Proceedings of the NACCQ 2000
Wellington NZ www.naccq.ac.nz

A Language Model Based Optical Character Recogniser (OCR) for Reading Incidental Text

Dr Malcolm McQueen
Dr Samuel Mann

Faculty of Art and Technology
Otago Polytechnic
Dunedin, New Zealand
malcolm@bit.tekotago.ac.nz

As part of a project to develop an environmental text reader for the blind, work is being done on the development of a suitable optical character reader. Currently used optical character readers only work well on well-defined text in a known font on a clear background. These conditions are not usually present in the environment and the performance of these optical character readers falls off very quickly with degradation in the conditions. This makes them unsuitable for our needs. A new and more robust type of template for representing letters has been developed that is less sensitive to variations in size, shape, and background of the characters. This paper describes this representation and testing on character forms typical of those found in the environment.

INTRODUCTION

This paper describes research towards the development of a robust Optical character recognition (OCR) system, suitable for use in a variety of applications but particularly the Simon-Sees devices for the visually impaired.

1. ENVIRONMENTAL TEXT

Mann *et al.* 1999 developed a specification for a tool that could be used by people with severe visual impairments in the reading of incidental or 'environmental text'. They found that there is a bewildering amount of text in the environment. It is only when one tries to look for it that it becomes apparent how much there is and how varied it is (Kaku 1997 called writing and words "invisible"). Even restricting the consideration to a single language, there are numerous font, sizes and styles. We add backgrounds, both intentionally and unintentionally. Using a number of techniques (including hand-writing) we write on different textures, shapes and orientation. Some important text is skewed and in perspective. Sometimes the text is only useful if we can place it in context, in association with other text or symbols. Often text is too far away but is still recognisably text, even when it is only single words. Text must be read with varying degrees of accuracy and within varying time limits.

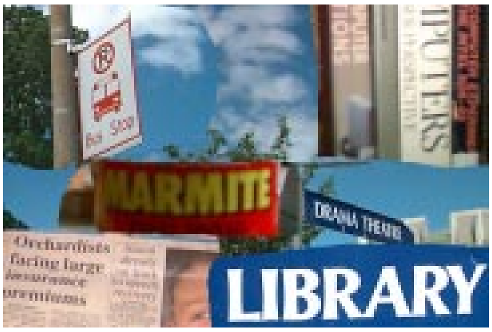

Figure 1: Sample environmental text (and non-text) From Mann (1999).

2. CURRENT OCR

Mann *et al.* (1999) tested commercial optical character recognition systems and found that none of the systems recognised any of the text at all.

A number of criteria must be adhered to in order for OCR to operate. These include:

- that you will have typed text documents placed neatly in a flat-bed scanner (or digitally generated)
- that there are predefined blocks of text
- that text has the same degree of skew and slant
- that there is black text on a white background
- and that there is text there in the first place.

These criteria are not met by the images of environmental text. Current OCR systems are not intended for the complex images involved in this project. If image falls outside the strict criteria then analysis fails.

Further there is a clearly definable series of events that the device would step through to process an image. This process can be divided up into four main steps.

1. Acquire image
2. Manipulation of captured image. This stage will take the image, find any text and filter out any extraneous interference.
3. Optical Character Recognition. The Optical Character Recognition software processes the ‘cleaned’ text.
4. Speech synthesis. The ASCII text is read by the speech synthesiser enabling the user to hear what they have ‘seen’.

The steps are not linear and there may be considerable interaction (e.g. instructions to move the camera 15 degrees to the right or 20% closer). Also the spoken words are to have emphasis according to accuracy etc.

Mann *et al.* concluded with the daunting conclusion that it was necessary to create a purpose built OCR system. This paper reports progress on that task.

2.1 Text Finding and Character Segmentation

Stage two in the process is finding the text. Wu *et al*. (1999) report a system, ‘Textfinder’ that uses multiscale texture segmentation and spatial cohesion constraints to identify areas of text. This system has achieved 95% of detection rate for characters and 93% for words over a large set of test images. While more complex that those normally read by OCR Wu’s images are of a higher quality, in terms of text readability than those described by Mann (1999). Nevertheless, this approach is worthwhile and significant progress in this direction has been made in partnership with Dr Chris Hendry of Otago University’s graphics lab. Progress on this stage will be reported elsewhere, we are confident that this process, using a derivative of the Hough transform will be successful.

Given an area believed to contain text, the next stage is to break the text into characters. While this is relatively simple for straight text, a simple segmentation will suffice, it is more complicated for angled or perspective text. The process uses an output of the transforms used in identifying the areas of text, the orientation angles. From this a bounding ‘rectangle’ is created although this is not necessarily square. The algorithm to create the polygon (or rather its reverse) is used in all subsequent processing. In this way, the skewed letter shapes are effectively returned to straight.

