Chapter 8: Xử lý tiếng nói

Date	@April 8, 2024
Status	Done

Lý thuyết

Tiếng nói - Speed

Tổng quan về tiếng nói và lĩnh vực phân tích tiếng nói trong thực tế

Thu tiếng nói

```
r = audiorecorder(fs,bits,channels)
```

`fs` tần số lấy mẫu, mặc định là 8 kHz

`bits`: (bit per sample): số bít mã hoá mỗi mẫu tiếng nói, mặc định là 8

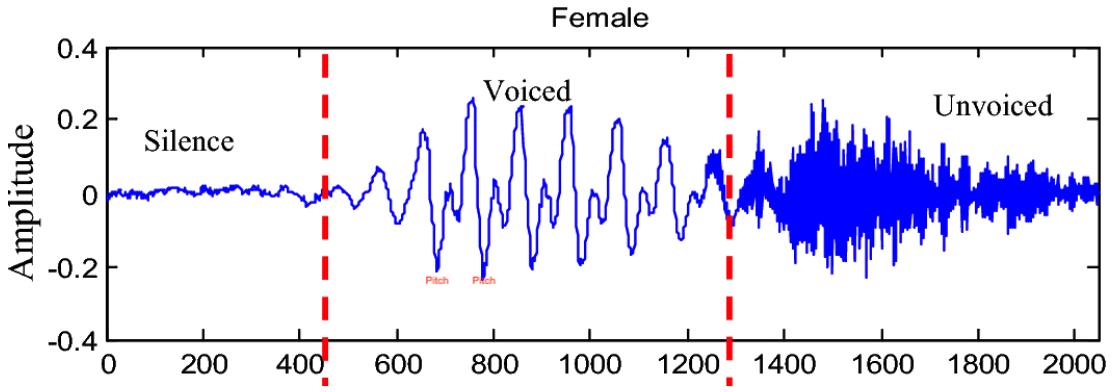
`channels`: quyết định việc ghi âm đơn kênh mono (=1) hay âm thanh lập thể stereo (=2)

`record (r)`	ghi âm
`pause (r)`	dừng (pause) quá trình ghi âm
`resume(r)`	tiếp tục lại quá trình ghi âm
`stop (r)`	dừng hẳn quá trình ghi âm
`recordblocking (r,10)`	ghi âm trong 10s
`play (r)`	nghe lại file ghi âm
`audiowrite("myvoice.wav", r, fs)`	ghi ra file myvoice.wav
`voice = getaudiodata(r)`	chuyển dữ liệu audio ra dạng vector dữ liệu
`plot(voice)`	Vẽ tín hiệu trong miền thời gian

Đặc điểm của tiếng nói

Tiếng nói, nói chung, gồm 3 giai đoạn:

- Giai đoạn im lặng
- Giai đoạn phát tiếng (voiced) có biên độ lớn, các chu kỳ, đỉnh rõ ràng
- Giai đoạn không phát tiếng- (unvoiced) có biên độ bé hơn, nhiễu hơn



