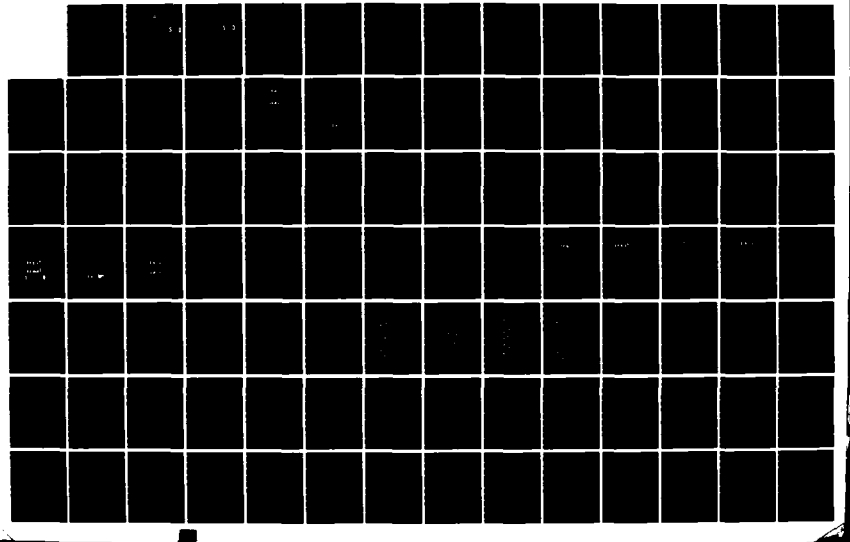
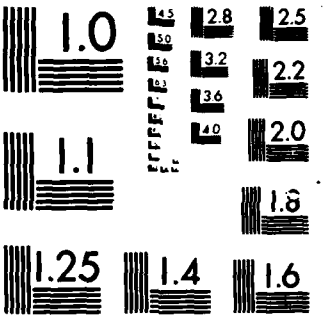
AD-A163 938 WORD SEGMENTATION ALGORITHM BASED ON RECOGNITION OF LETTER FEATURES(U) AIR FORCE INST OF TECH WRIGHT-PATTERSON AFB OH SCHOOL OF ENGINEERING 1/2

UNCLASSIFIED D V SOBOTA DEC 85 AFIT/GE/ENG/85D-43 F/G 9/2 NL





WORD SEGMENTATION ALGORITHM BASED ON RECOGNITION OF LETTER FEATURES

THESIS

David V. Sobota
First Lieutenant, USAF

AFIT/GE/ENG/85D-43

DEPARTMENT OF THE AIR FORCE
AIR UNIVERSITY
AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

(1)

AFIT/GE/ENG/85D-43

WORD SEGMENTATION ALGORITHM BASED ON RECOGNITION OF LETTER FEATURES

THESIS

David V. Sobota
First Lieutenant, USAF

AFIT/GE/ENG/85D-43

AFIT/GE/ENG/85D-43

WORD SEGMENTATION ALGORITHM BASED ON RECOGNITION OF LETTER FEATURES

THESIS

Presented to the Faculty of the School of Engineering

of the Air Force Institute of Technology

Air University

In Partial Fullfillment of the

Requirements for the Degree of

Master of Science in Electrical Engineering

David V. Sobota, B.S.E.E.

First Lieutenant, USAF

December 1985

PREFACE

The following thesis study was motivated by a desire to produce a more reliable segmentation algorithm in order to incorporate past thesis research using Fourier Transform techniques to recognize individual letters in a word.

I wish to express my gratitude to my thesis advisor, Dr. Matthew Kabrisky, for giving me the support and freedom to pursue this thesis topic. Also, special thanks to Capt. King and Dan Zambon for their dedication and patience in keeping the computer system running and helping me become familiar with it.

Most of all, I would like thank my fiancee Jude for her love and understanding while getting through the worst of times and for being the reason for enjoying the best of times here at the Air Force Institute of Technology.

TABLE OF CONTENTS

List of Figures

ABSTRACT

The purpose of this ~~study~~ thesis was to develop a computer algorithm to segment letters in a word so that pattern recognition techniques such as two dimensional Fourier Transforms could be used to recognize the individual letters. This study did not intend to cover various pattern recognition techniques but, as the algorithm developed letter features were recognized to gather information or clues on adjacent touching letters to decide on possible segmentation locations. This study was limited to the segmentation of lower case letters in the English alphabet excluding stylized print and italic print. Lower case print was chosen because it represents the worst case task for a segmentation algorithm. Two adjacent lower case letters often look like a third lower case letter. However, in upper case letters similiar occurrences are rare. Hand printed touching letters were selected to demonstrate the validity of the algorithm.

I. INTRODUCTION

1.1 BACKGROUND

Since the advent of high speed computers, users have had to enter hardcopy text into data files by physically retyping the text on a keyboard. For secretaries and other computer users, this retyping requires many hours of tedious manual labor at a time when labor is becoming more expensive while the cost of electronically using and storing that data is rapidly declining (7 : 105). The ideal solution would be a computerized reading machine or optical character reader (OCR) which electronically scans and digitizes text and converts the text into ASCII data to be stored on magnetic disk or manipulated by computer software.

OCR's have many applications besides just text storage. The U.S. Post Office has been trying to find a fast and reliable reading machine for sorting mail since the early 1960's. The Foreign Technology Division, at Wright Patterson A.F.B., Ohio, has been using a slow, error prone reading machine as a front end processor for computer translation of Russian text (4). A reading machine could also be used to preserve this nation's libraries of books. Presently, air pollution is deteriorating all books made of wood pulp. It would be impossible to reprint all these books and many would be lost for future generations if the technology to store or preserve them is not developed. Using present technology, it is already possible to store digitized pictures of each page on high density storage

disks but, the use of a reading machine using an entropy efficient code would greatly reduce the storage space and cost enabling entire libraries to be stored on a few disks. By reducing the data bits needed to store a book, this technology would make it feasible to quickly call up any book desired on a home computer without ever going to a library. Another application of OCR machines is aid to the blind (7 : 110). Kurzweil Computer Products introduced the Reading Machine in 1976 which scans text and reads it aloud. Although the error rate may be intolerable for storing text, the human listener can still understand letter errors and even whole word errors due to the context of the words and sentences, respectively (4). More will be said later in the text about the Kurzweil OCR machines.

1.2 PROBLEM

There are already many reading machines on the market. However, most of them can only read special typefaces such as MICR (Magnetic Ink Character Recognition) type, OCR-B, or OCR-A. These types have additional features which help the reading machine distinquish similiar characters. There are also several machines designed to handle large quantities of type written text. These machines, including models made by companies such as DEST, TOTEC, and Hendrix, read by doing template matching of known fonts and rely on the uniform width and spacing of type written text. Only the Kurzweil 4000 machine uses an Artificial Intelligence

variable width fonts found in books, magazines, and papers. However, the Kurzweil machine is only used for large quantities of the same type of text because the machine has to be "taught" and produces intolerable errors until the system has "learned."

The reading machine problem is difficult. Most machines have no problem reading the uniformly spaced text typed neatly on high quality white paper. In fact, most template matching reading machines only need half of the template's digital information to correctly recognize a character. The problem occurs when the letters in the text have nonuniform width and uneven spacing. The letters are also often overlapping, or touching each other. These variable width or proportionally spaced characters, as they are often called, confuse most reading machines. The width of letters can vary in range from one to four units of length. In most books, the letter "m" is four times as wide as the letter "i". Even proportional print characters could be read if the template of each letter was known before hand. But, most text is printed in varying fonts and in varying pitch. In these cases, template matching fails and a more sophisticated approach has to be used (7 : 106-108).

Noise is another dominant factor that makes the reading machine problem difficult. Type written characters are printed very neatly with adjacent characters separated from each other. A window can be drawn around each typed

character without including adjacent characters. In contrast, characters in books and newspapers are often touching or overlapping and a window cannot be draw around each character without including some of the adjacent characters. In addition, breaks in letters or dark spots near letters occur due to insufficient ink or poor printing plates. Also, threshold effects and noise produce random white and black spots in the image when the text is digitized for computer processing. All these types of noise not only interfere with attempts to recognize the letters but intefere with attempts to separate or segment the individual letters from each other.

1.3 SCOPE

The goal of this study was to develop a computer algorithm to segment letters in a word so that pattern recognition techiniques using either Fourier Transform or feature recognition could recognize the separate individual letters. This study did not intend to cover various pattern recognition techniques, but as the algorithm evolved feature recognition was used to gather information or clues on neighboring letters to decide on possible segmentation locations. This study was limited to the segmentation of lower case English alphabet excluding stylized print and italic characters. Lower case print was chosen because it represents a more difficult problem for a segmention algorithm. For example, in upper case print, two adjacent

capital letters rarely look like a third capital letter except possibly in the case of EI looking like B or VV looking like a W. However, in lower case print these problems and others are quite common and will be discussed later in the Theoretical Development. Handprinted, touching, lower case letters were selected in an attempt to create a worst case task for algorithm testing. Hand printed letter were also used to simulate the roughness of edges and the filling in of gaps due to the effects of digitization and noise.

1.4 Assumptions

The following list of assumptions were made because of time constraints in order to spend maximum time on the segmentation problem. Assumptions 3 and 4 were assumed to exist for all fonts after examining 22 standard type faces (2 : 189-215).

1. Noise filtering to remove isolated dark spots and appropriate thresholding has already been performed to produce a digitized binary (black and white) image of each word.

2. The probability that random noise occurring in isolated areas to cause one letter to look like another is negligible.

3. The width of vertical lines in printed fonts is approximately constant.

4. The height of small letters such as a,e,c,o, etc.

is approximately constant in printed fonts.

5. The lines of printed text will be placed horizontally and the letters will be place right side up.

6. Vertical lines in printed fonts are approximately perpendicular to the horizontal line of print.

1.5 APPROACH

Research proceeded from the idea that, except for six letters, English printed letters have a at least one region of vertical continuous lines at the edge of each letter. The exceptions are s,v,w,x,y, and z. Of these six exceptions, the last five occur with low probability. Taking advantage of this characteristic, letter features such as vertical continuous lines were used as the primary segmentation clues to predict where one letter ends and another begins. Additional letter features such as height, holes, valleys, and arches were also detected to get clues as to what letters were present on either side of the vertical continuous line. These segmentation clues along with any detected spaces in a word were used as inputs to a final segmentation decision algorithm.

1.6 EQUIPMENT AND MATERIAL

The equipment and material support was provided by the Signal Processing Lab at the Air Force Institute of Technology (AFIT). Most of the work was performed using the Data General Nova 2 minicomputer, the Data General Eclipse S/250 minicomputer, and the Octek 2000 Image Analyzer.

II. THEORETICAL DEVELOPMENT

2.1 CHOOSING THE APPROACH

The segmentation of English printed letters is a problem that most papers and articles describing reading machines and character recognition do not adequatedly address. Three theses by Bentkowski (1), Shum (5) and Simmons (6) at AFIT have tried to tackle this problem. All three tried to correlate the FT of the image inside a sliding window with the FT of a letter template being searched. However, in all cases, using upper case letters, correlation peaks occurred on letters that did not resemble the search template. The sliding window approach has several potential problems especially with lower case letters. Picking a window size could be a problem with letters varying in width by as much as 4 to 1 units of length. False correlations could occur because two adjacent letters often look like a third letter. For example, "o" and "l" might look like a "d". False correlations could also occur because shapes of one letter often exist in other letters. For example, the shape of "n" exist in "h" and "m". Also, the shape of "o" exist in "b","d","p", and "q". Figure 2. below shows other cases where letter shapes exist in more than one letter. For these reasons, the sliding window correlation approach could only work if extensive error correction postprocessing and "intelligence" is added.

c	-	o	b	e	d	p	q
l	-	b	d	k	h		
o	-	o	b	d	p	q	
n	-	h	m				
r	-	h	m	n			

Figure 2.1 Letter Shapes Which Occur in Other Letters

An algorithm to segment English letters should take full advantage of the characteristics inherent in the shape of the letters. This will not only enable the algorithm to be simpler but enable it to run quicker. At present the Kurzweil 4000, the only machine that can read the proportional print in books, requires a minicomputer and megabytes of memory. Although, it is proprietary information, the segmentation algorithm probably takes up a large percentage of the memory. The segmentation problem is complex. It is almost necessary to know what the individual letters are in a word to decide where to segment them. Take for example the case of a "on" letter combination. See Figure 2.2 below. Depending on where this combination is segmented, the machine might read a "cn", "on", "on", or even "cin" if noise adds a dot above the vertical line.

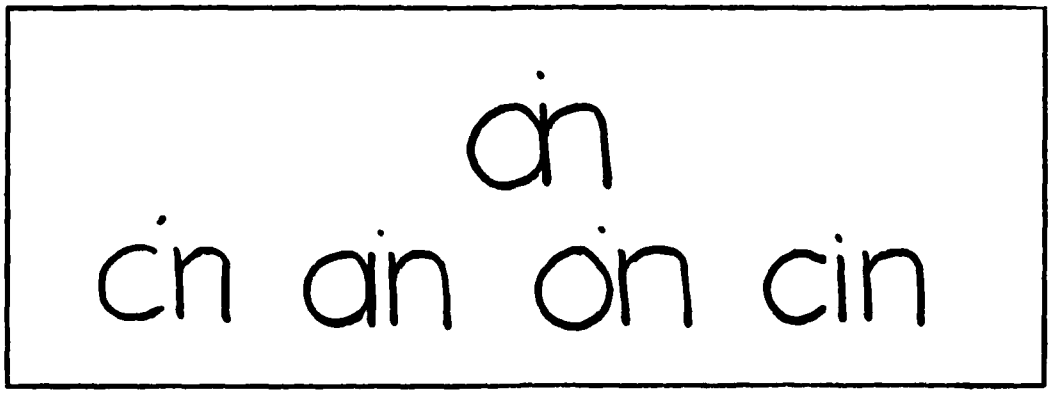


Figure 2.2 Example of Ambiguous Letter Combination

2.2 LETTER FEATURES AS SEGMENTATION CLUES

As mentioned in the "Approach", letter features such as vertical continuous lines (VCL) were used as the primary segmentation clues from which other segmentation clues on either side of the VCL were examined. VCL is defined as the greatest number of black pixels adjacent to other black pixels along a vertical line. The detection of VCL regions was chosen because of inherent noise immunity. The probability that random black pixels will line up in a vertical line is negligible. VCL regions occur along vertical lines in a letter and at the edge of curves due to the "squaring" effect of digitization and the thickness of the curved line. Figure 2.3 below shows the VCL's detected in the word " the ".

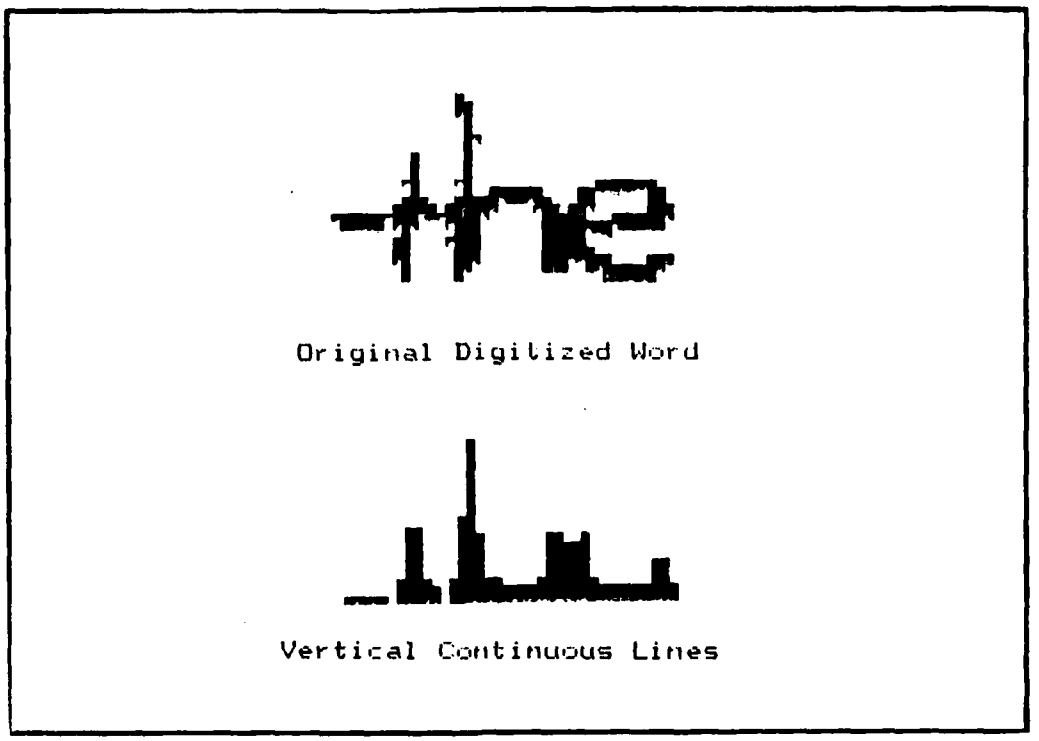


Figure 2.3 VCL's in the word "the"

Of the 676 two letter combinations, only 99 combinations do not have a VCL at or near the point where the two letters should be segmented. These cases are segmented using other letter features. Appendix A list these 99 cases.

In addition to VCL's, other letter features were also detected. The detection of holes (O), two stacked holes (8), arches (A), and valleys (U) were used to determine what letter combinations were possible on either side of the VCL. O-region is defined as consecutive columns where there is a concentration of black pixels followed by a space of continuously white pixels along a vertical line and then another concentration of black pixels. O-regions exist

in a,b,c,d,e,f,g,o,p, and q. U-region is defined as consecutive columns where there is a space along a vertical line followed by a concentration of black pixels which occur lower than top of small letters (TS) such as a,e,o,u, etc. U-regions exist in u,v,w and y. A-region is defined as consecutive columns where there is a concentration of black pixels along a vertical line followed by a space. In a A-region, the concentration of black pixels has to occur above the bottom of small letters (BS). A-regions exist in h,m,n,r, and t. 8-region is defined as consecutive columns where there are two O-regions, one above the other. 8-regions exist in letters a,e,g,s,and z. These features were also chosen for their noise immunity characteristics. The probability that random noise produce the same concentration of black pixels and continuous white pixels in consecutive columns of a digitized image is also very small. Figure 2.4 below shows the O, U, A, and 8-regions detected in "zebra."

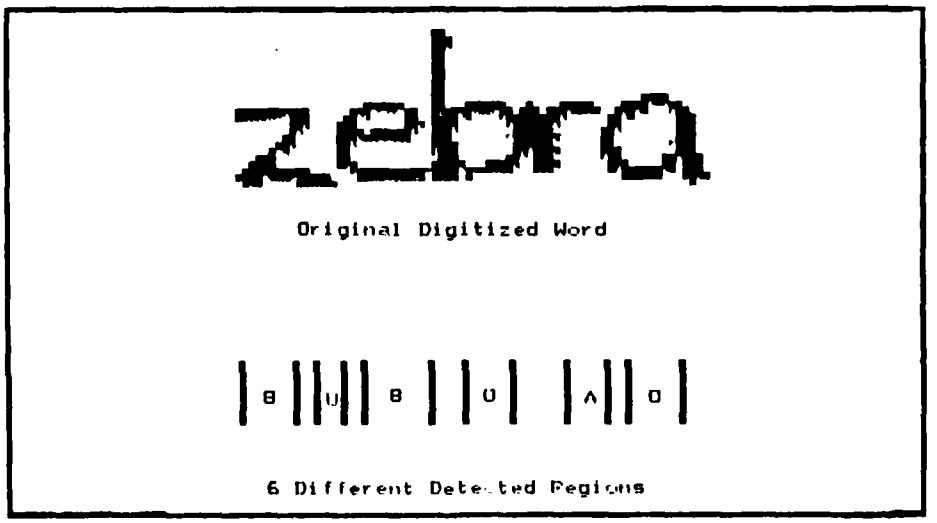


Figure 2.4 O,U,A, and 8 regions detected in "zebra"

For the 99 cases where VCL's do not exist at the segmentation point, a change from one region to another was used as a segmentation clue. For example, a change from an O-region to a U-region is used as a segmentation clue. This segmentation rule will handle all but twelve cases (ea, es, ez, vv, vw, wv, ww, yy, wy, yw, vy, yv, yy). Additional segmentation rules can handle eight of these twelve cases. The first three of these twelve cases is segmented at the middle of the letter combination since the width of the 8-region is too large to be caused by a single letter and e,a,s, and z are approximately the same width in all fonts. (Examples of common printed fonts are shown in Appendix B.) The last five cases (wy, yw, vy, yv, yy) are segmented between the U-regions by noting there is no letter in the English alphabet which has two U-regions and which extends below the line of print such as the letter "y". The last four cases (vv, ww, wv, vw) can only be distinquished if the context of the letters or the type of font used is known. If a gap or space does not exist between these letter combinations not even a human reader would be able to distinquish these letter combinations from false letters or letter combinations. (See Figure 2.5) Postprocessing of the recognized letters and the questionable letters needs to be used to consider probabilities of letter combinations and context to derive a " best guess " for these letter combinations.

Although the last four letter combinations, mentioned above, rarely occur in the English language, there are two more likely letter combinations which could also not be distinquished unless postprocessing is also used. The letter combinations "rn" and "cl" could not be distinquished from "m" and "d", respectively, unless a space exist between the letters. The Figure 2.5 below is a list of ambiguous letter combinations which could not be distinquished without a space between the letters.

rn - m
cl - d
vv - w
vw - vvv - wv
ww - wvv - vvw - vwv - wvv - vvvv

Figure 2.5 Indistinquishable Touching Letter Combinations

In addition to O,A,U,8 and VCL regions, the width and height of VCL regions were also used as segmentation clues. Take for example, the letter combination "db". If a gap is not detected between these two letters, one could not determine whether it was a "cb" or a "db" except by noticing that the width of the VCL region in the middle of the letter combination is too wide for only one letter. Using the width of the VCL regions as a segmentation clue takes advantage of the fact that for all fonts the width of vertical lines is nearly constant for all letters in the font (1). (See Appendix B for examples of printed fonts.) However, there are exceptions in handwritten print due to

font (1). (See Appendix B for examples of printed fonts.) However, there are exceptions in handwritten print due to the varying positions of the writing instrument. This algorithm will assume a near constant width for the VCL regions of each letter. If this assumption was not made, Figure 2.6 list additional letter combinations which could not be distinguished, by a human reader, unless a space existed between the letters.

ln	-	h
nn	-	m
lo	-	b
ol	-	d

Figure 2.6 Letter Combinations Distinquishable Using Width of VCL Regions

Besides constant width of VCL regions, height of VCL regions were also used as a segmentation clues. Take for example the letter combination "rh". This letter combination consist of two A regions on either side of a VCL region. If a gap is not detected, the same features would be detected in "rn" , "nn", "rln", or "m". By noting the greater height of the middle VCL region and the nonexistence of a letter that looks like the mirror image of an "h" , the algorithm can not only segment this letter combination but also determine that it is due to "rh" or "rln". Using width of VCL regions, the algorithm can further determine that the letter combinations is "rh" and not "rln".

2.3 NONEXISTENT LETTER FEATURES AS SEGMENTATION CLUES

advantage of the nonexistence of features in a single letter of the English alphabet. The following list of nonexistent letter features were considered in the final segmentation decision algorithm. The long vertical lines in these letters could be due to adjacent letters such as b,h,d,k,l,g,p, or q. In all these cases, the segmentation point is in between the A,O,U, or 8 regions and the longer vertical line.

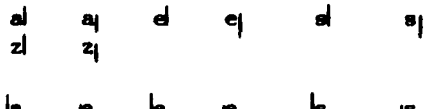
Nonexistent Features	Example Cases
A-TALL VCL	
TALL VCL-A	
TALL VCL-O	
8-TALL VCL	
TALL VCL-8	
U-TALL VCL	
TALL VCL-U	

Figure 2.7 Cases Segmented Using Nonexistent Features

One other feature that does not exist in a single letter is a vertical line that is longer than the length of tall letters such as the letter "1". Using this information, letters are segmented where a VCL region exists that is too tall. These cases exist when a tall letter and a letter which has a vertical line below the line of print are adjacent to each other. These cases are listed in Figure 2.8 below. This segmentation clue would be valid

even if the width of the VCL regions was not wide enough for two adjacent letters. In all these cases, the segmentation point is the middle of the tall VCL region.

Nonexistent Letter Feature	Example Cases
VCL region taller than "l"	gb gh gk gl qb qh qk ql jb jh jk jl dp lp

Figure 2.8 Cases Segmented Using Length of VCL

2.4 DETECT PROBLEM LETTERS

Two individual letter detection programs were also written to reduce the chances of error. A "t" detector was written because the letter "t" is the second most frequent letter in the English language. In addition, the previous segmentation clues did not adequately distinquish between VCL and A regions in a "t" to those found in "h","n","m", and "r". Because the "t" detector searches for the cross shape in a "t", it also detects the presence of an "f". Also, an "i" detector was written because the letter "i" occurs with relatively high probability and the VCL in an "i" could not be distinquished from the VCL in other letters. For example, a "ci" and a "ri" combination with no space in between the letters could look like "a" and "n", respectively. The "i" detector, which also detects the letter "j", is used to segment all letter combinations involving "i" or "j". The "t" detector is used to segment

all letter combinations involving "t" or "f".

