguida, validou-se a rede no contexto de localhost. Abaixo segue a representação gráfica dos dados do IONIK processados pelos algoritmos programados em Javascript. Para a obtenção dos gráficos, optou-se por utilizar a framework Chart.js.

Além disso, optou-se por representar $100S_4$ e a elevação do satélite em um mesmo gráfico a fim de melhor compreensão do resultado. Assim, pode-se notar, por exemplo, que para menores elevações o sinal tende a ser ruidoso e isso pode ser devido a outros efeitos que não a cintilação ionosférica, tais como multicaminhos, obstáculos (prédios, árvores, montanhas, etc). No procedimento de validação, exibiu-se também, a fim comparativo, o resultado obtido pelo Cigala mas nota-se que na versão definitiva da rede esse dado não será exibido concomitantemente.

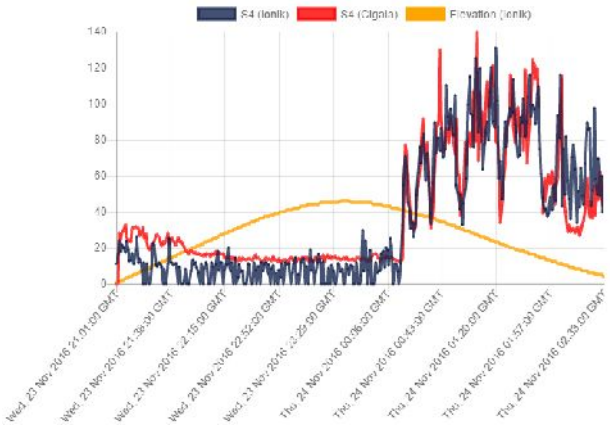

Figura 13 Validação Web para Satellite PRN = 26.

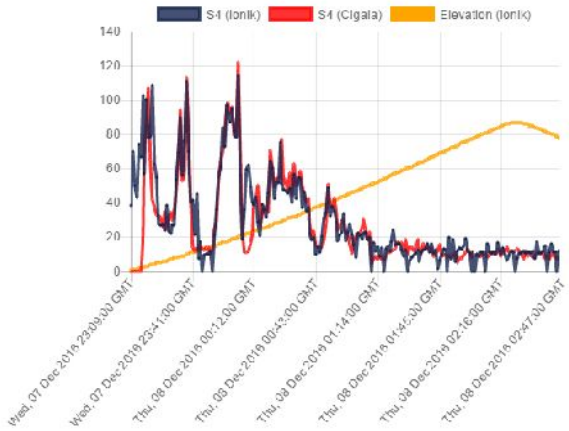

Figura 14 Validação Web para Satellite PRN = 14.

Conforme pode-se observar acima, os resultados obtidos pelo IONIK para o cálculo do índice ionosférico S_4 são satisfatórios diante do resultado obtido pela rede de monitores profissional Cigala.

4. Próximas Etapas

As próximas etapas do projeto consistem na integração do projeto com um domínio consolidado de um instituto como o INPE ou o próprio sistema Cigala. Além disso, após o deploy do IONIK, buscar-se-á dar início a montagem de uma rede de monitoramento ionosférico em larga escala, distribuindo-se protótipos em locais estratégicos a fim mapear o território brasileiro.

5. Conclusões

Após a validação do protótipo, pôde-se concluir que é possível a obtenção de dados satisfatórios relativos à estudos ionosféricos por meio de equipamentos de baixos custos baseados em tecnologias menos complexas do que equipamentos profissionais disponíveis no mercado. Além disso, o baixo custo do protótipo e sua flexibilidade de uso permite com que sejam distribuídos monitores em locais diferentes e assim obtido um mapeamento da ionosfera em um território amplo permitindo estudos mais complexos e de maior impacto sobre o território brasileiro.

6. Divulgação dos Resultados

1. Em 28 de julho de 2017, o projeto foi apresentado, em inglês, em um workshop promovido pelo INPE e com dois pesquisadores do *Deutsches Zentrum für Luft- und Raumfahrt* (DLR) relativo à estudos de efeitos da ionosfera em sistemas GNSS.
2. Abstract aceito na 32nd URSI (Union for Radio Science International) Conference em Montreal, Canadá.
3. Existe um interesse com base na divulgação deste trabalho de uma parceria entre o IAE e a Universidade do Texas-Dallas para um projeto de expansão da rede Ionik.

Referências

[1] http://arduino.cc/

[2] http://www.gpsinformation.org/dale/nmea.htm

[3] https://thenewboston.com/