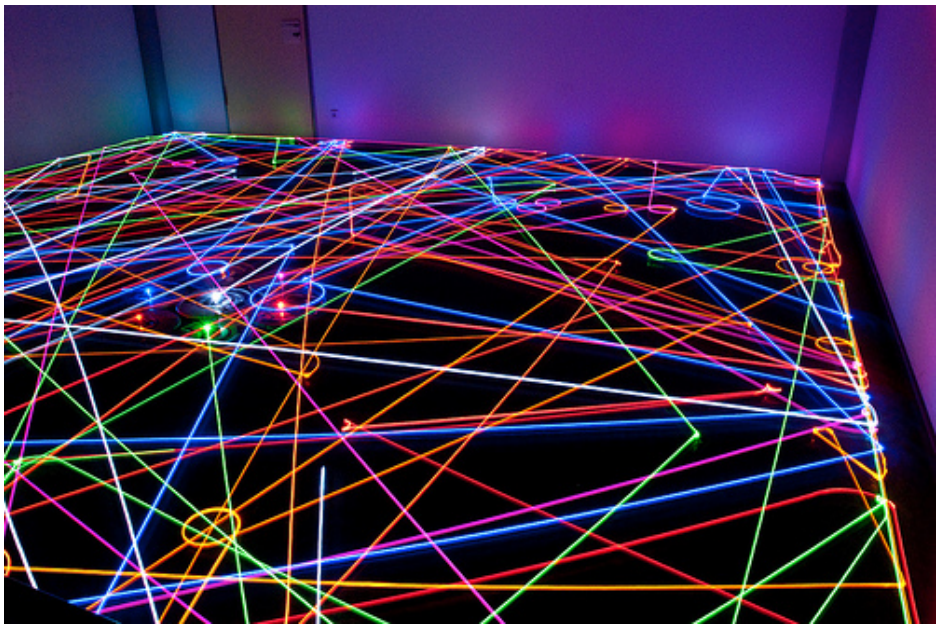
Programação 2014/2015

Mestrado em Engenharia Electrotécnica e de Computadores (MEEC)

Robot de Limpeza - iClean



Projeto de Programação – Entrega Intermédia

1 Introdução

O objetivo deste projeto é desenvolver uma aplicação que simula o funcionamento de um robot de limpeza intitulado como iClean. De certa forma, este programa simula o funcionamento de robots de limpeza que atualmente existem no mercado (e.g. roomba). O iClean deve-se movimentar num piso, limpando a área por onde passa. O objetivo dessa simulação é estimar quanto tempo um robot demora a limpar a área associada a uma divisão, assumindo que não existem objetos (móveis, mesas, etc).

Este programa deve permitir a geração de um mapa da divisão, simular o processo de limpeza por parte do robot, marcando os espaços já limpos, bem como a gravação de um ficheiro que indique o nome do robot, algumas estatísticas da simulação da limpeza e o caminho percorrido.

2 Funcionamento

Uma divisão é modelada através de uma grelha fixa de quadrados. Quando o robot iClean passa por um elemento da grelha (quadrado), este deve ser marcado como limpo. O iClean deve começar numa posição arbitrária da divisão e com uma direção aleatória de movimento.

Na Figura 1 é ilustrado o modelo para a divisão a limpar (grelha com um tamanho definido), a posição inicial do iClean e a marcação de cada quadrado da grelha como limpo à medida que a simulação de limpeza vai avançando.

