دانشکده مهندسی کامپیوتر

انتقال داده‌ها

پاییز ۱۴۰۰

پروژه فصل سوم

نظریه اطلاعات

استاد درس . . . . . دکتر دیانت

نام . . . . . علی صداقی

شماره دانشجویی ۹۷۵۲۱۳۷۸

فهرست مطالب

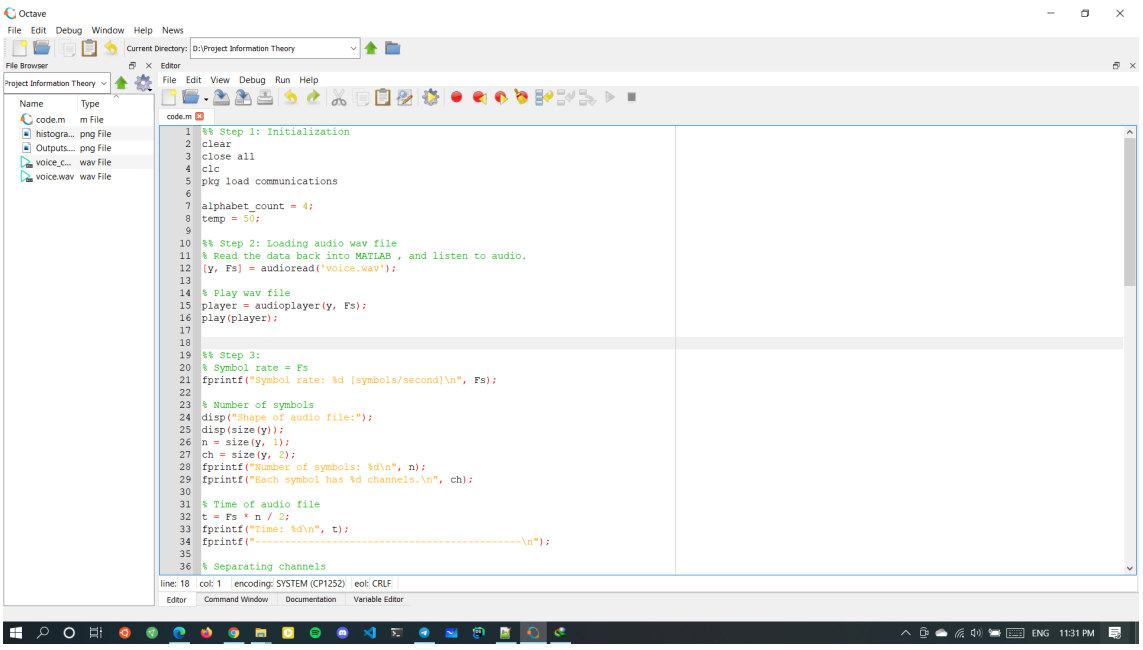
۱ گام اول

با توجه به حجم زیاد نرم‌افزار متلب و کمبود فضا، از نرم‌افزار GNU Octave استفاده کردم که دارای دو محیط کاربری متنی و گرافیکی است.

این برنامه تمامی دستورات متلب را پشتیبانی می‌کند و قابلیت بارگذاری پکیج‌ها و تولباکس‌های متلب را دارد.

برای مثال با دستور زیر می‌توانیم پکیج مربوط بهه انتقال داده متلب را بارگذاری کنیم:

```
pkg load communications
```



شکل ۱: محیط Octave

۲ گام دوم

یک فایل با فرمت WAV را از بازی جنگ‌های صلیبی بارگذاری می‌کنیم.

```
10  %% Step 2: Loading audio wav file
11  % Read the data back into MATLAB , and listen to audio.
12  [y, Fs] = audioread('voice.wav');
13
14  % Play wav file
15  player = audioplayer(y, Fs);
16  play(player);
```

شکل ۲: بارگذاری فایل صوتی

۳ گام سوم

سوالات پرسیده شده در متن را با استفاده از کد متلب پاسخ می‌دهیم.

```
% Symbol rate = Fs
fprintf("Symbol rate: %d [symbols/second]\n", Fs);

% Number of symbols
disp("Shape of audio file:");
disp(size(y));
n = size(y, 1);
ch = size(y, 2);
fprintf("Number of symbols: %d\n", n);
fprintf("Each symbol has %d channels.\n", ch);

% Time of audio file
t = Fs * n / 2;
fprintf("Time: %d\n", t);
fprintf("-----------------------------------------\n");

% Separating channels
% Comparing histograms shows the equality
% of two channels
y_1 = y(:, 1);

% Type of middle symbol
middle_symbol = y_1(floor(n/2))(1);
fprintf("Middle symbol type: %s\n", class(middle_symbol));

% Middle symbol
fprintf("Middle symbol value: %f\n", middle_symbol);
fprintf("-----------------------------------------\n");

% Min symbol
fprintf("Min symbol value: %f\n", min(y_1));

% Max symbol
fprintf("Max symbol value: %f\n", max(y_1));

% Mean symbol
fprintf("Mean of symbols: %f\n", mean(y_1));
```

