UNIVERSIDADE FEDERAL DE SANTA CATARINA - UFSC
CENTRO DE ENGENHARIA ELÉTRICA E ELETRÔNICA

LUCAS PEREIRA LUIZ
PEDRO HENRIQUE KAPPLER FORNARI

A HOLE MAP

FLORIANÓPOLIS
2016

LUCAS PEREIRA LUIZ

PEDRO HENRIQUE KAPPLER FORNARI

A HOLE MAP

Relatório técnico apresentado como requisito parcial para a obtenção de aprovação na disciplina Sistemas Embarcados, no Curso de Engenharia Eletrônica, na Universidade Federal de Santa Catarina.

Prof. Dr. Djones Vinicius Lettnin

FLORIANÓPOLIS

2016

SUMÁRIO

RESUMO

A HOLE MAP

Objetivo: Estudar e desenvolver um sistema capaz de detectar a presença de buracos em estradas usando uma matriz de sensores acoplada na parte inferior de um veículo. **Material e Método:** Três sensores ultrassônicos foram usados (o quarto não operou dentro dos limite esperado) para se fazer coletas de dados reais, em conjunto com um arduino. Para modelagem do sistema foram usados diagramas SysML. Os módulos e algoritmos então foram desenvolvidos em C++ e SystemC. O comportamento do sistema inicialmente foi observado por funções que mostravam as ocorrências no Terminal, mas, com o avanço do projeto, foi usado o GTKWave no final. **Resultados:** Não ocorreram erros de implementação. A maior taxa de acertos foi de 71.43%, ocorrendo apenas uma vez com um dos algoritmos. Vale lembrar que o desejado era 80%. **Conclusão**: O sistema desenvolvido não realizou uma detecção satisfatória porém, com um melhor estudo de filtros para se melhorar o algoritmo ou com o uso de outros tipos de sensores, pode-se melhorar a taxa de acertos. Tanto no cenário de novos sensores quanto no de um melhor algoritmo, não são necessárias mudanças na maioria do sistema já desenvolvido, visto que tentou-se realizar uma implementação genérica e de fácil manipulação.

Descritores: Sensores, Buracos em Estradas, Veículo, SysML, SystemC, C++, Filtros, Algoritmos.

1 INTRODUÇÃO

A anos os cidadãos sofrem com problemas ocasionados por vias públicas mal conservadas, pois além de fazer com que os motoristas sejam obrigados a desviar a atenção do trânsito para se atentar aos defeitos das pistas e autopistas [1]. Apesar de não existirem estatísticas concretas sobre qual a frequência com que pistas mal conservadas são a causa do acidente de trânsito, o portal [2] traz estatísticas fornecidas pelos cadastros realizados pelos órgãos DENATRAN, DATASUS e DPVAT, que indicam pessoas que deram entrada em algum serviço disponível à população que sofre acidentes de trânsito.

A responsabilidade de manter as pistas seguras, trafegáveis e uniformes é pública, ou, em alguns casos, da empresa que possui a concessão pela via. Isso é confirmado por [1], quando cita a defesa feita pelo Desembargador do Rio de Janeiro Nagib Slaibi Filho, tendo como base os artigos 29 e 30 da Constituição. [1] Ainda cita outros artigos e cartas políticas que levam os juízes a formular teorias, geralmente baseada no fato de a omissão da administração pública ocasionar um dano a um ou mais civis, o que fere a Responsabilidade Civil, culpando a administração e resultando assim na indenização por danos materiais e/ou morais a quem sofre acidente por esse problema.

Com essa motivação, este relatório visa apresentar uma abordagem para o desenvolvimento de um produto que mapeie os buracos das cidades, a partir de módulos facilmente aplicáveis e removíveis que possam ser instalados em automóveis públicos por determinado tempo. Assim, o sistema deverá identificar as posições dos buracos e armazená-las em uma memória, que, após a retirada dos equipamentos, será integrada, compilada e verificada, resultando em um relatório para a administração pública da posição onde se encontram os buracos das pistas.

Esse sistema poderá resultar em um prejuízo menor para a administração, visto que indenizações, ou até despesas de hospitalização, são muito mais prejudiciais e caras do que consertar a irregularidade na pista, além de melhorar a imagem da administração perante a população do município, diminuir o número de acidentes ao longo das cidades, e melhorando a qualidade de vida dos motoristas.

2 ESPECIFICAÇÕES E REQUISITOS

Ao longo desse trabalho foi desenvolvida apenas uma abordagem de sensoriamento para a detecção dos buracos - usando uma matriz de sensores ultrassônicos - a fim de avaliar se esse tipo de sensor atenderia aos requisitos do projeto. Assim, este capítulo é dividido em quatro subtópicos: inicialmente são apresentados os requisitos do sistema completo, após isso suas especificações e então fazemos a mesma abordagem especificamente para os sensores ultrassônicos. As especificações e requisitos desse projeto foram desenvolvidas com base em diagramas SysML, que serão apresentados a seguir.

2.1 REQUISITOS DO PROJETO COMPLETO.

Este subtópico é iniciado pela apresentação dos requisitos do sistema, os quais podem ser visualizados no diagrama de requisitos apresentado na Figura 1. O sistema deve conter 3 componentes principais, um processador, sensores e um GPS.

O papel do processador é de processar os dados provenientes dos sensores utilizando filtros e algoritmos avançados para decidir corretamente o que é ou não é um buraco, e armazenando em uma memória externa a posição dos buracos. Baseado nisso, seu principal requisito é ter uma capacidade de processamento suficiente para executar as funções necessárias para apresentar dados corretos. Logo os projetistas devem especificar compras de controladores com memória e frequência suficiente para executar os algoritmos projetados. Além disso ele deverá ter disponibilidade de entradas e saídas suficientes para atender a conexão dos sensores e dos módulos externos, como GPS e memória.

O principal requisito do GPS é atender a precisão da localização escolhida pelos projetistas, que, no caso desse projeto foi um raio de 10 metros. Assim, os projetistas devem verificar esse requisito antes e depois da compra do módulo.

