Exercícios de Programação Multimédia

Parte 5

1. Áudio - biblioteca Audio

1.1. Bateria

Crie um programa que reproduza graficamente (de forma aproximada apenas - com círculos e rectângulos) a Figura 1.

Os cliques em cada elemento da bateria devem desencadear o som correspondente.

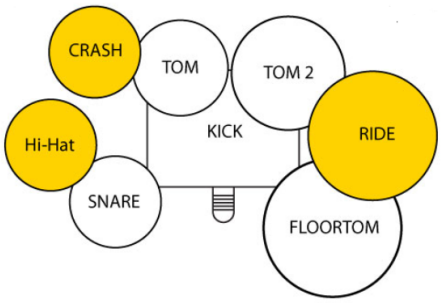

Figura 1: Bateria.

> **Nota:** Os ficheiros de áudio disponíveis não coincidem totalmente com os nomes na imagem. Use os mais aproximados.

1.2. Bola que colide com bordo

Crie um programa que anime uma bola saltitante na janela – a bola deve começar no centro da janela e mover-se numa direção aleatória, fazendo ricochete nos limites da janela.

Sempre que a bola bater no limite da janela e inverter a direção deve ser lançado um som (`ping.wav`).

1.3. Bola "panorâmica" que colide com bordo

Modifique o programa anterior de forma a que o som emitido quando a bola bate no limite da janela seja "espacializado" – a posição x da bola no momento

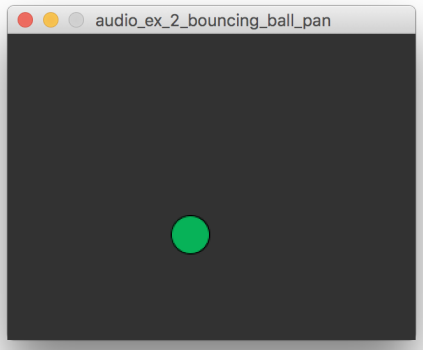

Figura 2: Bola Saltitante.

do embate deve ser usada para decidir a "posição" do som entre os canais esquerdo e direito (use o método `pan()`). A Figura 3 mostra alguns exemplos da espacialização.

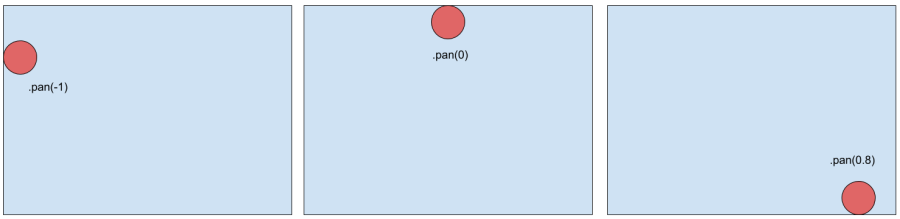

Figura 3: Diagram com exemplo da espacialização.

1.4. Múltiplas bolas "panorâmicas" que colidem com bordo

Usando uma classe `Bola` para representar a bola saltitante, refaça o exercício anterior e coloque várias bolas no ecrã. Construa a classe `Bola` de forma a que o seu programa principal seja igual ao seguinte:

```
import processing.sound.*;

Bola b[] = new Bola[15];

void setup() {
   size(640, 480);
   SoundFile som = new SoundFile(this, "ping.wav");
   for (int i = 0; i < b.length; i++) {
      b[i] = new Bola((int)random(20, 40), som);
   }
}
```

```
void draw() {
   background(50);

   for (int i = 0; i < b.length; i++) {
      b[i].desenha();
   }

   text("fps:␣" + frameRate, 10, 10);
}
```