شکل ۳: کد مربوط به پرینت مولفه‌های گام دوم

خروجی این قسمت به صورت زیر است:

Symbol rate: 22050 [symbols/second]
Shape of audio file: 87462 1
Number of symbols: 87462
Each symbol has 1 channels.
Time: 3.96653

Middle symbol type: double
Middle symbol value: 0.008942

Min symbol value: -0.928558
Max symbol value: 0.978363
Mean of symbols: -0.000313

به تعداد 87462 سمبل داریم که هر یک از جنس double است که دقت آن تا ۷ رقم اعشار هست. برای ذخیره سازی این فرمت نیاز به ۸ بایت داریم.

نرخ سمبل در ثانیه برابر 22050 شده است. یعنی در هر ثانیه آن تعداد سمبل داریم.

مقادیر هر سمبل میان عدد 1- تا 1 است.

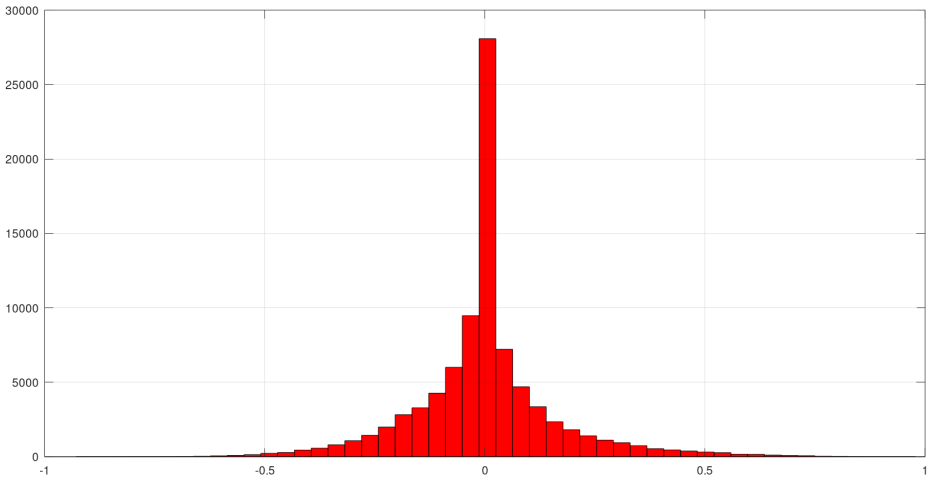
از تقسیم تعداد سمبل در نرخ سمبل بر ثانیه می‌توانیم به زمان فایل صوتی برسیم. که این عدد تقریبا نزدیک ۴ شده است.

طبق قضیه نایکویست با توجه به اینکه نرخ نمونه برداری بزرگتر از ۲ برابر بیشینه فرکانس بوده است، می‌توان از سیگنال ثانویه به سیگنال اولیه رسید. و پدیده آلیاسینگ نداریم به همین دلیل است که صوت گسستگی نداریم.

همچنین فرکانس شنوایی انسان از نرخ این صوت پایین تر است و ما آن را پیوسته می‌شنویم. در هنگام مشاهده فیلم نیز همین اتفاق می‌افتد زیرا چشم انسان نهایتن تا 24 فریم بر ثانیه را می‌تواند تشخیص دهد و اگر فریم ریت این فیلم بالاتر از این نرخ باشد ما آن را پیوسته مشاهده می‌کنیم.

۴ گام چهارم

با استفاده از دستور hist نمودار هیستوگرام را با 50 بازه رسم می‌کنیم.