2.5 FINAL DECISION RULES

General decision rules have been mentioned above along with specific examples. In deriving the final decision rules for all the two letter combinations consisting of A,O,U, or 8 regions on either side of a VCL region, each case had to be considered individually to derive more specific segmentation decision rules. Additional cases are those involving a A,O,U, or 8 region on either side of the letter "l" or tall VCL. The first two pages list more general decision rules based on various adjacent regions and width of VCL. The rest of the decision rules are more specific and consider nonexistent letter features. All these cases are listed on the following pages. The letter combinations involving a "i", "j", "f" or "t" are omitted because these cases are segmented using the "i" detector and "t" detector. In listing the different cases, two distinct shapes for the letters a, g, and y were recognized. These different shapes are shown in Figure 2.9 below.

a - ɑ

g - ɡ

y - ɣ

Figure 2.9 Letters Commonly Printed in Two Distinct Ways

Also, the following abbreviations were used in explaining the segmentation clues for the different letter combinations listed on the following pages:

thick - VCL region too wide to be caused by one single letter

thin - VCL region just wide enough for one letter

PVC - (previous VCL) VCL region preceding the VCL region be considered for segmentation

NVC - (next VCL) VCL region immediately following VCL region being consider for segmentation

TS - position which marks top of small letters such as a,c,e, or o

BS - position which marks bottom of small letters such as a,c,e, or o

TALL-VCL - VCL region whose height is taller than the distance from TS to BS such as the letter "l"

G - 8-region which extends below the line of print as in g

A/U - represents a A and U-region which overlap each other

U/A - represents a U and A-region which overlap each other

DECISION RULES

FEATURES	CASES							SEGMENTATION DECISION
8-VCL-0	ab	ac	ad	ap	ao	aq		middle VCL
thick	ga	gb	gc	gd	gp	go	gq	
8-VCL-0	ea	eb	ec	ed	ep	eo	eq	before VCL
thin	sa	sb	sc	sd	sp	so	sq	
	ga	gb	gc	gd	gp	go	gq	
	za	zb	zc	zd	zp	zo	zq	
O-VCL-0	aa	ab	ac	ad	ap	ao	aq	middle VCL
thick	ba	bb	bc	bd	bp	bo	bq	
	da	db	dc	dd	dp	do	dq	
	oa	ob	oc	od	op	oo	oq	
	pa	pb	pc	pd	pp	po	pq	
	qa	qb	qc	qd	qq	qo	qq	
O-VCL-0	ca	cb	cc	cd	cq	co	cq	before VCL
thin	ka	kb	kc	kd	kq	ko	kq	
	za	zb	zc	zd	zq	zo	zq	
U-VCL-0	ua	ub	uc	ud	uo	up	uq	middle VCL
thick	ya	yb	yc	yd	yo	yp	yq	
U-VCL-0	va	vb	vc	vd	vo	vp	vq	before VCL
thin	wa	wb	wc	wd	wo	wp	wq	
	ya	yb	yc	yd	yo	yp	yq	
A-VCL-0	ra	rb	rc	rd	ro	rp	rq	before VCL
thin	va	vb	vc	vd	vo	vp	vq	
	wa	wb	wc	wd	wo	wp	wq	
	ya	yb	yc	yd	yo	yp	yq	
A-VCL-0	t							do nothing
thin								
A-VCL-0	ha	hb	hc	hd	ho	hp	hq	middle VCL
thick	ma	mb	mc	md	mo	mp	mq	

8-VCL-A	eh	em	en	er	before VCL
thin	sh	sm	sn	sr	
	zh	zm	zn	zr	
8-VCL-A	ah	am	an	ar	middle VCL
thick	gh	gm	gn	gr	
	g^h	g^m	g^n	g^r	
A-VCL-A	rh	rm	rn	rr	before VCL
thin	vh	vm	vn	vr	
	wh	wm	wn	wr	
	yh	ym	yn	yr	
A-VCL-A	hh	hm	hn	hr	middle VCL
thick	mh	mm	mn	mr	
	nh	nm	nn	nr	
A-VCL-A	m and some t's				do nothing
thin					
U-VCL-A	uh	um	un	ur	middle VCL
thick	y^h	y^m	y^n	y^r	
U-VCL-A	vh	vm	vn	vr	before VCL
thin	wh	wm	wn	wr	
	yh	ym	yn	yr	

The Following Cases Need Additional Segmentation Clues to Distinguish Letter Combinations Marked * :

A-VCL-8 thick	he	hg	hg	hz	middle VCL
	me	mg	mg	mz	
	ne	ng	ng	nz	
A-VCL-8 thin	hs	ms	ns		after VCL
	ha	ma	na		
	hz	mz	nz		
A-VCL-8 thin *	re	rg	rg		before VCL
O-VCL-8 thick	oe	og	og		middle VCL
	be	bg	bg		
	de	dg	dg		
	oe	og	og		
	pe	pg	pg		
	qe	qg	qg		
O-VCL-8 thin	os				after VCL
	ba	bs	bz		
	da	ds	bz		
	oa	os	bz		
	pa	ps	pz		
	qa	qs	qz		
O-VCL-8 thin *	ce	cg	cg		before VCL
8-VCL-8 thin *	aa	as	az		after VCL
	ga	gs	gz		
8-VCL-8 thick	ae	ag	ag		middle VCL
	ge				
8-VCL-8 thin	ee	eg	eg		before VCL
	se	sg	ge		
	ze	zg	zg		
U-VCL-8 thick	ue	ug	ug		middle VCL
	ye	yg			

* See pages marked " Specific Segmentation Rules "

U-VCL-8 thin	ua	us	uz				after VCL
	ya	ys	yz				
U-VCL-8 thin *	ve	vg	vg	vz			before VCL
	we	wg	wg	wz			
	ye	yg	yg	yz			
U-VCL-U thick	uu	uy	yu				middle VCL
U-VCL-U thin	uv	uw	uy				after VCL
U-VCL-U thin *	vu	wu	yu				before VCL
	vy	wy					
O-VCL-U thick	au	bu	du	ou	pu	qu	middle VCL
	ay	by	dy	oy	py	qy	
O-VCL-U thin	av	bv	dv	ov	pv	qv	after VCL
	aw	bw	dw	ow	pw	qw	
	ay	by	dy	oy	py	qy	
O-VCL-U thin *	cu	cy					before VCL
A-VCL-U thick	hu	mu	nu				middle VCL
	hy	ny	ny				
A-VCL-U thin *	ru	ry					before VCL
A-VCL-U thin	hv	mv	nv				after VCL
	hw	mw	nw				
	hy	my	ny				
8-VCL-U thick	au	gu					middle VCL
	ay						
8-VCL-U thin	eu	gu	su	zu			before VCL
	ey	gy	sy	zy			
8-VCL-U thin *	av	gv					after VCL
	aw	gw					
	ay	gy					

* See pages marked " Specific Segmentation Rules "

O-VCL-A thin *	ch	cm	cn	cr		before VCL
O-VCL-A thin	av	bv	ov	pv	qv	after VCL
	aw	bw	ow	pw	qw	
	ay	by	oy	py	qy	
O-VCL-A thick	ah	am	an	ar		middle VCL
	bh	bm	bn	br		
	dh	dm	dn	dr		
	oh	om	on	or		
	ph	pm	pn	pr		
	qh	qm	qn	qr		

* See pages marked " Specific Segmentation Rules "

SPECIFIC SEGMENTATION RULES

ADDITIONAL RULES	CASES	SEGMENTATION DECISION
A-VCL-8, thin VCL		
Height PVC = (TS-BS) 8 region extends below BS No A region before thin PVC	rg rg	before VCL
Height PVC = (TS-BS) Height VCL < (BS-TS) No A region before thin PVC If 8 region between TS,BS, no thick NVC Space 2nd edge 8 region	re	before VCL
O-VCL-8, thin VCL		
Height PVC < (BS-TS) VCL or PVC not < BS or > TS If 8 region between TS,BS, no thick NVC Space 2nd edge 8 region	ce	before VCL
Height PVC < (BS-TS) 8 region extend below BS VCL or PVC not < BS or > TS	cg cg	before VCL
8-VCL-8, thin VCL		
1st 8 region extends below BS	ga gs	after VCL
No PVC at 1st edge of 1st 8 region No thick PVC NVC 2nd edge 2nd 8 region No 2nd 8 region below BS	aa	after VCL
No PVC at 1st edge of 1st 8 region No thick PVC Space after 2nd 8 region Space 1st edge 1st 8 region No 2nd 8 region below BS	as	after VCL

U-VCL-8, thin VCL

8 region extends below BS No thick PVC A region before VCL	vg vg wg wg yg yg	before VCL
No PVC at 1st edge U region No thick PVC No thick NVC No NVC 2nd edge 8 region A region before VCL	ve we ye	before VCL

U-VCL-U, thin VCL

No thick PVC No VCL 1st edge 1st U region Thick NVC NVC 2nd edge 2nd U region A region before VCL	vu wu yu vy wy	before VCL

O-VCL-U, thin VCL

No PVC or VCL > TS or < BS NVC thick NVC 2nd edge U region NVC < BS	cu cy	before VCL

A-VCL-U, thin VCL

PVC = (BS-TS) NVC < BS NVC thick NVC 2nd edge U region	ru ry	before VCL

8-VCL-U, thin VCL

No thick NVC No NVC 2nd edge U region VCL < BS 8 region extends below BS PVC 1st edge 8 region Thick PVC	gv gw gy	after VCL
No thick PVC No 8 region extending below BS. A region after VCL Space 1st edge 8 region	av aw ay	after VCL

FEATURES	CASES INVOLVING TALL VCL	SEGMENTATION DECISION
TALL VCL-A thick	lh lm ln lr	middle VCL
TALL VCL-A thin	lv lw ly	after VCL
TALL VCL-A thin *	h	do nothing
TALL VCL-O thick	la lb lc ld lo lp lq	middle VCL
TALL VCL-U thick	lu ly	middle VCL
TALL VCL-U thin	lv lw ly	after VCL
TALL VCL-8 thick	le lg lg	middle VCL
TALL VCL-8 thin	la ls lz lg	after VCL
A-TALL VCL thick	hl ml nl	middle VCL
A-TALL VCL thin	rl vl wl yl	before VCL
O-TALL VCL thick	al bl dl ol pl ql	middle VCL
O-TALL VCL thin	cl	before VCL
U-TALL VCL thick	ul yl	middle VCL
U-TALL VCL thin	vl wl yl	before VCL
8-TALL VCL thick	al gl	middle VCL
8-TALL VCL thin	el gl sl zl	before VCL

ENHANCED DECISION RULES

FEATURES	CASES							SEGMENTATION DECISION
8-VCL-O thick	ab	ac	ad	ap	ao	aq		middle VCL
8-VCL-O thin	ea	eb	ec	ed	ep	eo	eq	before VCL
	sa	sb	sc	sd	sp	so	sq	
	za	zb	zc	zd	zp	zo	zq	
O-VCL-O thick	aa	ab	ac	ad	ap	ao	aq	middle VCL
	ba	bb	bc	bd	bp	bo	bq	
	da	db	dc	dd	dp	do	dq	
	oa	ob	oc	od	op	oo	oq	
	pa	pb	pc	pd	pp	po	pq	
	qa	qb	qc	qd	qq	qo	qq	
O-VCL-O thin	ca	cb	cc	cd	cq	co	cq	before VCL
	ka	kb	kc	kd	kq	ko	kq	
	za	zb	zc	zd	zq	zo	zq	
U-VCL-O thick	ua	ub	uc	ud	uo	up	uq	middle VCL
	ya	yb	yc	yd	yo	yp	yq	
U/A-VCL-O thin	va	vb	vc	vd	vo	vp	vq	before VCL
	wa	wb	wc	wd	wo	wp	wq	
	ya	yb	yc	yd	yo	yp	yq	
A-VCL-O thin	ra	rb	rc	rd	ro	rp	rq	before VCL
A-VCL-O thin	t							do nothing
A-VCL-O thick	ha	hb	hc	hd	ho	hp	hq	middle VCL
	ma	mb	mc	md	mo	mp	mq	

8-VCL-A thin	eh	em	en	er			before VCL
	sh	sm	sn	sr			
	zh	zm	zn	zr			
8-VCL-A thick	ah	am	an	ar			middle VCL
A-VCL-A thin	rh	rm	rn	rr			before VCL
A-VCL-A thick	hh	hm	hn	hr			middle VCL
	mh	mm	mn	mr			
	nh	nm	nn	nr			
A-VCL-A thin	m and some t's						do nothing
U-VCL-A thick	uh	um	un	ur			middle VCL
	yh	ym	yn	yr			
U/A-VCL-A thin	vh	vm	vn	vr			before VCL
	wh	wm	wn	wr			
	yh	ym	yn	yr			
A-VCL-G thick	hg	mg	ng				middle VCL
	hg	mg	ng				
A-VCL-G	rg	rg					
O-VCL-G thick	og	bg	dg	og	pg	qg	before VCL
	og	bg	dg	og	pg	qg	
O-VCL-G thin	cg	cg					before VCL
8-VCL-G thick	ag	ag					middle VCL
8-VCL-G thin	eg	eg	sg	sg	zg	zg	before VCL
G-VCL-8 thick	ge	ge					middle VCL
G-VCL-8 thin	gs	gs	gz	gz			after VCL

G-VCL-G thick or thin	gg						middle VCL
G-VCL-U thick	gu	gu	gy				middle VCL
G-VCL-U thin	gu	gy					before VCL
G-VCL-A/U thin	gv gv	gw gw	gy gy				after VCL
U-VCL-8 thick	ue ye						middle VCL
U-VCL-8 thin	ua ya	us ys	uz yz				after VCL
U/A-VCL-8 thin	ve we ye	vz wz yz					before VCL
U-VCL-U thick	uu	uy	yu				middle VCL
U-VCL-A/U thin	uv	uw	uy				after VCL
U/A-VCL-U thin	vu vy	wu wy	yu				before VCL
O-VCL-U thick	au ay	bu by	du dy	ou oy	pu py	qu qy	middle VCL
O-VCL-A/U thin	av aw ay	bv bw by	dv dw dy	ov ow oy	pv pw py	qv qw qy	after VCL
O-VCL-U thin	cu	cy					before VCL
A-VCL-U thick	hu hy	mu ny	nu ny				middle VCL
A-VCL-U thin	ru	ry					before VCL

A-VCL-A/U	hv	mv	nv			after VCL
thin	hw	mw	nw			
	hy	my	ny			
8-VCL-U	au					middle VCL
thick	ay					
8-VCL-U	eu	su	zu			before VCL
thin	ey	sy	zy			
8-VCL-A/U	av					after VCL
thin	aw					
	ay					
O-VCL-A	ch	cm	cn	cr		before VCL
thin						
O-VCL-A/U	ov	bv	ov	pv	qv	after VCL
thin	ow	bw	ow	pw	qw	
	oy	by	oy	py	qy	
O-VCL-A	oh	om	on	or		middle VCL
thick	bh	bm	bn	br		
	dh	dm	dn	dr		
	oh	om	on	or		
	ph	pm	pn	pr		
	qh	qm	qn	qr		

The Following Cases Need Additional Segmentation Clues to Distinguish Letter Combinations Marked * :

A-VCL-8 thick	he	hz		middle VCL
	me	mz		
	ne	nz		
A-VCL-8 thin	hs	ms	ns	after VCL
	ha	ma	na	
	hz	mz	nz	
A-VCL-8 thin *	re			before VCL
O-VCL-8 thick	oe	be	de	middle VCL
	oe	pe	qe	
O-VCL-8 thin	os			after VCL
	ba	bs	bz	
	da	ds	bz	
	oa	os	bz	
	pa	ps	pz	
	qa	qs	qz	
O-VCL-8 thin *	ce			before VCL
8-VCL-8 thin *	aa	as	az	after VCL
8-VCL-8 thick	ae			middle VCL
8-VCL-8 thin	ee	se	ze	before VCL

* See pages marked " Specific Segmentation Rules "

III. SOFTWARE DEVELOPMENT

The following chapter explains the software which implements the segmentation algorithm described in Chapter 2. This chapter is divided into three sections. The first section describes the software, written by AFIT faculty, to store a digitized image into a video file and to manipulate the individual pixels in that file. The second section describes MAIN.FR, the main calling program which calls all the subroutines that processes the digitized image to produce segmentation clues described in Chapter 2. The final section describes the program, DECALG.FR, which uses all the segmentation clues to decide the segmentation points of individual letters in a word.

3.1 DIGITIZATION SOFTWARE

The Octek program and the PICBUF routines, written by Captain King, were used extensively as the main tool for capturing and processing a digitized image of a word (8).

The Octek program is a user friendly program which interfaces with the hardware of the video camera and the Octek 2000 Image Analyzer board. The Octek program enables the user to threshold a digital image, to interactively position a cursor to find coordinates of significant points on the monitor screen, and store the image in a video file.

PICBUF is a library of Fortran callable subroutines which hide the system dependent details of manipulating pixels in a digital image. The PICBUF subroutines were used

to read the value of individual pixels or an entire row of pixels from a video file. The PICBUF routines were also used to write to a video file in order to display, in the form of a digital image, the results of a software program.

3.2 SEGMENTATION CLUES SOFTWARE

The following section discusses the software which determines the segmentation clues explained in Chapter 2. Assume that the thresholded (binarized) digital image of a word has already been stored in a video file and that the coordinates of the word have been recorded using the OCTEK program.

The program, MAIN.FR, prompts the user for the name of the video file and the coordinates of a word to be segmented. The program then stores the digital image of the word in a buffer using the PICBUF routines. Next, MAIN calls nine subroutines and several PICBUF routines which implement the segmentation algorithm and output the results of each subroutine into a video file for display. The following paragraphs explain each of the segmentation subroutines called by MAIN.

VRTPROJ (vertical projection) counts the number of pixels in each column of the digital word and puts the result in an output buffer for display. This routine also locates columns which do not contain black pixels. These columns indicate spaces or gaps between letters in a word. Three types of gaps are located. These "GAP's" are weighted

arbitrarily for display purposes and also to indicate confidence in the location of letter edges.

1. 1 pixel in column at local minimum (GAP of weight 5)
2. No pixels in column (GAP of weight 10)
3. No pixels in two adjacent columns (GAP of weight 15)

GAP's of weight 10 or 15 are used to locate segmentation points and also to identify adjacent letters to determine neighboring segmentation points. GAP's of weight 5 have not been incorporated in the segmentation decision algorithm. Figure 3.1 below indicates two types of gaps found in the word " mat ".

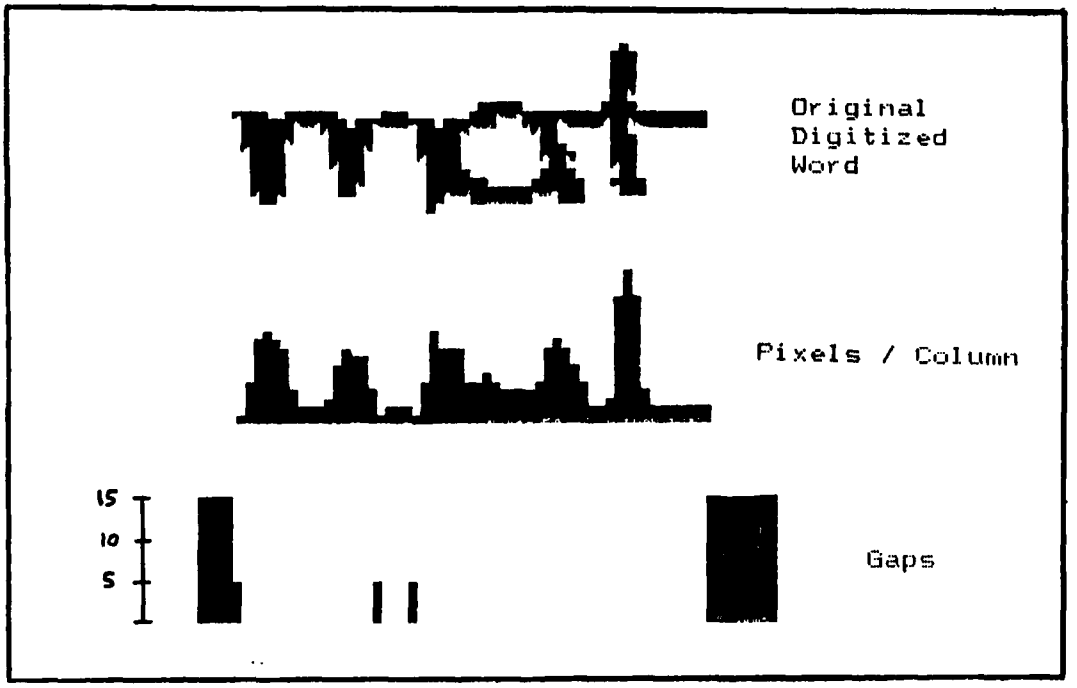


Figure 3.1 Display Showing Projection of Pixels Per Column And Two Types of Detected GAP's

HRZPROJ (horizontal projection) counts the number of

pixels in each row of the digital word and puts the result in an output buffer for display. The array HPROJ which contains the number of pixels per row is passed to FDLHT.

FDLHT (find letter height) finds the heights of various letters by taking advantage of the greater concentrations of black pixels per row that occur at the location of small letters such as a,o,u,e, etc. The various letter heights are explained below.

TT - top of tall letters such as b,d,f,h,k,l or t.
TS - top of small letters such as c,e,g,p, or o.
BS - bottom of small letters such as a,c,e, or o.
VBS - very bottom of small letters such as g,j,p, or q.

The letter height values are used in the final decision rule to determine whether a detected VCL is due to a specific letter. See Figure 3.2 below.

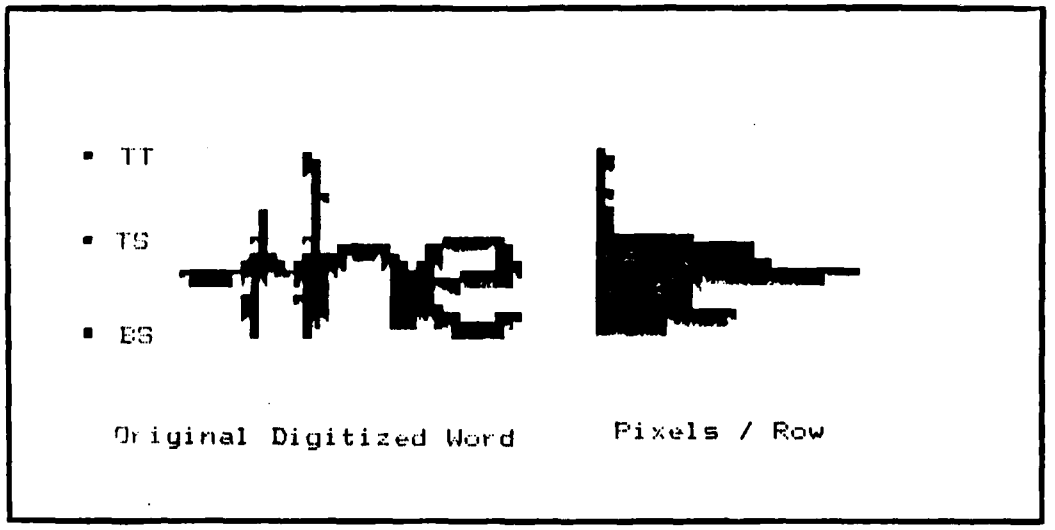


Figure 3.2 Display Showing Projection of Pixels Per Row And Three Detected Letter Heights

DVCLIN (detect continuous vertical lines) detects the longest vertical black line, which does not have a space of two or more pixels, for each column in the digital image of a word. This subroutine detects the VCL regions mentioned in Chapter 2.

EDVCL (edge of vertical continuous lines) determines the edges of significant VCL regions found in DVCLIN. Significant VCL regions are defined as those which have a height that is greater than or equal to 1/2 the distance from TS to BS. See Figure 3.3 below.

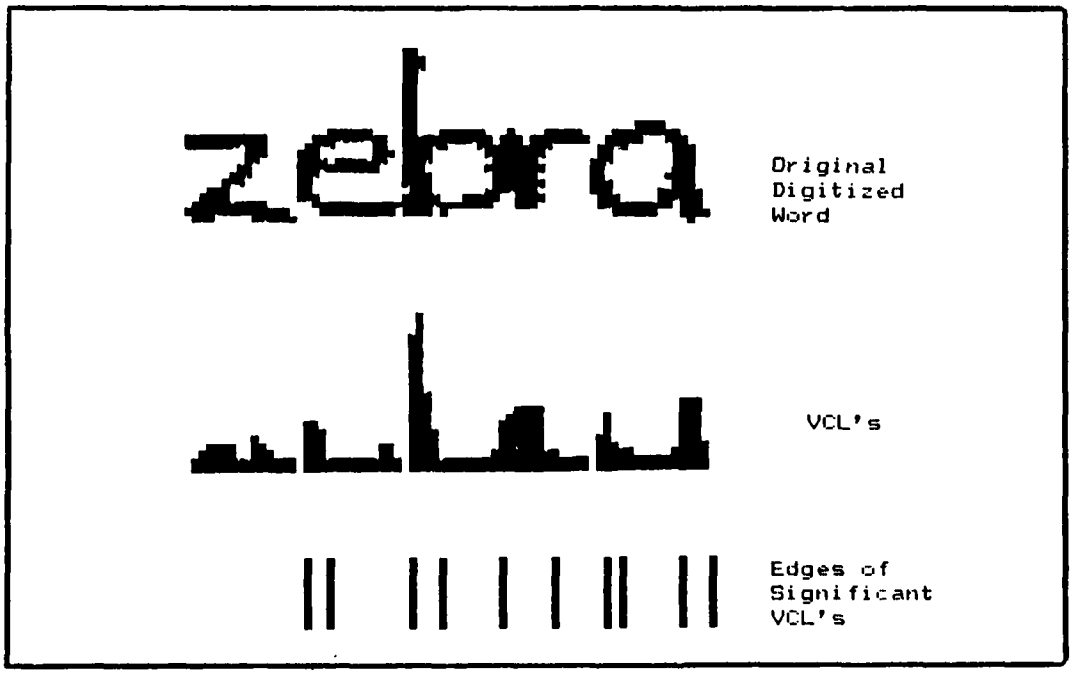


Figure 3.3 Display Showing VCL Regions And Detected Edges of VCL Regions

FINDT (find letter "t") finds the edges of the letter "t" or "f" by looking at all VCL regions which have a height

greater than the distance from TS to BS and which have a continuous horizontal line passing through the VCL region at a height near TS. FINDT also determines the edges of two adjacent "t/f" 's or the edges of a single "t/f" overlapping other letters.

IFIND ("i" find) finds the location of letters "i" or "j" by looking for a VCL region greater than or equal to the distance from TS to BS and which has a concentration of pixels above the the VCL region.

DTOUAE (detect "o","u",arches, and "e") detects the presence of O,U,A, and 8 regions defined in the Theoretical Development. This subroutine calls four additional subroutines. The first subroutine, DELOUA (delete "o","u", and arches), deletes all O,U, and A-regions which are too short or too narrow to be caused by a legitimate letter. The second subroutine called is DTEG (detect "e" or "g"). DTEG looks at all regions which meet the conditions for O-regions and also meet the conditions for 8 regions. DTEG then looks at each 8-region and O -regions and assigns it to one of six categories based on its vertical position relative to the various letter heights. These four types of O-regions and two types of 8-regions can be used to help recognize the letters in order to better determine the segmentation location. Similiarly, the last two subroutines called by DTOUAE, DTA and DTU, find two different types of A and U- regions, respectively. The edges of these regions

are put in an output buffer for display. The various types of O, 8, U, and A regions and their assigned numbers for display are given below.

Various Types of O, 8, U, and A Regions

Regions	Position	Assigned Number
O	TT - TS	1
	TT - BS	3
	TS - VBS	6
	TS - BS	9
8	TS - BS	12
	TS - VBS	15
U	TS - VBS	6
	TS - BS	9
A	TT - BS	6
	TS - BS	9

HTVCL (height of VCL) divides the VCL regions into two halves. This subroutine then finds the maximum position and mininum position of the vertical lines contained in the two halves of the VCL regions. The VCL regions are divided in half to account for cases where the touching vertical lines of two adjacent letters have two different lengths or positions as in the case of "gn" or "gh", respectively. These heights and positions of the vertical lines in the digitized image of the word are used as segmentation clues in the decision algorithm.

3.3 SEGMENTATION DECISION SOFTWARE

The last subroutine MAIN calls is DECALG which uses all the segmentation clues to decide where the letters in the

word should be segmented. DECALG gather all segmentation clues for each VCL into a one two dimension array IVCL. The height of each VCL and the presence of gaps are already stored in IVCL when HTVCL and VRTPROG are called, respectively. Another column in IVCL is used to indicate whether the VCL is either part of a "t" , near a "t" , part of a "i", part of a "v", or part of a "x". The DECALG then considers each VCL in the word individually while determining what regions are on both sides of the VCL. The program then makes a segmentation decision based on the decision rules outlined in Chapter 2. The segmentation location is either at the middle, the first edge, or the second edge of the VCL region. For those cases where the segmentation clues do not yield enough information to decide whether to segment before or after the VCL region, the individual letter recognizer would have to consider both segmentation locations and determine which one leaves legitimate letters on both sides of the segmentation point. Vertical lines are used to indicate below the word the segmentation points decided by DECALG.