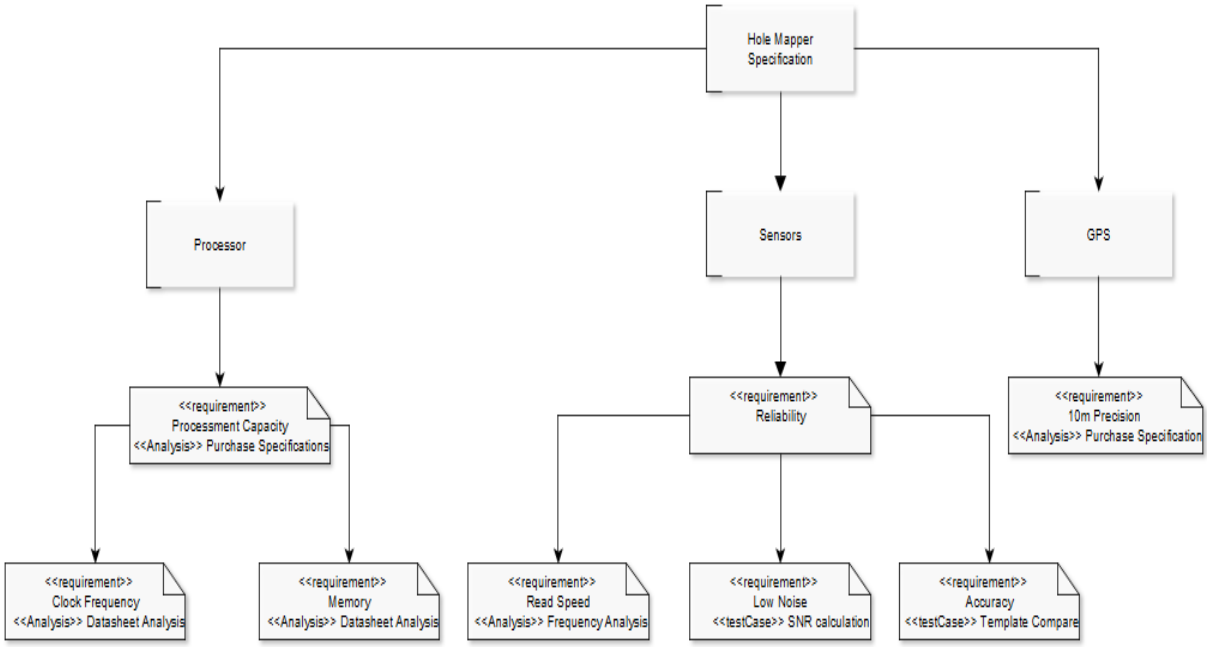
Por fim, os sensores devem ser confiáveis. Isso significa atender aos requisitos de:

- Resposta rápida, de modo a fornecer os dados em sequência em uma velocidade suficientemente alta para que o processador consiga identificar um buraco;

- Baixo ruído, ou seja, o sensor deve fornecer um dado que possa ser tratado pelos filtros, resultando em uma diferença entre amplitude e período de ruído e amplitude e período de buraco suficientemente grande para que o processador identifique os buracos;
- Precisão na aquisição dos dados, o que significa fornecer uma medida que indique realmente a diferença entre uma posição com buraco e uma posição sem buraco.

Esses requisitos serão exemplificados posteriormente para a matriz de sensores ultrassônicos, dando uma visão mais aplicada e de fácil entendimento de cada requisito citado acima.

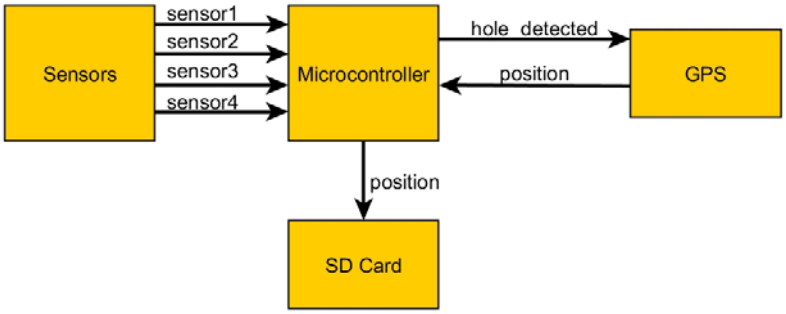
FIGURA 1 - DIAGRAMA DE REQUISITOS DO SISTEMA COMPLETO.



2.2 ESPECIFICAÇÕES DO PROJETO COMPLETO.

Nesse tópico serão abordados alguns diagramas que auxiliam no entendimento do projeto como um todo. Iniciando pelo diagrama de blocos do sistema (apresentado na Figura 2) como podemos observar existe um bloco para os sensores utilizados. Esse bloco, que simula todos os sensores utilizados, se conecta ao controlador a partir de um barramento de dados, e cada barramento representa um sensor diferente. O controlador recebe os dados dos sensores e está conectado a uma memória externa, nesse caso representada pelo cartão SD e a um GPS. Quando decidir que um buraco foi identificado o controlador transmite um sinal ao GPS, que responde a posição atual do veículo. Essa posição é então enviada para o cartão SD, que armazena o dado para que depois seja compilado para um relatório.

FIGURA 2 - DIAGRAMA DE BLOCOS DO SISTEMA COMPLETO.



A figura 4 ilustra o Diagrama de Sequência do projeto completo, ou seja, qual o fluxo que cada componente do sistema deve seguir para que ele funcione corretamente. Então, como é possível observar, o sistema é iniciado quando o veículo é ligado. Este inicializa o controlador, que por sua vez, inicializa os outros módulos do sistema como GPS, Cartão SD e os sensores. Após completa a inicialização, o veículo é considerado em movimento e o controlador fica constantemente se comunicando com os sensores, a fim de fazer a aquisição e processamento dos dados o mais rápido possível.