شکل ۴: هیستوگرام

ابتدا مقدار احتمالات را به ازای هر بازه محاسبه می‌کنیم، سپس با استفاده از فرمول آنتروپی مقدار آنتروپی صوت را محاسبه می‌کنیم.

```
%% Step 4:
% Plot histogram
hist(y_1, temp, 'FaceColor', 'red');
colormap(summer ());
grid on;

% Probability distribution
p = hist_quantized / sum(hist_quantized);
p_org = histo / sum(histo);
fprintf("Sum of probabilities: %d\n", sum(p));
fprintf("-----------------------------------------\n");

% Entropy
entropy = -sum(p_org .* log2(p_org));
fprintf("Entropy: %d\n", entropy);
fprintf("-----------------------------------------\n");
```

شکل ۵: کد مربوط به گام چهارم

خروجی به صورت زیر است:

Sum of probabilities: 1

Entropy: 3.74716

با توجه به قضیه شانون می‌توانیم این صوت را تا حد مقدار زیر فشرده کنیم. اما می‌دانیم شانون این مرز را به ما داده ولی روش رسیدن به آن نامعلوم است. به همین دلیل است که در عمل نمی‌توانیم به این عدد دست یابیم زیرا این عدد از تئوری محض به دست آمده است. منظور فشرده سازی بدون از دست دادن اطلاعات است.

$$3.74716 * 87462/8000 = 40.96[kB]$$

همچنین در صوت توزیع احتمال پیوسته است اما ما در درس با توزیع احتمال گسسته سروکار داشتیم پس باید از نمونه برداری استفاده کنیم، نرخ نمونه برداری متفاوت آنتروپی متفاوتی تولید میکند و آنتروپی متفاوت سایز فشرده شده متفاوت.

در عمل معمولا از روش‌هایی استفاده می‌شود که بخشی از اطلاعات از بین می‌رود اما به قدری نیست که تشخیص صوت اصلی ناممکن شود. مثلا در شبکه تلفن خانگی یک رنج خاص فرکانسی در نظر گرفته می‌شود اما ما در اینجا تمامی فرکانس‌ها را دخیل کردیم.

۵ گام پنجم

با استفاده از دستور huffmandict که یک الفبا و یک توزیع احتمال می‌گیرد، دیکشنری هافمن را می‌سازیم.

برای تولید الفبا و کوانتیزه کردن ورودی از دستور quantiz استفاده کردیم.

پس از تولید الفبا، با استفاده از دستور huffmanenco رشته کد شده را تولید و درون فایل صوتی جدید ذخیره می‌کنیم.

```
89  %% Step 5: Huffman encoding compress
90  dict = huffmandict(alphabet, p);
91  encoded = huffmanenco(quantized, dict);
92
93  % Save
94  audiowrite('voice_compressed.wav', encoded, Fs);
95
96  % Compare
97  org_bin = de2bi(quantized);
98  org_len = numel(org_bin);
99  comp_bin = de2bi(encoded);
100 comp_len = numel(comp_bin);
101 fprintf("Original file bit length: %d\n", org_len);
102 fprintf("Compressed file bit length: %d\n", comp_len);
```

شکل ۶: کد مربوط به گام پنجم

سپس طول بیتی که برای حالت اولیه و حالت فشرده سازی شده لازم است را محاسبه می‌کنیم. خروجی به صورت زیر است:

Original file bit length: 349848
Compressed file bit length: 181565

همانطور که مشاهده می‌شود تعداد بیت مورد نیاز پس از فشرده سازی کمتر شده است. اما پس از ذخیره سازی متوجه می‌شویم حجم فایل بیشتر شده است. دلیل آن این است که فایل اولیه خود مکانیزمی برای فشرده سازی خود داشته است که ما با باز کردن آن و تبدیل آن به بیت آن فشرده سازی اولیه را از بین برده ایم.

زمان لازم برای انتقال با لینک توصیف شده به صورت زیر به دست می‌آید:

$$Original = 349848/(64 * 1000) = 5.46[s]$$

$$Compressed = 181565/(64 * 1000) = 2.83[s]$$

که تقریبا این زمان نصف شده است.

نکات

۱. لینک پروژه لاتک درون فایل LaTex Link.txt موجود است.

```
Command Window
Symbol rate: 22050 [symbols/second]
Shape of audio file:
   87462       1
Number of symbols: 87462
Each symbol has 1 channels.
Time: 3.96653
-----------------------------------------------
Middle symbol type: double
Middle symbol value: 0.008942
-----------------------------------------------
Min symbol value: -0.928558
Max symbol value: 0.978363
Mean of symbols: -0.000313
-----------------------------------------------
Sum of probabilities: 1
-----------------------------------------------
Entropy: 3.74716
-----------------------------------------------
Original file bit length: 349848
Compressed file bit length: 181565
>>
```

شکل ۷: تصویر خروجی‌ها