IV. VALIDATION

In testing the validity of the segmentation decision algorithm, handwritten letters were used to simulate noise conditions. Time did not allow the testing of other types of print because more time was needed to perfect the decision rules and investigate more segmentation clues. In all four of the examples shown, the letters were deliberately written so that, in the digitized image of the word, each letter was not separated by a column which contained just white pixels. Also, no care was taken in writing the letters with perfectly vertical or unbroken lines. Looking at the word "the", the vertical lines in the letters "t" and "h" are slanted slightly to the right. Looking at the word "had", the vertical line in the letter "d" has two breaks in it.

The following four cases are presented to demonstrate the capabilities of the segmentation algorithm as discussed in Chapter 2. Except for the word "the", the four words were not chosen for any particular reason. The word "the" was chosen because it represents the most common word in the English language (3). Vertical lines are used to mark the segmentation points below the digitized image of the word. The height of the vertical line indicates the relative confidence of the segmentation decision. If two clues indicate a particular segmentation point a height of 10 pixels is given to the vertical line. The location of "i" or "t" is also given a 10 pixels segmentation mark. If only

one clue indicates a particular segmentation point a height of 5 pixels is given to the vertical line. In all cases the correct segmentation point was chosen based on the segmentation decision software. However, more rules need to added to the software. In the word " zebra " a U-region was found to exist at the 1st edge of the VCL in "e" which led to an incorrect segmentation location. This is easily corrected though if a U-region is required to have VCL's on both sides of the region. Alternatively, a U-region which does not have VCL's on both sides which are between TS and BS could be used as a segmentation clue since it represents an nonexistent letter feature.

Figures 4.1, 4.2, 4.3 and 4.4 show the segmentation points, the edges of the VCL regions, and the edges of the A,O,U,8 regions for the four cases. In the word "hod", a higher confidence was given to the first segmentation point because the vertical line in the "h" was determined to be greater than TS indicating that the A region was not due to an "r" or an "n" and that the VCL region between the "h" and the "o" should be segmented at the middle. If the tall vertical line of the "h" had not been detected the algorithm could not determine with confidence whether the A region was due to an "r" and segment the letter combination at the first edge of the VCL region or whether the A region was due to an "n" and segment at the middle. In the case of the words "mat" and "the", high confidence segmentation marks

were indicated for the edges of the two "t"'s detected by FINDT. Further validity testing needs to be conducted using other letter combinations considered in the theoretical development of the segmentation algorithm.

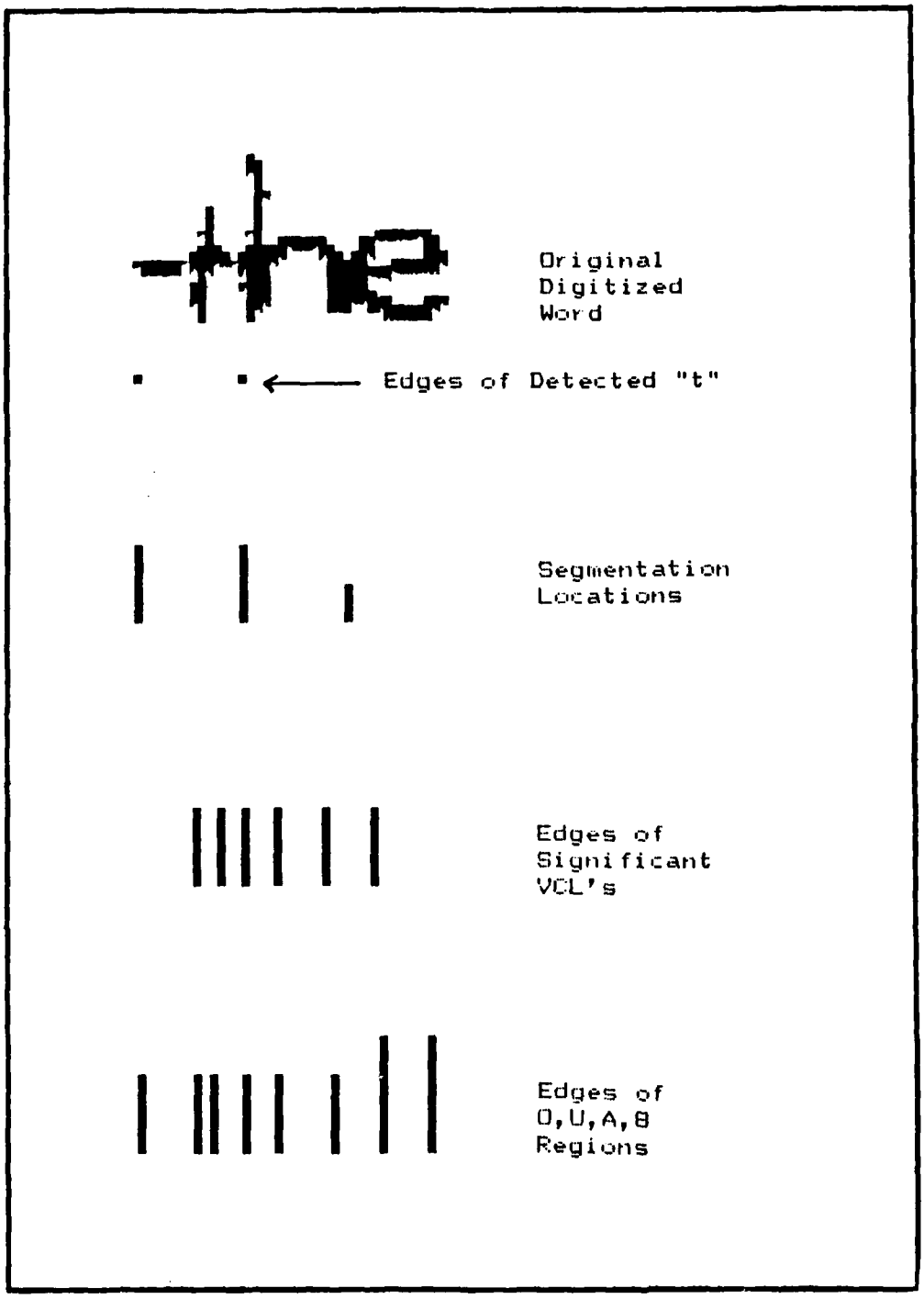


Figure 4.1 Segmentation Display Output for " the "

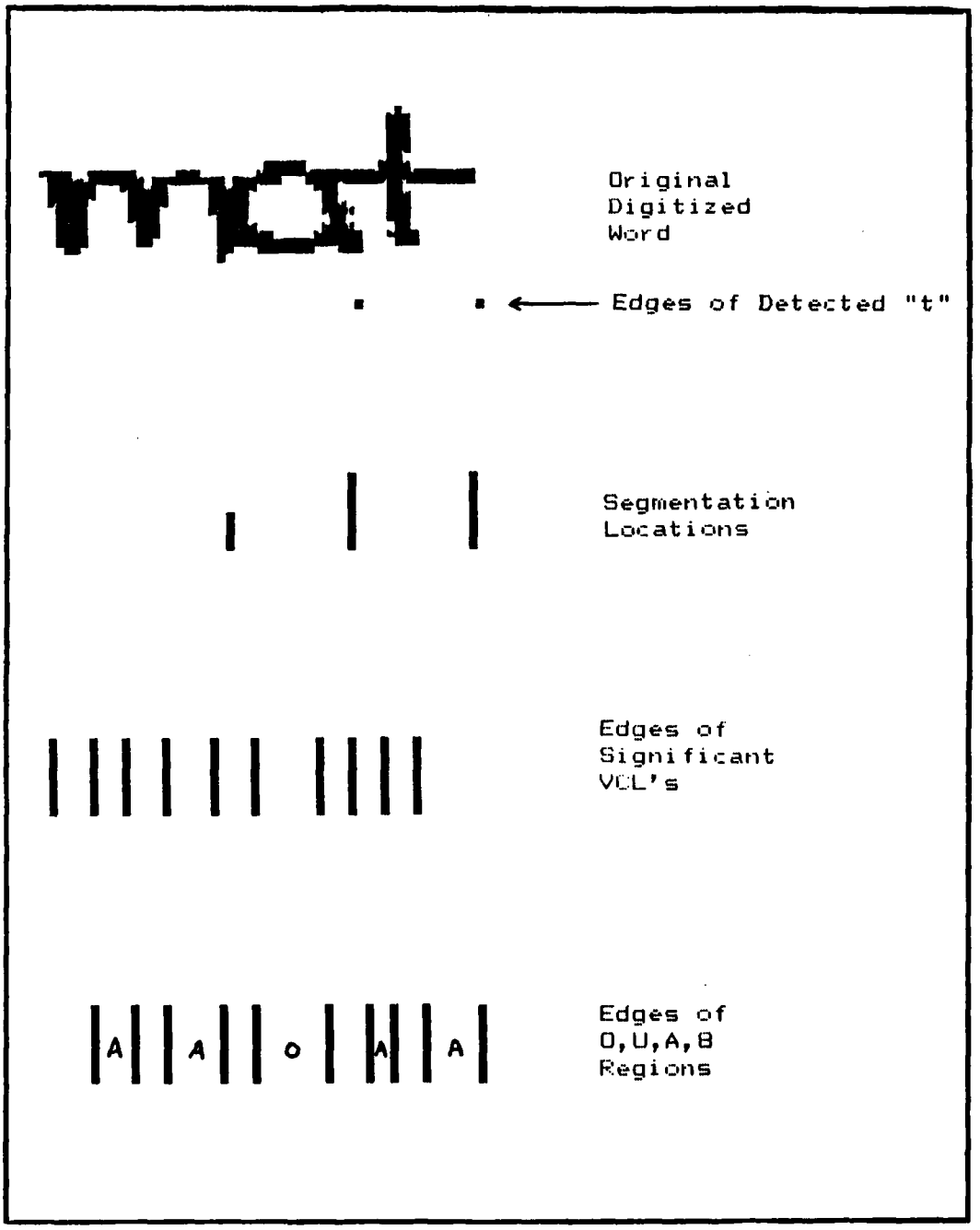


Figure 4.2 Segmentation Display Output for "mat"

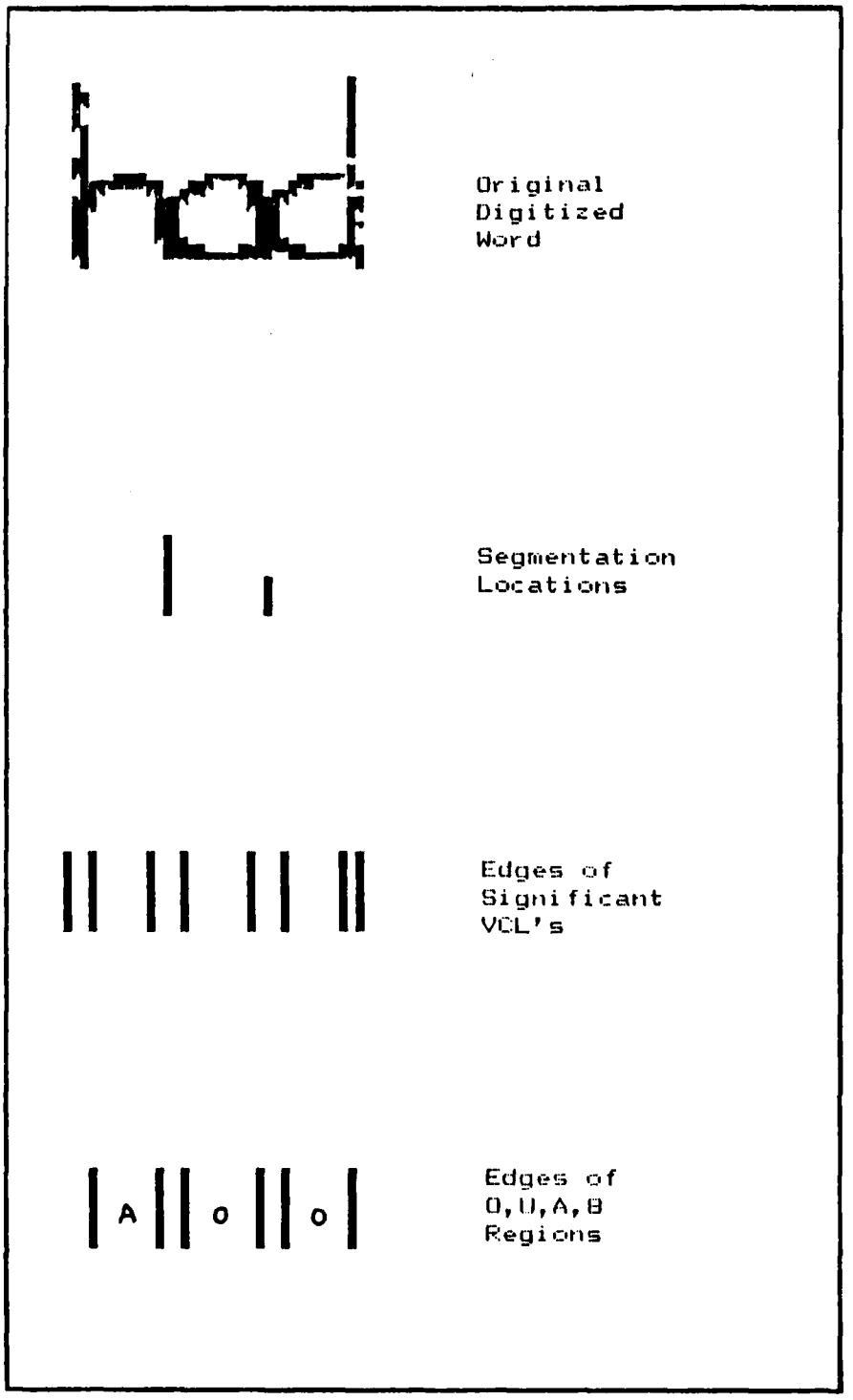


Figure 4.3 Segmentation Display Output for "had"

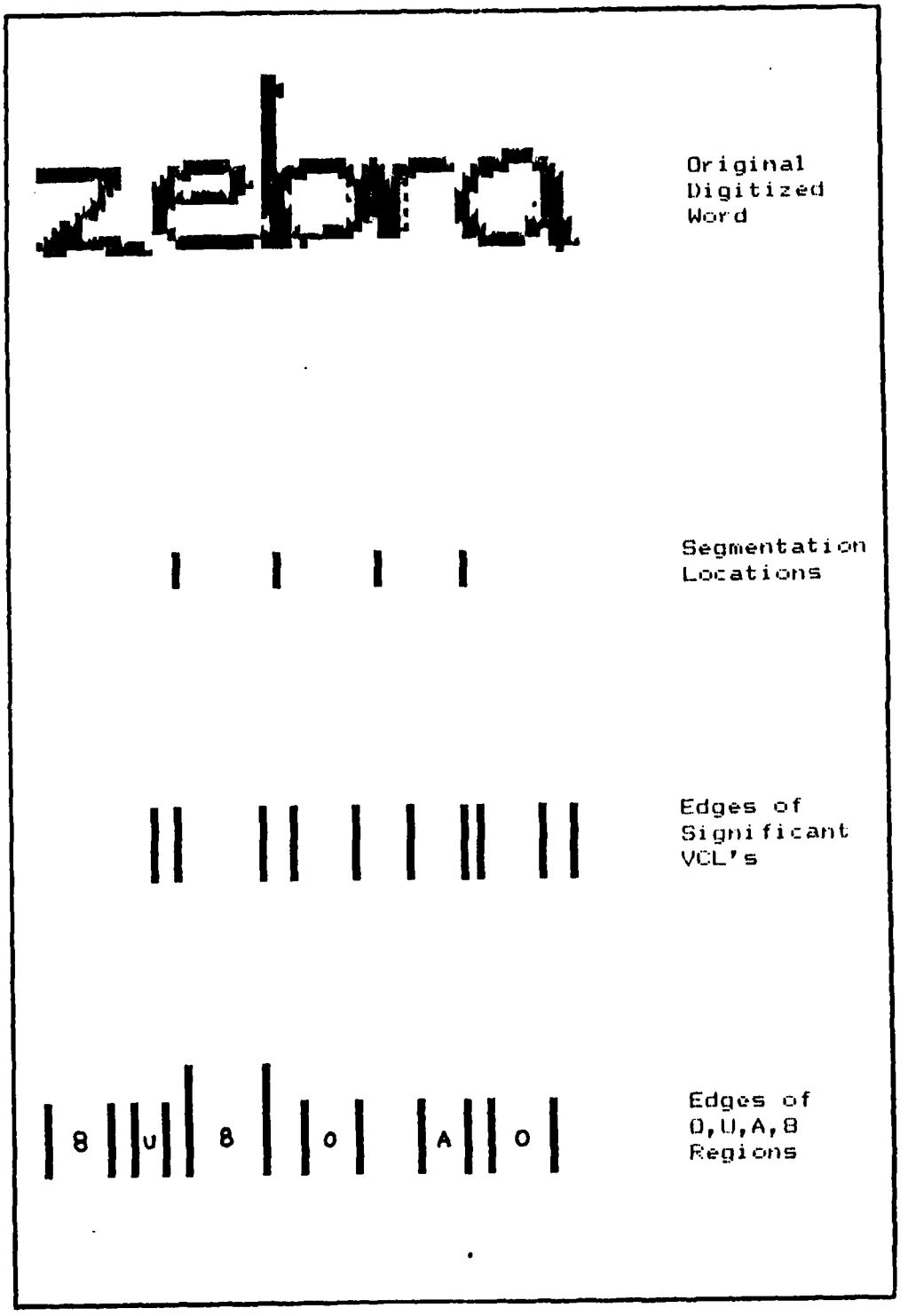


Figure 4.4 Segmentation Display Output for "zebra"

V. CONCLUSION AND RECOMMENDATIONS

The results in Chapter 4 indicate that it is possible to segment touching English letters based on the length and position of vertical lines, and VCL, A, O, U, and 8 regions. As mentioned in the introduction, it almost necessary to recognize the individual letters in order to segment letters which are not separated by a space. In effect, this thesis work has shown this to be true because the same features which were used to segment the letters could be used to help recognize the individual letters. This algorithm could be simplified and made more reliable by considering probabilities and locations of letter combinations. Many of the letter combinations considered in Chapter 2 may never occur in the English language. Also knowing where letter combinations can occur in a word could also improve the reliability. For example, by knowing that a "rn" combination can not begin a word, the segmentation algorithm could avoid falsely segmenting a "m" at the beginning of a word. One could also improve reliability by avoiding segmenting letter combinations, which proved difficult to segment, by considering these letter combinations as a single "new" letter. The "new" letter could then be recognized in the same manner as the other single letters in the alphabet.

In testing this algorithm, no spacing between any of the letters were assumed in order to create a worst case

condition. In reality, spaces between some letters in a word do occur. These spaces and the detected letter features could be used to recognize a letter which is touching an adjacent letter. By knowing one of the letters in a touching letter combination, the segmentation algorithm could more reliably segment the letter combination.

The work done in this thesis is just a starting point. Similiar software needs to be written to account for upper case print. Preprocessing software needs to be written to ignore " tails " at the edge of printed fonts. A pattern recognition approach has to be chosen to recognize the individual letters. Already, extensive work has been done at AFIT recognizing individual letters using two dimensional Fast Fourier Transforms. Also, extensive work has been done by IBM in the area of postprocessing to correct spelling mistakes or to predict a garbled letter in the context of a word (2). The foundations are already present for AFIT to someday demonstrate a reliable reading machine.

BIBLIOGRAPHY

1. Bentkowski, R. E. Segmentation of Touching English Letters MS Thesis. School of Engineering, Air Force Institute of Technology, Wright Patterson AFB, Ohio Dec. 1979.

2. Biegeleisen, J. I. The ABC of Lettering. New York: Harper & Row Publishers, 1976.

3. Dewey, Godfrey Relative Frequency of English Speech Sounds. Harvard University Press, Cambridge, MA, 1923.

4. Kabrisky, Matthew. Lecture materials distributed in EE 6.20, Pattern Recognition I. School of Engineering, Air Force Institute of Technology, Wright Patterson AFB, Ohio, Jan. 1985.

5. Shum, Lugene. Character Recognition Using Correlation Techniques MS Thesis. School of Engineering, Air Force Institute of Technology, Wright Patterson AFB, Ohio, Dec. 1984.

6. Simmons, Robin A. Machine Recognition of Unformatted Characters MS Thesis. School of Engineering, Air Force Institute of Technology, Wright Patterson AFB, Ohio, Dec. 1981.

7. Stanton, Tom. "Peripheral Vision: A Guide to Optical Character Readers," PC magazine, Vol. 4: 105-133 (July 9, 1985).

7. King, David A. OCTEK and PICBUF user's manual, Digital Signal Processing Lab, AFIT, Wright Pat. AFB, Ohio.

APPENDIX A

CASES WHERE NO MIDDLE VCL EXIST BETWEEN A,O,U, OR 8 REGIONS

ca	cg	cs	ct	cv	cw	cy	cz	cx
ea	eg	es	et	ev	ew	ey	ez	ex
ga	gg	gs	gt	gv	gw	gy	gz	gx
ka	kg	ks	kt	kv	kw	ky	kz	kx
ra	rg	rs	rt	rv	rw	ry	rz	rx
sa	sg	ss	st	sv	sw	sy	sz	sx
ta	tg	ts	tt	tv	tw	ty	tz	tx
va	vg	vs	vt	vv	vw	vy	vz	vx
wa	wg	ws	wt	wv	ww	wy	wz	wx
ya	yg	ys	yt	yv	yw	yy	yz	yx
xa	xg	xs	xt	xv	xw	xy	xz	xx

