1-

2- # A Spoken Language Interface For PC Control

3- Eng. Moatassem M. Abd Allah* Dr. Ismail Abd El-Ghaffar Farag** Prof. Dr. Matar Ali Matar***

4- ## ABSTRACT

This paper presents a study and implementation of a spoken language PC interface as an interactive man-machine communication facility that allows the voice control of IBM compatible PCs. This incorporates an acquisition subsystem, preprocessing and LPC-based feature extraction subsystem as well as a DTW pattern-matching and classification subsystem. The System is implemented using personal computer PC/AT-486 with 50 MHz processor speed. This results in recognition rate for a such specified system amounting from 92% to 98% for single candidate and third candidate choices respectively. The system works in near real time operations as it takes about 25 seconds to recognize the uttered command. The integrated system compares well to the commercially available counterparts.

I. INTRODUCTION

Human interfaces to computers are still residing an active area of research. Of these interfaces, the keyboard is considered the basic one for editing control as well as text information. Problems of correct typing on one hand and the typing speed on the other hand have urged the research for alternative means for its replacement or at least "resizing" its monopoly. Pointing devices (as mice) are developed and supporting software with icons are now widely used. Two other means are being developed and operationally tested, namely, the pen for handwriting texts, commands as well as drawings and the spoken language interface which is the subject of this work. This latter facility enjoys the following advantages :-

(1) High input speed : as the rate of information input by speech is about three times faster than keyboard input and eight times faster than inputting characters by hand.
(2) No training needed : since the generation of speech is a very natural human action.
(3) Parallel processing with other information as production of speech works quite well in conjunction with actions of the hands and feet or with visual perception of information.
(4) Simple and economical input sensor ; namely the microphone.
(5) coping with unusual circumstances of darkness, blindness, handicap.

The implementation under consideration is characterized by :-

- Limited number of required control commands. This fits the system into the category of isolated word or at most connected word recognition system.
- Presence of interfering noise as the input device has ability to pick up other sounds in the environment, making accurate recognition more difficult.
- Occurrence of band-width limitations and spectral distortions introduced by the acoustic environment.

5- *Graduate Student; Computers Department, **Staff Members; Computers Department, *** Professor; Department of Avionics, Military Technical College, Cairo, Egypt.

In this work, an isolated word recognizer for a set of vocabulary used in computer control is implemented. This incorporates an acquisition subsystem, preprocessing and feature extraction subsystem as well as a pattern-matching and classification subsystem. The acquisition subsystem acts as interfacing part, to convert the analog spoken words by the user to a digital form. The obtained speech data will be processed to reduce its noise part, through detecting the start and the end of the actual word (end point detection)[1]. As features characterizing the speech signal the linear predictive coefficients (LPC) will be extracted. LPC are used along with a dynamic time warping algorithm for pattern-matching and classification purposes.

In section II, system implementation is considered. Various modules are conceived to solve the problems of : data acquisition, preprocessing (end point detection), LPC extraction, dictionary building as well as pattern matching and classification. Section III considers the system assessment and comparison with its commercial counterparts. Section IV concludes the work and addresses the question of future work extension.

II. SYSTEM IMPLEMENTATION

The system block diagram is shown in Fig. 1. It comprises modules for Data Acquisition, Preprocessing, Feature Extraction, Dictionary Building-Up and Pattern Matching and Classification.