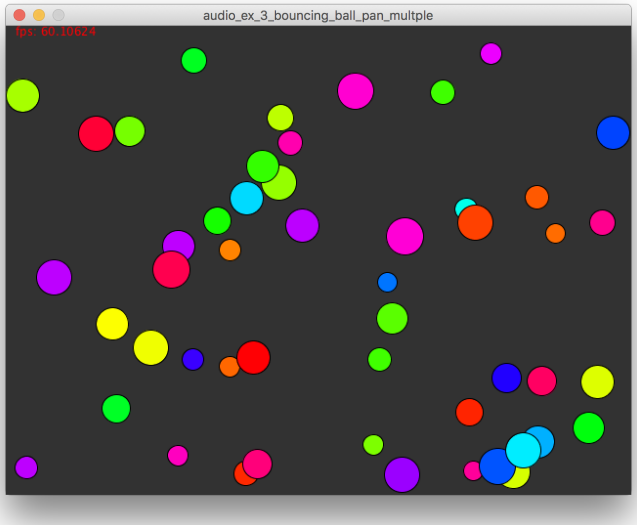

Figura 4: Exemplo do exercício 1.4 "Múltiplas Bolas Saltitantes Panorâmicas".

1.5. Bate Palmas

Crie um programa que capture áudio e desenhe um quadrado sempre que bater palmas – use a deteção do volume de som e desenhe um quadrado sempre que detectar um voluma acima de um determinado valor. Os quadrados devem ser desenhados com cores aleatórias e em posições aleatórias. O tamanho do quadrado deve ser proporcional ao volume do som que lhe deu origem.

1.6. Volume Visual

Crie um programa que capture áudio e represente visualmente o volume ao longo do tempo. O programa deve representar o volume num dado instante como uma linha vertical – desenhando várias linhas na janela (ver Figura 5). Quando chegar ao final da janela, esta deve ser apagada e o processo recomeça.

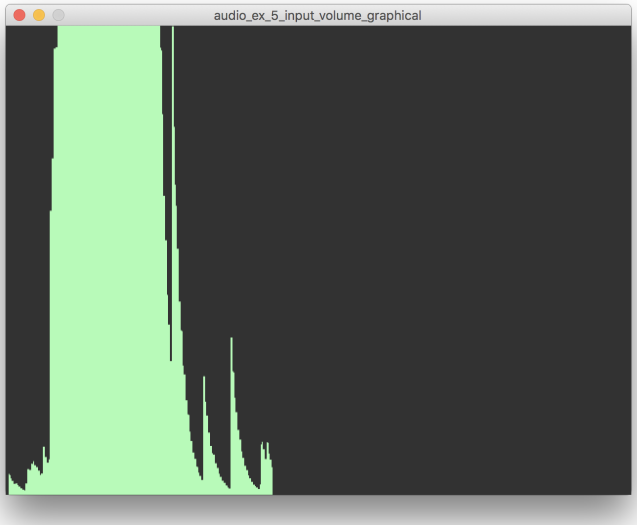

Figura 5: Exemplo do exercício 1.6 "Volume Visual".

1.7. Sequenciador

Crie um programa "sequenciador" de sons. O programa deve definir uma grelha no ecrã em que as linhas representam sons diferentes (pode usar os sons da bateria), e as colunas representam o tempo. De 1 em 1 segundo (ou outro intervalo definido por si), a coluna atual muda para a seguinte e nessa altura, todos os sons ativos nessa coluna são tocados. Quando não há mais colunas, volta-se ao início. Para ativar ou desativar um som numa dada célula, clica-se sobre a célula. O programa deve usar cores para representar o estado de cada célula (som ativo ou não ativo) e para representar a coluna actual.

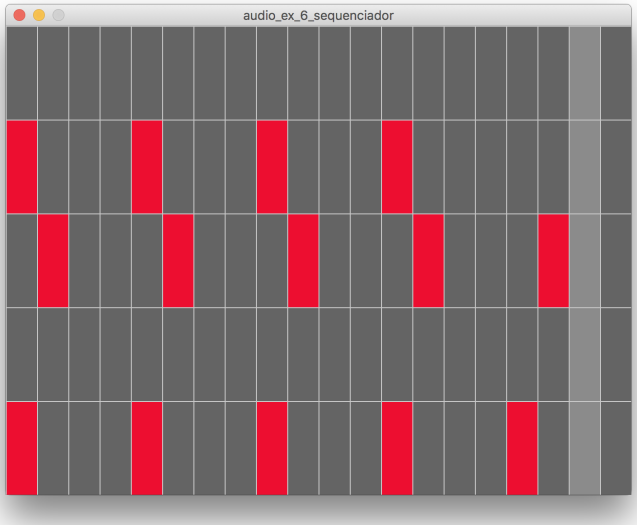

Figura 6: Exemplo do exercício 1.7 "Sequenciador".