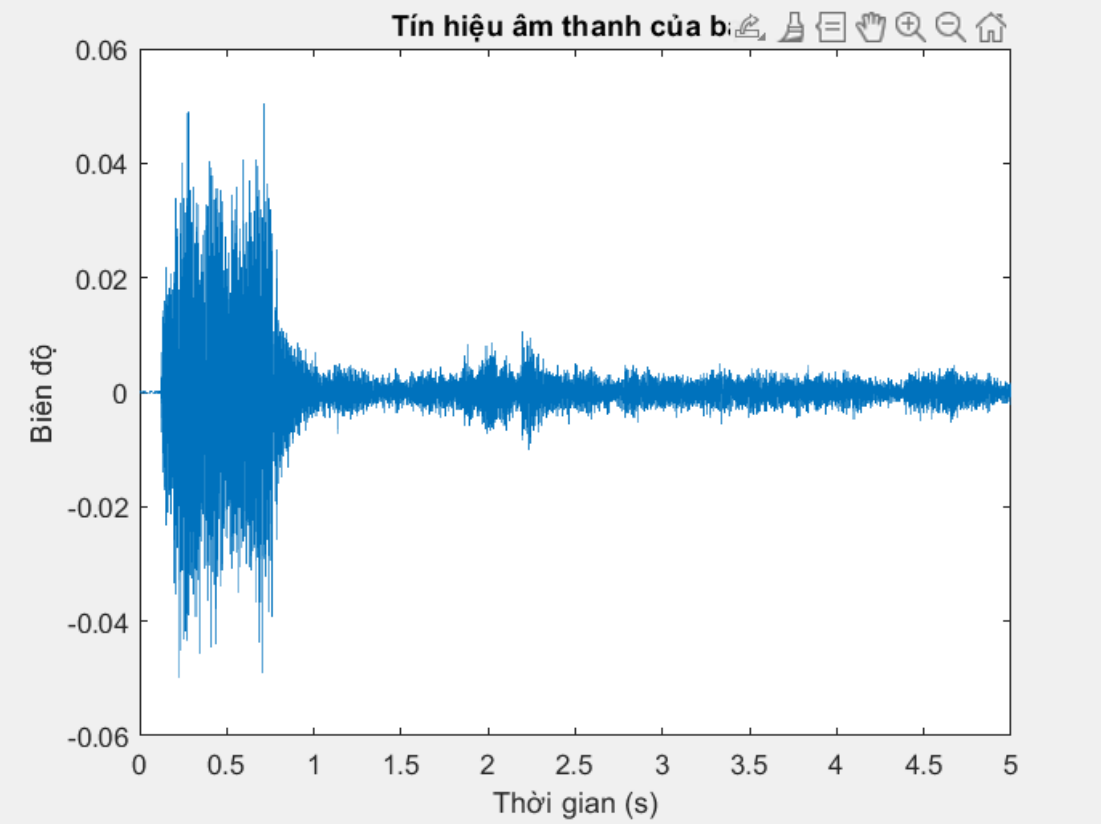
▼ Loại nhiễu trong file âm thanh

- Trong quá trình thu âm, âm thanh thường bị nhiễu do môi trường xung quanh gây ra, đặc biệt ở giai đoạn unvoiced.
- Giả sử, tín hiệu thu được là: s(n) = x(n) + d(n). với x(n) là tín hiệu sạch và d(n) là nhiễu của tín hiệu.
- **Phương pháp trừ phổ:** Khôi phục tín hiệu x(n) từ tín hiệu s(n) thu được sau khi ghi âm

▼ Bài tập

▼ Câu 1

```
>> Cau1bai8
Bắt đầu ghi âm. Hãy nói vào micro...
Kết thúc ghi âm.
Nghe lại âm thanh.
Đã lưu file âm thanh: myvoice1.wav
```



▼ Câu 2

Sử dụng lệnh plot, vẽ tín hiệu âm thanh trên trong file *start.au*. Khoảng voice và unvoice được xác định trong hình