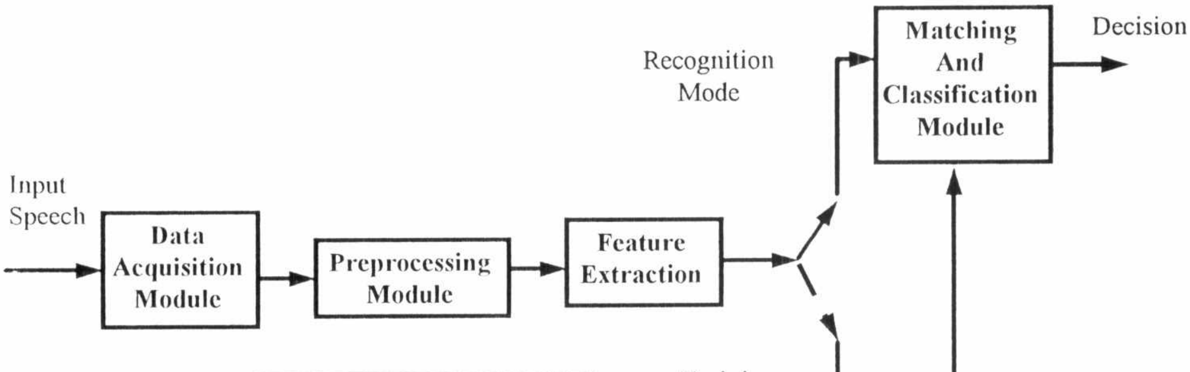

Fig. 1 System Block Diagram

II.1 Data Acquisition Module

The purpose of data acquisition is the capture of a speech utterance into the computer RAM . Words are uttered by the speaker into a microphone connected to an amplifier and a Low Pass Filter (LPF) with cutoff frequency of 5KHz. A 10KHz sampling rate is used along with an 8-bit A/D converter to digitize the amplified filtered speech signals. This A/D converter was clocked externally. Software, written in Turbo Pascal language, was used to read the output of the A/D converter and store it in the RAM of computer.

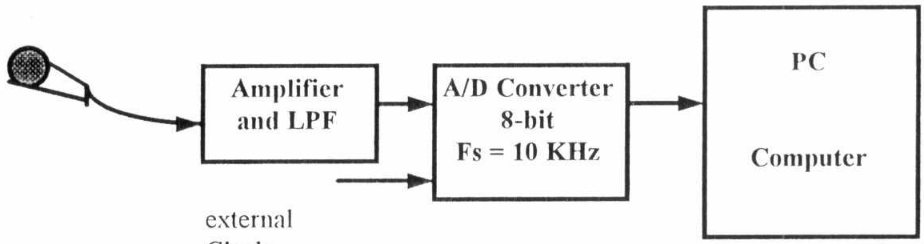

Fig. 2 Block Diagram of Data Acquisition Subsystem

II.2 Preprocessing Module

An important problem in speech processing is to detect the presence of speech in a background of noise and accurately determine the beginning and end of an utterance (end points) so that the amount of processing of speech data can be kept to a minimum. A simple technique for the end point detection (EPD) is to use the energy and Zero-Crossing Rate (ZCR) to distinguish between noise and speech[2]. Briefly, the idea is to determine experimentally two energy threshold levels: the lower threshold (ITL) and the upper threshold (ITU) as well as one ZCR threshold.

An estimate of the beginning or end of an utterance is obtained when the signal energy exceeds ITL and continues to increase until it exceeds ITU before falling below ITL. This number of frames is updated as the ZCR exceeds the set ZCR threshold, hence the beginning point is set back to the first point (in time) at which the threshold was exceeded.

The pseudo code of the used algorithm is as follows (see Fig.3):-

Variables

Input

X	Speech Data
n	Length of Speech Data
W	Hamming Window

Output

SP	Start Point
EP	End Point

Internal

E	Energy Function
S	Temporal Array
i,j,m,l	Counter

Line Procedure **End_Point**

```
i = 1
l = 0
For j = 0  to  n  do
Begin
    S[i] = X[j]
    m = X[i] * X[i+1]
    If (m <= 0) then
        INC(ZFRAME[l])
    EndIf
    INC(i)
    If (I = 100) then
        E[l] = 0
        For  m = 0  to  99  do
        Begin
            E[l] = E[l] + ABS(S[m]) * W[m]
        End_m
        INC(l)
I = 0

    EndIf
End_j
DEC(l)
IMAX = E[0]
IMIN = E[0]
For  i = 1  to  l  do
Begin
    If  (E[i] > IMAX) then
        IMAX = E[i]
    EndIf
End-i
For I = 1  downto  0 do
 Begin
 If (E[i] >= RTH) then
     EP = i
     Break
 EndIf
 End_i
 Zmean = Zframe[0]
 For i = 1 to 9 do
 Begin
 Zmean=Zmean/ZFRAME[i]
 End-i
 Zmean = Zmean /10
 Zvar = 0
 For I = 0 to 9 do
 Begin
     Zvar = Zvar + ((ZFRAME[i] -
 Zmean) * (ZFRAME[i] - Zmean))
 End-i
Zvar = Zvar / 10
Zstd = SQRT(Zvar)
IZCR = ROUND(2*Zstd) + Zmean
j = 0
For I = SP-1 downto 0 do
Begin
    If  (ZFRAME[i] >= IZCR) then
            INC(j)
    EndIf
```

```
    If (E[i] < IMIN) then
        IMIN = E[i]
    EndIf
End_j
I1 = 0.03 * (IMAX - IMIN) + IMIN
I2 = 4 * IMIN
If (I1 < I2) then
    ITL = I1
Else
    ITL = I2
EndIf
ITU = 5 * ITL
RTH = (ITL + ITU) / 2
SP = 0
EP = 1
For I = 1  to  1  do
Begin
    If (E[i] >= RTH) then
        SP = I
        Break
    EndIf
```

```
    If (j >= 0) then
            SP = i
            j = 0
    EndIf
End_i
j = 0
For i = EP to 1 do
Begin
    If (ZFRAME[i] >= IZCR) then
            INC(j)
    EndIf
    If (j >= 3) then
            SP = i
            j = 0
    EndIf
End_i
End EndPoint
```