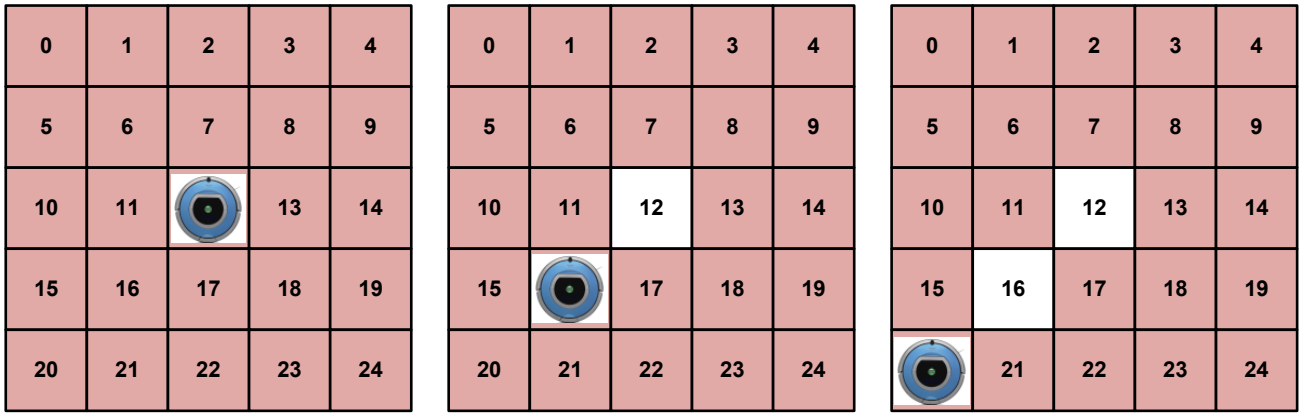


Figure 1: Exemplo de funcionamento do programa: os quadrados mais escuros correspondem à área suja e os quadrados brancos à área limpa.

O funcionamento do programa poderá ser sumarizado da seguinte forma:

1. Leitura do tamanho da divisão em número de quadrados na altura e largura (*stdin*).
2. Leitura do nome do robot de limpeza (*stdin*). Este nome não poderá ter mais do que três letras.
3. Simulação da limpeza da divisão: A simulação poderá ficar mais rápida ou mais lenta a pedido do utilizador (leitura de teclas de setas).
4. O mapa da divisão deverá ser atualizado para indicar os quadrados já limpos e a nova posição do robot.
5. Gravação do caminho percorrido pelo robot, i.e. o número de quadrados limpos e o número de quadrados percorridos (i.e. tempo demorado).

3 Simulação da Limpeza

A divisão é rectangular com uma altura A e largura L, valores pedidos ao utilizador no início do programa. Inicialmente, toda a divisão está suja e um robot não pode atravessar as paredes da divisão. Para guardar o mapa da divisão tem de ser usado um **vector M**, indicando quais os quadrados ocupados e sujos. O robot deve se movimentar de acordo com as seguintes regras:

- Na divisão, cada posição do vector M possui um índice tal como é ilustrado na Figura 1.
- O robot tem uma posição (Xc,Yc) que identifica o quadrado da divisão que ocupa.
- O robot tem uma posição (Xt,Yt) que pretende alcançar.

- Sempre que o robot alcançar a posição (Xt,Yt) deve ser gerada aleatoriamente uma nova posição para a qual se deve deslocar.
    - Deverá impor uma condição na posição gerada ? Experimente !
    - Não use movimentos determinísticos !
- O robot deve continuar a tentar alcançar uma dada posição até que a divisão esteja toda limpa.
- A velocidade da simulação deve poder ser controlado através do teclado (tecla ↑ para aumentar e tecla ↓ para diminuir).
- Quando toda a divisão estiver limpa, o robot não se deve movimentar.
- **Nota: o robot não conhece a altura A e largura L da divisão nem a sua topologia. Apenas pode assumir que conhece aproximadamente a área total da divisão (A*L).**