APPENDIX B

ABCDEFGHIJ
KLMNOP
QRSTUVWXYZ

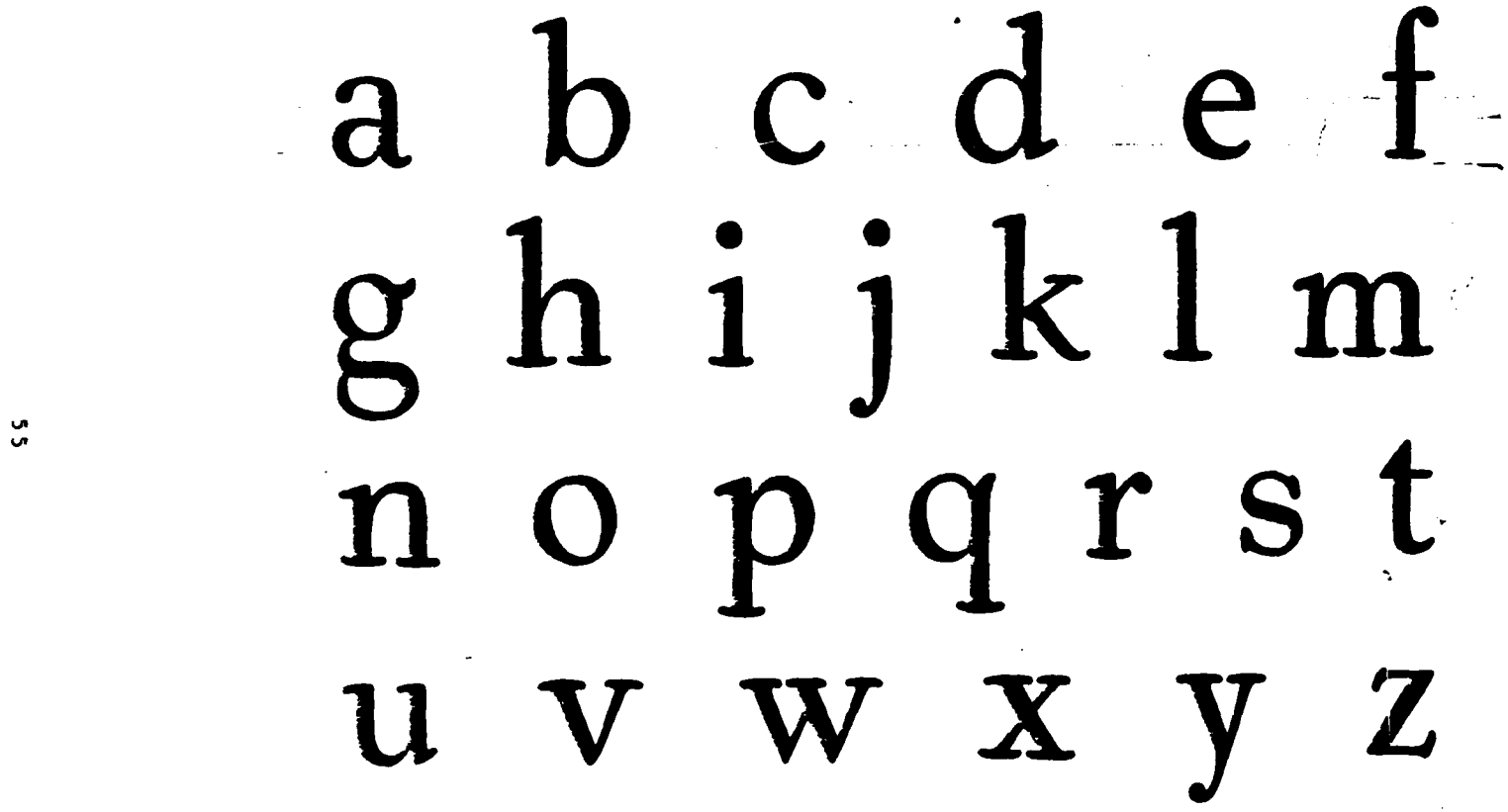
abcdefghijkl
mnopqrstuvwxyz

Caslon

ABCDEFGHIJKL
MNOPQRSTUVW
XYZ 1234567890

abcdefghijklmnop
qrstuvwxyz

Garamond Bold



Cooper, lower case

A B C D E F G H I
J K L M N O P Q R
S T U V W X Y Z

a b c d e f g h i j k l
m n o p q r s t
u v w x y z

Weiss Extra Bold

ABCDEFGHIJKL
MNOPQRSTUVW
XYZ 1234567890

abcdefghijklmnop
qrstuvwxyz

Garamond Bold

APPENDIX C

```
C*****************************************************************
C  SUBROUTINE NAME: MAIN PROGRAM ( MAIN.FR )
C  WRITTEN BY: 1LT. DAVID V. SOBOTA
C  PURPOSE:  THIS IS THE MAIN CALLING PROGRAM WHICH ASK THE
C  USER FOR THE FILE NAME AND COORDINATES OF THE WORD TO BE
C  SEGMENTED.  THIS PROGRAM THEN CALLS ALL THE SUBROUTINES
C  AND DISPLAYS THE BINARY WORD AND A DISPLAY OF ALL
C  SEGMENTATION CLUES DETERMINE BY THE SUBROUTINES.
C*****************************************************************

      PARAMETER NBUFSZ=300
      PARAMETER NUMCOLS=128
      PARAMETER NUMROWS=128
      PARAMETER KTTYOUT=10, KTTYIN=11
      INTEGER IBUF(NBUFSZ), INAME(20), NPIX
      INTEGER IBUF1(NBUFSZ)
      INTEGER IBUF3(NBUFSZ)
      INTEGER IBUF5(NBUFSZ)
      INTEGER ICOL, IROW, I, STROW, STCOL, TT,TS,BS,VBS
      INTEGER NCOLS, NROWS
      INTEGER VPROJ(128), GAP(128)
      INTEGER IARRAY(256)
      INTEGER IARRAY2(256)
      INTEGER HPROJ(128), RESULT(128), VCL(128)
C     INTEGER MXVHT(128), MXVHTX(128)
C     INTEGER MINVHT(128), MINVHTX(128)
      INTEGER VPW, CLW, MMW, AVCG(128), AVCW, MVC
      INTEGER EVC(128), ROW
      INTEGER VCLD(128),VCLU(128),IFND(128)
      INTEGER MARG, BGT(128), EDT(128)
      INTEGER BGVC(128),EDVC(128),BGI(128),EDI(128)
C     INTEGER BGO(128),EDO(128),BGU(128),EDU(128),EDE(128)
C     INTEGER BGA(128),EDA(128),BGT(128),EDT(128),BGE(128)
      INTEGER SEG(128)
      INTEGER MXHT1(128),MNHT1(128),MXHT2(128),MNHT2(128)
      INTEGER IOR(20,3),IUR(20,3),IER(20,3),IAR(20,3)
      INTEGER XIOR(20,3),XIUR(20,3),XIER(20,3),XIAR(20,3)
      INTEGER TBD(20,2),IVCL(20,10),IBD(20,2)
C
      EXTERNAL  VP
      EXTERNAL  DV
      EXTERNAL  HP
      EXTERNAL  FH
      EXTERNAL  EV
      EXTERNAL  HT
      EXTERNAL  DT
      EXTERNAL  IF
      EXTERNAL  TF
      EXTERNAL  DA
      CALL OVOPN(4,"MAIN.OL",IER); OVERLAY USED FOR MORE
                                 ; MEMORY
```

```
      .
      WRITE (KTTYOUT,10)
10    FORMAT (1X,"INPUT PICTURE FILE NAME=",Z)
      READ (KTTYIN,20) INAME(1)
20    FORMAT (S39)
      CALL PICFMT (IBUF,INAME,IHDR)
      CALL GFMT (IBUF,NROWS,NCOLS,NBITS,IMODE)
      CALL MAKB (IBUF)
      CALL PICIN (IBUF,INAME,IHDR)

C

      ACCEPT "NUMBER OF ROWS=",NROWS
      ACCEPT "NUMBER OF COLUMNS=",NCOLS
      ACCEPT "STARTING ROW=", STROW
      ACCEPT "STARTING COL.=", STCOL
C
C     STORE WORD IN INITIALIZED BUFFER IBUF5
C
      CALL PFMT (IBUF1,NROWS,NCOLS,4,IMODE)
      CALL MAKB (IBUF1)
C
      DO 26 I= 1,NROWS
      CALL GROW(IBUF,(STROW-1)+I,STCOL,NCOLS,IARRAY)
      CALL PROW(IBUF1,I,1,NCOLS,IARRAY)
26    CONTINUE
      CALL RELB(IBUF)
      CALL PFMT (IBUF ,NROWS,NCOLS,4,IMODE)
      CALL MAKB(IBUF )

C     MAKE OUTPUT BUFFER

      CALL PFMT (IBUF3,256,256,4,IMODE)
      CALL MAKB (IBUF3)

C     BLANK OUT THE ENTIRE BUFFER

      DO 65 I=1,256
            IARRAY(I)=15
65    CONTINUE
      DO 70 I=1,256
            CALL PROW(IBUF3,I,1,256,IARRAY)
70    CONTINUE
      MARG=5
      DO 80 I=1,NROWS
      CALL GROW(IBUF1,I,1,NCOLS,IARRAY)     ; ORIGINAL WORD
      CALL PROW(IBUF3,I,MARG,NCOLS,IARRAY) ; TOP LEFT
80    CONTINUE

      CALL PFMT(IBUF5,NROWS,NCOLS,4,IMODE)
      CALL MAKB(IBUF5)

      CALL OVLOD(4,VP ,1,IER)
      CALL VRTPROJ(IBUF1,NROWS,NCOLS,VPROJ,GAP,IBUF ,IBUF5)
```

```
      TYPE "VRTPROJ DONE"
                                                        ; PAINTS
      DO 90 I=1,NROWS                                   ; VRTPROJ
C     CALL GROW(IBUF ,I,1,NCOLS,IARRAY)                 ; #5 RT.
C     CALL PROW(IBUF3,4*(NROWS)+I,MARG*2+120,NCOLS,IARRAY)
90    CONTINUE
C                                                ; PAINTS
      DO 92 I=1,NROWS                            ; GAP
      CALL GROW(IBUF5,I,1,NCOLS,IARRAY)          ; #5 LFT.
      CALL PROW(IBUF3,4*(NROWS  )+I,MARG,NCOLS,IARRAY)
92    CONTINUE
      TYPE "92 DONE"

      CALL OVLOD(4,DV  ,1,IER)
      CALL DVCLIN(IBUF1,NROWS,NCOLS,VCLD,VCLU,VCL,IBUF )
      TYPE "DVCLIN DONE"
                                                        ;PAINTS
      DO 95 I=1,NROWS                                   ;DVCLIN2
C          CALL GROW(IBUF ,I,1,NCOLS,IARRAY)            ;#3 RT.
C          CALL PROW(IBUF3,2*(NROWS)+I,MARG*2+120,NCOLS,
     /     IARRAY)
95    CONTINUE
C
      CALL OVLOD(4,HP  ,1,IER)
      CALL HRZPROJ(IBUF1,NROWS,NCOLS,HPROJ,IBUF5)
      TYPE "HRZPROJ DONE"
C                                                       ; PAINTS
      DO 100 I=1,NROWS                                  ; HRZHIST
          CALL GROW(IBUF5,I,1,NCOLS,IARRAY)             ; #4 LFT
          CALL PROW(IBUF3,3*(NROWS  )+I,MARG,NCOLS,IARRAY)
100   CONTINUE
C
      CALL OVLOD(4,FH  ,1,IER)
      CALL FDLHT( NROWS,HPROJ,TT,TS,BS,VBS )
      TYPE "TOP TALL =",TT
      TYPE "TOP SHORT =",TS
      TYPE "BOTTOM SHORT =",BS
      TYPE "VERY BOTTOM SHORT =",VBS
C
      CALL PPNT(IBUF3,TT,MARG-2,0); MARKS PTS. WHERE TT,TS,
      CALL PPNT(IBUF3,TS,MARG-2,0); ETC. ARE LOCATED
      CALL PPNT(IBUF3,BS,MARG-2,0)
      CALL PPNT(IBUF3,VBS,MARG-2,0)
C
C     CALL SGAVC(IBUF ,TS,BS,NCOLS,NROWS,AVCG,VCL)
C     TYPE "SGAVC DONE"
C
C     CALL DDCRV(IBUF1,TS,BS,NCOLS,NROWS,MXVHT,MXVHTX,
     /  IBUF5,IBUF )
C     TYPE "DDCRV DONE"
C                                                       ;PAINTS
      DO 105 I=1,NROWS                                  ;DDCRV
C              CALL GROW(IBUF5,I,1,NCOLS,IARRAY)        ;#1 RT
```

```
C               CALL PROW(IBUF3,I,MARG*2+120,NCOLS,IARRAY)
105     CONTINUE
C                                                          ;PAINTS
        DO 106 I=1,NROWS                                   ;P. DDCRV
C               CALL GROW(IBUF ,I,1,NCOLS,IARRAY)    ;#1 RT.
C               CALL PROW(IBUF3,I,MARG*2+120,NCOLS,IARRAY)
106     CONTINUE
C
C       CALL DUCRV(IBUF1,TS,BS,NCOLS,NROWS,MINVHT,MINVHTX,
C    /  IBUF5,IBUF )
C       TYPE "DUCRV DONE"
C                                                          ;PAINTS
        DO 108 I=1,NROWS                                   ;DUCRV
C               CALL GROW(IBUF5,I,1,NCOLS,IARRAY)    ;#2 RT.
C               CALL PROW(IBUF3,2*(NROWS  )+I,MARG*2+120,
     /          NCOLS,IARRAY)
108     CONTINUE
C                                                          ;PAINTS
        DO 107 I=1,NROWS                                   ;P.DUCRV
C               CALL GROW(IBUF ,I,1,NCOLS,IARRAY)    ;#2 RT
C               CALL PROW(IBUF3,(NROWS  )+I,MARG*2+120,
C    /  NCOLS,IARRAY)
107     CONTINUE
C
        CALL OVLOD(4,EV ,1,IER)
        CALL EDVCL(IBUF ,NROWS,NCOLS,ROW,EVC,TS,BS,BGVC,EDVC,
     /  IVCL,IBUF5)
C                                                          ;PAINTS
        DO 160 I=1,NROWS                                   ;EDVCL
             CALL GROW(IBUF5,I,1,NCOLS,IARRAY)       ;#3 LFT.
             CALL PROW(IBUF3,2*(NROWS  )+I,MARG,NCOLS,IARRAY)
160     CONTINUE
        DO 155 I=1,NCOLS
C            IF (EVC(I).NE.10) GOTO 155
C            CALL PPNT(IBUF3,3*(NROWS  )-1,MARG*2+120-1+I,0)
155     CONTINUE
C
C       PUTS A PIXEL BY ROW OF DVCLIN SELECTED IN EDVCL
        CALL PPNT(IBUF3,2*(NROWS  )+ROW,MARG*2+120,0)
C
        CALL OVLOD(4,HT ,1,IER)
        CALL HTVCL(IBUF1,BGVC,EDVC,NROWS,NCOLS,MXHT1,MXHT2,
     /  MNHT1,MNHT2,BS,TS,IVCL)
C
        CALL OVLOD(4, DT ,1,IER)
        CALL DTOUAE(IBUF1,NROWS,NCOLS,TS,BS,IBUF3,IBUF2,
     /  XIOR,XIUR,XIAR,XIER,IOR,IUR,IAR,IER) ;
        TYPE "DTOUAE DONE"
C                                                         ; PAINTS
        DO 120 I=1,NROWS                                  ; DTOUAE
C            CALL GROW(IBUF5,I,1,NCOLS,IARRAY)       ; #4 RT.
C            CALL GROW(IBUF3,3*(NROWS)+I,MARG*2+120,NCOLS,
     /       IARRAY)
```

```
120     CONTINUE
C                                                       ; PAINTS
        DO 122 I=1,NROWS                                ; P. DTOUA
C               CALL GROW(IBUF2,I,1,NCOLS,IARRAY)       ; #4 LFT.
C               CALL PROW(IBUF3,3*(NROWS  )+I,MARG,NCOLS,
     /          IARRAY)
122     CONTINUE
C
        CALL OVLOD(4, IF ,1,IER)
        CALL IFIND(NCOLS,BGVC,EDVC,VCLD,VCLU,TS,BS,IFND,BGI,
     /  EDI,IBD)
        TYPE "IFIND DONE"
        DO 130 I=1,NCOLS
        IF (IFND(I).NE.10) GOTO 130
        CALL PPNT(IBUF3,NROWS+1,MARG-1+I,0)
130     CONTINUE
C
        CALL OVLOD(4, TF ,1,IER)
        CALL FINDT(IBUF1,NCOLS,NROWS,EVC,TS,BS,MXHT1,MXHT2,
     /  MNHT1,MNHT2,BGT,EDT,TBD)
        TYPE "FINDT DONE"
        DO 140 I=1,NCOLS
        IF ((BGT(I).NE.10).AND.(EDT(I).NE.10)) GOTO  140
        CALL PPNT(IBUF3,NROWS ,MARG-1+I,0)
140     CONTINUE

        CALL OVLOD(4, DA ,1,IER)
        CALL DECALG(NROWS,NCOLS,IVCL,IAR,IUR,IOR,IER,GAP,TBD,
     /  IBD,IBUF5)

        CALL CLOSE(4,IER)

        TYPE "DECALG DONE"
C                                                       ; PAINTS
        DO 150 I=1,NROWS                                ; DECALG
           CALL GROW(IBUF5,I,1,NCOLS,IARRAY)            ; #2 LFT.
           CALL PROW(IBUF3,(NROWS  )+I,MARG,NCOLS,IARRAY)
150     CONTINUE
        TYPE "OUTPUT DECALG DONE"
C       THE FOLLOWING SAVES BUFFER IBUF3 ON DISK AND RELEASES
C       OTHER BUFFERS.
C
        WRITE (KTTYOUT,110)
110     FORMAT ("OUTPUT PICTURE FILE NAME=",Z)
        READ (KTTYIN,20) INAME(1)
        CALL PICOUT (IBUF3,INAME,IHDR)
        CALL RELB(IBUF )
        CALL RELB(IBUF1)
        CALL RELB(IBUF3)
        CALL RELB(IBUF5)
        END
```

```
C*****************************************************************
C  SUBROUTINE NAME: VERTICAL PROJECTION (VRTPROJ.FR)
C  WRITTEN BY: 1LT. DAVID V. SOBOTA
C  PURPOSE:  THIS PROGRAM WILL DISPLAY A PROJECTION OF THE #
C  # OF PIXELS IN A COLUMN ON THE HORIZONTAL AXIS TO PRODUCE
C  VPROJ.   THE  VPROJ  ARRAY IS PROCESSED  TO  DETERMINE
C  POSSIBLE GAPS OR SPACES BETWEEN LETTERS.
C*****************************************************************
C
      OVERLAY VP
      SUBROUTINE VRTPROJ(IBUFI,NROWS,NCOLS,VPROJ,GAP,IBUFO1,
     /  IBUFO2)
C
      PARAMETER NBUFSZ=300
      PARAMETER NUMCL=256
      INTEGER IBUFI(NBUFSZ)
      INTEGER IBUFO1(NBUFSZ)
      INTEGER IBUFO2(NBUFSZ)
      INTEGER ICOL, IROW
      INTEGER NCOLS, NROWS
      INTEGER VPROJ(NUMCL)
      INTEGER IARRAY(NUMCL)
      INTEGER IARRY1(NUMCL)
      INTEGER IARRY2(NUMCL)
      INTEGER GAP(NUMCL)
C
C
      DO 30 ICOL=1,256          ; ZEROS VPROJ ARRAY
      VPROJ(ICOL)=0
30    CONTINUE
C
C     THIS LOOP COUNTS NUMBERS OF DARK PIXELS/COLUMN
C
      DO 35 IROW=1,NROWS
            CALL GROW(IBUFI,IROW,1,NCOLS,IARRAY)
            DO 40 ICOL=1,NCOLS
            IF (IARRAY(ICOL).EQ.0) VPROJ(ICOL)=
     /  VPROJ(ICOL)+1
40    CONTINUE
35    CONTINUE
C
C     THIS LOOP PRODUCES THE GAP ARRAY WHICH GIVE THREE
C     LEVELS OF CONFIDENCE AS TO WHETHER THERE IS A SPACE
C     BETWEEN THE LETTERS AT A  PARTICULAR COLUMN LOCATION.
C
      DO 60 K=1,NCOLS
      GAP(K)=0
      IF ((VPROJ(K).EQ.1).AND.((VPROJ(K+3).GE.5).OR.
     /  (VPROJ(K-3).GE.5))) GAP(K)=5
      IF ((VPROJ(K).EQ.0).AND.((VPROJ(K+3).GE.5).OR.
     /  (VPROJ(K-3).GE.5))) GAP(K)=10
      IF ((VPROJ(K).EQ.0).AND.((VPROJ(K-1).LE.1).OR.
     /  (VPROJ(K+1).LE.1))) GAP(K)=15
```

```
60      CONTINUE
C
C     THE FOLLOWING PUTS THE VPROJ DATA IN IBUFO1 FOR DISPLAY
C     THE FOLLOWING PUTS THE GAP DATA IN IBUFO2 FOR DISPLAY

        DO 50 IROW=1,NROWS
        DO 55 ICOL=1,NCOLS
                IARRY1(ICOL)=15
                IARRY2(ICOL)=15
                IF (VPROJ(ICOL).GE.IROW) IARRY1(ICOL)=0
                IF (GAP(ICOL).GE. IROW) IARRY2(ICOL)=0
55      CONTINUE
        CALL PROW(IBUFO1,NROWS-IROW+1,1,NCOLS,IARRY1)
        CALL PROW(IBUFO2,NROWS-IROW+1,1,NCOLS,IARRY2)
50      CONTINUE
C
        RETURN
        END
```

```
C ***************************************************************
C   SUBROUTINE NAME:  DETECT  VERTICAL  CONTINUOUS  LINES
C                       (DVCLIN.FR)
C   WRITTEN BY: 1LT. DAVID V. SOBOTA
C   PURPOSE:  THIS  SUBROUTINE  FINDS  SIGNIFICANT  VERTICAL
C   CONTINUOUS LINES (VCL'S) BY LOOKING AT EACH COLUMN IN THE
C   BUFFER  CONTAINING  THE WORD AND COUNTS  THE  NUMBER  OF
C   CONTINUOUS DARK PIXELS IN A COLUMN UNTIL IT SEES A GAP OF
C   2 WHITE PIXELS.  IT PERFORMS THIS COUNT STARTING FROM THE
C   BOTTOM AND THE TOP OF THE WORD AND CHOSES THE BIGGEST VCL
C   TO ACCOUNT FOR LETTERS LIKE "J" AND "I".
C***************************************************************

        OVERLAY DV
        SUBROUTINE DVCLIN(IBUFIN,NROWS,NCOLS,VCLD,VCLU,VCL,
     /  IBUFOUT)
C
        INTEGER INCR(256)
        INTEGER VCL(256),VCLD(256), VCLU(256)
        INTEGER IGAP(256)
        INTEGER IARRAY1(256)
        INTEGER IARRAY2(256)
        INTEGER IARRY1(256), IARRY2(256)
        INTEGER JARRAY(256)
        INTEGER START(256)
        INTEGER IBUFIN(300)
        INTEGER IBUFOUT(300)
        INTEGER NROWS,NCOLS
C
C       THIS CODE SCANS EACH COLUMN FROM TOP TO BOTTOM TO FIND
C       VCLD ARRAY
C
        DO 10 ICOL=1,NCOLS ; THIS LOOP INIT. COUNT PARAMETERS
        VCL(ICOL)=0        ; LARGEST VCL ARRAY
        VCLD(ICOL)=0       ; DOWNWARD VCL'S ARRAY
        IGAP(ICOL)=0       ; STORES COUNT OF GAPS OF DARK PIXELS
        INCR(ICOL)=1       ; INCREMENT FLAG
        START(ICOL)=0      ; START COUNT OF DARK PIXELS FLAG
10      CONTINUE
C
C
        DO 20 IROW=1,NROWS-1
                CALL GROW(IBUFIN,    IROW+1,1,NCOLS,IARRAY1)
                CALL GROW(IBUFIN,    IROW  ,1,NCOLS,IARRAY2)
                DO 30 ICOL=1,NCOLS
             IF (.NOT.((IARRAY1(ICOL).EQ.0).AND.
     /  (IARRAY2(ICOL).EQ.0))) GOTO 40
                START(ICOL)=1
                VCLD(ICOL)=VCLD(ICOL)+INCR(ICOL)
                GOTO 50
40      CONTINUE
                IF ( START(ICOL).EQ. 0) GOTO 30
                IGAP(ICOL)=IGAP(ICOL)+1
```

```
                IF ( IGAP(ICOL).GE.2 ) INCR(ICOL)=0
                GOTO 30
50      CONTINUE
30      CONTINUE
20      CONTINUE
C
C       THIS CODE PERFORMS THE SAME VCL COUNT AS ABOVE BUT
C       FROM OPPOSITE THE DIRECTION
C
        DO 55 I=1,NCOLS
        VCLU(I)=0       ; UPWARD VCL ARRAY IS ZEROED
        INCR(I)=1
        IGAP(I)=0
        START(I)=0
55      CONTINUE
C
        DO 60 IROW=1,NROWS-1
              CALL GROW(IBUFIN,NROWS-IROW+1,1,NCOLS,IARRAY1)
              CALL GROW(IBUFIN,NROWS-IROW  ,1,NCOLS,IARRAY2)
                DO 70 ICOL=1,NCOLS
                IF (.NOT.((IARRAY1(ICOL).EQ.0).AND.
     /  (IARRAY2(ICOL).EQ.0))) GOTO 80
                START(ICOL)=1
                VCLU(ICOL)=VCLU(ICOL)+INCR(ICOL)

                GOTO 90
80      CONTINUE
                IF ( START(ICOL).EQ. 0) GOTO 70

                IGAP(ICOL)=IGAP(ICOL)+1
                IF ( IGAP(ICOL).GE.2 ) INCR(ICOL)=0
                GOTO 70

90      CONTINUE
70      CONTINUE
60      CONTINUE
C
C       THIS LOOP COMPARES EACH COL. OF VCLU AND VCLD AND
C       PICKS THE BIGGEST AND STORES RESULTS IN THE VCL ARRAY
C
        DO 100 I=1,NCOLS
                VCL(I)=VCLU(I)
                IF (VCLU(I).GT. VCLD(I)) GOTO 100
                VCL(I)=VCLD(I)
100     CONTINUE
C
C       PUT VCL'S IN NEW BUFFER IBUFOUT FOR DISPLAY
C
        DO 110 IROW=1,NROWS
        DO 115 ICOL=1,NCOLS
                IARRAY2(ICOL)=15
                IF (VCL(ICOL).GE.IROW) IARRAY2(ICOL)=0
115     CONTINUE
```

```
      CALL  PROW(IBUFOUT,NROWS-IROW+1,1,NCOLS,IARRAY2)
110   CONTINUE
C
C     SMOOTHS ORIGINAL VCL TO FILL IN SMALL VALLEYS WHICH
C     PRODUCE ERRORS IN FINDING THE EDGES OF SIGNIFICANT
C     VCL'S (EDVCL.FR)
C
      DO 140  IROW=1,NROWS-1
              CALL GROW(IBUFOUT,IROW,1,NCOLS,IARRY1)
              CALL GROW(IBUFOUT,IROW+1,1,NCOLS,IARRY2)
      DO 150 J=1,NCOLS
      IF ((IARRY2(J).EQ.0).AND.((IARRY1(J-1).EQ.0).OR.
     /(IARRY1(J+1).EQ.0)))  IARRY1(J)=0
150   CONTINUE
      CALL PROW(IBUFOUT,IROW,1,NCOLS,IARRY1)
140   CONTINUE
C
      RETURN
      END
```

```
C*****************************************************************
C  SUBROUTINE NAME: HORIZONTAL PROJECTION (HRZPROJ.FR)
C  WRITTEN BY: 1LT. DAVID V.SOBOTA
C  PURPOSE: THIS ROUTINE COUNTS THE NUMBER OF DARK PIXELS IN
C  EACH ROW AND PROJECTS THE NUMBER ON THE VERTICAL AXIS.
C  THE RESULTS ARE USED IN FDLHT.FR TO FIND THE HEIGHTS OF
C  VARIOUS LETTERS.
C*****************************************************************
C
      OVERLAY HP
      SUBROUTINE HRZPROJ(IBUF1,NROWS,NCOLS,HPROJ,IBUF2)
      PARAMETER NUMROWS=256
      INTEGER HPROJ(NUMROWS)
      INTEGER IARRAY1(256)
      INTEGER IARRAY2(256)
C
      DO 10 IROW=1,NROWS ; ZEROS PROJECTION OF PIXELS/ROW
      HPROJ(IROW)=0
10    CONTINUE
C
      DO 20 IROW=1,NROWS
            CALL GROW(IBUF1,IROW,1,NCOLS,IARRAY1)
            DO 30 ICOL=1,NCOLS
            IF (IARRAY1(ICOL).EQ. 0) HPROJ(IROW)=
     /  HPROJ(IROW)+1
30    CONTINUE
20    CONTINUE
C
C     THE FOLLOWING PUTS HPROJ IN IBUF2 FOR DISPLAY
C
      DO 40 IROW=1,NROWS
      DO 50 ICOL=1,NCOLS
            IARRAY2(ICOL)=15
            IF (HPROJ(IROW).GE. ICOL) IARRAY2(ICOL)=0
50    CONTINUE
      CALL PROW(IBUF2,IROW,1,NCOLS,IARRAY2)
40    CONTINUE

      RETURN
      END
```

```
C*****************************************************************
C  SUBROUTINE NAME: FIND LETTER HEIGHT (FDLHT.FR)
C  WRITTEN BY : 1LT. DAVID V. SOBOTA
C  PURPOSE: THIS PROGRAM USES HPROJ PASSED BY HRZPROJ.FR TO
C  FIND FOUR ASSOCIATED HEIGHTS IN SMALL LETTERS:TT, TS, BS,
C  AND VBS. THESE DIFFERENT HEIGHTS ARE EXPLAINED BELOW. THESE
C  HEIGHTS ARE USED AS CLUES TO DETERMINE LOCATION OF POSSIBLE
C  LETTERS.
C*****************************************************************

      OVERLAY FH
      SUBROUTINE FDLHT(NROWS,HPROJ,TT,TS,BS,VBS )
      PARAMETER NUMROWS=256
      INTEGER HPROJ(NUMROWS), DTHR
      INTEGER I, IROW, NROWS, TPIX, NAVPX, TT, TS, BS, VBS
C
C     FIRST FIND AVE. # OF PIXELS/ROW TO DETERMINE THRESHOLD
C
      TPIX=0   ; COUNT OF TOTAL NUMBER OF PIXELS IN WORD
      N=0      ; COUNTS # OF ROWS W/ SIGNIFICANT # PIXELS
      DO 10 IROW=1,NROWS
            IF (HPROJ(IROW).LT. 2) GOTO 15
            TPIX=HPROJ(IROW)+TPIX
            N=N+1
15    CONTINUE
10    CONTINUE
      NAVPX=INT(TPIX/N) ; NAVPX = AVE. # OF PIXELS/ROW
C
C     TYPE "TPIX=", TPIX
C     TYPE "N=", N
C     TYPE "NAVPX=", NAVPX
C     TYPE "NROWS=", NROWS
C
C     TT = TOP OF TALL LETTERS ( h,f,l,b,d,ETC. )
C     TS = TOP OF SHORT LETTERS ( o,a,c,e,p,g,ETC. )
C     BS = BOTTOM OF SHORT LETTERS ( o,a,c,e,ETC. )
C     VBS = VERY BOTTOM OF SHORT LETTERS ( g,p,q,j,ETC. )
C
C     THIS CODE STARTS FROM THE TOP OF HPROJ TO DETERMINE
C     WHEN A SIGNIFICANT # OF PIXELS IS ENCOUNTER AND LESS
C     NAVPX TO FIND TT AND GREATER THAN NAVPX TO DETERMINE TS.
C
      TT=0     ; THESE VALUES ARE INITIALIZED
      TS=0
      BS=0
      VBS=0
      DO 30 I=1,NROWS
      IF (.NOT.(( HPROJ(I).LT. HPROJ(I+1)).AND.
     /  ( HPROJ(I+1).GE. 1).AND.(HPROJ(I+2).GT. 1))) GOTO 35
      IF ( TT .EQ. 0 ) TT=I+1
35    CONTINUE
      IF (.NOT.( ( HPROJ(I) .LT. HPROJ(I+1)) .AND.
     /  ( HPROJ(I+1).GT. NAVPX ).AND.(TT.NE.0) ))  GOTO 30
```

```
          IF ( TS .EQ. 0 ) TS=I
          IF ((TS-TT).LE.3) TS=TT
          IF (TS.NE.0) GOTO 40
30        CONTINUE
40        CONTINUE
C
C     THIS CODE STARTS FROM THE BOTTOM OF HPROJ TO DETERMINE
C     WHEN A SIGNIFICANT # OF PIXELS IS ENCOUNTER AND LESS
C     NAVPX TO FIND VBS AND GREATER THAN NAVPX TO DETERMINE BS
C
          DO 45 I=1,NROWS
          IF (.NOT.(( HPROJ(NROWS-I+1).LT. HPROJ(NROWS-I)).AND.
     /    ( HPROJ(NROWS-I).GE. 1).AND.(HPROJ(NROWS-I-1).GT. 1)
     /                                  ))  GOTO 50
          IF (VBS.EQ.0) VBS=NROWS-I
50        CONTINUE
          IF (.NOT.(( HPROJ(NROWS-I+1).LT. HPROJ(NROWS-I)).AND.
     /    ( HPROJ(NROWS-I).GT. NAVPX).AND.(VBS.NE.0))) GOTO 45
          IF ( BS .EQ. 0 ) BS=NROWS-I+1
          DTHR =(BS-TS)/3
          IF ((VBS-BS).LE.DTHR) BS=VBS
          IF (BS.NE.0) GOTO 55
45        CONTINUE
55        CONTINUE
          IF ((TS-TT).LT.DTHR) TS=TT
C
          RETURN
          END
```

```
C*****************************************************************
C  SUBROUTINE NAME: EDGES OF VERTICAL CONT. LINES (EDVCL.FR)
C  WRITTEN BY: 1LT. DAVID V. SOBOTA
C  PURPOSE: THIS SUBROUTINE TAKES THE RESULTS OF DVCLIN.FR
C  AND FINDS THE SIGNIFICANT EDGES OF THE VCL'S BY FINDING
C  VCL'S WHOSE HEIGHT IS AT LEAST EQUAL TO 1/2(BS -TS).
C*****************************************************************
C
      OVERLAY EV
      SUBROUTINE EDVCL(IBUFIN,NROWS,NCOLS,ROW,EVC,TS,BS,
     /  BGVC,EDVC,IVCL,IBUFOUT)
C
      INTEGER NCOLS,NROWS,N,NBIG,NTMP,SUM
      INTEGER IBUFIN(300),IBUFOUT(300),IARRY1(256)
      INTEGER WVC(256), STCL(256), IARRY2(256)
      INTEGER RWVC(256), RSTCL(256)
      INTEGER VCL(256)
      INTEGER EVC(256), PEDG1, PEDG2
      INTEGER ROW
      INTEGER NSEG(256)
      REAL AVG, STDTMP, SDSEG(256), STDSML
      INTEGER BGVC(256),EDVC(256)
      INTEGER TS,BS,MID,IVCL(20,10)
C
      NBIG=0
C
C     THIS LOOP LOOKS AT EACH ROW OF IBUFIN PASSED BY
C     DVCLIN.FR AND STORES THE STARTING POSITION AND WIDTH
C     OF EACH VCL ENCOUNTERED IN THE ROW.
C
      DO 20 I=3,NROWS          ; INITIALIZED PARAMETERS
            DO 10 N=1,(NCOLS/2)
            WVC(N)=0   ; WIDTH VCL'S ARRAY
            STCL(N)=1  ; STARTING COL. POSITION
10          CONTINUE
            N=1      ; INITIALIZED # OF VCL'S COUNT
            FLG=0
            CALL GROW(IBUFIN,NROWS-I+1,1,NCOLS,IARRY1)
            DO 30 J=1,NCOLS
          IF ((FLG.EQ.0).AND.(IARRY1(J).EQ.0)) STCL(N)=J
          IF (IARRY1(J).EQ.0)   GOTO 35
            GOTO 40
35    CONTINUE
            WVC(N)=WVC(N)+1
            FLG=1
40    CONTINUE
            IF ((IARRY1(J).EQ.15).AND.(FLG.EQ.1)) GOTO 45
      GOTO 30
45    CONTINUE
            N=N+1
            FLG=0
30    CONTINUE
C
```

```
        NTMP=N-1      ;NTMP IS NUMBER OF SEGMENTS IN GIVEN ROW
        NSEG(I)=NTMP ;STORES NTMP IN EACH ROW IN NSEG
        IF (NTMP .GT. NBIG) NBIG=NTMP  ;FINDS BIGGEST NTMP
C
        SUM=0
        DO 70 K=1,NTMP
                SUM=SUM+WVC(K)
70      CONTINUE
        AVG=(SUM/NTMP) ;FINDS AVE. WIDTH OF SEGMENTS