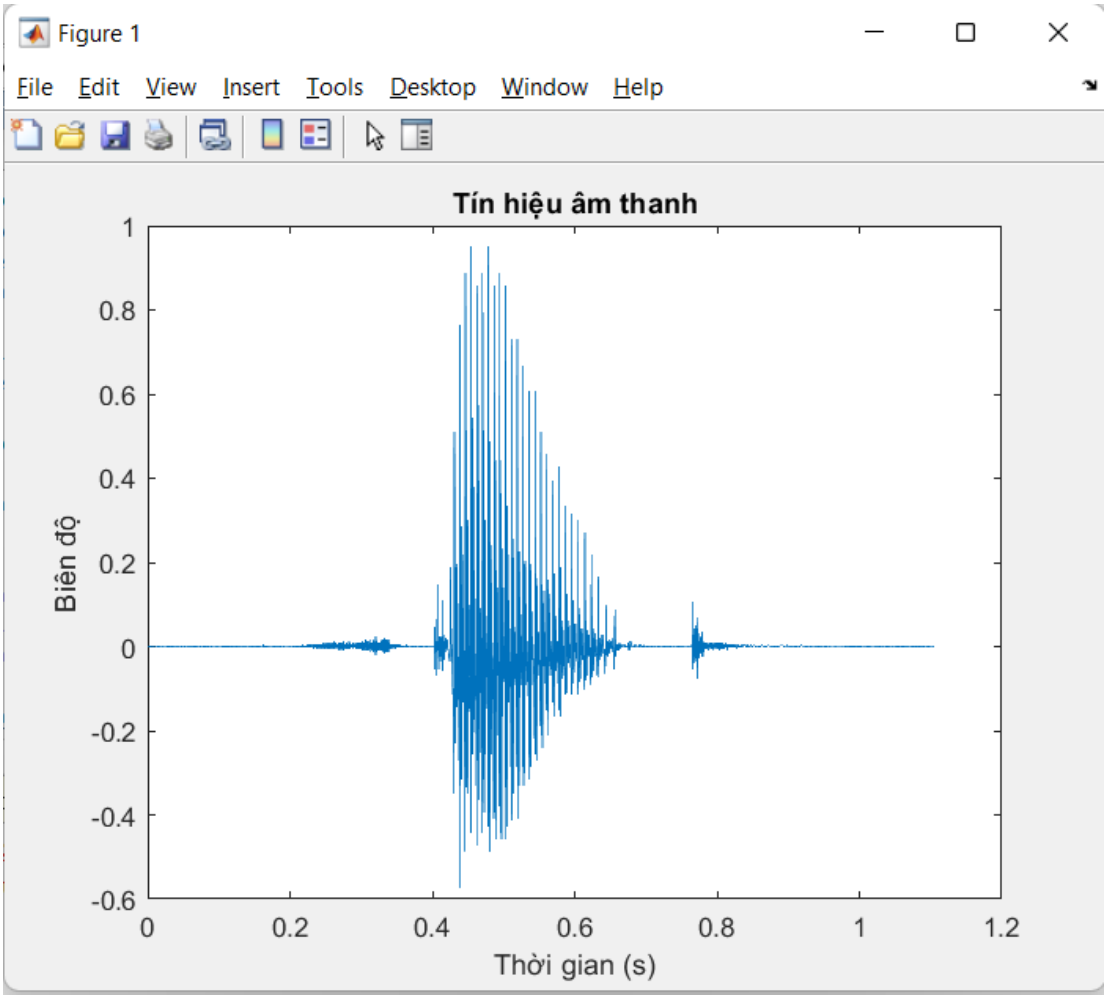
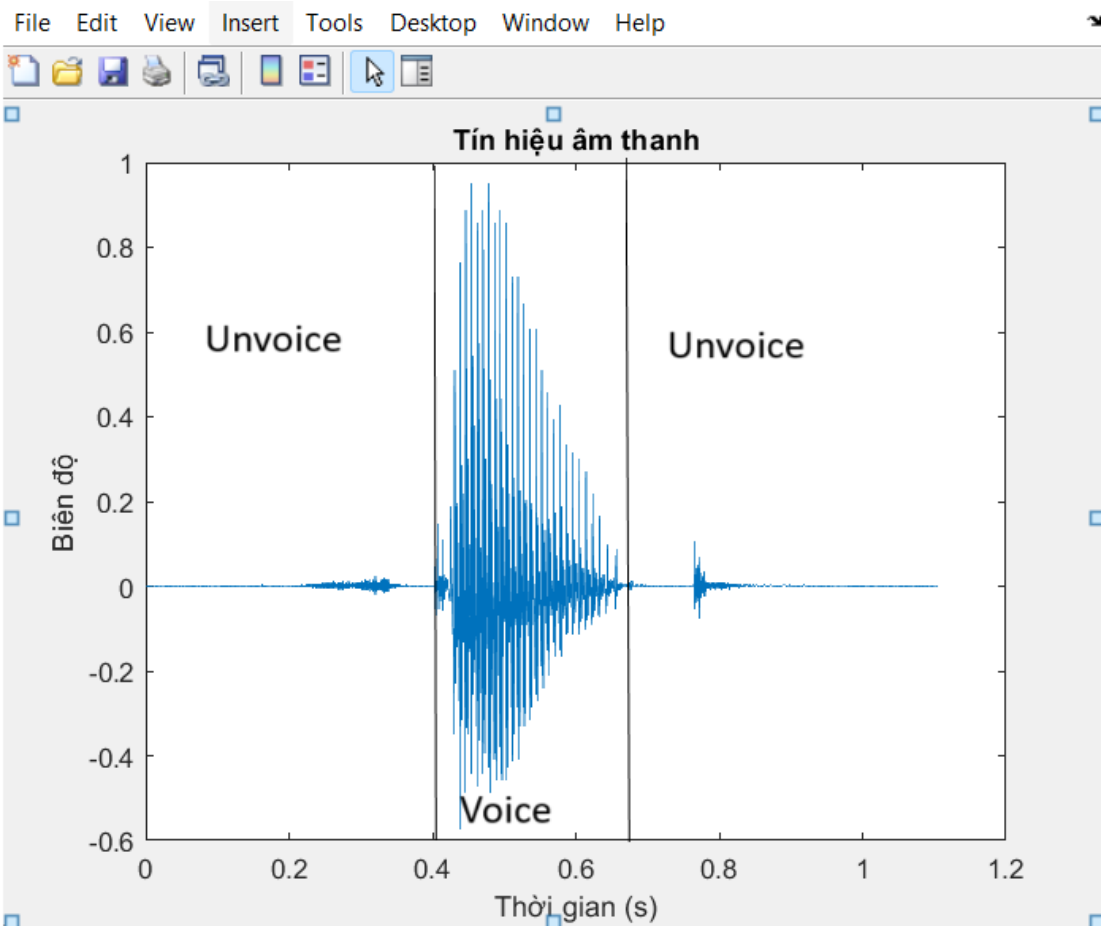