4 Escrita do Ficheiro

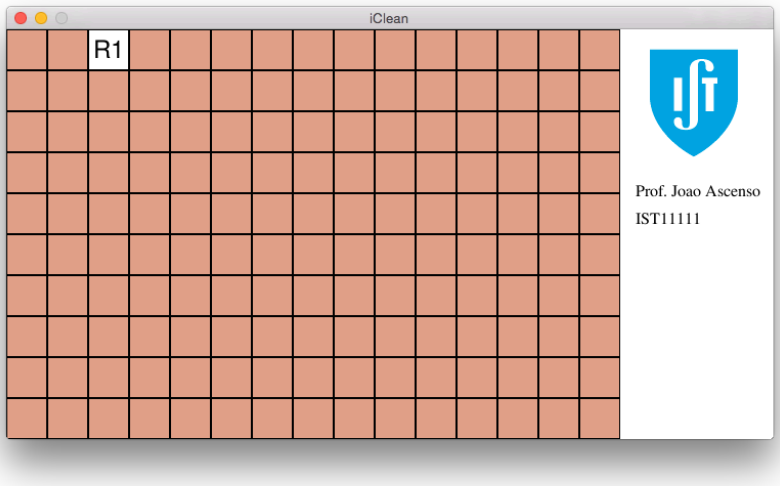
No final de cada simulação (robot parado) é necessário gravar um ficheiro com algumas estatísticas sobre o robot:

- Nome do robot
- Número de quadrados limpos
- Número de quadrados percorridos
- Lista de inteiros com um índice para cada quadrado percorrido pelo robot

Este ficheiro tem um nome predefinido intitulado estatísticas.txt.

5 Aspecto Gráfico

A aplicação iClean deverá ter um aspecto gráfico semelhante ao que está ilustrado na Figura 2. A aplicação deverá também mostrar o nome da aplicação (iClean), autor da aplicação e o nome do robot.



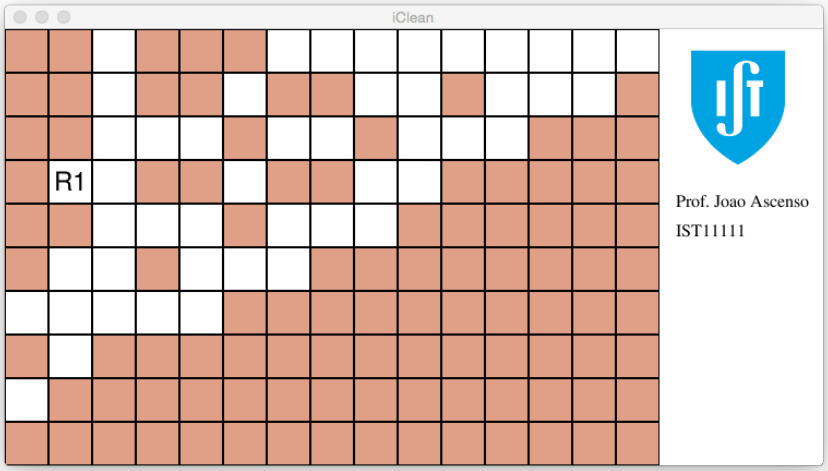


Figure 2: Aplicação iClean para uma divisão com A = 10 e L = 15, com o robot de limpeza na sua posição inicial (cima) após algum tempo de simulação (baixo).

6 Desenvolvimento do Programa iClean

Também é fundamental que os alunos cumpram as regras que se seguem no desenvolvimento do programa iClean.

6.1 Desenvolvimento Faseado

O desenvolvimento deste projeto deverá ser feito de uma forma faseada, devendo os alunos garantir que todas as funcionalidades codificadas até ao momento estão a funcionar corretamente.

É preferível um programa que implementa poucas funcionalidades, mas que funcionam corretamente, do que um programa totalmente desenvolvido mas que não faz nada.

Assim sugerem-se os seguintes passos, pela ordem apresentada, para realização do projeto:

- Leitura do nome do robot e das dimensões do mapa
- Inicialização da biblioteca SDL
- Criação da janela gráfica
- Movimentação do robot de limpeza segundo as regras indicadas
- Atualização do ecrã
- Reconhecimento das teclas para aumentar e diminuir a velocidade de simulação
- Escrita do ficheiro com o resultado da simulação

Os alunos deverão garantir a robustez da aplicação verificando todos os casos de erro (por exemplo quando o ficheiro com o mapa da divisão não existir).