        STDTMP=0.0
        DO 80 K=1,NTMP
                STDTMP=STDTMP+ABS(AVG-WVC(K))
80      CONTINUE
        SDSEG(I)=(STDTMP/NTMP)
20      CONTINUE
C
C     THIS LOOP PICKS THE ROW WITH NEAR GREATEST NUMBER OF
C     SEGMENTS AND SMALLEST STD. FOR ITS WIDTHS AND FINDS
C     THE EDGES AGAIN TO REPRESENT THE EDGES OF THE SIGNIFICANT
C     VCL'S. THE EDGES ARE THEN STORED IN BGVC AND EDVC.
C
        STDSML=500
        DO 120 I=3,NROWS
        IF ((NSEG(I).EQ. NBIG).OR.(NSEG(I).EQ.(NBIG-1)))
     /  GOTO 130
        GOTO 120
130     CONTINUE
C       TYPE I, " NSEG=", NSEG(I), " SDSEG=", SDSEG(I)
        IF (SDSEG(I).LE. STDSML) GOTO 140
        GOTO 120
140     CONTINUE
        STDSML=SDSEG(I)
        ROW=NROWS-I+1
120     CONTINUE
C
        DO 125 K=1,NCOLS
        WVC(K)=0
        STCL(K)=1
125     CONTINUE
                N=1
                FLG=0
        MID=NROWS-(BS-TS)/2 ; PREVIOUS CODE NULLED
        ROW=MID ; ******
                CALL GROW(IBUFIN,ROW ,1,NCOLS,IARRY1)  ;*****
                DO 150 J=1,NCOLS
               IF ((FLG.EQ.0).AND.(IARRY1(J).EQ.0)) STCL(N)=J
               IF (IARRY1(J).EQ.0)   GOTO 155
                GOTO 160
155     CONTINUE
                WVC(N)=WVC(N)+1
                FLG=1
160     CONTINUE
               IF ((IARRY1(J).EQ.15).AND.(FLG.EQ.1)) GOTO 165
```

```
        GOTO 150
165     CONTINUE
                N=N+1
                FLG=0
150     CONTINUE
        NUMSEG=N-1
C
        DO 90 K=1,NUMSEG
                RWVC(K)=WVC(K)
                RSTCL(K)=STCL(K)
90      CONTINUE
        DO 95 I=1,NUMSEG                        ;*
                TYPE I, " W", RWVC(I), " STC", RSTCL(I) ;*
95      CONTINUE                                ;*
        TYPE "STDSML", STDSML, " NTMP", NTMP  ," NBIG", NBIG
        TYPE "ROW= ", ROW  ;*
        DO 100 I=1,NCOLS
                EVC(I)=0
                BGVC(I)=0
                EDVC(I)=0
100     CONTINUE
C
C  EDGES OF SIGNIFICANT VCL'S IS THEN STORED IN IVCL ARRAY
        DO 200 J=1,20
        DO 210 K=1,10
        IVCL(J,K)=0
210     CONTINUE
200     CONTINUE
C
        DO 110 I=1,NUMSEG
        PEDG1=RSTCL(I)-1
        PEDG2=RSTCL(I)+RWVC(I)
        EVC(PEDG1)=10    ; STORES EDGES OF VCL'S IN EVC
        EVC(PEDG2)=10
        BGVC(PEDG1)=10   ; STORES BEGINNING OF VCL'S IN BGVC
        EDVC(PEDG2)=10   ; STORES END OF VCL'S IN EDVC
        IVCL(I,1)=PEDG1
        IVCL(I,2)=PEDG2
110     CONTINUE
C
C       RESULTS ARE PUT IN IBUFOUT FOR DISPLAY
C
        DO 250 I=1,NROWS
        DO 260 J=1,NCOLS
                IARRY2(J)=15
                IF (EVC(J).GE.I) IARRY2(J)=0
260     CONTINUE
        CALL PROW(IBUFOUT,NROWS-I+1,1,NCOLS,IARRY2)
250     CONTINUE

        TYPE "EDVCL DONE"
        RETURN
        END
```

```
C*****************************************************************
C  SUBROUTINE NAME: FIND HEIGHT OF VCL'S (HTVCL.FR)
C  WRITTEN BY: 1LT. DAVID V. SOBOTA
C  PURPOSE: THIS ROUTINE FINDS THE MAX. AND MIN. HEIGHTS OF
C  CONTINUOUS DARK PIXELS IN THE FIRST HALF AND SECOND HALF
C  OF EACH VCL REGION DETECTED IN EDVCL.FR. THIS INFORMATION
C  IS USED TO DETECTED T'S AND DISTINQUISH I'S FROM J'S AND
C  ALSO TO GIVE SEGMENTATION CLUES IN DECALG.FR.
C*****************************************************************
C
      OVERLAY HT
      SUBROUTINE HTVCL(IBUFI,BGVC,EDVC,NROWS,NCOLS,MXHT1,
     /  MXHT2,MNHT1,MNHT2,BS,TS,IVCL)
C
      INTEGER IBUFI(300)
      INTEGER NROWS,NCOLS,START,END
      INTEGER IARRY1(128), IARRY2(128)
      INTEGER STFLG(128),STZD(128)
      INTEGER STZU(128), NBZ
      INTEGER BGVC(128),EDVC(128)
      INTEGER MXROW, MNROW, MNHT1(128), MXHT1(128)
      INTEGER MNHT2(128),MXHT2(128), MDD
      INTEGER N,BS,TS,IVCL(20,10)
C
      DO 10 I=1,NCOLS ; INITIALIZE PARAMETERS
      STFLG(I)=0
      STZD(I)=0
      MNHT1(I)=0
      MXHT1(I)=0
      MNHT2(I)=0
      MXHT2(I)=0
10    CONTINUE
C
C     THIS LOOP FINDS THE BEGINNING AND END OF EACH VCL AND
C     THE MIDDLE OF EACH VCL.
C
      N=1
      DO 15 K=1,NCOLS
            IF (BGVC(K).EQ.10) START=K
            IF (EDVC(K).NE.10) GOTO 15
            END=K
            MXROW=256
            MNROW=1
            MDD=INT((END-START+1)/2)     ; MIDDLE OF VCL
C
C     THIS LOOP FINDS THE MAX. FOR THE FIRST HALF OF THE VCL
C
      DO 20 IROW=1,NROWS-1
            CALL GROW(IBUFI,      IROW  ,1,NCOLS,IARRY1)
            CALL GROW(IBUFI,      IROW+1,1,NCOLS,IARRY2)
            DO 30 ICOL=START,(START+MDD-1); 1ST HALF VCL
            NBZ=0
            IF ((IARRY1(ICOL-1).EQ.0).OR.
```

```
     /           (IARRY1(ICOL+1).EQ.0))  NBZ=1
                 IF (STFLG(ICOL).EQ.1) GOTO 30
                 IF ((IARRY1(ICOL).EQ.0).AND.
     /           ((IARRY2(ICOL).EQ.0).OR.(NBZ.EQ.1))) GOTO 35
                 GOTO 30
35       CONTINUE
                 STFLG(ICOL)=1
                 STZD(ICOL)=IROW
                 IF (STZD(ICOL).LT.MXROW) MXROW=STZD(ICOL)
30       CONTINUE
20       CONTINUE
C
         MXHT1(START)=MXROW ; MAX. HEIGHT PUT START POSITION

         DO 37 I=1,NCOLS
         STFLG(I)=0
         STZU(I)=0
37       CONTINUE
C
C    THIS LOOP FINDS THE MIN. FOR THE FIRST HALF OF THE VCL.
C
         DO 40 IROW=1,NROWS-1
                 CALL GROW(IBUFI,NROWS-IROW+1,1,NCOLS,IARRY1)
                 CALL GROW(IBUFI,NROWS-IROW  ,1,NCOLS,IARRY2)
                 DO 50 ICOL=START,(START+MDD-1)
                 NBZ=0
                 IF ((IARRY1(ICOL-1).EQ.0).OR.
     /           (IARRY1(ICOL+1).EQ.0)) NBZ=1
                 IF (STFLG(ICOL).EQ.1) GOTO 50
                 IF ((IARRY1(ICOL).EQ.0).AND.
     /           ((IARRY2(ICOL).EQ.0).OR.(NBZ.EQ.1))) GOTO 55
                 GOTO 50
55       CONTINUE
                 STFLG(ICOL)=1
                 STZU(ICOL)=NROWS-IROW
                 IF (STZU(ICOL).GT.MNROW) MNROW=STZU(ICOL)
50       CONTINUE
40       CONTINUE
C
         MNHT1(START)=MNROW ;MIN. HEIGHT PUT IN START POSTION
C
C  NUMBERS ARE STORED IN IVCL(N,3) ACCORDING TO DIFFERENT
C  HEIGHTS
         IF ((MXROW.GE.(TS-2)).AND.(MNROW.LE.(BS+2)))
       /  IVCL(N,3)=1
          IF ((MXROW.LE.(TS-2)).AND.(MNROW.LE.(BS+2)))
       /  IVCL(N,3)=2
          IF ((MXROW.GE.(TS-2)).AND.(MNROW.GE.(BS+2)))
       /  IVCL(N,3)=3
          IF ((MXROW.LE.(TS-2)).AND.(MNROW.GE.(BS+2)))
       /  IVCL(N,3)=4
C
                 MXROW=256
```

```
                MNROW=1
C
C    THIS LOOP FINDS THE MAX. HEIGHT FOR THE SECOND HALF
C    OF THE VCL
C
         DO 60 IROW=1,NROWS-1
                CALL GROW(IBUFI,       IROW  ,1,NCOLS,IARRY1)
                CALL GROW(IBUFI,       IROW+1,1,NCOLS,IARRY2)
                DO 70 ICOL=(END-MDD+1),END
                NBZ=0
                IF ((IARRY1(ICOL-1).EQ.0).OR.
     /          (IARRY1(ICOL+1).EQ.0)) NBZ=1
                IF (STFLG(ICOL).EQ.1) GOTO 70
                IF ((IARRY1(ICOL).EQ.0).AND.
     /          ((IARRY2(ICOL).EQ.0).OR.(NBZ.EQ.1))) GOTO 75
                GOTO 70
75       CONTINUE
                STFLG(ICOL)=1
                STZD(ICOL)=IROW
                IF (STZD(ICOL).LT.MXROW) MXROW=STZD(ICOL)
70       CONTINUE
60       CONTINUE
C
         MXHT2(END)=MXROW ;MAX. HEIGHT PUT IN END POSITION

         DO 77 I=1,NCOLS
         STFLG(I)=0
         STZU(I)=0
77       CONTINUE
C
C     THIS LOOP FINDS THE MIN. HT. FOR THE 2ND HALF OF THE VCL
C
         DO 80 IROW=1,NROWS-1
                CALL GROW(IBUFI,NROWS-IROW+1,1,NCOLS,IARRY1)
                CALL GROW(IBUFI,NROWS-IROW  ,1,NCOLS,IARRY2)
                DO 90 ICOL=(END-MDD+1),END
                NBZ=0
                IF ((IARRY1(ICOL-1).EQ.0).OR.
     /          (IARRY1(ICOL+1).EQ.0)) NBZ=1
                IF (STFLG(ICOL).EQ.1) GOTO 90
                IF ((IARRY1(ICOL).EQ.0).AND.
     /          ((IARRY2(ICOL).EQ.0).OR.(NBZ.EQ.1))) GOTO 95
                GOTO 90
95       CONTINUE
                STFLG(ICOL)=1
                STZU(ICOL)=NROWS-IROW
                IF (STZU(ICOL).GT.MNROW) MNROW=STZU(ICOL)
90       CONTINUE
80       CONTINUE

         MNHT2(END)=MNROW ; MIN. HEIGHT PUT IN END POSITION
C
C    NUMBERS ARE ASSIGNED TO IVCL(N,4) ACCORDING TO VARIOUS
```

```
C    HEIGHT CONDITIONS FOR 2ND HALF OF VCL
          IF ((MXROW.GE.(TS-2)).AND.(MNROW.LE.(BS+2)))
     /    IVCL(N,4)=1
          IF ((MXROW.LE.(TS-2)).AND.(MNROW.LE.(BS+2)))
     /    IVCL(N,4)=2
          IF ((MXROW.GE.(TS-2)).AND.(MNROW.GE.(BS+2)))
     /    IVCL(N,4)=3
          IF ((MXROW.LE.(TS-2)).AND.(MNROW.GE.(BS+2)))
     /    IVCL(N,4)=4
          N=N+1
15        CONTINUE
C
          TYPE "HTVCL DONE"
          RETURN
          END
```

```
C*****************************************************************
C  SUBROUTINE NAME: DETECT O'S,U'S,ARCHES, & E'S  (DTOUAE.FR)
C  WRITTEN BY: 1LT. DAVID V. SOBOTA
C  PURPOSE: THIS ROUTINE DETECTS THE PRESENCE OF LETTERS BY
C  FINDING THE REGIONS OF ZEROS( LACK OF DARK PIXELS ) NEAR
C  VCL'S AND HORIZONTAL LINES.
C  O :DEFINED AS LINE,SPACE,LINE NEXT TO VCL'S (o,a,p,b,ETC.)
C  U :DEFINED AS SPACE AND LINE NEXT TO VCL'S (u,v,w,y, ETC.)
C  A :DEFINED AS LINE AND SPACE NEXT TO VCL'S (h,n,m,r,t )
C  E :DEFINED AS LINE,SPACE,LINE,SPACE,LINE (a,e,g,s,z )
C  ( 8-REGIONS IN THESIS = E-REGIONS IN SOFTWARE )
C*****************************************************************
C
      OVERLAY DT
      SUBROUTINE DTOUAE(IBUFI,NROWS,NCOLS,TS,BS,IBUFO1,
     /  IBUFO2,XIOR,XIUR,XIUR,XIER,IOR,IUR,IAR,IER)
C
      INTEGER IBUFI(300)
      INTEGER IARRY1(128), IARRY2(128)
      INTEGER STFLG(128),STZD(128),EDZD(128)
      INTEGER STCTU(128),STZU(128),EDZU(128)
      INTEGER IBUFO1(300), IBUFO2(300)
      INTEGER STHL,ENHL,MAX,SFLG, MNHT
      INTEGER NZO(128), NZU(128), NZA(128), TS, BS, PMTSBS
C     INTEGER RNZO(128),RNZU(128),RNZA(128)
      INTEGER BGO(128),BGU(128),BGA(128), BGE(128)
      INTEGER EDO(128),EDU(128),EDA(128), EDE(128)
      INTEGER XEDO(128),XEDU(128),XEDA(128), XEDE(128)
      INTEGER XBGO(128),XBGU(128),XBGA(128), XBGE(128)
      INTEGER STOU(128), STOD(128)
      INTEGER IOR(20,3),IUR(20,3),IAR(20,3),IER(20,3)
      INTEGER XIOR(20,3),XIUR(20,3),XIAR(20,3),XIER(20,3)
C
      DO 10 I=1,NCOLS ; INITIALIZE SCAN DOWN COL. VAR.
      STFLG(I)=0 ; FLAG INDICATE DARK PIXELS ENCOUNTERED
      STZD(I)=0  ; INDICATES ROW WHITE PIXELS(ZEROS) BEGIN
      EDZD(I)=0  ; INDICATES ROW WHITE PIXELS(ZEROS) END
      STOD(I)=0
10    CONTINUE
C
C   THIS LOOP SCANS DOWN WORD TO FIND STZD AND EDZD FOR EACH
C   COLUMN IN FOLLOWING CODE: WHITE PIXEL-15, BLACK PIXEL-0
C
      DO 20 IROW=1,NROWS-1
            CALL GROW(IBUFI,      IROW  ,1,NCOLS,IARRY1 )
            CALL GROW(IBUFI,      IROW+1,1,NCOLS,IARRY2 )
              DO 30 ICOL=1,NCOLS
              IF (EDZD(ICOL).NE.0) GOTO 30
              NBZ=0
              IF ((IARRY1(ICOL-1).EQ.0).OR.
     /        (IARRY1(ICOL+1).EQ.0)) NBZ=1 ; PIXELS ARE DARK
C             IF ((IARRY1(ICOL).EQ.0).AND.((IARRY2(ICOL)
     /        .EQ.0).OR.(NBZ.EQ.1))) STFLG(ICOL)=1
```

```
                IF (.NOT.((IARRY1(ICOL).EQ.0).AND.
     /          ((IARRY2(ICOL).EQ.0).OR.(NBZ.EQ.1)))) GOTO 22
                  IF (STOD(ICOL).EQ.0) STOD(ICOL)=IROW
                  STFLG(ICOL)=1
22                CONTINUE
           IF ((IARRY2(ICOL).EQ.15).AND.
     /     (IARRY1(ICOL).EQ.15).AND.(STFLG(ICOL).EQ.1)) GOTO 25

           GOTO 27
25         CONTINUE
           STZD(ICOL)=IROW
           STFLG(ICOL)=0
           GOTO 30
27         CONTINUE
           IF ((IARRY1(ICOL).EQ.0).AND.((IARRY2(ICOL).EQ.0).OR.
     /     (NBZ.EQ. 1)).AND.(STZD(ICOL) .NE.0)) EDZD(ICOL)=IROW-1
30         CONTINUE
20         CONTINUE
C
C          THIS DOES THE SAME THING AS THE LOOP ABOVE BUT IN AN
C          UPWARD DIRECTION.
C
           DO 35 I=1,NCOLS
           STFLG(I)=0
           STZU(I)=0
           EDZU(I)=0
           STOU(I)=0
35         CONTINUE
C
           DO 40 IROW=1,NROWS-1
                 CALL GROW(IBUFI,NROWS-IROW+1,1,NCOLS,IARRY1 )
                 CALL GROW(IBUFI,NROWS-IROW  ,1,NCOLS,IARRY2 )
                   DO 50 ICOL=1,NCOLS
                   IF (EDZU(ICOL).NE.0) GOTO 50
                   NBZ=0
                   IF ((IARRY1(ICOL-1).EQ.0).OR.(IARRY1(ICOL+1)
     /             .EQ.0)) NBZ=1
C                  IF ((IARRY1(ICOL).EQ.0).AND.((IARRY2(ICOL)
     /             .EQ.0).OR.(NBZ.EQ.1))) STFLG(ICOL)=1
                 IF (.NOT.((IARRY1(ICOL).EQ.0).AND.((IARRY2(ICOL)
     /           .EQ.0).OR.(NBZ.EQ.1)))) GOTO 42
                   IF (STOU(ICOL).EQ.0) STOU(ICOL)=NROWS-IROW+1
                   STFLG(ICOL)=1
42                 CONTINUE
           IF ((IARRY2(ICOL).EQ.15).AND.
     /     (IARRY1(ICOL).EQ.15).AND.(STFLG(ICOL).EQ.1)) GOTO 45
           GOTO 47
45         CONTINUE
           STZU(ICOL)=NROWS-IROW + 1
           STFLG(ICOL)=0
           GOTO 50
47         CONTINUE
           IF ((IARRY1(ICOL).EQ.0).AND.((IARRY2(ICOL).EQ.0).OR.
```

```
     /   (NBZ.EQ. 1)).AND.(STZU(ICOL) .NE.0))
     /                              EDZU(ICOL)=NROWS-IROW+2
50       CONTINUE
40       CONTINUE
C
         DO 60 I=1,NCOLS ; INITIALIZE NUMBER OF ZEROS ARRAYS
         NZO(I)=0        ; INDICATES NUMBER OF "O" ZEROS
         NZU(I)=0        ; INDICATES NUMBER OF "U" ZEROS
         NZA(I)=0        ; INDICATES NUMBER OF "A" ZEROS
60       CONTINUE
C
         PMTSBS=INT((TS+BS)/2) ; POSITION MID. TS AND BS
         DO 70 I=1,NCOLS       ; TS= TOP OF SMALL LETTERS
C                              ; BS= BOTTOM OF SMALL LETTERS
C        TYPE  I, " STZU", STZU(I), " STZD" , STZD(I)
         IF (STZU(I).GT. STZD(I)) NZO(I)=(STZU(I)-STZD(I))+1
C        IF (.NOT.(STZD(I).GT.STZU(I))) GOTO 70
C        IF ((STZD(I).LT.( BS-2 )).AND.(EDZD(I).EQ.0))
C    /   NZA(I)=BS-STZD(I)
         IF ((STOU(I).LE.( BS-2 )).AND.(STOU(I).GT.0))
     /   NZA(I)=BS-2-STOU(I) ; STOD-STARTING DARK PIXEL DOWN
C        IF ((STZU(I).GT.( TS+2 )).AND.(EDZU(I).EQ.0))
C    /   NZU(I)=STZU(I)-TS
         IF ((STOD(I).GE.( TS+2 )).AND.(STOD(I).GT.0))
     /   NZU(I)=STOD(I)-(TS+2) ; STOU-STARTING DARK PIXEL UP
70       CONTINUE
C
C    DELOUA DELETES ALL NZO,NZU, OR NZA'S WHICH DO NOT HAVE
C    A MIN. WIDTH OR A MIN. HEIGHT .
C    TO WORK IN ALL CASES,THESE PARAMETERS WOULD HAVE TO BE
C    SCALED TO LETTER DIMENSIONS.
C
C        MINWD=2    ,    MINHT=(BS-TS)/2
         MNHT=(BS-TS)/2
         CALL DELOUA(NZO,NCOLS,2,MNHT,3,BGO,EDO)
         CALL DELOUA(NZA,NCOLS,2,MNHT,3,BGA,EDA)
         CALL DELOUA(NZU,NCOLS,2,MNHT,3,BGU,EDU)
C
C    DTEG TAKES ALL O-REGIONS AND DETECTS WHETHER THEY MEET
C    THE CONDITIONS FOR SIMPLE E-REGIONS,OR E-REGIONS WHICH
C    EXTEND BELOW BS.
C
         CALL DTEG(IBUFI,BGO,EDO,NROWS,NCOLS,TS,BS,BGE,EDE,
     /   IOR,IER) ;***
         CALL DTA(IBUFI,BGA,EDA,NROWS,NCOLS,TS,BS,IAR) ;***
         CALL DTU(IBUFI,BGU,EDU,NROWS,NCOLS,TS,BS,IUR) ;***
C
C        MINWD=50   ,    MINHT=0    ;  NO THRESHOLD
         MNHT=0
         CALL DELOUA(NZO,NCOLS,20,MNHT,3,XBGO,XEDO)
         CALL DELOUA(NZA,NCOLS,20,MNHT,3,XBGA,XEDA)
         CALL DELOUA(NZU,NCOLS,20,MNHT,3,XBGU,XEDU)
C
```

```
C     DTEG TAKES ALL O-REGIONS AND DETECTS WHETHER THEY MEET
C     THE CONDITIONS FOR SIMPLE E-REGIONS,OR E-REGIONS WHICH
C     EXTEND BELOW BS.
C
          CALL DTEG(IBUFI,XBGO,XEDO,NROWS,NCOLS,TS,BS,XBGE,XEDE
     /    ,XIOR,XIER) ;***
          CALL DTA(IBUFI,XBGA,XEDA,NROWS,NCOLS,TS,BS,XIAR) ;***
          CALL DTU(IBUFI,XBGU,XEDU,NROWS,NCOLS,TS,BS,XIUR) ;***
          DO 110 I=1,NROWS
          DO 120 J=1,NCOLS
                  IARRY1(J)=15
                  IF (NZO(J).GE.I) IARRY1(J)=0
120       CONTINUE
          CALL PROW(IBUFO1,NROWS-I+1,5*2+120,NCOLS,IARRY1)
110       CONTINUE
C
          DO 130 I=1,20
          DO 140 J=1,NCOLS
                  IARRY1(J)=15
                  IF (BGO(J).GE.I) IARRY1(J)=0
                  IF (EDO(J).GE.I) IARRY1(J)=0
                  IF (BGE(J).GE.I) IARRY1(J)=0
                  IF (EDE(J).GE.I) IARRY1(J)=0
140       CONTINUE
          CALL  PROW(IBUFO1,NROWS+20-I+1,5*2+120,NCOLS,IARRY1)
130       CONTINUE

          DO 150 I=1,NROWS
          DO 160 J=1,NCOLS
                  IARRY1(J)=15
                  IF (NZA(J).GE.I) IARRY1(J)=0
160       CONTINUE
          CALL PROW(IBUFO1,2*NROWS+20-I+1,5*2+120,NCOLS,IARRY1)
150       CONTINUE

          DO 170 I=1,20
          DO 180 J=1,NCOLS
                  IARRY1(J)=15
                  IF (BGA(J).GE.I) IARRY1(J)=0
                  IF (EDA(J).GE.I) IARRY1(J)=0
180       CONTINUE
          CALL PROW(IBUFO1,2*NROWS+40-I+1,5*2+120,NCOLS,IARRY1)
170       CONTINUE

          DO 190 I=1,NROWS
          DO 200 J=1,NCOLS
                  IARRY1(J)=15
                  IF (NZU(J).GE.I) IARRY1(J)=0
200       CONTINUE
          CALL PROW(IBUFO1,3*NROWS+40-I+1,5*2+120,NCOLS,IARRY1)
190       CONTINUE
          DO 210 I=1,20
          DO 220 J=1,NCOLS
```

```
                IARRY1(J)=15
                IF (BGU(J).GE.I) IARRY1(J)=0
                IF (EDU(J).GE.I) IARRY1(J)=0
220         CONTINUE
            CALL PROW(IBUFO1,3*NROWS+60-I+1,5*2+120,NCOLS,IARRY1)
210         CONTINUE
            RETURN
            END
```

```
C*****************************************************************
C  SUBROUTINE NAME: DELETE O,U,AND A REGIONS (DELOUA.FR)
C  WRITTEN BY: 1LT. DAVID V. SOBOTA
C  PURPOSE: THIS ROUTINE FINDS THE EDGES OF SIGNIFICANT O,A,U
C  REGIONS BY DELETING O,U,E, REGIONS WHICH DO NOT HAVE A MIN.
C  HEIGHT OF MINHT OF DO NOT MIN. WIDTH OF MINWD. THE EDGES
C  OF THE REMAINING REGIONS ARE THEN FOUND & GIVEN A WEIGHT
C  OF WT FOR DISPLAY IN ORDER TO DISTINQUISH THE DIFFERENT
C  REGIONS . THIS ROUTINE IS CALLED FOR O,A,AND U REGIONS.
C*****************************************************************
C
      SUBROUTINE DELOUA (NZ,NCOLS,MINWD,MINHT,WT,STCL,EDCL)
C
      INTEGER MAX, SFLG, START, END , NCOLS
      INTEGER NZ(128), RNZ(128), MINHT, MINWD
      INTEGER FLG, STCL(128), EDCL(128)
      INTEGER WT
C
      MAX=0
      SFLG=0
      START=0
      END=0
      DO 100  I=1,NCOLS
              RNZ(I)=NZ(I)
              IF (.NOT.((SFLG.EQ.0).AND.(NZ(I).GE.1)))
     /        GOTO 110
              START=I
              SFLG=1
110   CONTINUE
              IF (NZ(I).GT.MAX) MAX=NZ(I)
              IF ((SFLG.EQ.1).AND.(NZ(I).EQ.0)) END=I-1
              IF (END.EQ.0) GOTO 100
              DO 120 K=START,END
              IF (((END-START).LE.MINWD).OR.(MAX.LE.MINHT))
     /        RNZ(K)=0 ;****
120           CONTINUE
              SFLG=0
              MAX=0
              START=0
              END=0
100   CONTINUE
C
C     THE FOLLOWING FINDS THE EDGES OF THE HOLES, U'S, OR
C     ARCHES THAT SURVIVE THE CONDITIONS SET BY "DELOUA"
C
      DO 10 I=1,NCOLS
      STCL(I)=0
      EDCL(I)=0
10    CONTINUE
C
      FLG=0
      DO 20 I=1,NCOLS
         IF (.NOT.((FLG.EQ.0).AND.(RNZ(I).GT.0))) GOTO 25
```

```
                STCL(I)=WT
                FLG=1
25        CONTINUE
                IF (.NOT.((RNZ(I).EQ. 0).AND.(FLG.EQ.1)))
     /          GOTO 35
                EDCL(I-1)=WT
                FLG=0
35        CONTINUE
20        CONTINUE
C

          RETURN
          END
```

```
C*****************************************************************
C  SUBROUTINE NAME: DETECT A-REGIONS
C  WRITTEN BY: 1LT. DAVID V. SOBOTA
C  PURPOSE: THIS ROUTINE SCANS FROM BELOW THE LINE OF PRINT
C  LOOKS FOR A CONCENTRATION OF DARK PIXELS. IF THE PIXELS
C  ARE ABOVE THE LINE OF PRINT (BS) BY SOME THRESHOLD THEN
C  THIS REGION IS DEFINED AS A A-REGION.
C*****************************************************************

      SUBROUTINE DTA(IBUFI,BGA,EDA,NROWS,NCOLS,TS,BS,IAR)
C
      INTEGER IBUFI(300)
      INTEGER NROWS,NCOLS,START,END
      INTEGER IARRY1(128), IARRY2(128)
      INTEGER STFLG(128),STZD(128),EDZD(128)
      INTEGER STCTU(128),STZU(128),EDZU(128)
      INTEGER STHL,ENHL,MAX,MIN,SFLG
      INTEGER TS, BS
      INTEGER BGA(128),EDA(128)
      INTEGER WDTH,N,K,STOD(128),STOU(128),MID
      INTEGER IAR(20,3)
C
      DO 10 I=1,NCOLS ; INITIALIZE PARAMETERS
      STFLG(I)=0      ; MARKS HORIZ. LINE FOUND DOWN SCAN
      STZD(I)=0       ; STARTING ZERO(WHITE PIXEL) DOWN
      EDZD(I)=0       ; ENDING ZERO DOWN
      STOD(I)=0
10    CONTINUE
C
      DO 15 K=1,NCOLS
            IF (BGA(K).EQ.3) START=K
            IF (EDA(K).NE.3) GOTO 15
            END=K
C
      DO 35 I=1,NCOLS  ; INIT. PARAMETERS FOR UPWARD SCAN
      STFLG(I)=0
      STZU(I)=0
      EDZU(I)=0
      STOU(I)=0
35    CONTINUE
C
      DO 40 IROW=1,NROWS-1            ;SCANS FROM BELOW
            CALL GROW(IBUFI,NROWS-IROW+1,1,NCOLS,IARRY1 )
            CALL GROW(IBUFI,NROWS-IROW  ,1,NCOLS,IARRY2 )
            DO 50 ICOL=START,END
            IF (EDZU(ICOL).NE.0) GOTO 50
            NBZ=0
            IF ((IARRY1(ICOL-1).EQ.0).OR.(IARRY1(ICOL+1)
     /      .EQ.0)) NBZ=1
C           IF ((IARRY1(ICOL).EQ.0).AND.((IARRY2(ICOL)
     /      .EQ.0).OR.(NBZ.EQ.1))) STFLG(ICOL)=1
            IF (.NOT.((IARRY1(ICOL).EQ.0).AND.
     /      ((IARRY2(ICOL).EQ.0).OR.(NBZ.EQ.1)))) GOTO 42
```

```
                IF (STOU(ICOL).EQ.0) STOU(ICOL)=NROWS-IROW+1
                STFLG(ICOL)=1
42      CONTINUE
        IF ((IARRY2(ICOL).EQ.15).AND.
     /  (IARRY1(ICOL).EQ.15).AND.(STFLG(ICOL).EQ.1)) GOTO 45
        GOTO 47
45      CONTINUE
        STZU(ICOL)=NROWS-IROW
        STFLG(ICOL)=0
C       GOTO 50
47      CONTINUE
        IF ((IARRY1(ICOL).EQ.0).AND.((IARRY2(ICOL).EQ.0).OR.
     /  (NBZ.EQ. 1)).AND.(STZU(ICOL) .NE.0))
     /                              EDZU(ICOL)=NROWS-IROW+2
50      CONTINUE
40      CONTINUE
C
        WDTH=END-START+1
C       TYPE "START=",START," END=",END
        MAX=NROWS
        MIN=1
        MID=INT((BS+TS)/2)
        DO 70 I=START,END
C       TYPE "EDZU",EDZU(I)," EDZD",EDZD(I)
        IF ((STOU(I).LT.MAX).AND.(STOU(I).NE.0)) MAX=STOU(I)
        IF (STOU(I).GT.MIN) MIN=STOU(I)
70      CONTINUE
C       IF (.NOT.((MAX.LE.MID).AND.(MIN.GT.(BS+1)))) GOTO 75
        IF (.NOT.((MAX.LE.(TS-2)))) GOTO 75
        BGA(START)=6     ; INDICATE ABOVE TS
        EDA(END)=6
75      CONTINUE
C       IF (.NOT.((MAX.LT. MID  ).AND.(MIN.LT.(BS+1))))
     /  GOTO 80
        IF (.NOT.((MAX.GT.(TS-2)))) GOTO 80
        BGA(START)=9     ; INDICATE TS-BS
        EDA(END)=9
80      CONTINUE
15      CONTINUE
C
        DO 110 J=1,20
        DO 120 K=1,3
        IAR(J,K)=0
120     CONTINUE
110     CONTINUE
C
C
        N=1
        DO 100 K=1,NCOLS
        IF (BGA(K).GT.0) START=K
        IF (EDA(K).EQ.0) GOTO 100
        END=K
        IAR(N,1)=START
```

