Miguel Filipe Batista Prego

Licenciado em Engenharia Eletrotécnica e de Computadores

Detecção e registo automático de irregularidades no asfalto

Relatório intermédio para obtenção do Grau de Mestre em

Engenharia Eletrotécnica e de Computadores

Orientador: José Manuel Matos Ribeiro da Fonseca, Professor Associado, Faculdade de Ciências e Tecnologia da Universidade Nova de Lisboa

FACULDADE DE CIÊNCIAS E TECNOLOGIA
UNIVERSIDADE NOVA DE LISBOA

Setembro, 2016

Índice

Lista de Figuras

Lista de Tabelas

Capítulo 1

Trabalho desenvolvido

A solução proposta para a resolução do problema apresentado em XXX é um sistema separado em três fases, cada uma delas ligada a um elemento físico: o Arduino, o telemóvel e a base de dados (alocada num computador). Assim surgem as seguintes secções, referentes às três fases do sistema.

1.1 Funcionamendo do Arduino

Para que o código Arduino possa ser executado, este deve estar separado em três grupos chave: **uma zona de declarações** e **duas funções**, sendo que as duas funções têm os nomes de *setup* e *loop*.

- Na **zona de declarações** devem ser declaradas as bibliotecas que vão ser utilizadas durante o código. Estas bibliotecas são conjuntos de funções previamente definidas por um utilizador, que são habitualmente utilizadas e portanto são agrupadas num ficheiro para que outros as possam executar sempre que necessário. Também na zona de inicialização são declaradas variáveis globais, as quais conseguem armazenar um valor durante as várias execuções da função *loop* que será descrita mais à frente.

- Tal como o nome indica, a **função *setup*** é a função de preparação e inicialização do código. É aqui que se devem fazer declarações elementos físicos que serão utilizados como *leds* ou botões ou executar funções que apenas devem correr uma vez, por exemplo, para activar sistemas ou dispositivos.

- A **função *loop*** é uma função que é executada enquanto o Arduino estiver ligado e onde devem ser chamadas as funções centrais do código. No caso de existirem variáveis criadas dentro desta função, elas são inicializadas sempre que a função

chega ao fim, pelo que é benéfico que existam algumas variáveis globais no sistema de modo a acompanhar a evolução do mesmo e transportar informação entre os vários ciclos de execução da função *loop*

No caso específico desta dissertação a utilização dos três grupos chave é a seguinte:

- Na zona de declarações são incluídas as bibliotecas referentes ao acelerómetro, ao cartão de memória e ao GPS. São também definidos os modos de escrita no cartão de memória, os portos de ligação de comunicação do GPS e criadas variáveis globais referentes ao GPS e ao acelerómetro. Além disso, são criadas variáveis globais de controlo do código, variáveis de passagem de parâmetros entre vários ciclos da função *loop* e declaração de constantes que serão utilizadas diversas vezes ao longo do código, facilitando a alteração do seu valor, caso tal se mostre necessário.

- Na função *setup* são declaradas as bandas de comunicação entre o Arduino e o telemóvel e entre o GPS e o Arduino. São também inicializados dois *leds* como elementos de saída de informação e um botão como entrada de informação e é atribuído o estado de "desligado" a ambos os *leds*. Por fim, são executadas as funções de inicialização do leitor do cartão de memória e do acelerómetro, SD_init e AccelInit respectivamente, ambas explicadas mais á frente.

- A função *loop*, embora executada de uma forma constante, contém um código bastante simples mas deveras preciso. A função começa por verificar se existe informação disponível para leitura no módulo Bluetooth. Caso tal se verifique, essa informação é lida e comparada, para que seja possível determinar se a ligação ao módulo Bluetooth do Arduino sofreu alguma alteração, nomeadamente, quanto à ligação com a aplicação desenvolvida para esta dissertação. Na hipótese desta alteração se verificar, é avaliada se a ligação foi estabelecida ou cortada, sendo que no caso de ser estabelecida, é chamada a função ReadSD para leitura do cartão de memória (explicada mais à frente) e um *led* de controlo é aceso. Caso contrário, o mesmo *led* de controlo é desligado. Se não se verificar nenhuma alteração na ligação, o *led* mantém o estado em que se encontra. De seguida são lidos os valores de aceleração dos eixos X, Y e Z, provenientes do acelerómetro, armazenados e comparados com a leitura anterior para verificação da existência de alguma irregularidade no asfalto. A maneira como esta comparação será explicada mais à frente. Caso seja detectada alguma irregularidade, um dos *leds* de controlo pisca e são armazenados, numa variável **S**, os valores do nível da irregularidade, bem como o grupo data-hora proveniente do GPS. Seguidamente é verificada a ligação à aplicação e caso esta exista, os valores de **S** são enviados directamente para o telemóvel. Se a ligação não existir, os valores de **S** são transferidos para o cartão de memória para que possam ser transmitidos para o telemóvel assim que exista uma ligação estável com o mesmo. Por fim, os valores actuais do acelerómetro são armazenados para que possam ser comparados na próxima execução da função *loop*.