6.2 Documentação

O código produzido pelos alunos deverá ser comentado. Os comentários presentes no código deverão explicitar e explicar o funcionamento da aplicação assim como as decisões tomadas. As seguintes regras devem ser cumpridas:

- O código deve ser comentado sempre que realize alguma operação não óbvia.
- Os comentários devem ser claros, gramaticalmente corretos e usando frases simples.
- A declaração de todas as variáveis e constantes deve ser acompanhada de um comentário com uma breve descrição de para que servem.
- Cada bloco de código (seleção ou repetição) deve ser precedido de um breve comentário explicativo.
- Todos os programas devem ter um comentário inicial que identifique, no mínimo, o trabalho, o seu autor, o fim a que se destina e a data de realização.

6.3 Indentação

Um ponto fundamental na organização de escrita de código é a indentação, isto é, organização hierárquica das linhas de código, de acordo com âmbito onde elas se encontram. A indentação deixa o código fonte do programa mais organizado, mais legível, fácil de entender e de modificar, sendo uma parte essencial do trabalho.

6.4 Estrutura do Código

Todos os programas em C devem possuir a mesma estrutura genérica, composta pelas seguintes secções:

- Bloco de comentários
- Diretivas #include
- Constantes globais e variáveis globais (caso sejam necessárias)
- Declaração de funções
- Função main()
- Definição de funções

Como regra geral deve considerar que as funções devem caber num único ecrã, isto é, devem ter no máximo cerca de 30 linhas. Também deve cumprir as seguintes regras:

- Inicialize sempre as variáveis na sua declaração
- Teste a validade dos parâmetros recebidos por uma função
- Declare constantes e nunca use números no seu código
- Evite repetições de código, use funções, ciclos, etc.
- Evite o uso de variáveis globais
- Não use **goto**
- Escreva código simples e claro que um colega seu possa perceber !

6.5 Compilação

O compilador a usar na execução do projeto é o gcc em ambiente Linux. **Os projetos que não compilem, i.e. que tenham erros de sintaxe, não serão avaliados.** A existência de avisos durante a fase de compilação

poderá ser indício da existência de problemas no código. Estes deverão ser eliminados corretamente ou ignorados com cuidado extremo.

6.6 Decisões do Projeto

Como em qualquer projeto de informática, o funcionamento do programa não está totalmente definido no enunciado, existindo algumas ambiguidades e omissões. Para resolver essas omissões os alunos deverão tomar algumas decisões aquando do desenvolvimento do projeto. Estas decisões devem ser fundamentadas, sem nunca ir contra o definido no enunciado.

6.7 Biblioteca SDL

Durante o desenvolvimento deste projeto devera ser usada a biblioteca SDL2. A aplicação deverá ser compilada usando as bibliotecas SDL2, SDL2_image, SDL2_ttf. Mais informação disponível aqui:

- http://wiki.libsdl.org/APIByCategory
- https://wiki.libsdl.org/FrontPage

7 Submissão

Os alunos deverão submeter o código desenvolvido através do sistema FENIX até 6 de Abril. Apenas será necessário entregar o código correspondente ao programa desenvolvido.

Só será aceite um ficheiro de texto com extensão **.c**. Não utilize um processador de texto (e.g. Word) para formatar o seu programa.

8 Plágio

Os trabalhos serão objecto de um sistema de deteção de plágio. Os alunos podem conversar entre si para discutir possíveis soluções para algum problema que tenham mas não podem partilhar código fonte. Nesta entrega intermédia, todo o código deve ser realizado individualmente !

9 Avaliação da Entrega Intermédia

A avaliação do projeto terá em conta diversos parâmetros:

- Funcionalidades implementadas
- Qualidade do código produzido
- Estruturação da aplicação
- Comentários e legibilidade do código
- Tratamento de erros
- Rapidez na limpeza