WOJSKOWA AKADEMIA TECHNICZNA

LABORATORIUM

SYSTEMY DIALOGOWE

Stopień, imię i nazwisko prowadzącego	Imię oraz nazwisko słuchacza	Grupa szkoleniowa
dr inż. Andrzej Wiśniewski	***Grzegorz Pol***	***I7G2S1***

Data wykonania ćwiczenia
30.01.2010 r.

SPRAWOZDANIE
Z
PRACY LABORATORYJNEJ
NR 1

1. Założenia słownika

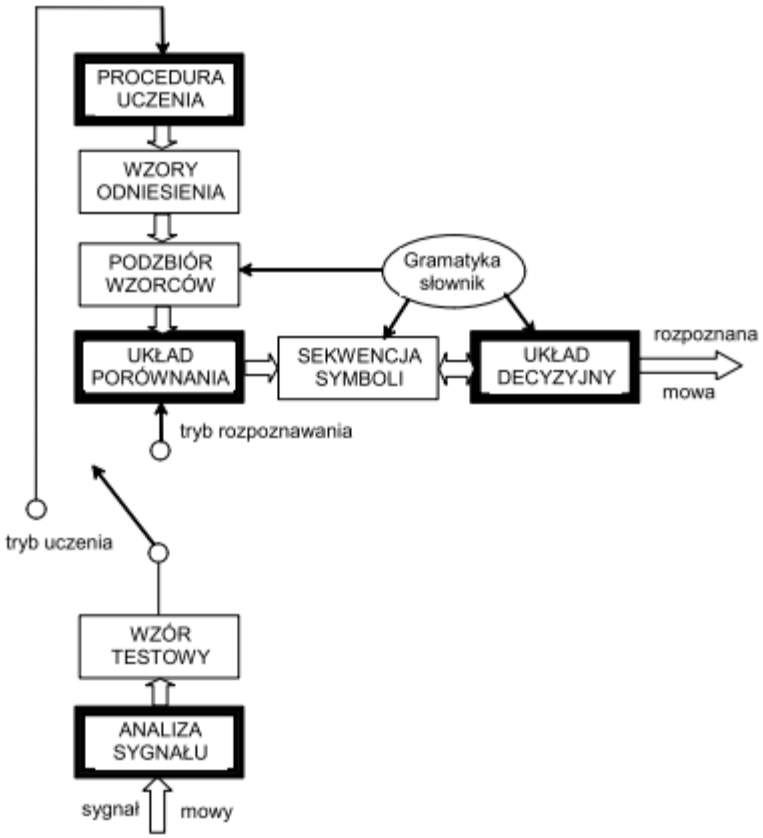
Moja aplikacja zgodnie z zadaniem posiada mały słownik wyrazów, które jest w stanie rozpoznać. Są to nazwy pięciu zwierząt:

- ✓ żyrafa
- ✓ aligator
- ✓ jeż
- ✓ kot
- ✓ orangutan

Jednostka fonetyczną jest jak widać całe słowo.

2. Struktura systemu oraz zdefiniowanie wzorca oraz teoretyczne założenia

Mój program został skonstruowany zgodnie ze standardową strukturą systemu automatycznego rozpoznawania mowy metodą rozpoznawania wzorców. Jego schemat prezentuje poniżej:



Wzorzec słów został zdefiniowany jako wektor, przetrzymujący średnie, charakterystyczne wielkości dla danego zbioru wyrazów. Użyte charakterystyczne wartości dla każdego słowa to:

Wartość:	Wzór
Energia sygnału:	$E = \sum_{n=1}^{N} z(n)^2$
Maksymalna wartość:	$Max = max[z(n)]$

Ilość przejść przez zero: $$PPZ = \frac{1}{2}\sum_{n=2}^{N}(sign(z(n)) - sign(z(n-1)))$$

oraz długość sygnału. Podsumowując wzór mojego wektora wygląda w następujący sposób:

WZORZEC = [maksymalna wartość, ilość przejść przez zero, energia sygnału, długość sygnału]

Poszczególne wartości wzorca zostały obliczone za pomocą poniższego kodu:

```
M=max(abs(z));
E=z*z';
L=length(z);
z2=z(2:L);
z1=z(1:L-1);
PPZ = sum(abs(sign(z2)-sign(z1)))/2;
WZTtemp = [M PPZ E L];
```