At present the segmentation is simple (Figure 2). We plan to convert this to a weight based system to deal with overlapping letters (e.g. ‘ry’) and touching letters (e.g. ‘mr’). The relative character size is determined and used to determine word spaces.

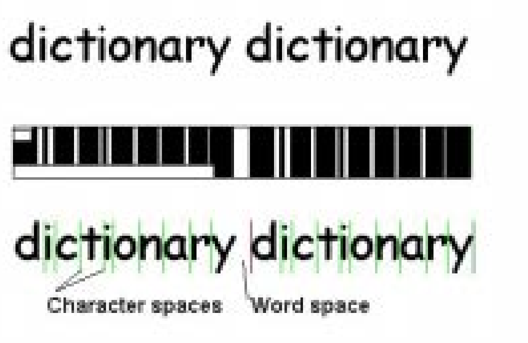

Figure 2: Character segmentation

2.2 Character Recognition

There are many potentially successful methods of identifying letters. For this project it was important that the method be robust and able to report accuracy. In order to maximise robustness, a line count method was chosen. This, it was felt would be less vulnerable to image condition, background and especially skew than other potential methods such as pixel based prototype or template methods (e.g. Xu and Nagy 1999).

Figure 3 shows the word 'dictionary' overlain with a 5 by 3 grid, localised for each letter. The system counts the number lines for each grid line. The system can be set for different thresholds to cope with fuzzy edges or interference. Unfortunately different fonts have different shaped letters (Figure 4). Serifs, emphasis and weightings cause quite different counts of lines. This however, works to our advantage. As we are aiming to read a wide range of fonts and text in varying condition, we can use this variability to our advantage. By identifying those attributes that are common across fonts, we can produce a scoring system based on the essence of each letter.

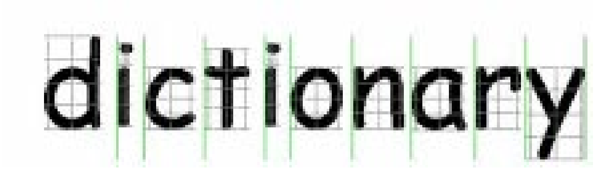

Figure 3: Line cross grids

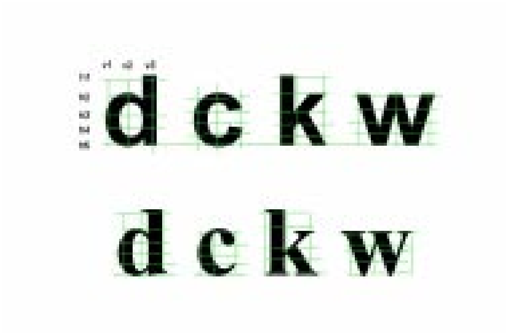

Figure 4. Different fonts have different linecounts. Arial and Times

Table 1 shows the combined line counts for nine different fonts. Those grid lines that score nine for a particular number of lines for letter are considered to be reliable indicators of that letter. For example, 'd' can be identified by having one line at h1, two at h3, two at h4, one at v1, two at v2 and one at v3. The number of lines at

Table 1: Combined line count scores for nine different fonts.

character	linecount	h1	h2	h3	h4	h5	v1	v2	v3
d	0	0	0	0	0	0	0	0	0
d	1	9	8	0	0	5	9	0	9
d	2	0	1	9	9	4	0	9	0
d	3	0	0	0	0	0	0	0	0
d	4	0	0	0	0	0	0	0	0
c	0	0	0	0	0	0	0	0	0
c	1	9	0	9	0	9	8	0	1
c	2	0	9	0	9	0	0	9	8
c	3	0	0	0	0	0	1	0	0
c	4	0	0	0	0	0	0	0	0
K	0	0	0	0	0	0	0	0	0
K	1	1	0	5	0	1	7	1	4
K	2	8	9	3	9	8	2	6	5
K	3	0	0	1	0	0	0	0	0
K	4	0	0	0	0	0	0	2	0
W	0	0	0	0	0	0	0	0	0
W	1	0	0	0	0	1	7	7	9
W	2	3	0	0	4	8	2	2	0
W	3	6	7	0	2	0	0	0	0
W	4	0	2	9	3	0	0	0	0

h5 is not useful in the identification of a ‘d’ as it may be either one or two.

When an unknown letter from an unknown font is presented to the system, the line counts are compared to the table (of which table 1 is a sample) and an overall score calculated. This method successfully identifies letters from the training fonts and many others but we felt it could be better.