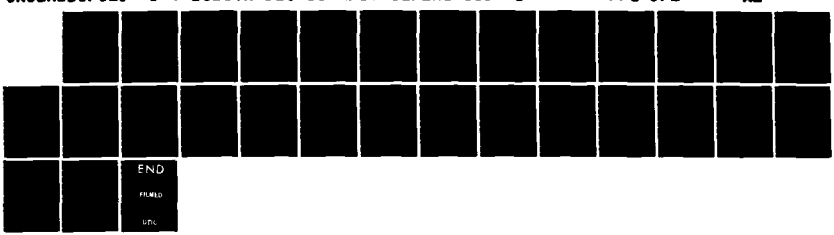
```
          IAR(N,2)=END
          IAR(N,3)=BGA(START)
          N=N+1
100       CONTINUE
C
          TYPE "DTA DONE"
          RETURN
          END
```

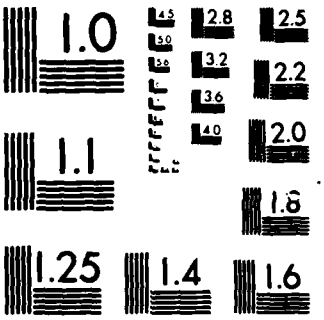
```
C*****************************************************************
C  SUBROUTINE NAME: DETECT U REGIONS
C  WRITTEN BY: 1LT. DAVID V. SOBOTA
C  PURPOSE: THIS ROUTINE SCANS DOWN FROM ABOVE THE WORD AND
C  LOOKS FOR A CONCENTRATION OF DARK PIXELS.  IF THE PIXELS
C  ARE  BELOW  TS  FOR SOME THRESHOLD THEN  THIS  REGION  IS
C  DEFINED AS A U-REGION.
C*****************************************************************

        SUBROUTINE DTU(IBUFI,BGU,EDU,NROWS,NCOLS,TS,BS,IUR)
C
        INTEGER IBUFI(300)
        INTEGER NROWS,NCOLS,START,END
        INTEGER IARRY1(128), IARRY2(128)
        INTEGER STFLG(128),STZD(128),EDZD(128)
        INTEGER STCTU(128),STZU(128),EDZU(128)
        INTEGER STHL,ENHL,MAX,MIN,SFLG
        INTEGER TS, BS
        INTEGER BGU(128),EDU(128)
        INTEGER WDTH,N,K,STOD(128),STOU(128),MID
        INTEGER IUR(?0,3)
C
        DO 10 I=1,NCOLS ; INITIALIZE PARAMETERS
        STFLG(I)=0      ; MARKS HORIZ. LINE FOUND DOWN SCAN
        STZD(I)=0       ; STARTING ZERO(WHITE PIXEL) DOWN
        EDZD(I)=0       ; ENDING ZERO DOWN
        STOD(I)=0
10      CONTINUE
C
        DO 15 K=1,NCOLS                      ;FINDS START AND END
                IF (BGU(K).EQ.3) START=K     ; IN A REGIONS
                IF (EDU(K).NE.3) GOTO 15
                END=K
        DO 20 IROW=1,NROWS-1                          ; SCANS DOWN
                CALL GROW(IBUFI,    IROW  ,1,NCOLS,IARRY1 )
                CALL GROW(IBUFI,    IROW+1,1,NCOLS,IARRY2 )
                DO 30 ICOL=START,END
                IF (EDZD(ICOL).NE.0) GOTO 30
                NBZ=0
                IF ((IARRY1(ICOL-1).EQ.0).OR.(IARRY1(ICOL+1)
     /          .EQ.0)) NBZ=1    ; NEIGHBORING PIXELS ARE DARK
C               IF ((IARRY1(ICOL).EQ.0).AND.((IARRY2(ICOL)
     /          .EQ.0).OR.(NBZ.EQ.1))) STFLG(ICOL)=1
                IF (.NOT.((IARRY1(ICOL).EQ.0).AND.
     /          ((IARRY2(ICOL).EQ.0).OR.(NBZ.EQ.1)))) GOTO 22
                IF (STOD(ICOL).EQ.0) STOD(ICOL)=IROW
                STFLG(ICOL)=1
22      CONTINUE
        IF ((IARRY2(ICOL).EQ.15).AND.
     /  (IARRY1(ICOL).EQ.15).AND.(STFLG(ICOL).EQ.1)) GOTO 25

        GOTO 27
25      CONTINUE
```

