U.PORTO
FEUP FACULDADE DE ENGENHARIA
UNIVERSIDADE DO PORTO

SPACE JETPACK

Projeto de Laboratório de Computadores

Realizado por:

Turma 2 Grupo 10

Marina Tostões Fernandes Leitão Dias
(up201806787)

Raquel Filipa Sepúlveda Figueiredo
(up201806664)

janeiro 2021

ÍNDICE

1. Instruções de Utilização do Programa

1.1. Menu

Fig. 1 - Menu

Ao iniciar o programa, é apresentado um ecrã de menu com 3 botões, onde ao selecionar cada um com o botão esquerdo do rato, o utilizador é redirecionado para novos ecrãs. Se o utilizador selecionar a opção *Começar*, este será redirecionado para o ecrã do jogo e o jogo iniciará imediatamente. Se o utilizador pressionar a opção *Instruções,* este será redirecionado para uma ecrã com todas as instruções sobre a jogabilidade do jogo. E por fim, se o utilizador selecionar a opção Sair, o jogo termina.

1.2. Instruções

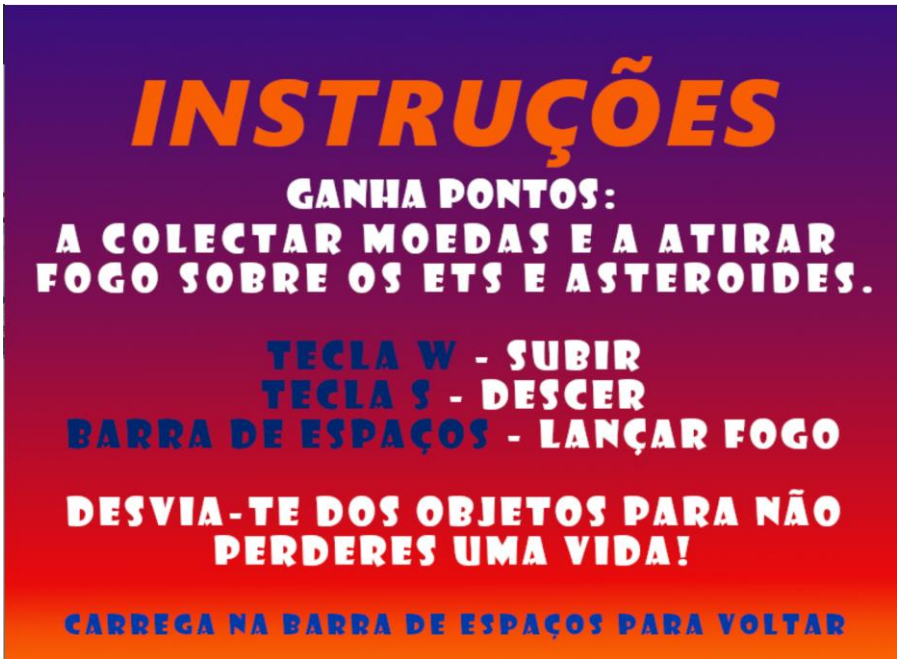


Fig. 2 - Instruções

Uma vez que o utilizador se encontre no ecrã Instruções, será apresentado um texto explicativo sobre a jogabilidade do jogo. O utilizador poderá voltar ao ecrã *Menu* sempre que desejar, pressionando a tecla ESPAÇO.