Fig. 3 Algorithm For End Point Detection

II.3 Feature Extraction Module

This module extracts the desired features over equal segments. of duration 20 msec.. Thus, approximately 50 speech samples are used to derive the linear prediction coefficients via correlation coefficients. The module is divided into the following procedures :-

Autocorrelation procedure.

Linear predictive coding procedure [3].

It should be noted that these procedures work sequentially without user intervention.

II.3.1 Autocorrelation Procedure

It extracts the correlation between samples in each of the K frames used in this processing. The length of each frame is defined by the length of the window (M ms).

II.3.2 Linear Predictive Coding (LPC) Procedure

This procedure produces the LPC parameters of the speech signal model comprising 12 predictor coefficients for each time frame of an utterance. The pseudo code of the Levinson-Durbin algorithm is as follows (see Fig.4) :-

Variables

Inputs

x	input time series
n	length of x
iorder	order of predictor

Outputs

a	predictor coefficients
rc	reflection coefficients
r0	power in x
ierr	outcome code
	0 - normal
	1 - .zero power in input

```
                    2 - prediction error power ≤ 0
     Internal
        r            array of autocorrelations
        pe           prediction error
        akk          most recent reflection coefficient
line Procedure Levinson_Durbin
// Compute Autocorrelation //
For i = 0 to iorder do
Begin
    sum = 0
    For k = 1 to n - i do
        sum = sum + x[k] * x[k+i]
      r[i] = sum
End_i
r0 = r[0]
If (r0 = 0) then
    ierr = 1
     exit
EndIf
// Compute Reflection Coeff. & Predictor Coeff. //
pe = r0
a[0] = 1
For k = 1 to iorder do
Begin
     // New Reflection Coefficient //
      sum = 0
       For i = 1 to k do
        sum = sum - a[k-i] * r[i]
       akk = sum / pe
rc[k] = akk
 // New Predictor Coefficients //
a[k] = akk
For i = 1 to k / 2 do
Begin
     ai = a[i]
     aj = a[k-i]
     ai = ai + akk * aj
     a(k-i) = aj + akk * ai
End_i
 //New Prediction Error //
pe = pe * (1 - akk ** 2)
If (pe < 0) then
  ierr = 2 //Non+ve prediction-error return//
   exit
EndIf
End_k
ierr = 0 //Normal return //
 End Levinsion_Durbin
```

Fig. 4 Pseudo code For Levinson Durbin Algorithm

II.4 Dictionary Building Up (Construction)

In template matching speech recognition systems, it is assumed that reference templates for the vocabulary are available in the machine. Usually the templates must be constructed from the speech of the intended user and thus a training session is required for enrollment of each new user in which versions of all the vocabulary words are given. If the same user regularly uses the machine, the templates can be stored in some back-up memory for using each time the recognizer is switched on and enrollment then merely consists of re-loading the correct set of stored templates. Such approach is adopted in speaker-dependent recognizers.