AD-A163 938 WORD SEGMENTATION ALGORITHM BASED ON RECOGNITION OF 2/2
LETTER FEATURES(U) AIR FORCE INST OF TECH
WRIGHT-PATTERSON AFB OH SCHOOL OF ENGINEERING
UNCLASSIFIED D V SOBOTA DEC 85 AFIT/GE/ENG/85D-43 F/G 9/2 NL





```
          STZD(ICOL)=IROW +1
          STFLG(ICOL)=0
C         GOTO 30
27        CONTINUE
          IF ((IARRY1(ICOL).EQ.0).AND.((IARRY2(ICOL).EQ.0).OR.
     /   (NBZ.EQ. 1)).AND.(STZD(ICOL) .NE.0))
     /   EDZD(ICOL)=IROW-1     ;MARKS END OF ZERO REGION
30        CONTINUE
20        CONTINUE
C
          WDTH=END-START+1
C         TYPE "START=",START," END=",END
          MAX=NROWS
          MIN=1
          MID=INT((BS+TS)/2)
          DO 70 I=START,END
C         TYPE "EDZU",EDZU(I)," EDZD",EDZD(I)
          IF ((STOD(I).LT.MAX).AND.(STOD(I).NE.0)) MAX=STOD(I)
          IF (STOD(I).GT.MIN) MIN=STOD(I)
70        CONTINUE
C         IF (.NOT.((MAX.LE.MID).AND.(MIN.GT.(BS+1)))) GOTO 75
          IF (.NOT.((MIN.GT.(BS+1)))) GOTO 75
          BGU(START)=6
          EDU(END)=6
75        CONTINUE
C         IF (.NOT.((MAX.LT.MID).AND.(MIN.LT.(BS+1)))) GOTO 80
          IF (.NOT.((MIN.LT.(BS+1)))) GOTO 80
          BGU(START)=9
          EDU(END)=9
80        CONTINUE
15        CONTINUE
C
          DO 110 J=1,20
          DO 120 K=1,3
          IUR(J,K)=0
120       CONTINUE
110       CONTINUE
C
          N=1
          DO 100 K=1,NCOLS
          IF (BGU(K).GT.0) START=K
          IF (EDU(K).EQ.0) GOTO 100
          END=K
          IUR(N,1)=START
          IUR(N,2)=END
          IUR(N,3)=BGU(START)
          N=N+1
100       CONTINUE

C
          TYPE "DTU DONE"
          RETURN
          END
```

```
C*****************************************************************
C  SUBROUTINE NAME: DETECT "E" AND "G"'S  ( ALSO s,z, AND a )
C  WRITTEN BY: 1LT. DAVID V. SOBOTA
C  PURPOSE: THIS ROUTINE DETECTS THE PRESENCE OF LETTERS WHICH
C  HAVE AT LEAST THREE HORIZONTAL LINE IN THEM BY LOOKING AT
C  LETTERS THAT FIRST MEET THE CONDITIONS FOR AN "O" SET IN
C  DTOUAE.FR. IT THEN SCANS FROM ABOVE AND  BELOW LOOKING FOR
C  THE FIRST AND SECOND HORIZONTAL LINE. IF THE BEGINNING OF
C  THE SECOND HORIZONTAL LINE ON THE ABOVE AND BELOW SCAN
C  DON'T OVERLAP THEN A e/g/s IS DETECTED.
C*****************************************************************

      SUBROUTINE DTEG(IBUFI,BGO,EDO,NROWS,NCOLS,TS,BS,BGE,
     /  EDE,IOR,IER)
C
      INTEGER IBUFI(300)
      INTEGER NROWS,NCOLS,START,END
      INTEGER IARRY1(128), IARRY2(128)
      INTEGER STFLG(128),STZD(128),EDZD(128)
      INTEGER STCTU(128),STZU(128),EDZU(128)
      INTEGER STHL,ENHL,SFLG
      INTEGER TS, BS
      INTEGER BGO(128),EDO(128)
      INTEGER BGE(128)
      INTEGER EDE(128),WDTH,N,K
      INTEGER MIN,MAX,MID
      INTEGER IER(20,3),IOR(20,3)
C
      DO 10 I=1,NCOLS ; INITIALIZE PARAMETERS
      STFLG(I)=0   ; MARKS HORIZ. LINE FOUND IN DOWN SCAN
      STZD(I)=0    ; STARTING ZERO(WHITE PIXEL) DOWN
      EDZD(I)=0    ; ENDING ZERO DOWN
      EDE(I)=0     ; MARKS END OF E REGION
      BGE(I)=0     ; MARKS START OF E REGION
10    CONTINUE
C
      DO 15 K=1,NCOLS   FINDS START AND END O-REGIONS
            IF (BGO(K).EQ.3) START=K
            IF (EDO(K).NE.3) GOTO 15
            END=K
      DO 20 IROW=1,NROWS-1                    ; SCANS DOWN
            CALL GROW(IBUFI,    IROW  ,1,NCOLS,IARRY1 )
            CALL GROW(IBUFI,    IROW+1,1,NCOLS,IARRY2 )
            DO 30 ICOL=START,END
            IF (EDZD(ICOL).NE.0) GOTO 30
            NBZ=0
            IF ((IARRY1(ICOL-1).EQ.0).OR.(IARRY1(ICOL+1)
     /      .EQ.0)) NBZ=1 ; NEIGHBORING PIXELS ARE DARK
            IF ((IARRY1(ICOL).EQ.0).AND.((IARRY2(ICOL)
     /      .EQ.0).OR.(NBZ.EQ.1))) STFLG(ICOL)=1
      IF ((IARRY2(ICOL).EQ.15).AND.
     /  (IARRY1(ICOL).EQ.15).AND.(STFLG(ICOL).EQ.1)) GOTO 25
```

```
          GOTO 27
25        CONTINUE
          STZD(ICOL)=IROW    ;****
          STFLG(ICOL)=0
27        CONTINUE
          IF ((IARRY1(ICOL).EQ.0).AND.((IARRY2(ICOL).EQ.0).OR.
     /   (NBZ.EQ. 1)).AND.(STZD(ICOL) .NE.0))
     /   EDZD(ICOL)=IROW-1     ;MARKS END OF ZERO REGION
30        CONTINUE
20        CONTINUE
C
          DO 35 I=1,NCOLS ; INIT. PARAMETERS FOR UPWARD SCAN
          STFLG(I)=0
          STZU(I)=0
          EDZU(I)=0
35        CONTINUE
C
          DO 40 IROW=1,NROWS-1              ;SCANS FROM BELOW
                  CALL GROW(IBUFI,NROWS-IROW+1,1,NCOLS,IARRY1 )
                  CALL GROW(IBUFI,NROWS-IROW  ,1,NCOLS,IARRY2 )
                  DO 50 ICOL=START,END
                  IF (EDZU(ICOL).NE.0) GOTO 50
                  NBZ=0
                  IF ((IARRY1(ICOL-1).EQ.0).OR.(IARRY1(ICOL+1)
     /            .EQ.0)) NBZ=1
                  IF ((IARRY1(ICOL).EQ.0).AND.((IARRY2(ICOL)
     /            .EQ.0).OR.(NBZ.EQ.1))) STFLG(ICOL)=1
          IF ((IARRY2(ICOL).EQ.15).AND.
     /   (IARRY1(ICOL).EQ.15).AND.(STFLG(ICOL).EQ.1)) GOTO 45
          GOTO 47
45        CONTINUE
          STZU(ICOL)=NROWS-IROW+1   ; ****
          STFLG(ICOL)=0
47        CONTINUE
          IF ((IARRY1(ICOL).EQ.0).AND.((IARRY2(ICOL).EQ.0).OR.
     /   (NBZ.EQ. 1)).AND.(STZU(ICOL) .NE.0))
     /                              EDZU(ICOL)=NROWS-IROW+2
50        CONTINUE
40        CONTINUE
C
          N=0 ;INIT. COUNT EDZU GT. EDZD (INDICATES E-REGIONS)
          M=0 ;INIT. COUNT 2ND O-REGION BELOW BS (OCCURS IN g)
          WDTH=END-START+1
          MAX=NROWS
          MIN=1
          MID=INT((BS+TS)/2)
C         TYPE "START=",START," END=",END
          DO 70 I=START,END
C         TYPE "EDZU",EDZU(I)," EDZD",EDZD(I)
          IF (STZU(I).GE.(BS+2)) M=M+1
          IF (EDZU(I).GT.EDZD(I)) N=N+1   ; COUNTS # OF TIMES
C                                         ; AT LEAST 2 O-REGION
C                                         ; DETECTED
```

```
          IF (.NOT.(START.EQ.END)) GOTO 69
          TYPE "M=",M," N=",N," BS=",BS," TS=",TS
          TYPE "EDZU",EDZU(I)," EDZD",EDZD(I)
          TYPE "STZU",STZU(I)," STZD",STZD(I)
69        CONTINUE
          IF ((STZD(I).LT.MAX).AND.(STZD(I).NE.0)) MAX=STZD(I)
          IF (STZU(I).GT.MIN) MIN=STZU(I)
70        CONTINUE
          IF (.NOT.((MAX.LT.(TS-2)).AND.(MIN.LT.(MID-1))))
         /  GOTO 60
          BGO(START)=1
          EDO(END)=1
60        CONTINUE
          IF (.NOT.((MAX.LT.(TS-3)).AND.(MIN.GT.(MID+3))))
         /  GOTO 65
          BGO(START)=3
          EDO(END)=3
65        CONTINUE
          IF (.NOT.((MAX.LT.MID).AND.(MIN.GT.(BS+3)))) GOTO 75
          BGO(START)=6
          EDO(END)=6
75        CONTINUE
          IF (.NOT.((MAX.LT.(MID-2)).AND.(MIN.GT.(MID+2))))
         /  GOTO 80
          BGO(START)=9
          EDO(END)=9
80        CONTINUE
C   IF 3 HORIZONTAL LINES DETECTED FOR A THIRD OF THE LETTER
C   WIDTH THEN A E-REGION IS DETECTED.
          IF ((N.LT.(WDTH/3)).OR.(N.LT.3)) GOTO 76
          EDE(END)=12
          BGE(START)=12
          BGO(START)=0
          EDO(END)=0
76        CONTINUE
          IF ((M.LT.(WDTH/3)).OR.(M.LT.3).OR.(BGE(START).EQ.0))
         /  GOTO 95
          BGE(START)=15
          EDE(END)=15
          BGO(START)=0
          EDO(END)=0
95        CONTINUE
15        CONTINUE
C
          DO 110 J=1,20
          DO 120 K=1,3
          IOR(J,K)=0
          IER(J,K)=0
120       CONTINUE
110       CONTINUE
C
          N=1
          DO 100 K=1,NCOLS
```

```
            IF (BGO(K).GT.0) START=K
            IF (EDO(K).EQ.0) GOTO 100
            END=K
            IOR(N,1)=START
            IOR(N,2)=END
            IOR(N,3)=BGO(START)
            N=N+1
100         CONTINUE
            N=1
            DO 130 K=1,NCOLS
            IF (BGE(K).GT.0) START=K
            IF (EDE(K).EQ.0) GOTO 130
            END=K
            IER(N,1)=START
            IER(N,2)=END
            IER(N,3)=BGE(START)
            N=N+1
130         CONTINUE
C
            TYPE "DTEG DONE"
            RETURN
            END
```

```
C*****************************************************************
C  SUBROUTINE NAME: FIND LOCATION OF "I"'S (IFIND.FR)
C  WRITTEN BY: 1LT. DAVID V. SOBOTA
C  PURPOSE: BECAUSE THE LETTER "I" IS ONE OF THE FEW LETTERS
C  WHERE THE VCL DOES NOT OCCUR AT THE EDGE OF THE LETTER,
C  THIS SPECIAL LETTER DETECTOR HAS TO BE USED TO ACCOUNT FOR
C  THIS EXCEPTION. THIS ROUTINE WILL ALSO FIND A SMALL CASE J.
C*****************************************************************
C
      OVERLAY IF
      SUBROUTINE IFIND(NCOLS,BGVC,EDVC,VCLD,VCLU,TS,BS,IFND,
     /  BGI,EDI,IBD)
C
      INTEGER VCLD(256), VCLU(256)
      INTEGER NCOLS, START, END, SVCLD, SVCLU
      INTEGER IFND(256), BGI(256), EDI(256)
      INTEGER TS,BS,MAX,ITHR ;*******
      INTEGER BGVC(128),EDVC(128),IBD(20,2),N
C
C     START -USED TO INDICATE THE START OF A VCL
C     END -USED TO INDICATE THE END OF A VCL
      SVCLD=0 ; SUM OF PIXELS IN VCL DOWN
      SVCLU=0 ; SUM OF PIXELS IN VCL UP
C
      DO 5 I=1,NCOLS  ; INITIALIZED PARAMETERS
      IFND(I)=0 ; MARKS BEGINNING AND END OF I FOR DISPLAY
      BGI(I)=0  ; MARKS BEGINNING OF I
      EDI(I)=0  ; MARKS END OF I
5     CONTINUE
C
      DO 7 J=1,20
      DO 8 K=1,2
      IBD(J,K)=0
8     CONTINUE
7     CONTINUE
C
C   THIS LOOP FINDS THE EDGES OF EACH VCL AND THEN SUMS THE
C   VCL'S DOWN AND VCL'S UP AND COMPARES THE TWO.  IF SUM UP
C   IS GREATER THAN OR EQUAL TO TWICE SUM DOWN THAN AN LOWER
C   CASE "I" IS DETECTED.
C
      N=1
      DO 10 I=1,NCOLS
            IF (BGVC(I).EQ.10) START=I
            IF (EDVC(I).NE.10) GOTO 10
            END=I
C
      MAX=0 ;********
            DO 30 J=START,END
      IF (VCLD(J).LT.2) GOTO 25  ; DOT ON I HAS TO BE
            SVCLD=SVCLD+VCLD(J); TWO PIXELS
25    CONTINUE
            SVCLU=SVCLU+VCLU(J)
```

```
          IF (VCLU(J).GT.MAX) MAX=VCLU(J)
30        CONTINUE
          TYPE "SUMD",SVCLD, " SUMU",SVCLU
          ITHR=(BS-TS)/2+1 ;  STRAIGHT LINE THRESHOLD FOR I
C         IF ((INT(SVCLU/SVCLD)).LT.2)  GOTO 35    ;***
          IF (.NOT.(((INT(SVCLU/SVCLD)).GE.2).AND.(MAX.GT.ITHR)
     /    .AND.(MAX.LT.(BS-TS+2)))) GOTO 35   ;***
                    IFND(START)=10
                    IFND(END)=10
                    BGI(START)=10
                    EDI(END)=10
                    IBD(N,1)=START
                    IBD(N,2)=END
          N=N+1
35        CONTINUE
                    SVCLU=0
                    SVCLD=0
10        CONTINUE

          RETURN
          END
```

```
C*****************************************************************
C  SUBROUTINE NAME: FIND LOCATION LETTER LOWER CASE "T"
C  WRITTEN BY: 1LT. DAVID V. SOBOTA
C  PURPOSE: BECAUSE THIS LETTER IS ONE OF THE FEW LETTERS
C  WHERE A VCL DOES NOT OCCUR AT THE EDGE OF THE LETTER,
C  THIS SPECIAL ROUTINE IS USE TO DETECT THIS LETTER AND
C  HANDLE THIS EXCEPTION.
C*****************************************************************
C
      OVERLAY TF
      SUBROUTINE FINDT(IBUFIN,NCOLS,NROWS,EVC,TS,BS,MXHT1,
     /  MXHT2,MNHT1,MNHT2,BGT,EDT,TBD)
C
      INTEGER IBUFIN(300)
      INTEGER EVC(128)
      INTEGER LNGTH, TS, BS, END(128), START(128), NCOLS
      INTEGER SCROSS(128), ECROSS(128),TFLG
      INTEGER IARRY1(128), IARRY2(128)
      INTEGER TBEG, TEND, NROWS
      INTEGER SCOL,ECOL
      INTEGER  MAX
      INTEGER ENDTH, LTHR
      INTEGER T, M, MTOTAL, STTH
      INTEGER BGT(128),EDT(128), MDTT, THR
      INTEGER MXHT1(128),MNHT1(128), K, DIF
      INTEGER MXHT2(128),MNHT2(128), KST,KED, MN, MX
      INTEGER TSRCH,BSRCH
      INTEGER TBD(20,2),NUMT
C
      N=1       ; COUNTS NUMBER OF T'S FOUND
      DO 5 I=1,NCOLS   ; INITIALIZE T PARAMETERS
      SCROSS(I)=0    ;MARKS START OF HORIZONTAL LINE IN T
      ECROSS(I)=0    ;MARKS END OF HORIZONTAL LINE IN T
5     CONTINUE
      DO 10 I=1,NCOLS
      BGT(I)=0          ;MARKS BEGINNING OF T
      EDT(I)=0          ;MARKS END OF T
      START(I)=0        ;MARKS START OF EACH VCL
      END(I)=0          ;MARKS END OF EACH VCL
10    CONTINUE
C
C     THIS LOOP PUTS THE START & END OF EACH VCL IN AN ARRAY
C
      M=1       ; COUNTS NUMBER OF VCL'S FOUND IN EVC ARRAY
      DO 20 I=1,NCOLS
              IF (EVC(I).NE.10) GOTO 20
              IF (START(M).NE.0) GOTO 30
              START(M)=I
              GOTO 20
30    CONTINUE
              END(M)=I
              M=M+1     ; INCREMENT COUNT
20    CONTINUE
```

```
C
C    THIS LOOP LOOKS AT EACH VCL AND SEES IF IF HAS A
C    HORIZONTAL LINE THAT EXTENDS BEYOND THE VCL AROUND THE
C    "TS" ROW.
C
         MTOTAL=M-1       ; TOTAL NUMBER OF VCL'S
         DO 15 T=1,MTOTAL
         TYPE "        START", START(T), " END", END(T)
         TYPE "MX1", MXHT1(START(T))," MX2", MXHT2(END(T)),
     /   " MN1", MNHT1(START(T))," MN2",MNHT2(END(T))
C
C    MXHT1,MXHT2 CONTAIN THE MAX. HEIGHT OF CONTINUOUS PIXELS
C    IN THE FIRST HALF OF THE VCL AND THE SECOND HALF OF THE
C    VCL. THESE PARAMETERS ARE PASSED FROM HTVCL.FR
C
         THR=BS-TS
         KST=START(T)
         KED=END(T)
         MX= MXHT1(KST)
         IF (MXHT2(KED).LT.MX) MX=MXHT2(KED)
         MN= MNHT1(KST)
         IF (MNHT2(KED).GT.MN) MN=MNHT2(KED)
         DIF=MN-MX
         TYPE "(BS-TS)", THR , " (MX-MN)", DIF
         IF (DIF.LE.THR) GOTO 15 ; CHECK IF HEIGHT OF VCL BIG
                                 ; ENOUGH FOR T
         TSRCH=TS-5
         IF (TSRCH.LT.1) TSRCH=1 ; THIS PREVENTS SEARCH FROM
                                 ; EXCEEDING BUFFER LIMITS
         BSRCH=TS+5
                 DO 40 J=TSRCH,BSRCH  ; SEARCH WINDOW HORIZ.
                 LNGTH=0              ; COUNTS LENGTH OF HORIZ.
                 TFLG=0  ; INDICATES WHEN COND. FOR T IS MET
                 CALL GROW(IBUFIN,J,1,NCOLS,IARRY1)
                 CALL GROW(IBUFIN,J+1,1,NCOLS,IARRY2)
                 TBEG=START(T)-10    ; SETS UP SEARCH WINDOW
                 TEND=END(T)+10      ; FOR HORIZ. LINE
                 IF (TBEG.LE.0) TBEG=1
                 IF (TEND.GT.NCOLS) TEND=NCOLS
         TYPE "#0", "TBEG", TBEG, " TEND", TEND
C
C     THIS LOOP COUNTS LENGTH OF HORIZONTAL LINE UNTIL A
C     VERTICAL CONTINUOUS GAP OF TWO PIXELS OCCUR. IT THEN
C     CHECKS IF THE LINE EXTENDS BEYOND THE VCL TO FORM THE
C     T SHAPE.
C
                 DO 50  K= TBEG,TEND
                    IF ((IARRY1(K).EQ.15).AND.
     /              (IARRY2(K).EQ.15)) GOTO 60
                    LNGTH=LNGTH+1
         TYPE "#1", " J=", J, " K=", K, " L=", LNGTH
                    LTHR=2*(END(T)-START(T))
                    STTH=START(T)-2
```

```
                ENDTH=END(T)+2
          IF (((K-LNGTH).LT.STTH).AND.
         /  (LNGTH.GT.LTHR ).AND.(K.GT.ENDTH)) TFLG=1
          TYPE "#2"," TFLG", TFLG," J=",J," K=",K, IARRY1(K),
         /  IARRY2(K)
          GOTO 55
60        CONTINUE
          TYPE "#3"," TFLG",TFLG," L", LNGTH," J=",J," K=",K,
         /  IARRY1(K),IARRY2(K)
                IF (TFLG.NE.1)   GOTO 65
                ECROSS(N)=K-1
                SCROSS(N)=K-LNGTH
C    THE NEXT TWO STATEMENTS MAKE SURE THE DETECTED T DOES NOT
C    OVERLAP THE NEXT VCL REGION WHEN THE END OF THE CROSS IN
C    THE T IS FOUND.
          IF (((K-LNGTH).LT.END(T-1)).AND.((T-1).NE.0))
         /  SCROSS(N)=END(T-1)
          IF (((K-1).GT.START(T+1)).AND.((T+1).LE.MTOTAL))
         /  ECROSS(N)=START(T+1)
          TYPE "#4", " N=", N," ST.-T", SCROSS(N), " END-T",
         /  ECROSS(N)
                N=N+1
                GOTO 15
65        CONTINUE
                LNGTH=0
55        CONTINUE

50        CONTINUE
          TYPE "#6", " N=", N," ST.-T", SCROSS(N), " END-T",
         /  ECROSS(N)
          TYPE "#7", " J=",J," K=",K, " TFLG", TFLG
          IF (.NOT.((ECROSS(N).EQ.0).AND.(TFLG.EQ.1))) GOTO 40
          ECROSS(N)=TEND
C
C    THE NEXT THREE STATEMENTS MAKE SURE EDGES OF THE T DO NOT
C    EXTEND INTO INTO NEXT VCL REGION WHEN END OF CROSS IS NOT
C    FOUND.
          IF ((T+1).LE.MTOTAL) ECROSS(N)=START(T+1)
          SCROSS(N)=TEND-LNGTH
          IF (((TEND-LNGTH).LT.END(T-1)).AND.((T-1).NE.0))
         /  SCROSS(N)=END(T-1)
          N=N+1 ; ***
          GOTO 15
40        CONTINUE
15        CONTINUE
C
          DO 100 J=1,20  ;*****
          DO 110 K=1,2
          TBD(J,K)=0
110       CONTINUE
100       CONTINUE
C
          DO 90 I=1,N-1 ;*****
```

```
      TYPE "#8"," N=",I," ST. OF T", SCROSS(I), " END OF T",
     /  ECROSS(I)
C     THIS CODES HANDLES SPECIAL CASE OF TWO T'S IN A ROW.
      IF (SCROSS(I+1).LE.0) GOTO 85
      IF  (.NOT.(ECROSS(I).GT.SCROSS(I+1)))  GOTO 85  ;  TT
      MDTT=(ECROSS(I)+SCROSS(I+1))/2  ; MARKS MIDDLE OF TT
      ECROSS(I)=MDTT
      SCROSS(I+1)=MDTT
      TYPE "#9", " N=", I," ST.-T", SCROSS(I), " END-T",
     /  ECROSS(I)
85    CONTINUE
      SCOL=SCROSS(I)
      ECOL=ECROSS(I)
      TYPE "#10"," SCOL",SCOL," ECOL",ECOL
      IF ((ECOL.LE.0).OR.(SCOL.LE.0)) GOTO 90
      BGT(SCOL)=10     ; MARKS BEGINNING OF T
      EDT(ECOL)=10     ; MARKS END OF T
      TBD(I,1)=SCOL ;
      TBD(I,2)=ECOL ;
90    CONTINUE

      RETURN
      END
```

```
C*****************************************************************
C  SUBROUTINE NAME: DECISION ALGORITHM (DECALG.FR)
C  WRITTEN BY: 1LT. DAVID V. SOBOTA
C  PURPOSE: THIS ROUTINE LOOKS AT EACH VCL IN A WORD AND ITS
C  NEIGHBORING VCL'S AND USES INFORMATION ON THE HEIGHT OF
C  THE VCL'S AND NEIGHBORING LETTER FEATURES(O,U,E,ARCH,I,T)
C  TO DECIDE ON WHERE TO SEGMENT LETTERS THAT ARE TOUCHING.
C  IT ALSO USES THE PRESENCE OF I AND T TO EVALUATE WHETHER
C  A VCL IS DUE TO TWO LETTERS NEXT TO EACH OTHER. THIS
C  DECISION ALGORITHM IS A REWRITTEN VERSION OF THE ORIGINAL
C  DECISION  ALGORITHM  DEMONSTRATED  IN  THE  THESIS.  THIS
C  VERSION  INCORPORATES THE ENHANCED DECISION  RULES  WHICH
C  RELY  ON  DETECTING VARIOUS TYPES OF O,U,A,8  REGIONS  IN
C  ORDER  TO  GET  A  BETTER IDEA  AS  TO  WHAT  THE  LETTER
C  COMBINATION  IS  SO THAT IT CAN BE  SEGMENTED  WITH  MORE
C  CONFIDENCE.   IT  ATTEMPTS  TO  RECOGNIZE  X,V,W,  LETTER
C  FEATURES SO THAT FALSE SEGMENTATION WILL NOT OCCUR DUE TO
C  LETTER COMBINATIONS THAT HAVE SIMILIAR FEATURES. IT ALSO
C  HAS AN U DETECTOR (LU1,2) WHICH CAN BE USED TO ELIMINATE
C  FALSE SEGMENTATION CLUES DUE TO U-REGIONS IN V AND "TAILS"
C  OF SOME LETTERS. DUE TO TIME CONSTRAINTS, THE SEGMENTATION
C  OF LETTER COMBINATIONS WITH NO VCL AT THE MIDDLE HAVE NOT
C  BEEN INCORPORATED AND FURTHER REFINING NEEDS TO BE DONE.
C*****************************************************************
C
      OVERLAY DA
      SUBROUTINE DECALG(NROWS,NCOLS,IVCL,IAR,IUR,IOR,IER,
     /  XIAR,XIUR,XIOR,XIER,GAP,IBD,TBD,IBUFO)
C
      INTEGER NCOLS, NROWS, IBUFO(300), IARRAY(128)
      INTEGER SEG(128),GAP(128)
      INTEGER P2,N1,V1,V2, MDVC
      LOGICAL AO,OA,EA,AE,OE,EO,OO,AA,EE,UU,OU,AU,EU,
     /  UO,UA,UE
      LOGICAL VIT, SPP2,SPV1,SPV2,SPN1
      LOGICAL P2BL,V1BL,V2BL,N1BL
      LOGICAL P2AB,V1AB,V2AB,N1AB , ABBL
      LOGICAL VOK,AOL1,UOL1,AOL2,UOL2,VAU,U9,UVA
      INTEGER IVCL(20,10),IAR(20,3),IUR(20,3),IOR(20,3),
     /  IER(20,3)
      INTEGER XIAR(20,3),XIUR(20,3),XIOR(20,3),XIER(20,3)
      INTEGER MVCW,VCW,TWD,IWD,SUM,TBD(20,2),IBD(20,2),WDVC
      INTEGER MAX,V2A,V2U,V2E,V2O,V1A,V1U,V1E,V1O
      INTEGER WD,PWD,NWD,RW2A,RW2O,RW1A,RW1O,AU2,UA1
      INTEGER WDO,WDO1,WDO2,WDA,LU1,LU2
C
      DO 5 I=1,NCOLS ; INIT. SEG. ARRAY WHICH INDICATES
      SEG(I)=0       ; WHERE LETTERS EDGES ARE LOCATED.
      IF (GAP(I).GT.5) SEG(I)=10 ; GAPS ARE DISPLAYED ON
5     CONTINUE                   ; SEGMENTATION DISPLAY
C
C  FIND MINIMUM WIDTH OF VCL REGION
C  BEST IF TAKEN FROM LARGE SAMPLE OF WORDS TO INSURE THAT
```

```
C  MVCW IS DUE TO VCL IN ONE LETTER.
      MVCW=NCOLS
      DO 10 N=1,20
            IF (IVCL(N,1).EQ.0) GOTO 20
            VCW=IVCL(N,2)-IVCL(N,1)+1
         IF (VCW.LT.MVCW) MVCW=VCW ; MVCW- MIN. VCL WIDTH
10    CONTINUE
20    CONTINUE
C
C   *****MAIN LOOP FOR INCORPORATING FEATURES IN IVCL*****
C   FIND AVG. WIDTH OF VCL IN I'S AND T'S.
      TWD=NCOLS
      IWD=NCOLS
C   ******** BEGINNING OF MAIN LOOP*********
      DO 30 N=1,20
      IF  (IVCL(N,1).EQ.0) GOTO 55
      K=0
      SUM=0
      DO 35 J=1,20
      IF (TBD(J,1).EQ.0) GOTO 40
      IF (.NOT.((TBD(J,1).LT.IVCL(N,1)).AND.(TBD(J,2).GT.
     /  IVCL(N,2))))  GOTO 35
      SUM=SUM+(IVCL(N,2)-IVCL(N,1)+1)
      K=K+1
      IVCL(N,7)=2 ; 2-INDICATES VCL IS MID. VCL LETTER T.
35    CONTINUE
40    CONTINUE
      IF (SUM.NE.0) TWD=(SUM/K) ; AVG. WIDTH OF VCL'S IN T'S
C
      K=0
      SUM=0
      DO 45 J=1,20
      IF (IBD(J,1).EQ.0) GOTO 50
      IF (.NOT.(IBD(J,1).EQ.IVCL(N,1))) GOTO 45
      SUM=SUM+(IBD(J,2)-IBD(J,1)+1)
      K=K+1
      IVCL(N,7)=1 ; 1-INDICATES VCL IS SAME VCL IN LETTER I.
45    CONTINUE
50    CONTINUE
      IF (SUM.NE.0) IWD=(SUM/K) ; AVG. WIDTH OF VCL'S IN I'S
C
      MAX=0
      DO 75 I=IVCL(N,1)-3,IVCL(N,1)
      IF ( GAP(I).GT.MAX ) MAX =GAP(I) ;STORES MAX GAP NEAR
75    CONTINUE                         ;FIRST EDGE OF VCL
      IVCL(N,5)=MAX  ; STORES GAP IN POSITION 5
      MAX=0
      DO 85 I=IVCL(N,2),IVCL(N,2)+3
      IF (GAP(I).GT.MAX  ) MAX =GAP(I) ;STORES MAX GAP NEAR
85    CONTINUE                         ;2ND EDGE OF VCL
      IVCL(N,6)=MAX
C
C     CHECKS TO SEE IF VCL NEXT TO LETTER T
```

```
C        3-INDICATES BEG. OF VCL NEXT TO T
C        4-INDICATES END OF VCL NEXT TO T
C
         DO 90 I=1,20
         IF (IVCL(N,1).EQ.0) GOTO 100
         IF ((IABS(IVCL(N,1)-TBD(I,2))).LE.3) IVCL(N,7)=3
         IF ((IABS(IVCL(N,2)-TBD(I,1))).LE.3) IVCL(N,7)=4
90       CONTINUE
100      CONTINUE
C
         V2A=0    ; INDICATES 2ND EDGE VCL NEXT TO A-REGION
         V2O=0    ; INDICATES 2ND EDGE VCL NEXT TO O-REGION
         V2U=0    ; INDICATES 2ND EDGE VCL NEXT TO U-REGION
         V2E=0    ; INDICATES 2ND EDGE VCL NEXT TO E-REGION
         V1A=0    ; INDICATES 1ST EDGE VCL NEXT TO A-REGION
         V1O=0    ; INDICATES 1ST EDGE VCL NEXT TO O-REGION
         V1U=0    ; INDICATES 1ST EDGE VCL NEXT TO U-REGION
         V1E=0    ; INDICATES 1ST EDGE VCL NEXT TO E-REGION
         RW2A=0   ; INDICATES ROW IN IAR,IOR, ETC.
         RW2O=0   ;
         RW1A=0   ;
         RW1O=0
C      **** LOOP 70 CHECKS WHAT REGIONS ARE NEXT TO VCL ****
         DO 70 I=1,20
         IF (IVCL(N,1).EQ.0) GOTO 110
C      **** 2ND EDGE OF VCL LOOKED AT ****
         IF (.NOT.((IABS(IVCL(N,2)-IAR(I,1))).LE.3)) GOTO 71
         V2A=IAR(I,3) ; SPECIFIC TYPE OF REGION STORED
         RW2A=I       ; ROW POS. STORED FOR LATER V DETECTION
71       CONTINUE
         IF (.NOT.((IABS(IVCL(N,2)-IOR(I,1))).LE.3)) GOTO 72
         V2O=IOR(I,3)
         RW2O=I       ; ROW STORED FOR LATER X DETECTION
72       CONTINUE
         IF ((IABS(IVCL(N,2)-IUR(I,1))).LE.3) V2U=IUR(I,3)
         IF ((IABS(IVCL(N,2)-IER(I,1))).LE.3) V2E=IER(I,3)
C      **** 1ST EDGE OF VCL LOOKED AT ****
         IF (.NOT.((IABS(IVCL(N,1)-IAR(I,2))).LE.3)) GOTO 73
         V1A=IAR(I,3)
         RW1A=I
73       CONTINUE
         IF (.NOT.((IABS(IVCL(N,1)-IOR(I,2))).LE.3)) GOTO 74
         V1O=IOR(I,3)
         RW1O=I
74       CONTINUE
         IF ((IABS(IVCL(N,1)-IUR(I,2))).LE.3) V1U=IUR(I,3)
         IF ((IABS(IVCL(N,1)-IER(I,2))).LE.3) V1E=IER(I,3)
70       CONTINUE
110      CONTINUE
C
C  16 LOGICAL VARIABLE INDICATE REGIONS NEXT TO VCL
         AO=((V1A.GT.0).AND.(V2O.GT.0))  ; 1
         OA=((V1O.GT.0).AND.(V2A.GT.0))  ; 2
```

```
        EA=((V1E.GT.0).AND.(V2A.GT.0))   ; 3
        AE=((V1A.GT.0).AND.(V2E.GT.0))   ; 4
        OE=((V1O.GT.0).AND.(V2E.GT.0))   ; 5
        EO=((V1E.GT.0).AND.(V2O.GT.0))   ; 6
        OO=((V1O.GT.0).AND.(V2O.GT.0))   ; 7
        AA=((V1A.GT.0).AND.(V2A.GT.0))   ; 8
        EE=((V1E.GT.0).AND.(V2E.GT.0))   ; 9
        UU=((V1U.GT.0).AND.(V2U.GT.0))   ; 10
        OU=((V1O.GT.0).AND.(V2U.GT.0))   ; 11
        AU=((V1A.GT.0).AND.(V2U.GT.0))   ; 12
        EU=((V1E.GT.0).AND.(V2U.GT.0))   ; 13
        UO=((V1U.GT.0).AND.(V2O.GT.0))   ; 14
        UA=((V1U.GT.0).AND.(V2A.GT.0))   ; 15
        UE=((V1U.GT.0).AND.(V2E.GT.0))   ; 16
C
C  CHECKS TO SEE IF FEATURES MEET THE CONDITIONS FOR A VCL
C  IN X.
C  1ST O-REGION IS CHECK TO SEE IT MEETS CORRECT CONDITIONS
      IF (.NOT.(OO.AND.(V2O.EQ.9).AND.(V1O.EQ.9))) GOTO 125
        TYPE N," X O-REGION CONDITION CHECK"
        DO 120 I=IOR(RW1O,1)-3,IOR(RW1O,1)
        IF (GAP(I).GE.5) IVCL(N,7)=6
        DO 130 J=1,20
        IF ((XIAR(J,2).EQ.I).OR.(XIUR(J,2).EQ.I).OR.
     /  (XIER(J,2).EQ.I))  IVCL(N,7)=6
130     CONTINUE
120     CONTINUE
        IF (IVCL(N,7).NE.6) GOTO 119
        TYPE N," MEETS 1ST O-REGION X CONDITION "
119     CONTINUE
C
C       2ND O-REGION IS CHECKED AS ABOVE.
        DO 129 I=IOR(RW1O,2),IOR(RW1O,2)+3
        IF (GAP(I).GE.5) IVCL(N,7)=6
        DO 139 J=1,20
        IF ((IAR(J,2).EQ.I).OR.(IUR(J,2).EQ.I).OR.
     /  (IER(J,2).EQ.I))  IVCL(N,7)=6
139     CONTINUE
129     CONTINUE
        IF (IVCL(N,7).NE.6) GOTO 138
        TYPE N," MEETS 2ND O-REGION X CONDITION "
138     CONTINUE
C
C  CHECK WIDTH OF 2 O-REGIONS TO SEE IF TOO NARROW o,b,c,ETC.
C  ALSO, CHECK WIDTH OF A-REGION BETWEEN O-REGIONS TO SEE
C  IF WIDER THAN O-REGIONS.
        MAX=0
        WDO1=IOR(RW1O,2)-IOR(RW1O,1)+1 ; MAX WIDTH OF O-REGION
        WDO2=IOR(RW2O,2)-IOR(RW2O,1)+1
        DO 140  I=1,20  ; DETERMINE MAX. WIDTH OF O-REGION
        WDO=IOR(I,2)-IOR(I,1)+1
        IF ((WDO.GT.MAX).AND.(IOR(I,3).EQ.9)) MAX=WDO
        IF ((XIAR(I,2).LE.IOR(RW2O,2)).AND.(XIAR(I,1).GE.
```

```
     /   IOR(RW10,1)))
     /   WDA=XIAR(I,2)-XIAR(I,1)+1 ; WIDTH OF A-REGION
140      CONTINUE                  ; BETWEEN O-REGIONS
         IF ((WDO1.LT.MAX).AND.(WDO2.LT.MAX)) IVCL(N,7)=6
         IF ((WDA.GT.WDO1).OR.(WDA.GT.WDO2)) IVCL(N,7)=6
         IF (IVCL(N,7).NE.6) GOTO 123
         TYPE N," MEETS WIDTH O-REGION X CONDITION "
123      CONTINUE
125      CONTINUE
C
C    CHECKS TO SEE IF FEATURES MEET THE CONDITIONS FOR A VCL
C    IN V,W,OR Y.
C    5-INDICATES POSSIBLE VCL IN V,W,OR Y.

         IF (AU.OR.UA) GOTO 111
         VOK=(((IVCL(N,3).EQ.1).AND.(IVCL(N,4).EQ.1)).OR.
     /   ((IVCL(N,3).EQ.3).AND.(IVCL(N,4).EQ.3))); VCL IS OK.
         DO 105 I=1,20
         AOL1=((IAR(I,2).GE.IVCL(N,1)).AND.(IAR(I,1).LT.
     /   IVCL(N,1))); A-VL
         UOL2=((IUR(I,1).LE.IVCL(N,2)).AND.(IUR(I,2).GT.
     /   IVCL(N,2))); VL-U
         UOL1=((IUR(I,2).GE.IVCL(N,1)).AND.(IUR(I,1).LT.
     /   IVCL(N,1))); U-VL
         AOL2=((IAR(I,1).LE.IVCL(N,2)).AND.(IAR(I,2).GT.
     /   IVCL(N,2))); VL-A
         IF (((AOL1.AND.UOL2).OR.(AOL2.AND.UOL1)).AND.VOK)
     /   IVCL(N,7)=5
105      CONTINUE
111      CONTINUE
C
30       CONTINUE ; ***** END OF MAIN LOOP
55       CONTINUE
C
C        ******2ND MAIN LOOP FOR SEGMENTATION OF VCL******
C
         DO 500 N=1,20
         IF (IVCL(N,1).EQ.0) GOTO 550
         IF (IVCL(N,7).GE.5) GOTO 500
         IF (IVCL(N,7).NE.1) GOTO 21
         SEG(IVCL(N,1))=5
         SEG(IVCL(N,2))=5
         TYPE N," VCL MEETS I CONDITIONS"
21       CONTINUE
         IF (IVCL(N,7).NE.2) GOTO 22
         TYPE N," VCL MEETS T CONDITIONS"
         DO 23 J=1,20
         IF (.NOT.((TBD(J,1).LT.IVCL(N,1)).AND.(TBD(J,2).GT.
     /   IVCL(N,2))))  GOTO 24
         SEG(TBD(J,1))=5
         SEG(TBD(J,2))=5
24       CONTINUE
23       CONTINUE
```

```
22      CONTINUE
C
        V2A=0    ; INDICATES 2ND EDGE VCL NEXT TO A-REGION
        V2O=0    ; INDICATES 2ND EDGE VCL NEXT TO O-REGION
        V2U=0    ; INDICATES 2ND EDGE VCL NEXT TO U-REGION
        V2E=0    ; INDICATES 2ND EDGE VCL NEXT TO E-REGION
        V1A=0    ; INDICATES 1ST EDGE VCL NEXT TO A-REGION
        V1O=0    ; INDICATES 1ST EDGE VCL NEXT TO O-REGION
        V1U=0    ; INDICATES 1ST EDGE VCL NEXT TO U-REGION
        V1E=0    ; INDICATES 1ST EDGE VCL NEXT TO E-REGION
        RW2A=0   ; INDICATES ROW IN IAR,IOR, ETC.
        RW2O=0   ;
        RW1A=0   ;
        RW1O=0
        LU1=0    ; INDICATES IF U-REGION IS PART OF LETTER
        LU2=0    ; U OR Y
C
        DO 60 I=1,20
        IF (IVCL(N,1).EQ.0) GOTO 95
        IF (.NOT.((IABS(IVCL(N,2)-IAR(I,1))).LE.3)) GOTO 61
        V2A=IAR(I,3) ; SPECIFIC TYPE OF REGION STORED
        RW2A=I       ; ROW POSITION STORED
61      CONTINUE
        IF (.NOT.((IABS(IVCL(N,2)-IOR(I,1))).LE.3)) GOTO 62
        V2O=IOR(I,3)
        RW2O=I
62      CONTINUE
        IF ((IABS(IVCL(N,2)-IUR(I,1))).LE.3) V2U=IUR(I,3)
        IF ((V2U.EQ.9).AND.((IABS(IVCL(N+1,1)-IUR(I,2))).LE.3)
     /  .AND.(IVCL(N,4).EQ.1).AND.((IVCL(N+1,3).EQ.1).OR.
     /  (IVCL(N+1,3).EQ.3))) LU2=1 ; U-REGION IN LETTER U OR Y
        TYPE N," LU2=",LU2
        IF ((IABS(IVCL(N,2)-IER(I,1))).LE.3) V2E=IER(I,3)
        IF (.NOT.((IABS(IVCL(N,1)-IAR(I,2))).LE.3)) GOTO 63
        V1A=IAR(I,3)
        RW1A=I
63      CONTINUE
        IF (.NOT.((IABS(IVCL(N,1)-IOR(I,2))).LE.3)) GOTO 64
        V1O=IOR(I,3)
        RW1O=I
64      CONTINUE
        IF ((IABS(IVCL(N,1)-IUR(I,2))).LE.3) V1U=IUR(I,3)
        IF ((V1U.EQ.9).AND.((IABS(IVCL(N-1,2)-IUR(I,1))).LE.3)
     /  .AND.(IVCL(N-1,4).EQ.1).AND.((IVCL(N,3).EQ.1).OR.
     /  (IVCL(N,3).EQ.3))) LU1=1 ; U-REGION IN LETTER U OR Y
        TYPE N," LU1=",LU1
        IF ((IABS(IVCL(N,1)-IER(I,2))).LE.3) V1E=IER(I,3)
60      CONTINUE
95      CONTINUE
C
        AO=((V1A.GT.0).AND.(V2O.GT.0))   ; 1
        OA=((V1O.GT.0).AND.(V2A.GT.0))   ; 2
        EA=((V1E.GT.0).AND.(V2A.GT.0))   ; 3
```

```
      AE=((V1A.GT.0).AND.(V2E.GT.0))   ; 4
      OE=((V1O.GT.0).AND.(V2E.GT.0))   ; 5
      EO=((V1E.GT.0).AND.(V2O.GT.0))   ; 6
      OO=((V1O.GT.0).AND.(V2O.GT.0))   ; 7
      AA=((V1A.GT.0).AND.(V2A.GT.0))   ; 8
      EE=((V1E.GT.0).AND.(V2E.GT.0))   ; 9
      UU=((V1U.GT.0).AND.(V2U.GT.0))   ; 10
      OU=((V1O.GT.0).AND.(V2U.GT.0))   ; 11
      AU=((V1A.GT.0).AND.(V2U.GT.0))   ; 12
      EU=((V1E.GT.0).AND.(V2U.GT.0))   ; 13
      UO=((V1U.GT.0).AND.(V2O.GT.0))   ; 14
      UA=((V1U.GT.0).AND.(V2A.GT.0))   ; 15
      UE=((V1U.GT.0).AND.(V2E.GT.0))   ; 16
C
C     CHECK IF VCL TOO WIDE TO BE CAUSED BY ONE LETTER
      WDVC=IVCL(N,2)-IVCL(N,1)+1
      WD=0
      IF ((WDVC.GT.(MVCW*1.5)).OR.(WDVC.GT.(TWD*1.25)).OR.
     /  (WDVC.GT.(IWD*1.25))) WD=1
C   CHECK IF PREVIOUS VCL TOO WIDE TO BE CAUSED BY ONE LETTER
      PWD=-1
      IF ((N-1).LT.1) GOTO 83
      PWDVC=IVCL(N-1,2)-IVCL(N-1,1)+1
      PWD=0
      IF ((PWDVC.GT.(MVCW*1.5)).OR.(PWDVC.GT.(TWD*1.25)).OR.
     /  (PWDVC.GT.(IWD*1.25))) PWD=1 ; PREVIOUS VCL WIDE
83    CONTINUE
C     CHECK IF NEXT VCL TOO WIDE TO BE CAUSED BY ONE LETTER
      NWD=-1
      IF (IVCL((N+1),1).EQ.0) GOTO 84
      NWDVC=IVCL(N+1,2)-IVCL(N+1,1)+1
      NWD=0
      IF ((NWDVC.GT.(MVCW*1.5)).OR.(NWDVC.GT.(TWD*1.25)).OR.
     /  (NWDVC.GT.(IWD*1.25))) NWD=1
84    CONTINUE
C
C  FOLLOWING CODE CHECKS TO SEE IF A-REGION AFTER VCL IS DUE
C  TO V, W, OR Y.
C     AU2=1 ; INDICATES A/U REGION AFTER VCL.
      AU2=0 ; USED TO DISTINGUISH V,W,Y FROM U'S
      IF ((V2A.NE.9)) GOTO 80
      DO 79 J=1,20
      IF (IUR(J,1).EQ.0) GOTO 80
      VAU=((IUR(J,1).LE.IAR(RW2A,2)).AND.(IUR(J,2).GT.
     /  IAR(RW2A,2)))
      U9=(IUR(J,3).EQ.9)
      VOK=((NWD.EQ.0).AND.((IVCL(N+1,3).EQ.1).OR.
     /  (IVCL(N+1,3).EQ.3)))
      IF (U9.AND.(VAU.OR.(VOK.AND.(IVCL(N+1,7).EQ.5)))) AU2=1
79    CONTINUE
80    CONTINUE
C
C   FOLLOWING CODE CHECKS TO SEE IF A-REGION BEFORE VCL IS  DUE
```

```
C    TO V, W, OR Y.
C         UA1=1 INDICATE U/A REGION BEFORE VCL.
          UA1=0
          IF ((V1A.NE.9)) GOTO 109
          DO 99 J=1,20
          IF (IUR(J,1).EQ.0) GOTO 109
          VAU=((IUR(J,2).LT.IAR(RW1A,1)).AND.(IUR(J,1).GT.
     /    IAR(RW1A,1)))
          U9=(IUR(J,3).EQ.9)
          VOK=((PWD.EQ.0).AND.((IVCL(N-1,3).EQ.1).OR.
     /    (IVCL(N-1,3).EQ.3)))
          IF (U9.AND.(VAU.OR.(VOK.AND.(IVCL(N-1,7).EQ.5)))) UA1=1
99        CONTINUE
109       CONTINUE
C
C         SP-- INDICATE IF SPACE IS NEAR VARIOUS EDGES OF VCL'S
          SPP2=((IVCL(N-1,6).GT.5).OR.(IVCL(N-1,7).EQ.1))
          SPV1=((IVCL(N,7).EQ.3).OR.(IVCL(N,5).GT.5))
          SPV2=((IVCL(N,7).EQ.4).OR.(IVCL(N,6).GT.5))
          SPN1=((IVCL(N+1,5).GT.5).OR.(IVCL(N+1,7).EQ.1))
C
          VIT=((IVCL(N,7).EQ.2).OR.(IVCL(N,7).EQ.1)); IS I OR T
          P2BL=(IVCL(N-1,4).EQ.3)
          P2AB=(IVCL(N-1,4).EQ.2)
          V1BL=(IVCL(N,3).EQ.3)
          V1AB=(IVCL(N,3).EQ.2)
          V2BL=(IVCL(N,4).EQ.3)
          V2AB=(IVCL(N,4).EQ.2)
          N1BL=(IVCL(N+1,3).EQ.3)
          N1AB=(IVCL(N+1,3).EQ.2)
          ABBL=((V1AB.AND.V2BL).OR.(V1BL.AND.V2AB).OR.
     /    (IVCL(N,3).EQ.4).OR.(IVCL(N,4).EQ.4))
C
          MDVC=(IVCL(N,1)+IVCL(N,2))/2
          V1=IVCL(N,1)     ; POSITION OF 1ST EDGE VCL
          V2=IVCL(N,2)     ; POSITION OF 2ND EDGE VCL
          P2=IVCL(N-1,2)   ; POSITION OF 2ND EDGE PREVIOUS VCL
          N1=IVCL(N+1,1)   ; POSITION OF 1ST EDGE NEXT VCL
          IF (.NOT.(ABBL)) GOTO 26 ; SEG. ON IMPOSSIBLE FEATURE
          SEG(MDVC)=10
          TYPE N," ABBL CONDITION"
26        CONTINUE
C
          IF (WD.EQ.0) GOTO  112
          SEG(MDVC)=2
          GOTO 460
112       CONTINUE
C
C   IF 2 VCL TOO CLOSE TO BE CAUSED BY ONE LETTER SEG. MIDDLE
          IF (N1.EQ.0) GOTO 150
          IF ((N1-V2).LE.MVCW) SEG(((V2+N1)/2))=3
150       CONTINUE
          IF (P2.EQ.0) GOTO 155
```

```
          IF ((V1-P2).LE.MVCW) SEG(((V1+P2)/2))=3
155       CONTINUE

          IF ((VIT).OR.(IVCL(N,7).GE.5)) GOTO 500
C   THE FOLLOWING CASES NEED NO ADDITIONAL SEGMENTATION
C   FEATURES OTHER THAN A,O,U, AND 8 REGIONS AND WIDTH OF VCL:

C         EO CASE
          IF (.NOT.((EO).AND.(V1E.EQ.9).AND.(WD.NE.1))) GOTO 300
          SEG(V1)=5
          TYPE N,"EO"
300       CONTINUE

C         OO CASE
          IF (.NOT.((@O).AND.(V2O.EQ.9).AND.(V1O.EQ.9).AND.
     /    (WD.NE.1)))  GOTO 305
          SEG(V1)=5
          TYPE N,"OO"
305       CONTINUE
C
C         UO CASE AND AO CASE ( JUST UO CASE SEGMENTED AT MDVC )
          IF (.NOT.((UO).AND.(AO).AND.(V2O.EQ.9).AND.(WD.NE.1)))
     /    GOTO 310
          SEG(V1)=5
          TYPE N,"U/AO"
310       CONTINUE

C         AO CASE AND UA1=1
          IF (.NOT.((AO).AND.(.NOT.(UO)).AND.(V2O.EQ.9).AND.
     /    (UA1.EQ.1).AND.(WD.NE.1))) GOTO 315
          SEG(V1)=5
          TYPE N,"VO"
315       CONTINUE

C         AO CASE
          IF (.NOT.((AO).AND.(.NOT.(UO)).AND.(V2O.EQ.9).AND.
     /    (WD.NE.1))) GOTO 320
          SEG(V1)=5
          TYPE N,"AO"
          IF (.NOT.((V1AB.OR.V2AB).AND.(NWD.EQ.0))) GOTO 318
          TYPE "MISTAKE COULD TO DUE TO SOME T'S"
318       CONTINUE
320       CONTINUE

C         OA CASE
          IF (.NOT.((OA).AND.(.NOT.(OU)).AND.(AU2.EQ.0).AND.
     /    (V1O.EQ.9).AND.(WD.EQ.0))) GOTO 325
          SEG(V1)=5
          TYPE N,"OA"
325       CONTINUE

C         OA CASE AND OU CASE
          IF (.NOT.((OA).AND.(OU).AND.(V1O.EQ.9).AND.(WD.NE.1)))
```

```
     /   GOTO 330
         SEG(V2)=5
         TYPE N,"OA/U"
330      CONTINUE
C
         OA CASE AND AU2=1
         IF (.NOT.((OA).AND.(.NOT.(OU)).AND.(AU2.EQ.1).AND.
     /   (WD.NE.1))) GOTO 335
         SEG(V2)=5
         TYPE N,"OV"
335      CONTINUE
C
C        EA CASE
         IF (.NOT.((EA).AND.(.NOT.(EU)).AND.(V1E.EQ.9).AND.
     /   (WD.NE.1))) GOTO 340
         SEG(V1)=5
         TYPE N,"EA"
340      CONTINUE
C
C        AA CASE
C        IF ((AA).AND.(NIT).AND.(WD.EQ.1)) SEG(MDVC)=5
         IF (.NOT.((AA).AND.(UA1.NE.1).AND.(.NOT.(UA.OR.AU)).AND.
     /   (WD.NE.1))) GOTO 345
         SEG(V1)=5
         TYPE N,"AA"
345      CONTINUE
C
C        AA CASE AND UA CASE
         IF (.NOT.((AA).AND.(UA).AND.(UA1.NE.1).AND.
     /   (WD.NE.1))) GOTO 350
         SEG(V1)=5
         TYPE N,"U/AA"
350      CONTINUE
C
C        AA CASE AND UA1=1
         IF (.NOT.((AA).AND.(UA1.EQ.1).AND.(WD.NE.1))) GOTO 355
         SEG(V1)=5
         TYPE N,"VA"
355      CONTINUE
C
C        UA CASE
         IF (.NOT.((UA).AND.(.NOT.(AA.OR.UU)).AND.(AU2.NE.1).AND.
     /   (WD.NE.1))) GOTO 360
         SEG(V1)=5
         TYPE N,"UA"
360      CONTINUE
C
C        AG CASE
         IF (.NOT.((AE).AND.(.NOT.(UE.OR.(UA1.EQ.1))).AND.
     /   (V2E.EQ.15).AND.(WD.NE.1))) GOTO 365
         SEG(V1)=5
         TYPE N,"AG FEATURE"
365      CONTINUE
```

```
C
C       OG CASE
        IF (.NOT.((OE).AND.(V2E.EQ.15).AND.(WD.NE.1)))
     /  GOTO 370
        SEG(V1)=5
        TYPE N,"OG FEATURE"
370     CONTINUE
C
C       EG CASE
        IF (.NOT.((EE).AND.(V1E.EQ.12).AND.(V2E.EQ.15).AND.
     /  (WD.NE.1)))  GOTO 375
        SEG(V1)=5
        TYPE N,"EG"
375     CONTINUE
C
C       GE CASE
        IF (.NOT.((EE).AND.(V1E.EQ.15).AND.(V2E.EQ.12).AND.
     /  (WD.NE.1)))  GOTO 380
        SEG(V2)=5
        TYPE N,"GE"
380     CONTINUE
C
C       GG CASE
        IF (.NOT.((EE).AND.(V1E.EQ.15).AND.(V2E.EQ.15).AND.
     /  (WD.NE.1)))  GOTO 385
        SEG(MDVC)=5
        TYPE N,"GG"
385     CONTINUE
C
C       GU CASE
        IF (.NOT.((EU).AND.(.NOT.(EA.OR.(AU2.EQ.1))).AND.
     /  (V1E.EQ.15).AND.(WD.NE.1))) GOTO 390
        SEG(MDVC)=5
        TYPE N,"GU"
390     CONTINUE
C
C       GA CASE AND GU CASE OR AU2=1
        IF (.NOT.((EA).AND.((AU2.EQ.1).OR.(EU)).AND.(V1E.EQ.15)
     /  .AND.(WD.NE.1))) GOTO 395
        SEG(V2)=5
        TYPE N,"GV"
395     CONTINUE
C
C       AE CASE *********
        IF (.NOT.((EA).AND.(.NOT.(UE.OR.(UA1.EQ.1))).AND.
     /  (V2E.EQ.12).AND.(WD.NE.1))) GOTO 400
        SEG(V2)=5
        TYPE N,"AE"
        IF (P2AB.OR.P2BL) GOTO 397
        TYPE "POSSIBLE MISTAKE IF re"
397     CONTINUE
400     CONTINUE
C
```

```
C         OE CASE
          IF (.NOT.((OE).AND.(V1O.EQ.9).AND.(V2E.EQ.12)
     /    .AND.(WD.NE.1))) GOTO 405
          SEG(V2)=5
          TYPE N,"OE"
405       CONTINUE
C
C         EE CASE  ********
          IF (.NOT.((EE).AND.(V1E.EQ.12).AND.(V2E.EQ.12)
     /    .AND.(WD.NE.1))) GOTO 410
          SEG(V1)=5
          TYPE N,"EE- POSSIBLE MISTAKE DUE TO aa or as"
410       CONTINUE
C
C         UE CASE
          IF (.NOT.((UE).AND.(V1E.EQ.12).AND.(.NOT.(AE.OR.
     /    (UA1.EQ.1))).AND.(WD.NE.1))) GOTO 415
          SEG(V2)=5
          TYPE N,"UE"
415       CONTINUE
C
C         VE CASE
          IF (.NOT.((AE).AND.((UE).OR.(UA1.EQ.1)).AND.(V2E.EQ.12)
     /    .AND.(WD.NE.1))) GOTO 420
          SEG(V1)=5
          TYPE N,"VE"
420       CONTINUE
C
C         UV CASE
          IF (.NOT.((UA).AND.((UU).OR.(AU2.EQ.1))
     /    .AND.(WD.NE.1))) GOTO 425
          SEG(V2)=5
          TYPE N,"UV"
425       CONTINUE
C
C         VU CASE
          IF (.NOT.((AU).AND.((UU).OR.(UA1.EQ.1))
     /    .AND.(WD.NE.1))) GOTO 430
          SEG(V1)=5
          TYPE N,"VU"
430       CONTINUE
C
C         OV CASE
          IF (.NOT.((OA).AND.((OU).OR.(AU2.EQ.1)).AND.(V1O.EQ.9)
     /    .AND.(WD.NE.1))) GOTO 435
          SEG(V2)=5
          TYPE N,"OV"
435       CONTINUE
C
C         OU CASE
          IF (.NOT.((OU).AND.(.NOT.(OA.OR.(AU2.EQ.1))).AND.
     /    (V1O.EQ.9 ).AND.(WD.NE.1))) GOTO 440
          SEG(V1)=5
```

```
          TYPE N,"VE"
440       CONTINUE
C
C         AU CASE
          IF (.NOT.((AU).AND.(.NOT.((UU).OR.(UA))).AND.(V1A.EQ.9 )
     /    .AND.(WD.NE.1))) GOTO 445
          SEG(V1)=5
          TYPE N,"AU"
445       CONTINUE
C
C         AV CASE
          IF (.NOT.((AA).AND.((AU).OR.(AU2.EQ.1)).AND.(V1A.EQ.9 )
     /    .AND.(WD.NE.1))) GOTO 450
          SEG(V1)=5
          TYPE N,"AV"
450       CONTINUE
C
C         EU CASE
          IF (.NOT.((EU).AND.(.NOT.(EA.OR.(AU2.EQ.1))).AND.
     /    (V1E.EQ.12).AND.(WD.NE.1))) GOTO 455
          SEG(V1)=5
          TYPE N,"EU"
455       CONTINUE
C
C         EV CASE
          IF (.NOT.((EA).AND.((EU).OR.(AU2.EQ.1)).AND.(V1E.EQ.12)
     /    .AND.(WD.NE.1))) GOTO 460
          SEG(V2)=5
          TYPE N,"EV"
460       CONTINUE
C
C   THE FOLLOWING RULES DETERMINE SEGMENTATION LOCATION BASED
C   ON THE NONEXISTENCE OF FEATURES IN LETTERS IN THE ENGLISH
C   ALPHABET:

C                              A-REGION FOLLOWED BY LONG VCL
          IF ((V1A.EQ.9).AND.(WD.EQ.1).AND.((V2BL.OR.V2AB)))
     /    SEG(MDVC)=SEG(MDVC)+5
          IF ((V1A.EQ.9).AND.(WD.EQ.0).AND.((V1BL.OR.V1AB)))
     /    SEG(V1)=SEG(V1)+5
C                              LONG VCL FOLLOWED BY A-REGION
          IF ((V2A.EQ.9).AND.(WD.EQ.1).AND.((V1BL.OR.V1AB)))
     /    SEG(MDVC)=SEG(MDVC)+5
          IF ((V2A.EQ.9).AND.(WD.EQ.0).AND.(AU2.EQ.0).AND.
C      /    ((V1AB.OR.V2AB)))
C      /    SEG(IAR(RW2A,2)+3)=5 ; **** POSSIBLE k OR h
C                              LONG VCL FOLLOWED BY O-REGION
          IF ((V2O.EQ.9).AND.(WD.EQ.1).AND.((V1BL.OR.V1AB))
     /    .AND.(SPN1)) SEG(MDVC)=SEG(MDVC)+5
C                              E-REGION FOLLOWED BY LONG VCL
          IF ((V1E.GE.12).AND.(WD.EQ.1).AND.((V2BL.OR.V2AB)))
     /    SEG(MDVC)=SEG(MDVC)+5
          IF ((V1E.GE.12).AND.(WD.EQ.0).AND.((V1BL.OR.V1AB)))
```

```
     /   SEG(V1)=SEG(V1)+5
C                              LONG VCL FOLLOWED BY E-REGION
         IF ((V2E.GE.12).AND.(WD.EQ.1).AND.((V1BL.OR.V1AB)))
     /   SEG(MDVC)=SEG(MDVC)+5
         IF ((V2E.GE.12).AND.(WD.EQ.0).AND.((V2BL.OR.V2AB)))
     /   SEG(V2)=SEG(V2)+5
C                              U-REGION FOLLOWED BY LONG VCL
         IF ((V1U.EQ.9).AND.(WD.EQ.1).AND.((V1BL.OR.V1AB)))
     /   SEG(MDVC)=SEG(MDVC)+5
         IF (((V1A.EQ.9).OR.(V1U.EQ.9)).AND.(WD.EQ.0).AND.
     /   (UA1.EQ.1).AND.((V2BL).OR.(V2AB))) SEG(V1)=SEG(V1)+5
C                              LONG VCL FOLLOWED BY U-REGION
         IF ((V2U.EQ.9).AND.(WD.EQ.1).AND.((V1BL.OR.V1AB)))
     /   SEG(MDVC)=SEG(MDVC)+5
         IF (((V2A.EQ.9).OR.(V2U.EQ.9)).AND.(WD.EQ.0).AND.
     /   (AU2.EQ.1).AND.((V2BL).OR.(V2AB))) SEG(V2)=SEG(V2)+5

500      CONTINUE
550      CONTINUE
C
C        OUTPUT RESULTS

         DO 200  I=1,NROWS
         DO 210  J=1,NCOLS
                 IARRAY(J)=15
                 IF (SEG(J).GE.I) IARRAY(J)=0
210      CONTINUE
         CALL PROW(IBUFO,NROWS-I+1,1,NCOLS,IARRAY)
200      CONTINUE

         RETURN
         END
```