A aquisição e processamento dos dados se repete até que o algoritmo identifique um buraco, onde então deixa a aquisição de dados de lado por um pequeno período de tempo a fim de fazer a requisição da posição do GPS. Assim que o GPS retorna a posição atual do veículo, o controlador deve armazenar a posição recebida

em um buffer interno e tentar abrir o arquivo do cartão SD para armazenar a posição da irregularidade. Se nenhum problema acontecer nessa tentativa, a posição é armazenada no cartão SD, o controlador apaga a posição do buffer interno e retoma a comunicação com os sensores a fim de encontrar outra irregularidade na pista. Caso as tentativas para abrir o arquivo ultrapassem um limite de tempo, a posição é mantida na memória interna do controlador e o mesmo retoma a comunicação com os sensores. Se o problema persistir e a memória interna não for mais suficiente para armazenar as posições, o controlador irá salvar uma flag para identificação do problema e sobrescreverá dados, perdendo informação.

Ainda foi desenvolvido para modelar esse projeto um diagrama de estados, o qual pode ser verificado na figura 3. Nele podemos observar que após o sistema estar inicializado os sensores transmitem dados para o algoritmo de detecção, porém enquanto nenhum buraco for identificado pelo algoritmo, os únicos dois estados acessados são o de aquisição de dados e o estado de detecção, onde os dados passam pelo algoritmo que decide se existe ou não um buraco naquela determinada posição.

Caso um buraco seja detectado, o sistema passará para um estado de aquisição de posição. Feita aquisição de posição, o sistema armazena a posição em um buffer interno, e então, conforme dito no diagrama de sequência, são feitas algumas tentativas de armazenamento da posição em uma memória externa. Se o tempo máximo disponível para essas tentativas for excedido, o sistema volta a executar a leitura dos sensores e processamento dos dados.

FIGURA 3 - DIAGRAMA DE ESTADOS DO PROJETO COMPLETO.

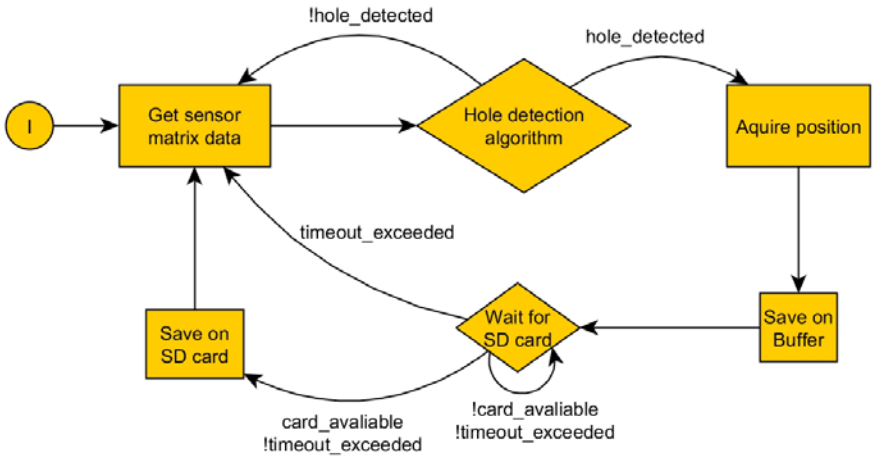
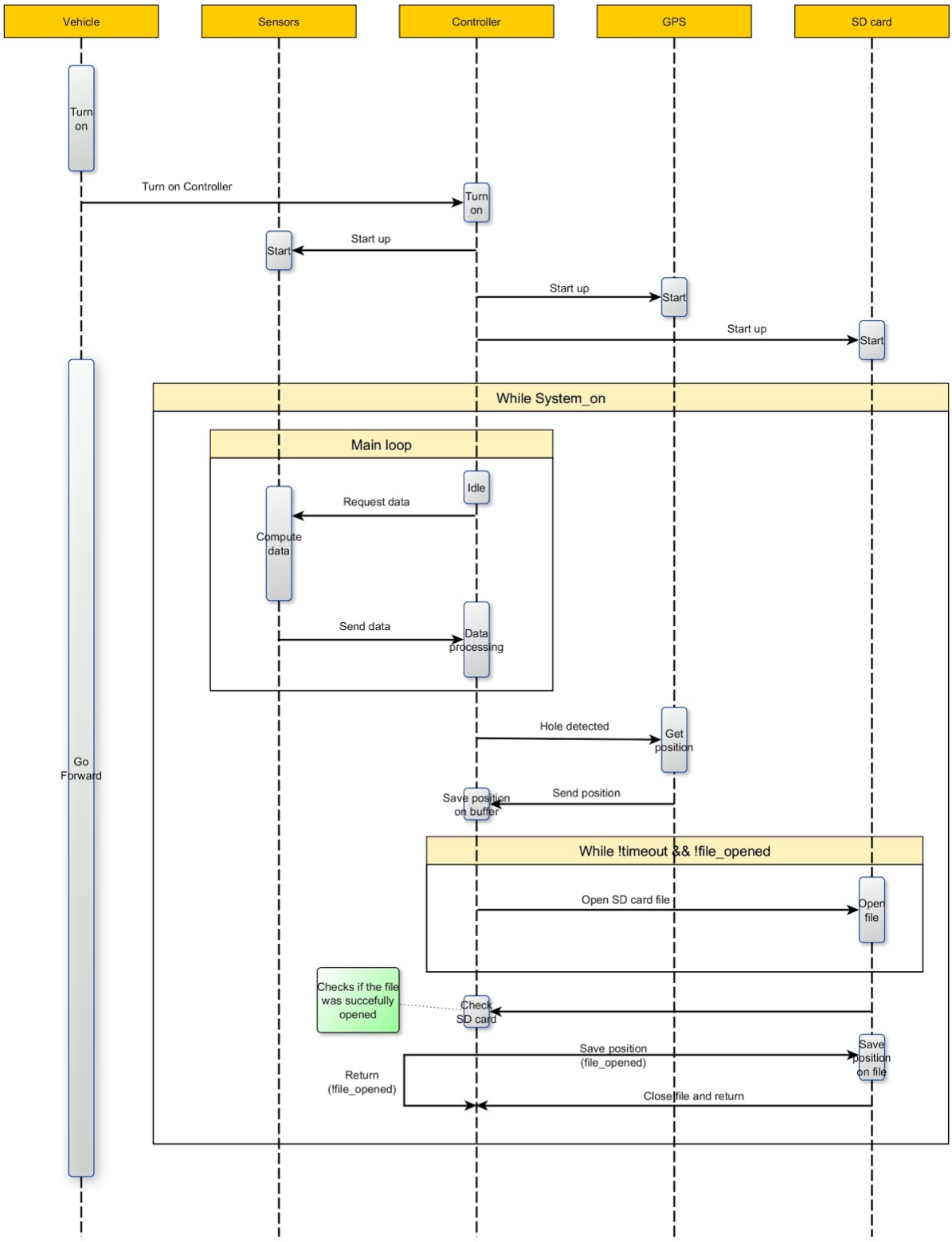


FIGURA 4 - DIAGRAMA DE SEQUÊNCIA DO PROJETO COMPLETO.