1.3. Jogo

Fig. 3 – Início do jogo

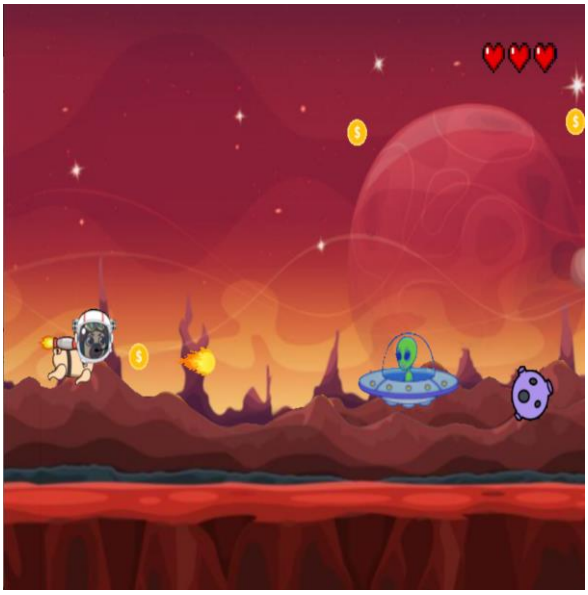


Fig. 4 – Fogo lançado

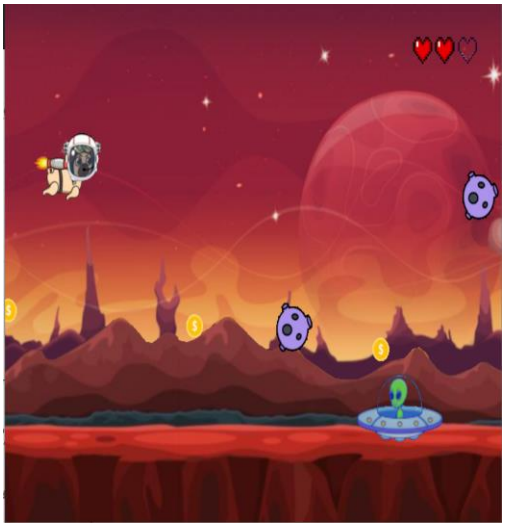


Fig. 5 – Vida perdida

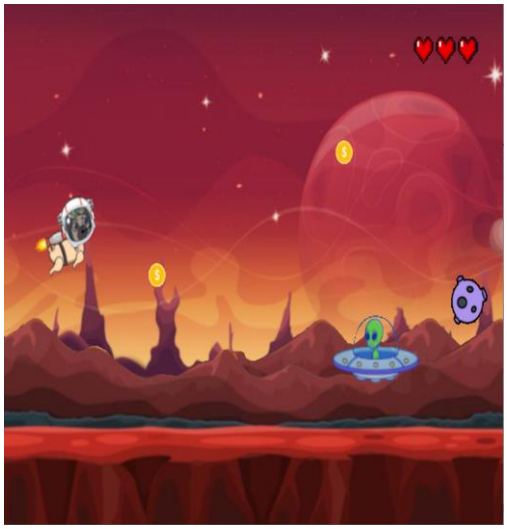


Fig. 6 – Personagem a subir

Fig. 7 – Personagem a descer

Uma vez que o utilizador se encontre no ecrã do jogo, é-lhe apresentado o seu personagem que está representado por um cão com um *JetPack*, moedas, asteroides e ET's (Fig. 3). O objetivo do jogo é que o jogador apanhe o maior número de moedas, colidindo com elas, e que destrua o maior número de asteroides e ET's com o seu jato de fogo, este último aparece somente quando o jogador pressiona a tecla ESPAÇO (Fig. 4). Caso o jogador colida com asteroides ou ET's este irá perder uma vida (Fig. 5). Para se desviar destes últimos, o jogador deve pressionar as teclas W e S, sendo W para subir (Fig. 6) e S para descer (Fig. 7). Se o jogador perder todas as vidas ou se pressionar na tecla ESC, este será redirecionado para um ecrã de *Game Over*.

1.4. Game Over

Fig. 8 – Game Over

Uma vez que o utilizador se encontre no ecrã *Game Over*, é-lhe apresentado o total de pontos que adquiriu durante a partida. O utilizador pode voltar ao ecrã de *Menu* pressionando na tecla ESPAÇO.

2. Estado do Projeto

As funcionalidades implementadas foram as seguintes:
Timer, Keyboard, Mouse e Video Card.