VITA

Lt. David V. Sobota was born on 15 January 1960 in Lincoln, Nebraska but lived most of his life in Hinton In The Hedges, England. He is a first generation American; son of a Polish father and a British mother. He graduated from Upper Heyford H.S. , England in 1978 and attended the United States Air Force Academy where he received a Bachelor of Science degree in Electrical Engineering in June 1982. Upon graduation, he was commissioned in the Regular Air Force and assigned to Systems Engineering Facility, Wright Patterson AFB. He later worked in the F-16 System Program Office before entering the School of Engineering, Air Force Institute of Technology in May 1984.

Permanent Address: Carisbrooke, Hinton In
The Hedges,Nr.Brackley,
Northamptonshire,
England

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE

AD-A163938

REPORT DOCUMENTATION PAGE

1a. REPORT SECURITY CLASSIFICATION: UNCLASSIFIED	1b. RESTRICTIVE MARKINGS
2a. SECURITY CLASSIFICATION AUTHORITY	3. DISTRIBUTION/AVAILABILITY OF REPORT: Approved for public release; distribution unlimited.
2b. DECLASSIFICATION/DOWNGRADING SCHEDULE	
4. PERFORMING ORGANIZATION REPORT NUMBER(S): AFIT/GE/ENG/85D-43	5. MONITORING ORGANIZATION REPORT NUMBER(S)
6a. NAME OF PERFORMING ORGANIZATION: School of Engineering; 6b. OFFICE SYMBOL (If applicable): AFIT/ENG	7a. NAME OF MONITORING ORGANIZATION
6c. ADDRESS (City, State and ZIP Code): Air Force Institute of Technology, Wright-Patterson AFB, Ohio 45433	7b. ADDRESS (City, State and ZIP Code)
8a. NAME OF FUNDING/SPONSORING ORGANIZATION; 8b. OFFICE SYMBOL (If applicable)	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER
8c. ADDRESS (City, State and ZIP Code)	10. SOURCE OF FUNDING NOS. — PROGRAM ELEMENT NO. / PROJECT NO. / TASK NO. / WORK UNIT NO.
11. TITLE (Include Security Classification): See Box 19	
12. PERSONAL AUTHOR(S): David Sobota, B.E., 1st Lt. USAF	
13a. TYPE OF REPORT; 13b. TIME COVERED FROM _____ TO _____	14. DATE OF REPORT (Yr., Mo., Day): 1985 December; 15. PAGE COUNT: 120
16. SUPPLEMENTARY NOTATION	

17. COSATI CODES			18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number)
FIELD	GROUP	SUB. GR.	Reading Machine, Character Recognition, Letter Segmentation
06	04		

19. ABSTRACT (Continue on reverse if necessary and identify by block number)

Title: Word Segmentation Algorithm Based on Recognition of Letter Features

Thesis Advisor: Matthew Kabrisky, Ph.D.
Professor of Electrical Engineering

Approved for public release IAW AFR 190-1.
LYNN E. WOLAVER 16 JAN 86
Dean for Research and Professional Development
Air Force Institute of Technology (AFIT)
Wright-Patterson AFB OH 45433

20. DISTRIBUTION/AVAILABILITY OF ABSTRACT	21. ABSTRACT SECURITY CLASSIFICATION
UNCLASSIFIED/UNLIMITED ☒ SAME AS RPT. ☐ OTIC USERS ☐	UNCLASSIFIED

22a. NAME OF RESPONSIBLE INDIVIDUAL	22b. TELEPHONE NUMBER (Include Area Code)	22c. OFFICE SYMBOL
Matthew Kabrisky, Ph.D.	513-255-5276	AFIT/ENG

DD FORM 1473, 83 APR EDITION OF 1 JAN 73 IS OBSOLETE.

SECURITY CLASSIFICATION OF THIS PAGE

This thesis presents an algorithm for segmentation of concatenated English letters in a word. The algorithm consist of recognizing vertical lines, and letter features to decide on the most likely segmentation location for each letter in a word.

END

FILMED

3-86

DTIC