2.3 REQUISITOS DO TRABALHO DESENVOLVIDO.

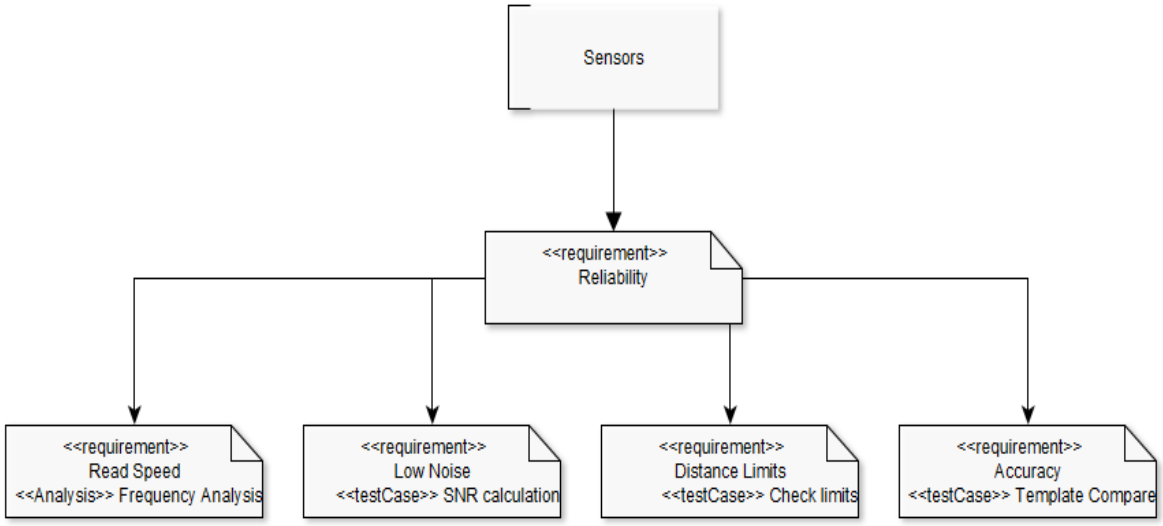
Seguindo o mesmo fluxo de apresentação do projeto completo, o primeiro diagrama apresentado é o diagrama de requisitos do projeto, que pode ser visualizado na Figura 5. Este diagrama representa especificação de requisitos do projeto completo voltado ao sensor utilizado neste caso, que é uma matriz de 3 sensores de distância ultrassônicos. Inicialmente a matriz seria de 4 sensores, porém um deles foi eliminado pois seu requisito de distância mínima do chão não foi obedecida.

Assim, como é possível observar, a frequência de disponibilidade dos dados pelos sensores deveria identificar buracos de, no mínimo, 30 centímetros de comprimento por 25 centímetros de largura. Essas dimensões são buracos nos quais rodas mais comuns já podem ser danificadas. Então, dado que foi estimada uma velocidade máxima de 72km/h, o tempo máximo para ler corretamente seria de 0.3m/(20m/s * 2) que equivale a 7.5ms, ou seja uma frequência mínima de 134Hz, já respeitando os critérios de Nyquist. Também sabemos que essa é a frequência em que todos os sensores devem concluir a leitura, logo cada sensor deve fornecer uma saída estável à uma frequência de, no mínimo 402Hz.

Outro requisito é o de baixo ruído, que, como dito antes, deve ser solucionado após a aplicação de filtros. Nesse projeto foram utilizados filtros de média móvel, filtros FIR passa baixas e um filtro de Kalman, que serão melhor explicados posteriormente.

Por fim temos o requisito de precisão, que no caso dos sensores escolhidos é de, aproximadamente 0.3cm, e no máximo de 0.5cm, o que é bastante aceitável para essa aplicação, que pretende identificar profundidades de, no mínimo 5cm.

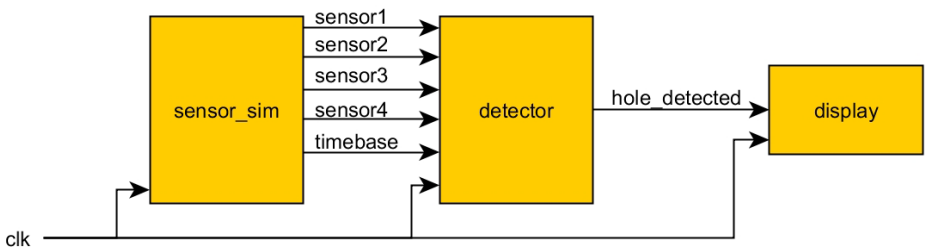
FIGURA 5 - DIAGRAMA DE ESTADOS DO PROJETO COMPLETO.



2.4 ESPECIFICAÇÕES DO TRABALHO DESENVOLVIDO.

A partir de agora serão apresentadas as especificações desse trabalho. Conforme a ordem que foi apresentado o projeto completo, o primeiro diagrama apresentado é o de blocos. A figura 6 apresenta esse diagrama e ilustra os módulos utilizados para o teste. O bloco sensor_sim irá simular a aquisição dos dados, provenientes de um arquivo de texto previamente preenchido com dados reais dos sensores. Assim, a cada ciclo de clock, o bloco sensor_sim transmite um dado de cada sensor e uma base de tempo. Então esse dado é processado pelo algoritmo contido no módulo detector e então o display recebe e apresenta ao usuário se aquela determinada amostra representa ou não um buraco.