Dispositivo	Utilidade	Interrupções
Timer	Refrescar o ecrã Desenhar frames e controlar frequência de imagem	Sim
Keyboard	Movimentar o personagem e lançar fogos	Sim
Mouse	Selecionar opções no ecrã Menu	Sim
Video Card	Visualizar os ecrãs	Não

Timer

O dispositivo Timer foi usado para controlar a frequência de imagem. A cada interrupção do timer o ecrã é atualizado. Foi utilizada a variável COUNTER, que é alterada a cada interrupção do Timer, para determinar quando são desenhados os asteroides, ET's e as moedas.

Este dispositivo foi utilizado na função principal proj main_loop().

Codigo usado: timer_subscribe_int(), timer_unsubscribe_int(), timer_interrupt_handler() e a variavél COUNTER.

Keyboard

O dispositivo Keyboard foi usado para controlar os movimentos do personagem principal, bem como os fogos lançados por este. A cada interrupção do Keyboard, é guardado num *array* o *scancode* ou *breakcode* da tecla pressionada. Se o primeiro elemento desse array for o *scancode* da tecla W, a posição do personagem diminui em y. Se for o scancode da tecla S a posição do personagem aumenta em y. Por fim, se o *scancode* for o da tecla ESPAÇO, é acrescentado um novo fogo ao *array* de fogos, se o utilizador estiver no ecrã do jogo. Caso este esteja no ecrã Game Over ou Instruções este volta ao ecrã Menu.

Este dispositivo foi utilizado na função principal proj main_loop().

Código usado: kbd_subscribe_int(), kbd_unsubscribe_int(), kbd_ih(), get_data() e o *array* arr[2]. E Estas funções encontram-se no ficheiro keyboard.c e são usadas na função principal principal proj main loop().

Mouse

O dispositivo Mouse foi usado para selecionar opções no ecrã Menu. A cada interrupção do Mouse, é guardado num *array* um pacote com os bytes respetivos a cada tecla do rato, estando o byte a 1 se a tecla foi pressionada, e é guardado no *array* igualmente o deslocamento do rato en x e em y. Para ir para o ecrã selecionado nas opções do Menu é usada uma *enum game_state {MENU, PLAY*, INSTRUCTIONS, EXIT, GAMEOVER}, logo quando a posição do rato se encontra sobre uma das opções e que o seu botão esquerdo é posicionado o *game_state* muda respetivamente.

Este dispositivo foi utilizado na função mouse_pos(...) no ficheiro *mouse.c*.

Código usado: mouse_subscribe_int(), mouse_unsubscribe_int(), mouse_ih(), get_mouse_status(), isSyncronized(), packet_assemble(), mouse_pos() e a variávell packet_byte_cnt. Estas funções encontram-se no ficheiro mouse.c e parte delas são usadas na função principal principal proj main loop().

Video Card

O dispositivo Video Card foi usado para visualizar todas as imagens xpm do jogo. Quando o projeto é iniciado é aberta uma janela no modo 115 usando a função vg_init(0x115). Para representar alterações nos ecrãs é usado o *double buffering* usando a função change_buffer() e por fim para representar cada imagem é usada a função vg_draw_xpm_proj(…).

Codigo usado: vg_init(…), change_buffer(), vg_draw_xpm_proj(…) e pixel_color(…). Estas funções encontram-se no ficheiro video_gr.c e parte delas são usadas na função principal principal proj main loop().

3. Organização do Código e Detalhes de Implementação

O código do projeto foi maioritariamente realizado na função proj_main_loop().

Nessa função foi feito un switch case para cada caso da game_state.

Quando o game_state é PLAY, nessa secção do código é feita toda a parte importante para o funcionamento do jogo.