3. Sposób przygotowania danych

Na początku pracy formułuję słownik powyższych rozpoznawalnych wyrazów. Kolejnym krokiem było utworzenie 15 plików *.waw dla każdego powyższego słowa. Są to nagrania mono o częstotliwości 16 kHz o rozmiarze 16 bitów na próbkę. Po tej czynności zająłem się skonstruowaniem filtru usuwającego składową stała oraz ciszę z początku i końca sygnału. Realizuję to za pomocą poniższego kodu:

```
clc; clear all; close all;

n=15;
m=5;
A = ['zyrafa';'jez';'kot';'aligator';'orangutan'];
path='probki';
path2 = [path, 'czyste'];

Esuma=zeros(1,4);
Etemp=zeros(1,4);
Ewszystkie=zeros(5,4);
%Ewszystkie(2:2,:)

for i=1:m
    for j=1:n
        [z,fs]=wavread(strcat(path,'/',deblank(A(i,:)),int2str(j),'.wav'));
        %obciecie sygnalu:
        dlugosc_sygnalu = length(z);
        prog = 0.025;
        x=1;
        %obciecie poczatkowej ciszy
        for y=1:dlugosc_sygnalu/2;
            if abs(z(y))<prog;
                x=x+1;
            else
                break;
            end;
        end;
        q=length(z);
        %obciecie koncowej ciszy
        while q>dlugosc_sygnalu/2;
            q=q-1;
            if abs(z(q))>prog;
                break;
            end;
        end;
        d=z(x:q);
        M=max(abs(d));
```

```
        E=0;
        L=length(d);
        d2=d(2:L);
        d1=d(1:L-1);
        PPZ = sum(abs(sign(d2)-sign(d1)))/2;
        E = [M PPZ E L];
        Esuma=Esuma+E;
        %zapis plikow:
        wavwrite(d, fs, 16,strcat(path2,'/',deblank(A(i,:)),'_',int2str(j),'.wav'));

    end;
    Ewszystkie(i:i,:)=Esuma/n;
end;
Ewszystkie;
```

Jak da się wywnioskować z powyższego kodu ucinam sygnał jeżeli wartość jest mniejsza niż 0.025.

4. Procedura uczenia.

Następnym krokiem było przygotowanie wzorców – czyli etap nauki naszego programu. Program pobiera przefiltrowane wcześniej przez nas pliki *.waw i wylicza wzorce zgodnie ze wzorem przedstawionym dwa podpunkty wyżej. Aplikacja z naszych piętnastu wzorców wylicza jeden, który jest średnia arytmetyczną:

$$\text{wzorzec}_{\text{globalny}} = \frac{\text{wzorzec}_1 + \text{wzorzec}_2 + \text{wzorzec}_3 + \ldots + \text{wzorzec}_{13} + \text{wzorzec}_{14} + \text{wzorzec}_{15}}{15}$$

Realizuje to poniższy kod:

```
clc; clear all; close all;

n=15;
m=5;
b=4; %elementy we wzorcu
A = ['zyrafa';'jez';'kot';'aligator';'orangutan'];
path='probki';
path2 = [path, 'czyste'];

Esuma=zeros(1,b);
Etemp=zeros(1,b);
Etest=zeros(1,b);
Ewszystkie=zeros(m,b);

%tworzenie wzorcow
for i=1:m
        Esuma=zeros(1,b);
    for j=1:n
        [z,fs]=wavread(strcat(path2,'/',deblank(A(i,:)),'_',int2str(j),'.wav'));
        M=max(abs(z));
        E=z*z';
        L=length(z);
        z2=z(2:L);
        z1=z(1:L-1);
        PPZ = sum(abs(sign(z2)-sign(z1)))/2;
        Etemp = [M PPZ E L];
        Esuma=Esuma+Etemp;
    end;
    Ewszystkie(i,:)=Esuma/(n-5);
end;
Ewszystkie;
```

5. Procedura rozpoznawania

Ostatnią procedurą mojego programu jest rozpoznawanie. Zasada jej działania oparta jest na dwóch macierzach:

- ✓ Macierzy znanej – zbudowanej na podstawie naszego ciągu testowego
- ✓ Macierzy rozpoznania – macierzy do której wpisujemy rozpoznane słowa