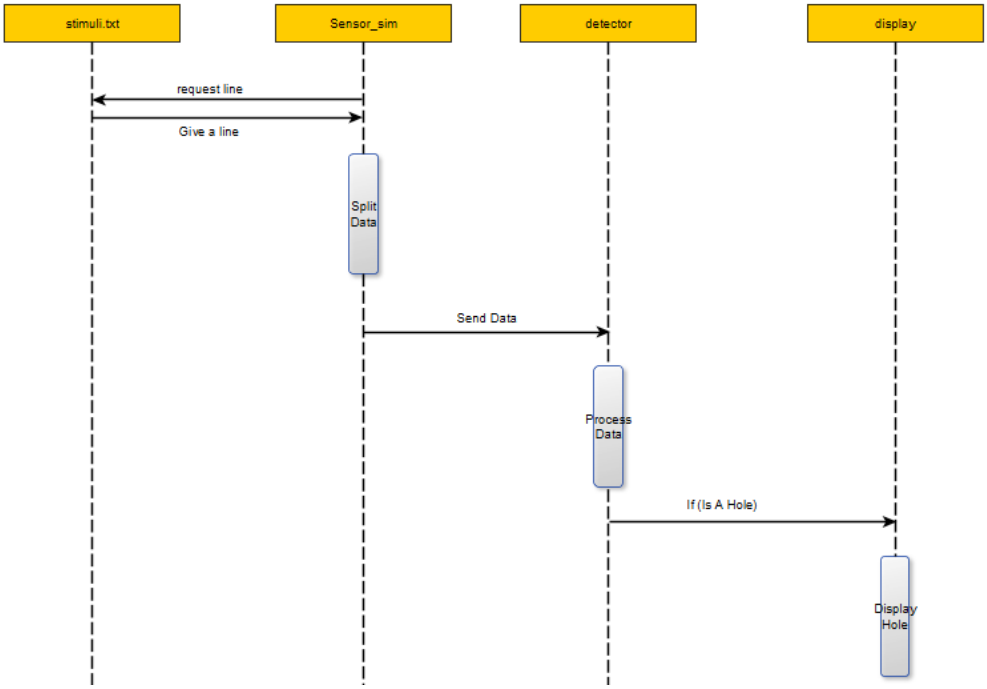
FIGURA 6 - DIAGRAMA DE BLOCOS DO TRABALHO DESENVOLVIDO.



Feito isso, na figura 7 se encontra o diagrama de sequência desse projeto. Nele pode-se verificar que o fluxo desse teste é relativamente mais simples do que o projeto completo. O módulo sensor_sim faz a aquisição de uma linha do arquivo de

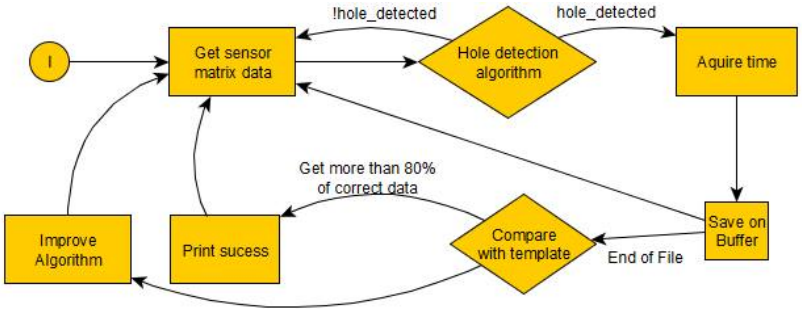
estímulos, separa o dado em vetores de saída e o transmite para o módulo detector, o qual processa o dado e transmite ao display o resultado. Quando um buraco é detectado o display apresenta alguns dados interessantes sobre aquele buraco, como valor das amostras do sensor, tempo da detecção, entre outros fatores que irão servir para verificação posteriormente. Quando o processo termina, também termina o ciclo de clock, então o processo se repete, até o final do arquivo.

FIGURA 7 - DIAGRAMA DE SEQUÊNCIA DO TRABALHO DESENVOLVIDO.



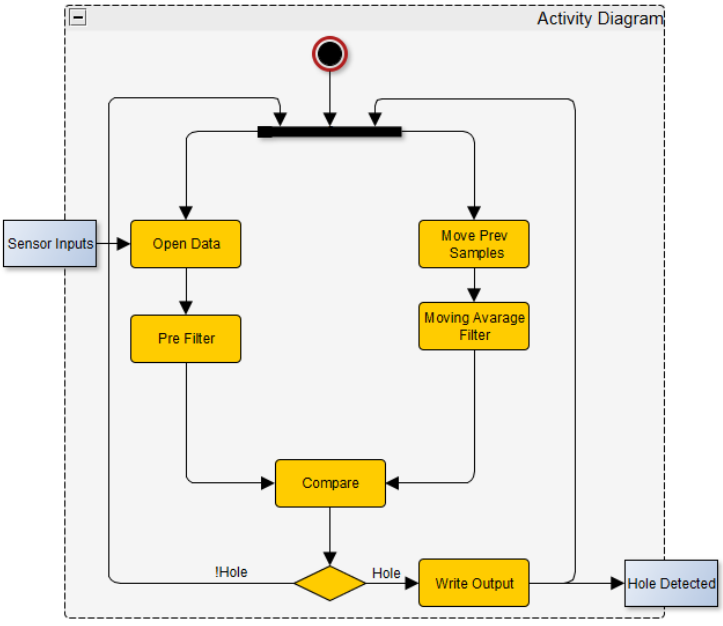
O diagrama de estados é apresentado em sequência, pela figura 8. Nele é possível observar que o fluxo básico dos testes realizados é: adquirir os dados; processá-los por um dos algoritmos desenvolvidos; e, caso algum buraco for detectado, o tempo dessa detecção é salvo. Assim, posteriormente comparamos esses dados com o gabarito e então, caso o requisito de 80% de acerto for obedecido, o sistema conclui por um sucesso do algoritmo. Caso contrário o algoritmo é melhorado, até encontrar a melhor solução. Feito isso o processo se repete com outro conjunto de amostras.

FIGURA 8 - DIAGRAMA DE ESTADOS DO TRABALHO DESENVOLVIDO.