Os asteroides, os ET's, e as moedas são criados e representados da mesma maneira. Primeiramente é usado o counter do timer para determinar quando adicionar um elemento novo. Esse novo elemento será então adicionado a um array com os restantes elementos do seu tipo. De seguida, esse array é percorrido e cada elemento seu é desenhado utilizando a função vg_draw_xpm_proj(…). As posições de cada elemento são também guardadas em arrays usando as funções asteroides_pos_arr(), ets_pos_arr(), fogos_pos_arr(), coins_pos_arr().Estas funções encontram-se no ficheiro moves.c. A representação dos fogos é idêntica, mas em vez de ser adicionado um novo fogo consoante o counter do timer, é adicionado quando a tecla ESPAÇO é pressionada.

Para movimentar estes elementos, o array das posições em x é percorrido e para cada elemento aumenta-se ou diminui-se esse valor como pretendido.

Para realizar os movimentos do cão, verifica-se se as teclas W ou S foram pressionadas. Se estas foram pressionadas são executadas as funções up(…) e down(…) respetivamente, onde a posição em y é alterada consoante o movimento pretendido.

Para determinar se houve alguma colisão dos asteroides ou dos ET's com os fogos, os arrays dos elementos que se pretende verificar a colisão são percorridos e verifica-se se se encontram na mesma posição, se sim, apaga-se ambos elementos e a pontuação aumenta.

Para determinar se houve alguma colisão dos asteroides, dos ET's ou das moedas com o cão, o array do elemento que se pretende verificar a colisão com o cão é percorrido e verifica-se se se encontram na mesma posição, se sim, apaga-se o elemento e as vidas diminuem de um valor.

Esta parte é a mais significativa para o nosso projeto, logo tem uma importância de 40%.

(realizada por Raquel Sepúlveda)

Quando o game_state é MENU, é desenhado o ecrã de Menu e o rato usando a função vg_draw_xpm_proj(…) e todas a variáveis importantes para o jogo são inicializadas.

Esta parte é importante para o nosso projeto, mas não é fundamental, logo tem uma importância de 7%.

(realizada por Marina Dias)

Quando o game_state é GAMEOVER, é desenhado o ecrã de Game Over e a pontuação usando a função vg_draw_xpm_proj(…) e verifica-se se a tecla ESPAÇO é pressionada, se for o game_state volta a ser MENU.

Esta parte é importante para o nosso projeto, pois informa ao jogador quando perdeu e os pontos que adquiriu, sem esta o jogo não estaria completo, logo tem uma importância de 10%.

(realizada por Marina Dias)

Quando o game_state é INSTRUÇÕES, é desenhado o ecrã de Intruções usando a função vg_draw_xpm_proj(…) e verifica-se se a tecla ESPAÇO é pressionada, se for o game_state volta a ser MENU.