2. Áudio - biblioteca Minim

Documentação em `http://code.compartmental.net/minim/index.html`

2.1. ID3 tags & Loops

Crie um programa que permita a reprodução em *loop* de um ficheiro escolhido aleatoriamente entre um conjunto de ficheiros de áudio. A reprodução contínua realizar-se-á entre duas a quatro vezes (valor escolhido aleatoriamente). Durante a primeira reproducão do ficheiro, deverá surgir a seguinte mensagem na janela: "A reproduzir (inserir título aqui) interpretado por (inserir autor aqui)." (Fig. 7, lado esquerdo). Se a reprodução já corresponde a uma repetição da faixa, deve surgir a seguinte mensagem na janela: "Resta ouvir (inserir contagem de loops em falta) vezes." (Fig. 7, lado direito). Ao clicar no rato, deverá ser escolhido aleatoriamente um outro ficheiro e repetir o processo descrito anteriormente.

Para a contagem de *loops*, recomenda-se o recurso ao método `loopCount()` da classe `AudioPlayer`.

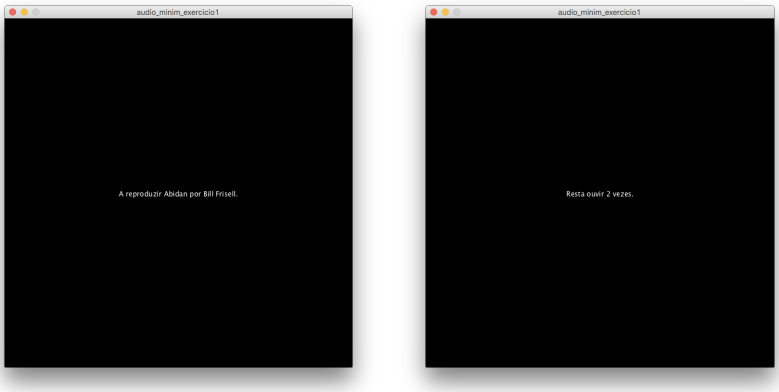

Figura 7: Mensagens a escrever. Lado esquerdo: mensagem aquando da primeira reprodução; lado direito: mensagem quando faltam duas reproduções do ficheiro para terminar.

2.2. Gravação

Crie um programa tendo por base o código apresentado para a classe AudioRercorder para implementar o programa que grava do microfone para o ficheiro 8. O programa, para além de mostrar as ondas do nível do som de entrada, deve apresentar frases centradas que o utilizador deve ler enquanto tem a gravação ativa.

Os cliques do rato permitem mudar as frases. A gravação é controlada através do teclado em que a tecla 'r' inicia e pára uma gravação, enquanto que a tecla 's' permite guardar a mesma no disco.

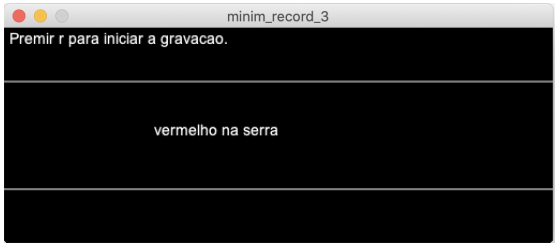

Figura 8: Gravação.

Deve ainda adicionar a funcionalidade de gravar o som com efeito de playback. Para tal recomenda-se a utilização do efeito *BandPass* com uma frequência central de 440 e uma largura de banda de 10. Para a leitura do texto recomenda-se que tenha por base o exercício 2 da ficha 2 (com Strings).

2.3. Múltiplas bolas "panorâmicas" que colidem com bordo 2

Resolva o exercício 1.4 recorrendo a classes da biblioteca Minim.

2.4. Análise FFT

Refaça o exemplo 4 (FFT) apresentado nas aulas de forma a ter a análise espectral baseada em FFT do canal esquerdo a surgir no topo da janela, enquanto a mesma análise, mas desta vez do canal direito, surge no fundo da janela, tal como se ilustra na Fig. 9.