Przebieg procedury wygląda następująco: wartości poszczególnych składowych dźwięku podanego przez nas na wejściu są porównywane z wartościami składowych we wzorcu każdego ze słów. Rozpoznanym wyrazem zostaje ten którego różnica jego parametrów z parametrami wzorca określonego wyrazu jest najmniejsza. Realizuje to poniższy kod:

```
%procedura rozpoznawania:
i=1;
j=1;
Rozpoznana=zeros(m,n);
Znana=[1,1,1,1,1,1,1,1,1,1,1,1,1,1,1;
       2,2,2,2,2,2,2,2,2,2,2,2,2,2,2;
       3,3,3,3,3,3,3,3,3,3,3,3,3,3,3;
       4,4,4,4,4,4,4,4,4,4,4,4,4,4,4;
       5,5,5,5,5,5,5,5,5,5,5,5,5,5,5];
liczbawszystkichwyrazow=75;
liczbapoprawnierozpoznanych=0;
prawdopodobienstwo =0;
ilosc_bledow=0;
for i=1:m
    for j=1:n
        [w,fs]=wavread(strcat(path2,'/',deblank(A(i,:)),'_',int2str(j),'.wav'));
        M=max(abs(w));
        E=w*w';
        L=length(w);
        w2=w(2:L);
        w1=w(1:L-1);
        PPZ = sum(abs(sign(w2)-sign(w1)))/2;
        Etest = [M PPZ E L];
        Ewynikporownania2=zeros(m,b); %maciez z porownaniem
        k=zeros(m,1); %maciez sum wierszy
        for s=1:m %petla po wszystkich wzorcach
            Ewynikporownania2(s,:)=abs(Ewszystkie(s,:)-Etest);
            k(s)=sum(Ewynikporownania2(s,:));
        end;
        [C,I]=min(k);
        Rozpoznana(i,j)=I;
    end;
end;
```

Na koniec zliczamy poprawnie rozpoznane wyrazy i dzielimy je przez liczbę wszystkich wyrazów (5 x 15 = 75) mnożąc razy 100 w celu otrzymania wyniku w procentach. Kod realizujący:

```
for y=1:m
    for t=1:n
        if Rozpoznana(y,t)==Znana(y,t);
            liczbapoprawnierozpoznanych=liczbapoprawnierozpoznanych+1;
        else ilosc_bledow=ilosc_bledow+1;
        end;
    end;
end;
prawdopodobienstwo=(liczbapoprawnierozpoznanych/liczbawszystkichwyrazow)*100
ilosc_bledow
```

6. Wnioski

Mój program osiągnął wg mnie dość dobry wynik rozpoznania wynoszący 80% wykorzystując tylko 4 wskaźniki. W tabelce poniżej przedstawiam wynik jaki otrzymałem:

macierz znana

1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
2	2	2	2	2	2	2	2	2	2	2	2	2	2	2
3	3	3	3	3	3	3	3	3	3	3	3	3	3	3
4	4	4	4	4	4	4	4	4	4	4	4	4	4	4
5	5	5	5	5	5	5	5	5	5	5	5	5	5	5

macierz rozpoznana

1	1	1	1	1	1	1	1	**2**	1	**3**	1	1	1	1
2	**1**	2	2	2	2	**5**	2	2	**4**	2	2	2	2	2
3	3	3	3	**1**	3	3	3	**2**	3	3	3	3	3	**5**
4	4	4	4	4	**5**	4	**3**	4	4	4	**3**	4	4	4
2	5	**1**	5	5	5	5	5	5	**3**	5	5	5	5	**2**

Ilość błędów: 15
Prawdopodobieństwo: 80%

Ilość błędów wskazuje, że co 5 wyraz został przez mój program, źle rozpoznany. Wynika to z niedoskonałości wzoru, decydującego o rozpoznaniu wyrazu, gdyż uważam, że np. maksymalna wartość jest mniej ważna od długości (słowo można donośniej wymówić, ale czas jego trwania nie zmieni się w znaczącym stopniu). Na potwierdzenie mojej tezy podam, że w zaawansowanych programach rozpoznawania mowy na podstawie wzorców możemy znaleźć zdecydowanie inne wskaźniki (np. FFT, LPC, funkcje autokorelacji).

Inną przyczyną nie 100% wyniku rozpoznania jest słabej jakości mikrofon wbudowany w mój komputer i problem wyszukania odpowiedniej wartości progu przy docinaniu dla wszystkich wyrazów.