Por fim, ainda foi desenvolvido um diagrama de atividade para o módulo detector, visto que é o único módulo que tem um algoritmo que realmente modifica as entradas para gerar a saída desejada. Esse diagrama pode ser visualizado na figura 9. Nele fica mais fácil entender o procedimento básico do algoritmo de detecção, dado que os dados de entrada são recebidos, passam por um pré filtro, que será melhor explicado futuramente, e então é comparado com uma média móvel realizada com as amostras anteriores do mesmo sensor. A partir dessa comparação um buraco é identificado ou não, e isso é transmitido para a saída.

FIGURA 9 - DIAGRAMA DE ATIVIDADE DO MÓDULO DETECTOR.



3 AQUISIÇÃO DOS DADOS

Esse trabalho foi desenvolvido basicamente a partir de simulações, porém para essas simulações foram utilizados dados reais adquiridos de sensores instalados em um Volkswagen Up. Assim, esse capítulo é dedicado a explicação de como foi feita a instalação dos sensores e a metodologia seguida para a aquisição dos dados.

Iniciando pela instalação dos sensores, um mecânico foi questionado sobre o melhor local para instalação e assim o local de instalação foi escolhido. Todos os carros possuem alguns buracos com uma vedação removível, que é utilizado para remover a água do carro em caso de uma possível inundação ou entrada excessiva de água, a figura 10 apresenta a localização dos sensores no veículo utilizado para os testes.

Com isso foi projetado um invólucro que encaixasse exatamente nesses buracos e que tivesse espaço para que os sensores ficassem seguros, fixos e resistentes a água. O software utilizado para modelar esse invólucro foi o SolidWorks. Após isso, o modelo foi impresso em uma impressora 3D, este pode ser visualizado na figura 11. Após alguns ajustes finos, os sensores foram instalados no carro, como já apresentava a figura 10 e, com uma visão da parte inferior do veículo na figura 12. Isso possibilitou uma aquisição de dados reais muito mais fidedigna e precisa para alcançar o objetivo do trabalho, que é verificar se esses sensores podem ser utilizados nessa aplicação.

FIGURA 10 - LOCALIZAÇÃO DOS BURACOS PARA ESCOAMENTO DE ÁGUA.

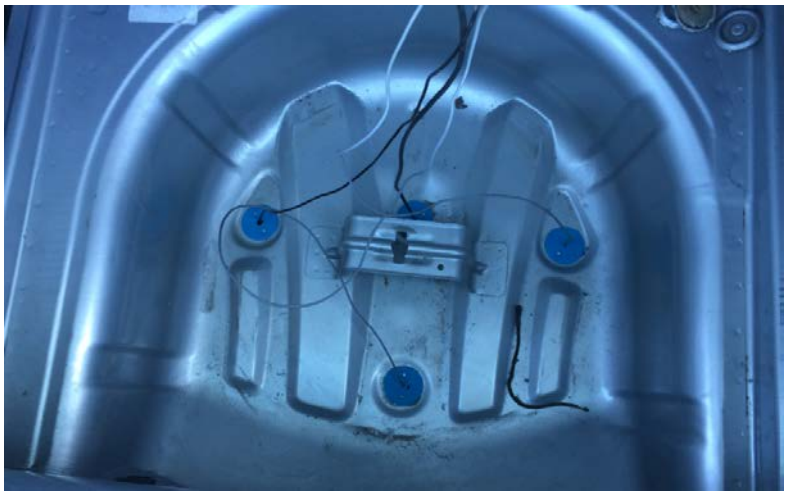


FIGURA 11 - INVÓLUCRO DESENVOLVIDO PARA O SENSOR.

FIGURA 12 - SENSORES INSTALADOS NO CARRO (VISÃO INFERIOR).

Assim, como indicado na proposta do trabalho, a aquisição dos dados foi feita utilizando um arduino e um cartão SD conectados aos sensores. O código utilizado para a aquisição está disponível no git e utiliza as bibliotecas para cartão SD, sensor ultrassônico e utiliza a função millis para criar uma base de tempo. O programa armazena 10 amostras de cada sensor, juntamente com 10 amostras de tempo e então armazena isso no cartão SD em cada linha de um arquivo de texto.

A metodologia seguida foi que em cada arquivo criado com dados adquiridos dos sensores, também foi gerada uma base de dados utilizando um cronômetro e marcando exatamente o tempo em que o carro passava por cima de um buraco. Essa base de dados posteriormente será utilizada para ser comparado com os buracos e tempos de detecção do algoritmo, sendo essa a verificação do sistema simulado.

4 MODELAGEM DA ARQUITETURA

Após obter dados com confiabilidade razoável, foi modelada a arquitetura dos testes, que, como já foi apresentado nos diagramas, contempla três módulos descritos em SystemC que são interligados conforme a figura 6. O bloco sensor_sim faz aquisição dos dados contidos nos arquivos de texto adquiridos a partir de dados reais dos sensores. Esse bloco possui 5 saídas, uma para cada sensor utilizado e 1 para a base de tempo, lembrando que um dos sensores não foi utilizado, logo os dados do Sensor 4 são descartados.

O bloco detector contempla os algoritmos que foram desenvolvidos ao longo do trabalho, que foram constantemente melhorados a fim de aumentar a precisão da detecção. Os algoritmos serão melhor explicados no próximo capítulo. Esse módulo recebe os dados do bloco sensor_sim e tem como saída apenas uma flag, que indica se foi ou não detectado um buraco.

O bloco display simplesmente monitora a flag enviada pelo bloco detector e quando essa se encontra ativa ele disponibiliza as informações do buraco, como tempo e distância do carro até o chão.

Esses módulos são executados de forma síncrona, por um sinal de clock que não necessariamente representa o real tempo de execução em um controlador real, este tempo pode ser observado pelas amostras do sinal timebase. Ao longo da execução, um arquivo vcd é criado, sendo que ele monitora os dados transmitidos pelos sensores, a base de tempo, o clock e a flag de detecção de buracos. Isso facilita bastante a visualização de qual o tempo de detecção de cada buraco.

5 DESENVOLVIMENTO DOS MÓDULOS

Este capítulo será dividido em 3 subtópicos, um para cada módulo, e explicará razoavelmente o que foi feito para obter os resultados obtidos a partir do trabalho. Para mais detalhes o leitor poderá acessar os arquivos do trabalho, que estão bastante detalhados e comentados, além de ser possível acessar a documentação gerada pelo DoxyGen, que dá uma explicação mais visual do programa.