A recognizer may be required to be used by a wide variety of speakers without re-training (speaker-independent recognizers). In this case, the available templates must be representative of the speech of any of the expected speakers, and so several templates must be provided for each word. The usual method of constructing such templates is to collect many utterances of vocabulary words spoken by a wide variety of speakers. The use of all these different pronunciations of a word as references can increase the vocabulary size to such an extent as to be unfeasible. However, for special application (when the required vocabulary is small) this method results in high recognition performance.

Another procedure for constructing speaker-independent templates is to collect a wide variety of pronunciations for the word as mentioned above.

The resultant word patterns are then aligned with each other so that each one can be represented by a single point in the same multidimensional feature-by-time space. These points are then clustered into a fairly small number of groups that are chosen to represent the spread of the word patterns in the space. Templates are then made to represent the average properties in each cluster[4]. The problem with such training techniques is that, during recognition, the clustered templates are not always likely to match the utterances of a particular speaker. Consequently, there is a much greater chance that one of the several alternative templates available for competing words will yield a smaller cumulative distance.

II.5 Pattern Matching And Classification Module

In the recognition session of a recognizer, the unknown input word is compared in turn with all the templates in the store, and the nearest neighbour is assumed to be the correct one. The problem with this process is that the words to be compared are not all of the same length.
A technique that is capable of matching one word with another on the basis of optimum non-linear time scale distortion has to be used. Dynamic Programming (DP) is applied to attain this objective and referred to as Dynamic Time Warping (DTW)[5,6].The pseudo code of the algorithm is as follows (see Fig.5) :-

```
Variable
   rmax       max. no. of references.
   mframe     no. of frame in each reference.
   DISTCOST   distance. measure between test pattern and references pattern.
   Input
      REFWORD      all references words
      TESTWORD     test word
      REFAction    action of each word
   Output
      Run          Action
line Procedure Dynamic Time Warping
//  Initialize Final Cost And Action Will Be Taken //
   Cost = HUGE_VALUE
   Action = No_Action
   // Starting To Calculate Distance //
   For  k = 1  to rmax  do
   Begin
      // Initialize Distance Cost Arrary //
      For  i = 1  to  N  do
      Begin
         For  j = 1   to  M  do
            DISTCOST[i,j] = HUGE_VALUE
      End_i
      DISTCAOST[0,0] = 0
      For  i = 1  to  N    do
      Begin
         For  j = 1   to   M  do
         Begin
            If  ((ABS(i-j) > TRAC)  then
               Continue
            EndIf
            sum = 0
            For  P = 1   to  LPCCoef    do
               sum = sum + ((REFWORD[k,i,P] - TESWORD[j,P])
                  * (REFWORD[k,i,P] - TESTWORD[j,P])
            MINM = DISTCOST[i-1,j]
            If  (MINM > DISTCOST[i-1,j-1]) then
               MINM = DISTCOST[i-1,j-1]
            EndIf
            If  (MINM > DISTCOST[i,j-1]) then
               MINM = DISTCOST[i,j-1]
            EndIf
            DISTCOST[i,j]= SQRT(sum)+MINM
         End_j
   End_i
   If  (Cost > DISTCOST[N,M])  then
      Cost = DISTCOST[N.M]
      Action = REFAction[k]
   EndIf
   End_k
   Run Action
End Dynamic_Time_Warping
```

Fig. 5 Pseudo Code For Dynamic Time Warping Algorithm

III. SYSTEM ASSESSMENT

Conceived to be a spoken language PC-interface, the DOS commands were selected for validation of the implemented PC-interface. A system was configured around an IBM PC with 486 processor (but 386 processor

would suffice) with 4MByte RAM, VGA Card and VGA monitor. An Add-on voice card (COVOX) employing an 8 bit A/D converter, sampling rate up to 20KHz and buffer size of 64KByte is used in conjunction with a microphone or a combination of microphone-speaker set. Moreover a Start / stop (wake-up / sleep) facility is added to push the system in a "sleep" state that terminates by "Wake Up" command.
The implemented system was subjected to recognition tests under laboratory conditions and under several options of windowing and segmentation in LPC extraction and backtracking in DTW. Tests were run at 10KHz sampling rate, 20 msec. frame duration with neither windowing nor overlapping.