A problem with the lines shown in Figure 3 is that they were placed in an arbitrary manner. The outside lines are placed by being ‘in a few pixels from the edge’. How many is a few pixels becomes crucial where serifs are concerned. For example the right hand vertical in the ‘c’ (see Figure 4) just catches the upper arm in the Arial but misses it in the Times. The placement of h2 and h3 are also sensitive, it is easy to just miss the curve of a letter. It was felt that these placement issues were affecting the reliability of letter identifications.

In order to rectify these issues a further table was generated, again based on nine training fonts. This time fewer lines were used but the locations of these was calculated by placing very large number of grid lines over the letters at a variety of angles. The optimum placement of the lines was considered to maximally characterise the letter. This suggested the use of 12 lines (three each of horizontal, vertical and left and right diagonal). Each line was broken into three segments, giving an array of 36 values for each letter. A template was created for each letter based on a logical addition with majority weighting. Again, all letters making up the training set were uniquely matched, but more importantly, the system also recognised a variety of text styles and fonts, including hand printing (Figure 5). The system did not have unique success with all fonts or the hand printing but, as Table 2 shows, the correct letters are almost always in the top five potential letters.

Table 2: Correct letter usually in top five (from characters in Figure 5)

Select	Result
A	G H A q P
E	E F T B f
K	R X K S g
O	O o 0 c C
S	s S E J G
a	8 s a 3 R
g	g 6 q y E

3. LETTER FREQUENCY MODEL

We believe that perfect recognition is unlikely. It would be foolhardy to rely on character identification alone. The text of interest is rarely of sufficient quality to generate correct word on the first try. We have found, however, that it is rare that for the correct letters not to be found in the top five possibilities. Table 3 shows that for dictionary and education, all correct letters are in there somewhere. The next task is to identify them quickly and reliably.

Table 3: Miss identified but correct letters as potentials

Dictionary identified as dic1iouayy
d (bahkg) i (jtfcf) c (crgka) l (tilfj) i (jtfcf) o (oddpq) u (hnkuv) a (rdnga) y (yrvvh) y (yrvvh)

Education identified as ekucaliou
e (bssez) k (gardb) u (hnkuv c (crgka) a (rdnga) l (tilfj) i (jtfcf) o (oddpq) u (hnkuv)

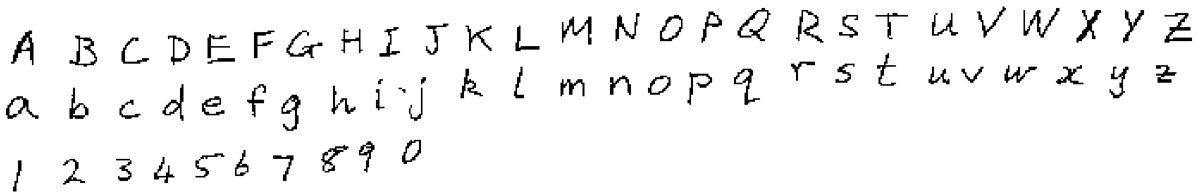
A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
a b c d e f g h i j k l m n o p q r s t u v w x y z
1 2 3 4 5 6 7 8 9 0

Figure 5: Hand printing recognised

One approach would be to use a look up with a dictionary. This proved prohibitively slow. If we use the combinations for the top five identified characters for each letter in the word ‘dictionary’, there are 9,765,625 possible combinations. With word checks requiring a database look up, this was not a sensible approach.

An alternative approach was to use letter frequencies. A very large text file was created and all the frequency of all three-letter combinations were calculated. Table 4 shows some of these results. Of the 9915 combinations found, ‘the’ is most common, making up 1.4% of the letter threesomes whereas ‘blz’ was very unlikely (though strangely not impossible).

Table 4

3-letter combination	Frequency percent
the	1.362879734
and	0.550760469
her	0.266397112
hat	0.191445778
pho	0.019785353
eld	0.017557457
nny	0.003464481
pud	0.000797137
dic	0.018753163
dih	0.000163515
dig	0.008124667
dif	0.025968276
die	0.024660154
did	0.017485919
blz	4.08788E-05

Using letter frequencies it is possible to allocate letters among the top three by picking the highest scoring combination of letters. Thus the incorrectly identified ‘uamed’ in Table 5 could be returned to ‘named’ according to the letter frequencies.

Unfortunately it does not always work. For example in ‘office’ the ‘off’ scores 0.00027 and loses out to ‘ott’ with 0.00017. It is tempting to say that we obviously wouldn’t change it if it was right already, but unfortunately the computer doesn’t know that. Combining the letter frequencies with the identification scores (e.g. those in Table 5) does not help.