5.1 SENSOR_SIM.

O módulo sensor_sim é, como já explicado anteriormente, o módulo que abre o arquivo de texto que contém a informação da coleta de dados e separa cada amostra dos sensores e transmite para o módulo detector.

Para isso esse módulo utiliza as bibliotecas de C++ fstream e vector. Com elas cada linha do arquivo texto é armazenada em um vetor e, a cada sinal de clock, é coletado um conjunto de dados desse vetor e esses dados são transmitidos para a saída. Quando o vetor é enviado completamente uma nova linha é adquirida e passa pelo mesmo processo, até o final do arquivo. A biblioteca fstream permite a abertura dos arquivos de texto e dá acesso a funções para adquirir linhas subsequentes, assim como encontrar o final do arquivo. A biblioteca vector é utilizada para armazenar os dados de forma dinâmica, o que facilita bastante o trabalho do desenvolvedor. Ainda são utilizadas bibliotecas padrão para entrada e saída e para identificação de caracteres, utilizadas para identificar os delimitadores de cada amostra.

5.2 DETECTOR.

O módulo detector é a principal área de trabalho desse projeto, nele foram implementados os algoritmos de detecção de buracos, cada vez mais aprimorados. Os primeiros algoritmos utilizam procedimentos computacionais básicos para identificar variações, como limites fixos, o que já esperávamos que não iria trazer bons resultados, mas foi bastante interessante para fazer uma primeira integração com o módulo de estímulos.

A complexidade dos algoritmos aumentou conforme filtros de media móvel, passa baixas FIR foram adicionados, o que ajudou a eliminar erros de calibraçao do sensor, desniveis na pista e vibrações no carro, entre outros fatores.

Visto que isso não foi suficiente para eliminar amostras em que o sensor obteve leituras totalmente fora do padrão, por efeitos de reflexão da onda sonora ou, até mesmo a não captação da mesma, foi aplicada uma saturação para valores que se encontravam fora do padrão, o que é feito bastante no projeto de filtros digitais quando a aquisição de dados sofre muito com o ambiente externo. Essa saturação eliminou amostras indesejadas que contribuem apenas para desestruturar a detecção e os filtros de média móvel ainda aplicados.

Também tentou-se aplicar um filtro FIR, porém sem muita metodologia de projeto, o que gerou resultados não muito satisfatórios. Por isso foi decidido utilizar mais critérios no desenvolvimento de um filtro mais interessante, passando a coleta de dados para o MATLAB e projetando um filtro.

Após o estudo de algumas topologias e da simulação das mesmas, foi verificado que o filtro que atendeu melhor os requisitos de enfatizar as posições de buracos e atenuar os picos desnecessários, resultando assim em um sinal muito mais limpo e confiável, foi o filtro de Kalman, seguindo os passos apresentados em [3], seguido de uma saturação. Ainda foi muito interessante utilizar o MATLAB para ter uma noção melhor de como o sinal se comportava. Lembrando ainda que após o filtro apresentado o sinal ainda passa por uma saturação de picos extremos. Também, para o leitor que tem interesse em entender melhor cada filtro desenvolvido, na documentação gerada em forma Doxy e até mesmo nos códigos do projeto é possível encontrar essas informações.

Os algoritmos são abordados mais afundo posteriormente.

5.3 DISPLAY.

O módulo display meramente recebe a variável booleana hole_detected (provinda do módulo detector) e mostra ela, de forma que vale 0 quando não detectou um buraco, e 1 quando detectou. Este módulo foi implementado pensando-se em usar futuramente a variável booleana que é mostrada para requerer a posição do GPS a fim de se gravá-la no cartão SD.

6 INTEGRAÇÃO (SIMULAÇÃO)

Os módulos sensor_sim, detector e display foram integrados na função main.cpp. Essa integração foi feita descrevendo-se os sinais dos sensores (sensor1 a sensor4), a base de tempo (timebase), a flag de detecção dos buracos (hole_detected), e o clock (clk). Os blocos foram interligados como se mostrou na figura 6. Todos os sinais são registrados em um arquivo ‘wave.vcd’ para visualização das formas de onda.

Foram testados 6 algoritmos: kalman_filter; first_algorithm; p1_algorithm; l1_algorithm; l2_algorithm; p2_algorithm. Cada um aborda a detecção de buracos e as filtragens de maneira diferente, de modo a se testar possíveis alternativas para o sistema.

A metodologia para realização dos testes utilizou, inicialmente o primeiro arquivo de dados, nomeado de Teste1.txt, o qual já apresentou um resultado muito interessante e possibilitou a criação dos quatro algoritmos iniciais.

Após isso foi realizada mais uma coleta de dados, dessa vez mais metódica, avaliando um mesmo circuito a diferentes velocidades, gerando os arquivos: 10kmph.txt, 20kmph.txt, 30kmph.txt, 40kmph.txt, 10kmphb.txt. Desses arquivos, o 30kmph.txt foi escolhido para fazer uma pré seleção dos dois melhores filtros, por ser a velocidade intermediária entre o mínimo e máximo dos requisitos do projeto. Assim, após escolhidos os dois melhores filtros, todos os arquivos foram simulados nesses dois filtros, avaliando se o sensor é aplicável ou não para este sistema.

7 VALIDAÇÃO FINAL DO SISTEMA

Após executar as simulações foram obtidos os resultados apresentados na Tabela 1, onde foi verificado que ou o sensor ou os algoritmos desenvolvidos ainda não obedecem ao requisito estipulado de obter 80% de acerto na detecção dos buracos. Isso acontece principalmente pelo fato de que acontecem muitos falsos positivos, o que provavelmente pode ser melhorado trabalhando melhor os algoritmos de detecção. Outro ponto interessante de se observar foi que os erros, falsos positivos, geralmente acontecem próximos aos buracos, geralmente entre uma faixa de mais ou menos 5 segundos.