1.2 Funções

1.2.1 SD_init

Esta função serve para inicializar o leitor de cartões SD. Nela é chamada a função *begin* da biblioteca *SD* e verificada a resposta dessa mesma função. Caso a resposta seja o valor 10, um *led* de controlo pisca para que o utilizador saiba que a inicialização ocorreu com sucesso e é verificada a existência de um ficheiro com informação previamente armazenada sobre irregularidades detetadas anteriormente. Caso exista esse ficheiro, este é aberto de modo a que se possam juntar novos valores de irregularidades detetadas, caso contrário, o ficheiro é criado para futuros armazenamentos.

1.2.2 WriteSD

1.2.3 ReadSD

1.2.4 AccelInit

Nesta função o acelerómetro é ligado e são determinados os valores de sensibilidade do mesmo. É também feita uma inicialização de valores para queda livre e batida que não são utilizados para esta dissertação. Esta funcionalidade foi mantida para possíveis aplicações futuras e uma deteção mais pormenorizada das irregularidades. Seguidamente, um *led* de controlo pisca para o utilizador saber que a inicialização foi bem sucedida.

Bibliografia

Chan, C. K., Y. Gao, Z. Zhang e N. Dahnoun (2014). "Implementation and evaluation of a pothole detection system on TI C6678 digital signal processor". Em: *EDERC 2014 - Proceedings of the 6th European Embedded Design in Education and Research Conference*.

Chen, K., M. Lu, X. Fan, M. Wei e J. Wu (2011). "Road condition monitoring using on-board three-axis accelerometer and GPS sensor". Em: *Proceedings of the 2011 6th International ICST Conference on Communications and Networking in China, CHINACOM 2011*.

Fouad, M. M., M. A. Mahmood, H. Mahmoud, A. Mohamed e A. E. Hassanien (2014). "Intelligent road surface quality evaluation using rough mereology". Em: *2014 14th International Conference on Hybrid Intelligent Systems*.

He, Y., J. Wang, H. Qiu, W. Zhang e J. Xie (2011). "A research of pavement potholes detection based on three-dimensional projection transformation". Em: *Proceedings - 4th International Congress on Image and Signal Processing, CISP 2011*.

Hegde, S., H. Mekali e G. Varaprasad (2015). "Pothole detection and inter vehicular communication". Em: *2014 IEEE International Conference on Vehicular Electronics and Safety, ICVES 2014*.

Jang, J., A. W. Smyth, Y. Yang e D. Cavalcanti (2015). "Road surface condition monitoring via multiple sensor-equipped vehicles". Em: *2015 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*.

Kattan, A. e M. F. Aboalmaaly (2014). "A smartphone-cloud application as an aid for street safety inventory". Em: *Proceedings of the 11th International Conference on Electronics, Computer and Computation, ICECCO 2014*.

Madli, R., S. Hebbar, P. Pattar e V. Golla (2015). "Automatic Detection and Notification of Potholes and Humps on Roads to Aid Drivers". Em: *IEEE Sensors Journal*.

Mednis, A., G. Strazdins, R. Zviedris, G. Kanonirs e L. Selavo (2011). "Real time pothole detection using Android smartphones with accelerometers". Em: *2011 International Conference on Distributed Computing in Sensor Systems and Workshops, DCOSS'11*.

Moazzam, I., K. Kamal, S. Mathavan, S. Usman e M. Rahman (2013). "Metrology and visualization of potholes using the microsoft kinect sensor". Em: *IEEE Conference on Intelligent Transportation Systems, Proceedings, ITSC*.

Yu, X. e E. Salari (2011). "Pavement pothole detection and severity measurement using laser imaging". Em: *IEEE International Conference on Electro Information Technology*.

Zhang, Z., X. Ai, C. K. Chan e N. Dahnoun (2014). "An efficient algorithm for pothole detection using stereo vision". Em: *IEEE International Conference on Acoustic, Speech and Signal Processing (ICASSP)*.