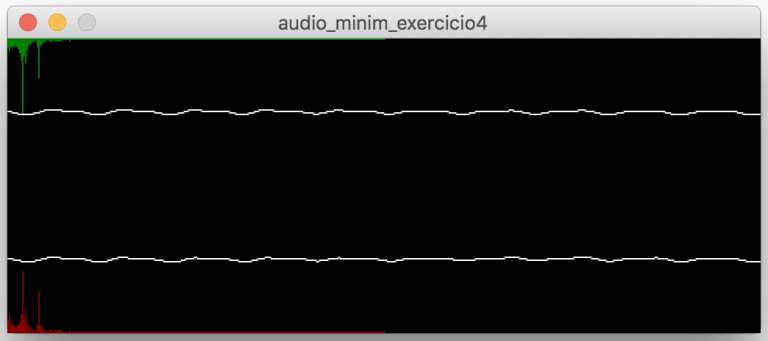

Figura 9: Análise espectral baseada em FFT.

2.5. Análise FFT 2

Numa janela 600× 600 represente as várias bandas do espectro de um ficheiro áudio usando triângulos dispostos circularmente, tendo o centro da janela como o vértice comum (ver Fig. 10). A área de cada triângulo deve variar com a amplitude da banda que represente, enquanto a cor de preenchimento é aleatória. Adicionalmente, o fundo (em níveis de cinzento) deverá variar de acordo com a amplitude do som, tendo a intensidade como valor mínimo 180 e máximo 255.

3. Desafios

3.1. Another air theremin

A partir do código do exemplo 5 da biblioteca Minim (*Air theremin*), construa um programa que define a frequência de um oscilador em função da altura de um dado objeto ou cor. Recomenda-se a utilização da biblioteca OpenCV para o seguimento (*tracking*) do objetos e/ou da cor.

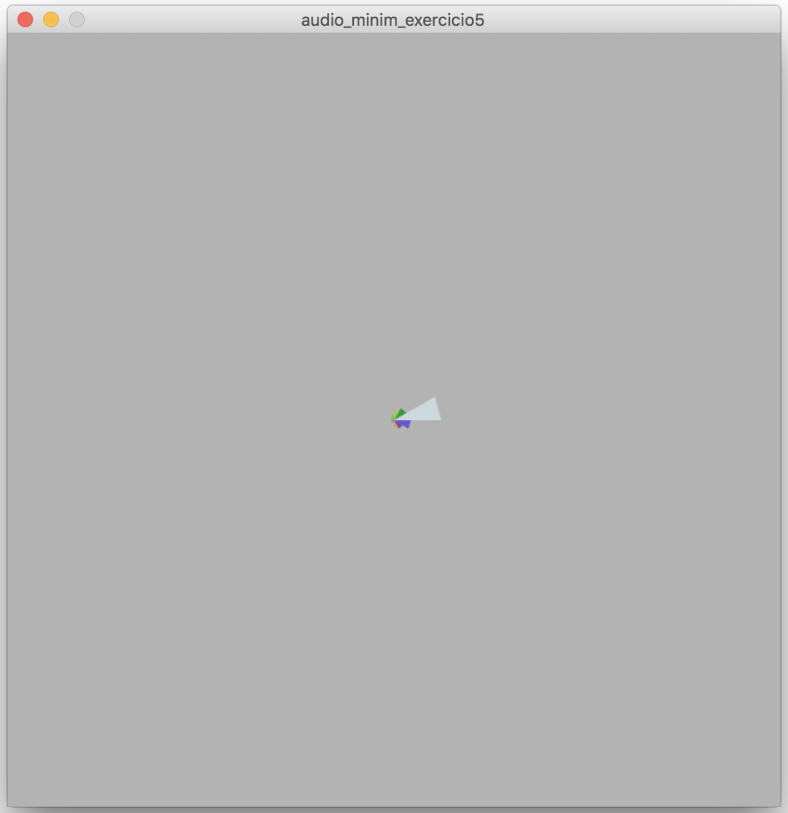

Figura 10: Análise espectral baseada em FFT.