Na tabela 1, ‘V’ representa o número de acertos, ‘X’ representa o número de erros e ‘T’ representa o número total de buracos indicados no gabarito, os algoritmos selecionados para o teste completo foram p1_algorithm e p2_algorithm, visto melhor desempenho para30km/h. Isso pode ser associado a esses algoritmos se basearem em filtros passa altas, onde existe uma marcação de pontos para cada detecção fora do padrão e quando um limite é atingido, um buraco é marcado, o que faz sentido quando se lembra que variações bruscas se devem a altas frequências.

TABELA 1 - RESULTADOS OBTIDOS A PARTIR DA EXECUÇÃO DAS SIMULAÇÕES.

Análise de acertos	Algoritmo	V/(X+T)	V/T
30kmph	**kalman_filter**	38,10%	66,67%
	first_algorithm	36,36%	66,67%
	p1_algorithm	71,43%	83,33%
	l1_algorithm	33,33%	50,00%
	l2_algorithm	29,03%	75,00%
	p2_algorithm	45,00%	75,00%
10kmph	**p1_algorithm**	27,27%	54,55%
	p2_algorithm	17,65%	54,55%
10kmphb	**p1_algorithm**	30,43%	46,67%
	p2_algorithm	32,00%	53,33%
20kmph	**p1_algorithm**	29,17%	50,00%
	p2_algorithm	32,14%	50,00%

40kmph	**p1_algorithm**	50,00%	66,67%
	p2_algorithm	30,43%	66,67%

A tabela 2 apresenta os dados do gabarito que foram coletados para se comparar com as simulações. Os tempos estão no formato “minutos:segundos.milissegundos”.

TABELA 2 - GABARITO MODELADO PARA OS ARQUIVOS DE TESTE.

Gabaritos					
Teste1.txt	**10kmph.txt**	**20kmph.txt**	**30kmph.txt**	**40kmph.txt**	**10kmphb.txt**
00:57.460	00:19.350	00:09.330	00:09.100	00:07.150	00:09.310
01:52.810	00:38.850	00:12.310	00:09.230	00:07.910	00:10.700
01:56.520	00:41.370	00:27.480	00:21.090	00:15.850	00:28.750
03:36.990	00:51.330	00:28.400	00:24.600	00:16.050	00:32.930
03:45.640	00:54.520	00:29.610	00:35.290	00:19.170	00:34.870
03:47.470	01:08.460	00:33.660	01:11.970	00:22.910	00:37.280
03:52.650	02:00.380	00:37.740	01:14.930	00:28.920	00:51.470
04:03.820	02:06.230	00:39.640	01:16.140	00:57.430	01:34.940
05:22.650	02:11.060	00:51.210	01:22.830	01:02.830	01:41.960
05:32.820	02:31.450	01:29.290	01:25.050	01:13.680	01:46.420
	02:53.770	01:34.190	01:34.920	01:14.180	01:47.870
		01:36.670	01:41.900	01:20.580	01:51.790
		01:52.560			02:27.520
		02:13.580			02:30.110
					02:37.270

8 CONCLUSÃO

Foi possível a partir desse trabalho determinar características importantes sobre os sensores de distância ultrassônicos. Uma delas é a facilidade em obter um erro devido a reflexão da onda sonora em pistas não pavimentadas, pela grande irregularidade da pista. Em dias de chuva buracos preenchidos por água não são identificados. Outra situação (já esperada), foi a variação relativa a vibração e balanço do automóvel, inclusive acusando detectar buracos em curvas acentuadas. Tendo isso em mente, pode-se tomar dois caminhos a fim de chegar no resultado esperado: adquirir sensores melhores ou melhorar os algoritmos de detecção.

Outro ponto importante desse trabalho foi o aprendizado adquirido de ferramentas de C++, além do aprimoramento das técnicas utilizando SystemC. Foram estudadas bibliotecas para extrair informações de arquivos e utilizá-las para realizar diferentes processamentos. Também foi aplicada a biblioteca vector, que é uma ferramenta importantíssima quando se trabalha com matrizes. Outro ponto importante estudado foi o projeto de filtros, relembrando alguns filtros digitais como FIR, média móvel, IIR, além de estudar uma nova topologia de filtro, o filtro de Kalman, muito utilizado para eliminar ruído na leitura de sensores.

O conteúdo das aulas também foi muito aplicado e aprimorado, visto que o projeto foi um sistema utilizando os conceitos de UML e se reforçando com diagramas SysML. Esse estudo foi de extrema importância para levantar características antes não pensadas sobre o projeto, além de dar um fluxo de projeto muito mais fácil. Além disso, foram utilizados conceitos das aulas de C++ e SystemC para o desenvolvimento dos módulos. Também foi utilizado o gtkwave para conferir se o sistema estava realizando o que deveria, utilizando os diagramas, os gabaritos gerados e os resultados fazer a verificação.

Por fim, pode-se dizer que ainda há muito trabalho a ser feito para ter um produto confiável. Outros tipos de sensores podem ser avaliados, porém esse trabalho já possibilitou uma plataforma de testes rápida e simples para que próximos testes possam ser realizados para essa mesma aplicação. Isso porque a única modificação necessária seria no arquivo detector.cpp, onde um novo algoritmo pode ser incluído e testado. Também foi possível verificar que é um projeto que pode ser concretizado com mais tempo de desenvolvimento, e que foi um aprendizado incrível.

REFERÊNCIAS

[1] PEIXOTO, L. S. et al. Acidentes decorrentes de vias públicas urbanas danificadas: a responsabilidade civil do município. In: **Âmbito Jurídico**, Rio Grande, XV, n. 101, jun 2012. Disponível em: <http://www.ambito-juridico.com.br/site/?n_link=revista_artigos_leitura&artigo_id=11884>. Acesso em nov 2016.

[2] POR VIAS SEGURAS. Estatísticas nacionais de acidentes de trânsito. In: **Por Vias Seguras,** mai 2016. Disponível em: <http://www.vias-seguras.com/os_acidentes/estatisticas/estatisticas_nacionais>. Acesso em nov 2016.

[3] INTERACTIVE MATTER LAB. Filtering Sensor Data with a Kalman Filter. In: **Interactive Matter Lab,** dec 2009. Disponível em: <http://interactive-matter.eu/blog/2009/12/18/filtering-sensor-data-with-a-kalman-filter/>. Acesso em nov 2016.