TD - effets audionumériques

Prénom : Alfred Nom : Gaillard Groupe : ECIB 1

Point 1 :

```
s=audioread('toms.wav');
```

Point 2 :

```
l=length(s);
```

La valeur de l est 418000, on divise cette valeur par la fréquence :
418000/44100= 9.4785s

Point 3 :

On crée un nouveau son destination

```
d=zeros(l,1);
```

Point 4 :

Sachant que N=1024 la longueur d'une trame est donc de H=512, la boucle va dépasser la longueur en échantillon du son de 512-l%512.
Pour solutionner ce problème, il suffit alors de ne pas traiter la dernière trame en faisant :

```
for n=1:H:L-H
```

Point 5 :

Lors de traitement on peut avoir des disparité entre deux trames de 1024, afin de les "lisser" pour chaque demi trames on fusionne la demi trame courante avec celle d'avant et celle d'après. Ce qui correspond à se déplacer en demi trame N=512.

Point 6 :

```
H=1024;
N=512;
for n=1:H:length(s)-N+1
      x=s(n:n+N-1);
      s2(n:n+N)=s2(n:n+N)+x;
endfor
```

Point 7 :

Comme vu précédemment le son dure 9,478s et contient 418000 échantillons.
Or une trame contient 1024 échantillons, le son contiendra 418000/1024=408,203 trames.

D'ou 9,478 / 408 = 0,023s
Une trame dure 0,023 secondes.

Point 8 :
Le code correspondant est : `c = 0.5*(1-cos((2*pi*(0:N-1))/N));`

Point 9 :
On rajoute donc dans le for la ligne suivante :

```
for n=1:H:length(s)-N+1
      x=s(n:n+N-1);
      c = 0.5*(1-cos((2*pi*(0:N-1))/N));
      x=toms(n+1:n+H).*c;
      s2(n:n+N)=s2(n:n+N)+x;
endfor
```

Point 10 :
Pour chaque trame du son, il faut replacer la référence de temps au milieu de la trame, on le fait avec le code suivant : `graphe = fftshift(trame);`

Point 11 :
Passage dans le domaine fréquentiel par application de la FFT :

```
X = fftshift(trame);
graphe = fft(X);
```

Point 12 :
Affichage avec les fonctions abs et plot :

```
for n=1:H:length(s)-N+1
      t = toms(n:n+H);
      r = fftshift(t);
      freq = fft(r);
      a = abs(freq);
      plot(a(1:H/8));
      pause(0.1);
endfor
```

On obtient le spectre de chaque trame du son par tranche de 0.1 seconde.
Le spectre représente l'évolution du son au fil du temps.

Point 13 :
La boucle for devient :

```
trameTemporel = abs(TrameFrequentiel);

for n=1:H:L-H+1
      w = transpose(0.5*(1-cos((2*pi*(0:N))/N)));
      trameTemporaire = toms(n:n+H);
      tramePondere = trameTemporaire .* w;
      trameCentre = fftshift(tramePondere);
      TrameFrequentiel = fft(trameCentre);
      trameTemporel = abs(TrameFrequentiel);
      plot(trameTemporel(1:H/8));
      pause(0.1);
endfor
```

Point 14 :
On fait la FFT inverse au spectre avec :

```
trameTemporel = ifft(abs(TrameFrequentiel));
for n=1:H:L-H+1
      w = transpose(0.5*(1-cos((2*pi*(0:N))/N)));
      trameTemporaire = toms(n:n+H);
      tramePondere = trameTemporaire .* w;
      trameCentre = fftshift(tramePondere);
      TrameFrequentiel = fft(trameCentre);
      trameTemporel = ifft(abs(TrameFrequentiel));
      plot(trameTemporel(1:H/8));
      pause(0.1);
endfor
```

Point 15 :
Pour être sûr qu'il n'y a pas de partie imaginaire, on ajoute :

```
trameReal = real(trameTemporel);
for n=1:H:L-H+1
      w = transpose(0.5*(1-cos((2*pi*(0:N))/N)));
      trameTemporaire = toms(n:n+H);
      tramePondere = trameTemporaire .* w;
      trameCentre = fftshift(tramePondere);
      TrameFrequentiel = fft(trameCentre);
      trameTemporel = ifft(abs(TrameFrequentiel));
      trameReal = real(trameTemporel);
      plot(trameReal(1:H/8));
```

```
      pause(0.1);
endfor
```

Point 16 :
Pour recentrer la trame real on utilise à nouveau la fonction fftshift :

```
trameRealCentre = fftshift(trameReal);
for n=1:H:L-H+1
      w = transpose(0.5*(1-cos((2*pi*(0:N))/N)));
      trameTemporaire = toms(n:n+H);
      tramePondere = trameTemporaire .* w;
      trameCentre = fftshift(tramePondere);
      TrameFrequentiel = fft(trameCentre);
      trameTemporel = ifft(abs(TrameFrequentiel));
      trameReal = real(trameTemporel);
      trameRealCentre = fftshift(trameReal);
      plot(trameRealCentre(1:H/8));
      pause(0.1);
endfor
```

Point 17 :

On accumule cette trame dans le son destination avec une addition pour ne pas écraser le son originel en ajoutant :

```
destination(n:n+H) = destination(n:n+H) + trameRealCentre;
for n=1:H:L-H+1
      w = transpose(0.5*(1-cos((2*pi*(0:N))/N)));
      trameTemporaire = toms(n:n+H);
      tramePondere = trameTemporaire .* w;
      trameCentre = fftshift(tramePondere);
      TrameFrequentiel = fft(trameCentre);
      trameTemporel = ifft(abs(TrameFrequentiel));
      trameReal = real(trameTemporel);
      trameRealCentre = fftshift(trameReal);
      destination(n:n+H) = destination(n:n+H) + trameRealCentre;
endfor
```

Point 18 :

Sauvegarde du fichier son destination grâce à la fonction audiowrite :

```
nom = 'sonDestination.wav';
audiowrite(nom, destination, 44100);
```

Point 19 :

Si on n'utilise plus la fonction abs, le son de destination redevient le même que le son d'origine.
La fonction abs est donc bien responsable du son robotique.

Point 20 :

On applique une fenêtre de Hann au spectre. Le but étant d'appliquer un filtre passe-haute. Le spectre ne laisse alors apparaître que les hautes fréquences.