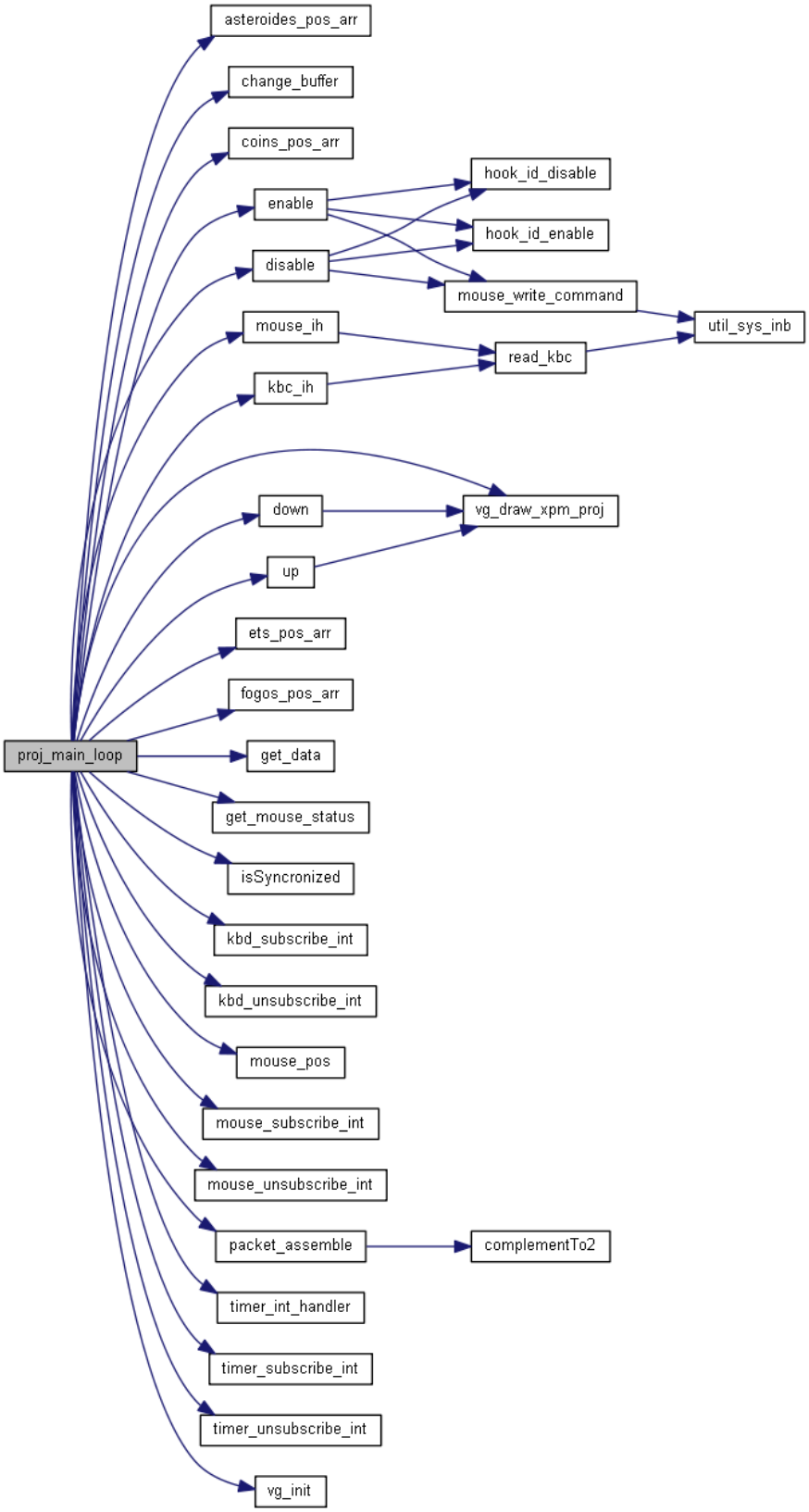
Esta parte não é fundamental para o nosso projeto é bastante útil, principalmente para novos jogadores, logo tem uma importância de 3%.

(realizada por Marina Dias)

Por fim, todas as funções usadas relativas aos dispositivos têm igual importância para o funcionamento do jogo, logo têm uma importância de 40%.

(As funções relativas aos dispositivos foram realizadas em conjunto exceto a função vg_draw_xpm_proj(…) que foi realizada por Marina Dias)

Function call graph



4. Conclusões

A unidade curricular LCOM é muito trabalhosa e requer bastante tempo e dedicação da parte dos estudantes. Tendo em conta que a maioria dos estudantes que realizam esta unidade curricular não têm muita preparação no que toca à programação em C, os laboratórios são um pouco abstratos, porque espera-se dos estudantes que implementem código em C sem sequer aprenderem a linguagem primeiro. Os slides não são suficientemente explicativos para alunos que nunca estiveram em contacto com este tipo de programação, logo os alunos têm de aprender sobre os temas abordados por si próprios, pesquisando na Internet e estudando trabalhos de alunos antigos, a fim de poder entender o que lhes é realmente pedido.

Visto que os laboratórios não são obrigatórios nem avaliados, isso torna a unidade curricular, apesar de tudo, mais acessível.

5. Instruções de Instalação

Para executar o nosso projeto, é necessário fazer *make* a partir da pasta proj/src e seguidamente executar o comando lcom_run proj.