function [fala,ruido] = detetor\_fala(fich\_audio, noise\_len, pause\_len, espor\_noise\_len)

% detetor\_fala -> função que analisa o sinal de áudio e separa o ruido da

% fala através da criação de 2 vetores

% Argumentos de entrada:

% fich\_audio: vetor com o sinal áudio a analisar

% noise\_len: número de amostras do segmento do ruido

% pause\_len: duração do tempo de pausa permitido no segmento de

% voz(amostras), normalmente 50

% espor\_noise\_len: número máximo de amostras que um ruido esporádico pode tomar

% Argumentos de saída:

% ruido: vetor que guarda o segmento de ruido (ruido removido)

% fala: vetor que guarda o segmento de fala sem ruido

%Inicializações

ruido=0; %vetor segmento de ruido inicializado a 0

fala=0; %vetor segmento de fala inicializado a 0

temp\_espor=0; %variável contador do número de amostras para saber se o sinal obtido é esporádico ou não

temp\_fala\_pausa=0; %variável contador do número de amostras para saber se o sinal obtido é uma pausa entre as falas ou não

n\_amostras=0; %variável auxiliar contador do número de amostras

SNR=(mean(fich\_audio))/(std(fich\_audio)); %SNR=media/desvio\_padrao

threshold=SNR\*10; %threshold calcula um limite, que indica

if abs(threshold) < 1 %qual é a probalidade de encontrar ruido ou fala

threshold=2;

end

media=mean(fich\_audio(1:noise\_len)); %calculo da média

dp=std(fich\_audio(1:noise\_len)); %calculo do desvio-padrão

%Algoritmo da função

for i=1:length(fich\_audio) %percorre-se o ficheiro de áudio e compara-se

if abs(fich\_audio(i)) > (media + abs(threshold)\*dp) %se o valor do sinal é maior ao limite calculado é um segmento de fala

temp\_espor = temp\_espor+1;

n\_amostras = n\_amostras+1;

temp\_fala\_pausa=0;

if(temp\_espor) > espor\_noise\_len %se o contador de amostras de ruido esporadico for maior que o valor recebido por parâmetro

fala=cat(1,fala,fich\_audio(i-n\_amostras+1:i)); %conta-se como um segmento de fala, senão é removido

n\_amostras=0;

end

else %senão é um segmento de ruido

temp\_fala\_pausa=temp\_fala\_pausa+1;

n\_amostras = n\_amostras+1;

if temp\_fala\_pausa > pause\_len %se o contador da pausa for maior que o valor recebido por parametro assume-se que é ruido

ruido=cat(1,ruido,fich\_audio(i-n\_amostras+1:i));

temp\_espor=0;

n\_amostras=0;

temp\_fala\_pausa=0;

end

end

end

end

function [fala,ruido] = detetor\_fala(fich\_audio, noise\_len, pause\_len, espor\_noise\_len)

% detetor\_fala -> função que analisa o sinal de áudio e separa o ruido da

% fala através da criação de 2 vetores

% Argumentos de entrada:

% fich\_audio: vetor com o sinal áudio a analisar

% noise\_len: número de amostras do segmento do ruido

% pause\_len: duração do tempo de pausa permitido no segmento de

% voz(amostras), normalmente 50

% espor\_noise\_len: número máximo de amostras que um ruido esporádico pode tomar

% Argumentos de saída:

% ruido: vetor que guarda o segmento de ruido (ruido removido)

% fala: vetor que guarda o segmento de fala sem ruido

%Inicializações

ruido=0; %vetor segmento de ruido inicializado a 0

fala=0; %vetor segmento de fala inicializado a 0

temp\_espor=0; %variável contador do número de amostras para saber se o sinal obtido é esporádico ou não

temp\_fala\_pausa=0; %variável contador do número de amostras para saber se o sinal obtido é uma pausa entre as falas ou não

n\_amostras=0; %variável auxiliar contador do número de amostras

SNR=(mean(fich\_audio))/(std(fich\_audio)); %SNR=media/desvio\_padrao

threshold=SNR\*10; %threshold calcula um limite, que indica

if abs(threshold) < 1 %qual é a probalidade de encontrar ruido ou fala

threshold=2;

end

media=mean(fich\_audio(1:noise\_len)); %calculo da média

dp=std(fich\_audio(1:noise\_len)); %calculo do desvio-padrão

%Algoritmo da função

for i=1:length(fich\_audio) %percorre-se o ficheiro de áudio e compara-se

if abs(fich\_audio(i)) > (media + abs(threshold)\*dp) %se o valor do sinal é maior ao limite calculado é um segmento de fala

temp\_espor = temp\_espor+1;

n\_amostras = n\_amostras+1;

temp\_fala\_pausa=0;

if temp\_espor > espor\_noise\_len %se o contador de amostras de ruido esporadico for maior que o valor recebido por parâmetro

if (fala==0)

fala=fich\_audio(i-n\_amostras+1:i); %conta-se como um segmento de fala, senão é removido

n\_amostras=0;

else

fala(length(fala)+1:length(fala)+ n\_amostras)=fich\_audio(i-n\_amostras+1:i);

n\_amostras=0;

end

end

else %senão é um segmento de ruido

temp\_fala\_pausa=temp\_fala\_pausa+1;

n\_amostras = n\_amostras+1;

if temp\_fala\_pausa > pause\_len %se o contador da pausa for maior que o valor recebido por parametro assume-se que é ruido

temp\_espor=0;

if(n\_amostras~=0)

if (ruido==0)

ruido=fich\_audio(i-n\_amostras+1:i);

else

ruido(length(ruido)+1:length(ruido)+ n\_amostras)=fich\_audio(i-n\_amostras+1:i);

end

end

n\_amostras=0;

end

end

end

end