III.1. Anatomy of the DOS Commands and the Dictionary Structure

DOS commands can be classified according to several viewpoints : user's viewpoint, system viewpoint and programmer's viewpoint. We consider only the user's viewpoint and go ahead for further classifications for the sake of simplifying the dictionary constructions and searching over.

From the set of DOS commands those ones (either dedicated for communication with the system or for Input_Output management) are generally single word commands. These commands are exemplified by: Date, Time, Ver, Exit, Cls, More.

Commands dedicated for working with disks and directories or for setting up the system environment or working with assembler are generally two utterance commands. This subset is exemplified by: Chkdsk, Label, Vol, Sys, Country, Lastdrive, Buffers, Md, Rd.

The remaining subset is dedicated for file management and in using batch files are generally multiword commands. This subset is exemplified by : Copy, Rename, Recover, Restore, Xcopy.

For constructing the dictionary, a list of records has been established. Each record has three fields as shown in Fig. 6. The first field contains the command features, while the second is the associated DOS command. The third field contains a pointer to a linked list that contains the other command words attributes and switches. The list actually represents the syntax of the command. This attribute will have value NULL if the command is a single word.

In practice, the system will get a spoken word and extract its features. The word features are compared with all templates stored in dictionary. The nearest neighbour is assumed to be the correct one. The DOS command associated with the selected entry in the dictionary will be the one to be executed. The pointer field of this entry in dictionary is checked for NULL value. If this is true, the command is a single word and the system will start the execution. The system has been tested for this category of commands as Time, Ver, CLS, ... etc.

When the command is a multiword, the system will start parsing the entire command by traversing the associated linked list. The system will recognize that in some of the commands the whole entire list is optional. In that case the system will wait for certain time and then execute the default command if there is no other spoken words. The work for parsing of multiword command is not complete yet.

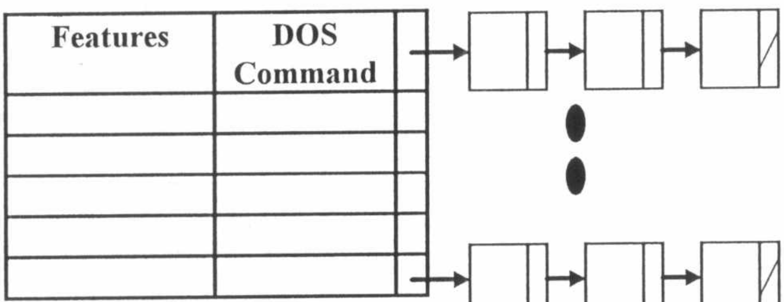

Fig. 6 Dictionary Structure

III.2 Performance Scores

The system was tested for the following list of DOS commands :- CLS, Command, DosKey, DosShell, Date, Dir, Time, Tree, Vsafe, Edit, Label, Ver.

Ten versions of each command were spoken and one is used as reference. Results have shown average recognition error of 8% if just the first candidate word is accepted and reduces to 4% and 2% if the second or third candidate respectively are accepted.

The obtained results compare well to those obtained for airline vocabulary[7], where LPC and DTW are used. For the system given in[4], a 10% recognition error are obtained if the first candidate is accepted and 4.6% if the second candidate whereas 2.9% are obtain for the third candidate. The present system and that of reference[7] give definitely lower performance than those tuned for digits only[8] when error approaches 0.1%.

III.3 Comparison with Commercial Counterparts

Table 1 is used to position the implemented system with respect to five commercial counterparts operating as isolated word recognizers. These system are : Rover 1.0 ,Microsoft Sound System 2.0,ExecuVoice, VoiceServer 2.0 and Creative VoiceAssist[9]. Other commercially available voice control (or Navigation) Systems as well as dictation and development systems are in [10].