Figure 1
File Edit View Insert Tools Desktop Window Help
Tín hiệu âm thanh
Biên độ
Thời gian (s)
1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
0
0.2
0.4
0.6
0.8
1
1.2



▼ Câu 3

Tiếng nói của nam và nữ có các khoảng cách giữa pitch khác nhau. Với nam, khoảng cách giữa 2 pitch liên tiếp là 8ms còn với nữ là 4ms

```
Khoảng cách giữa các pitch:
  -19.0476
         0
         0
         0
   19.0476
         0
         0
```

```
Giới tính: Nam
Khoảng thời gian giữa 2 pitch liên tiếp (ms):
   -2.3810
         0
         0
         0
    2.3810
         0
         0
```

▼ Câu 4

Trong MATLAB, hàm `stft` (short-time Fourier transform) có nhiệm vụ chia tín hiệu thành các khung (frames) và tính FFT cho từng khung. Đây là một công cụ quan trọng trong xử lý tín hiệu và phân tích âm thanh.

Để sử dụng hàm `stft` trong MATLAB, bạn có thể làm như sau:

1. Tải tín hiệu âm thanh vào MATLAB hoặc tạo một tín hiệu âm thanh mới.
2. Xác định các thông số cần thiết cho việc chia khung (framing) và tính FFT. Các thông số bao gồm:
   - Kích thước của mỗi khung (frame size): Đây là số mẫu (samples) trong mỗi khung. Thông thường, kích thước này được chọn là một lũy thừa của 2 để tận dụng tính chất của FFT (ví dụ: 256, 512, 1024).
   - Bước nhảy giữa các khung (frame hop): Đây là số mẫu giữa hai khung liên tiếp. Thông thường, bước nhảy này nhỏ hơn kích thước của khung để có sự chồng lắp giữa các khung.
3. Sử dụng hàm `stft` để thực hiện chia khung và tính FFT cho từng khung. Ví dụ:

   matlab

   Copy

   ```matlab
   % Xác định các thông số
   frameSize = 256;
   frameHop = 128;

   % Thực hiện STFT
   [S, f, t] = stft(audio, frameSize, frameHop);
   ```

Trong đoạn mã trên, `audio` là tín hiệu âm thanh đầu vào, `S` là ma trận biểu diễn STFT, `f` là vector các tần số và `t` là vector thời gian tương ứng với các khung.

Lưu ý rằng hàm `stft` có thể có các tham số và tùy chọn khác nhau trong MATLAB, nhưng các tham số cơ bản được đề cập ở trên là quan trọng nhất.

Hàm `stft` giúp bạn phân tích tín hiệu âm thanh thành miền tần số và miền thời gian, cung cấp thông tin về biên độ và pha của tín hiệu trong các khung thời gian nhỏ. Điều này rất hữu ích trong nhiều ứng dụng xử lý tín hiệu âm thanh, như phân tích âm vực, trích xuất đặc trưng, và xử lý tín hiệu thời gian thực.