So, the letter frequency works most of the time but becomes very complex when words get longer. It is clearly not as simple as picking highest value. Take ‘dictionary’ for example. The ‘ion’ is always picked, but ‘ona’ is less frequent than other combinations. It is possible to keep a running score and aim to maximise, but the multiple constraints get very complex.

Say we do not know if the seventh letter in dictionary is a ‘u’ or an ‘n’. We can look at the surrounding letters and score them (Table 6). But this still leaves a problem if the ‘n’ prevents the next letter from being assigned. We then would have to back a group and say, ‘no, this is not an n’. We attempted to develop code for this but the iteration and function calling became too complex.

Table 6. Maximise on surrounding letters

n	u
ion = 0.004	iou = 0.0002
ona = 0.0003	oua = 0.000002
nar = 0.0001	uar = 0.00001
0.0044	0.000212

4. KICLIOUAYY MODEL

The dictionary is too slow, the letter frequency unreliable. It is not, however, necessary to look up every combination, some can be discarded after a letter frequency check and others because there are no words with that character in that position.

Table 5: Identification Scores and Letter Combination Frequencies

Correct	Identified	ID Score	ID 2	ID2 Score	ID3	ID 3 Score	ID 4	ID4 Score
N	U	85	H	59	N	58	K	41
A	A	71	R	60	D	59	N	49
M	M	90	N	32	W	23	W	15
E	E	69	B	63	S	57	S	49

Letter frequencies
Uam = 0.0060
Ham = 0.0964
Nam = 0.2026
Kam = 0.0120

By combining the results with the original letter identification score, it is possible to not only quickly identify words matching the letters, but give them a likelihood/accuracy score as well.

By building words from certainties, letter frequencies and dictionary lookups, the system can now take letter combinations from the character identification and produce coherent words. Table 7 shows that 'kicliouayy' becomes 'dictionary', 'educaliou' becomes 'education' and 'couteyeuce' becomes 'conference'. These words can now be spoken via speech synthesis.

Table 7: Combined Process Generates Words From Identified Letters

dictionary
kicliouayy k (8- gardb-) i (14- jtfcf-) c (7- crgka-) l
(1- tilfj-) i (14- jtfcf-) o (2- oddpq-) u (2- hnkuv-) a
(6- rdnga-) y (7- yrvvh-) y (7- yrvvh-) dictionary
education educaliou education
conference
couteyeuce c (7- crgka-) o (2- oddpq-) u (2- hnkuv-) t
(3- ijfil-) e (3- bssez-) y (7- yrvvh-) e (3- bssez-) u
(2- hnkuv-) c (7- crgka-) e (3- bssez-)

The whole process, from an original image takes around 5 seconds on a 166 Pentium, most of the time is in finding the characters in the first place. It may be possible to speed the process by not examining characters where a word is already confirmed. An iterative process may also help in clarifying words that originally could not be confirmed. For example 'blind' originally identifies as 'dliukd' and resolves to 'blink', scoring 367 and 'blind' scoring 355, too close to call, but it may be worth directing the system to have another look at the dubious character in fifth position. We are trying to avoid a sentence context process to decide on such words but this may be unavoidable.

5. CONCLUSION

This paper has presented progress to date on generating a robust OCR system for use in reading incidental or environmental text. The line count method of identifying characters has proved successful, as has a combination letter frequency/dictionary method of generating words. Current work focuses on improving the algorithms, more rigorous testing and integrating components of the process. When the text finding component is integrated, we should be able to read environmental text in a manner suitable for supporting the needs of the visually impaired.

6. REFERENCES

Kaku, M. (1997) Visions: How Science Will Revolutionize the 21st Century. New York, Anchor Books (Doubleday), 403pp.

Mann, S. (1999). Databank of images for Simon Sees, Otago Polytechnic.http://bit.tekotago.ac.nz/~sam/research/webpage/is/blind_ocr.html

Mann, S., Brook, P. and Fogarty, S. (1999). Goals for supporting the education needs of the visually impaired with an integrated reading device. Disability in Education: Maximising Everyone's Potential, University of Otago: 10.

Mann, S., Brook, P., Gray, J. T. and Fogarty, S. (1999). General purpose reading device for the visually impaired. Proceedings of the 12th Annual Conference of the National Advisory Committee on Computing Qualifications, Dunedin, National Advisory Committee on Computing Qualifications: 173-181.

Wu, V., Manmatha, R. and Riseman, E. M. (1999) "TextFinder: An automatic system to detect and recognise text in images." *IEEE Transactions on Pattern Analysis and Machine Intelligence* 21(11): 1224-1229.

Xu, Y. and Nagy, G. (1999) "Prototype Extraction and Adaptive OCR." *IEEE Transactions on Pattern Analysis and Machine Intelligence* 21(12): 1280-1296.