Table 1 Voice Control Packages Comparison

	Rover	Sound System 2.0	ExecuVoice 1.0	Creative VoiceAssist	VoiceServer 2.0	System
Display						
Issue Disk	3.5 & 5.25	3.5	3.5	3.5	3.5	3.5 & 5.25
Min. Required	H.D, 386, 4MRAM Windows 3.1	H.D, 386, 4MRAM Windows 3.1	H.D, 386, 4MRAM Windows 3.1	H.D, 386, 4MRAM Windows 3.1	H.D, 386, 4MRAM Windows 3.1	H.D, 386, 4MRAM
Optional H/W	8MRAM	8MRAM	8MRAM	8MRAM	None	None
Sleep Command	✓	✓	✓	✓	✓	✓
Automatic Sleep	×	×	×	×	×	×
Multiple user	✓	✓	✓	✓	✓	✓
Application Specific	✓	✓	✓	✓	✓	✓
Automatic Extraction	×	✓	×	✓	✓	×
Context Sensitive	×	×	×	×	✓	✓
Learning While In Use	×	×	×	×	✓	✓
User Define Commands	✓	✓	✓	×	✓	✓

Built-in Macro Recorder	✓	×	×	×	✓	×
Commands per user	Unlimited	Unlimited	220	6,752	1,000	DOS
Active In One Time	Unlimited	Unlimited	64	256	500	300
For	Relatively Simple To Use	Good Microphone Design Worked	Very Easy To Use	Include Proper Macro Recorder	Good Voice Recognition Dedicated	Easy To Use

IV. CONCLUSIONS AND FUTURE WORK

This paper gives a study and implementation of a spoken language PC interface as an interactive man-machine communication facility that allows the voice control of IBM compatible PCs. The System is implemented using personal computer PC/AT-486 with 50 MHz processor speed. This results in recognition rate for a such specified system amounting from 92% to 98% for single candidate and third candidate choices respectively. The system works in near real time as it takes about 25 seconds to recognize the uttered command. Improvement of such system (and its commercial counterparts) should be an outcome of concurrent "digging" deeper in the multidisciplinary domain of signal processing, algorithms, VLSI, computers..etc.

Issues for future research directions are given in a recent review paper[11]. For the specific task of voice control (or Navigation), the following points seem to be the immediate extension of this work :

(a) Use of multiword DOS commands that may entails techniques of connected word recognition.

(b) Classification algorithms based on Neural Networks (NN).

(c) Comparison of DTW, (NN) and Hidden Markov models under conditions of extended LPC feature vector (through use of energy as well as Zero Crossing Rate).

REFERENCES

[1] Wilson J., "Voice Processing - The New Revolution", *Proceedings of the International Conference on Applications Of Speech Recognition In Industrial And Military Environments*, London, July, 1986.

[2] L. R. Rabiner and S.E. Levinson, Isolated and Connected Word Recognition_Theroy and Selected Applications, *IEEE Trans. Communications*, COM-29(5): 621-659, May 1981.

[3] L. R. Rabiner, and Ronald W. Schafer, Digital Processing Of Speech Signals, *Prentice-Hall Inc.*, 1978.

[4] Rabiner L. R., S. E. Levinson and M. M. Sondhi, "On the Application of Vector Quantization and Hidden Markov Models to Speaker-Independent, Isolated Word Recognition", *American Telephone and Telegraph Company*, Bell System, September, 1983.

[5] L. Breiman, J.H. Freedman, R.A. Olshen, and C.J. Stone, Classification and Regression Trees, *Wadsworth & Brooks*, Monterey, CA, 1984.

[6] C. Myers, L.R. Rabiner, and A. E. Rosenberg, Performance tradeoffs in dynamic time warping algorithms for isolated word recognition, *IEEE Trans. Acoustics*, Speech, Signal Proc., ASSP-28(6): 623-635, December 1980.

[7] Rabiner L. R., K. C. Pan and F. K. Soong, "On the Performance of Isolated Word Speech Recognizers Using Vector Quantization and Temporal Energy Contours," *AT&T Bell Labs Tech.*, February, 1984.

[8] L.R. Rabiner and B.H. Juang, "Fundamentals of Speech Recognition", *Prentice-Hall Inc.*, Chap. 9, 1993.

[9] M.Nicholson, "Talk to Me; On Test :Voice Recognition Systems", *PC_Plus Magazine*, pp.276-280, March 1994.

[10] Mat Beard, Ed. "Voice Recognition" , Pc Magazine (Middle, Near East), Vol. 1, No. 3, pp. 88-93, Feb. 1995.

[11] M. Bush et al, "The Challenge of Spoken Language Systems : Research Dirctions For the Nineties", IEEE Trans. Speech and Audio Processing, Vol. 3, No. 1, pp. 1-21, Jan. 1995.