COMPUTER VISION BASED TEXT AND EQUATION EDITOR FOR LaTeX

Özcan Öksüz, Uğur Güdükbay

Dept. of Computer Eng.
Bilkent University
06800 Bilkent Ankara Turkey
{oksuz,gudukbay}@cs.bilkent.edu.tr

Enis Çetin

Dept. of Electrical and Electronics Eng.
Bilkent University
06800 Bilkent Ankara Turkey
cetin@ee.bilkent.edu.tr

Abstract - In this paper, we present a computer vision based text and equation editor for LaTeX. The user writes text and equations on paper and a camera attached to a computer records actions of the user. In particular, positions of the pen-tip in consecutive image frames are detected. Next, directional and positional information about characters are calculated using these positions. Then, this information is used for on-line character classification. After characters and symbols are found, corresponding LaTeX code is generated.

1. INTRODUCTION

In this paper, we present a computer vision based text and equation editor for LaTeX. In this system, the user writes text and equations on paper and a camera attached to a computer records actions of the user. Written characters are recognized on-line and the corresponding LaTeX code is generated at the end of each page. Handwriting recognition can be classified into two categories, on-line and off-line, which differ in the form the data is presented to the system. On-line recognition has some benefits over off-line recognition, mostly due to the extended amount of information that is obtainable. In addition to the spatial properties of the stylus, also the number of strokes, their order, the direction of writing for each stroke, and the speed of writing are available. Another advantage of on-line recognition is that there is close interaction between the user and the machine. The user can thus correct any recognition error immediately when it occurs. Moreover, on-line handwriting recognition promises to provide a dynamic means of communication with computers through a pen-like stylus, not just a keyboard. This seems to be a more natural way of entering data into computers [1, 2, 4, 5, 6].

In our system the user writes on a regular paper or a white board and a USB CCD camera attached to the computer captures the video while he or she writes characters. We detect the pen-tips in image frame and determine the bounding boxes of each character. Then chain and region codes are calculated and characters are recognized. Finally, LaTeX codes corresponding to the recognized characters are generated.

The rest of the paper is organized as follows: first, key features of our character recognition system are explained. Next, experiments conducted and their results are shown. Finally, conclusions are given.

This work is partially supported by European Commission 6th Framework Program with grant number FP6-507752 (MUSCLE Network of Excellence Project) and by Turkish State Planning Organization (DPT) with grant number 2004K120720.

2. CHARACTER RECOGNITION SYSTEM

2.1. Preliminary Classification

In our recognition system it is assumed that whole paragraphs or documents are written on a stripped paper, one line at a time. In addition, each line is divided into 3 strips formed by 4 reference lines. By using these reference lines the need for character size normalization is eliminated and the classification of letters is performed by using relative positions of them in the three-strip frame. By performing preliminary classification we tried to reduce the number of possible candidates for an unknown character to a subset of the total character set. For this purpose, the selected domain is categorized into five groups as shown in Figure 1. Table 1 lists all the characters allowed in our system, classified into the five pre-classification groups.

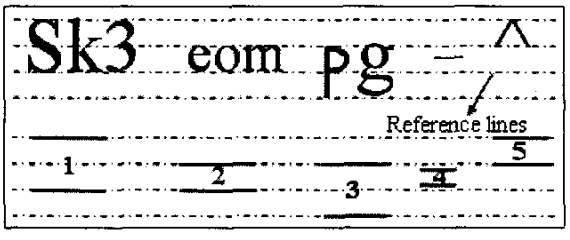


Figure 1: Reference lines and possible locations of five character groups.

Group	Characters of the group
1	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,b,d,f,h,k,l,0,1,2, 3,4,5,6,7,8,9,(,),[,],{,}, $\sqrt{}$,$\sum$,$\prod$,$\int$,Δ,λ, Θ,γ,δ,β,Ω,Φ
2	a,c,e,i,m,n,o,r,s,u,v,w,x,z,*,+,/,<,>, =,',",`,\,π,θ,α,ϵ,σ
3	g,j,p,q,y
4	-
5	^

Table 1: Characters in five groups

2.2. Chain codes

The user writes the characters using an ordinary board marker having black color. In order to trace the pen movements easily, a blue band is attached near its tip and at each frame the position of this blue band is recorded. After pen tip points are found they are processed for chain code extraction. A chain code is a sequence of numbers between 0 and 7 obtained from the quantized angle of the pen tip's point in an equally timed manner. Chain code values for angles and chain coded representation of character E are shown in Figure 2.

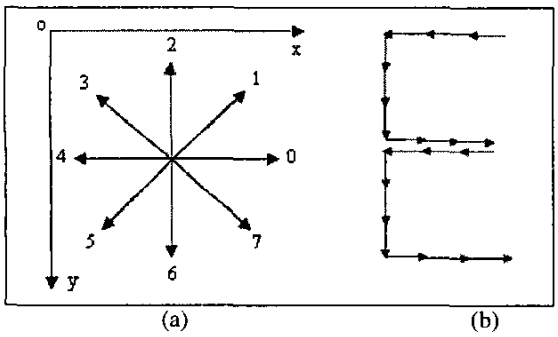


Figure 2: (a) Chain codes (b) Chain coded representation of character E=444666000444666000

2.3. Bounding boxes and their sub-rectangles

After characters are written, the last image of the video is converted into a binary image using thresholding. Then, this binary image is processed to determine the bounding box of each character, which is the minimum rectangle enclosing the written character. Bounding boxes are used to distinguish each character's coordinates on the paper. Then using these coordinates each character's movement points are separated and corresponding chain codes are calculated.

Pen movement points have another usage different from chain code generation. Their positions in the bounding box are used during the classification period. First of all, each calculated bounding box is divided into sub-rectangles. Bounding boxes belonging to the character groups 1 and 3 are divided into 9 different sub-rectangles, and bounding boxes belonging to the character groups 2 and 5 are divided into 5 sub-rectangles, as shown in Figure 3. Then, region coded representation of each character is described using the idea that each separate character stroke should be described with minimum bounding sub-rectangle, as shown in Figure 4.

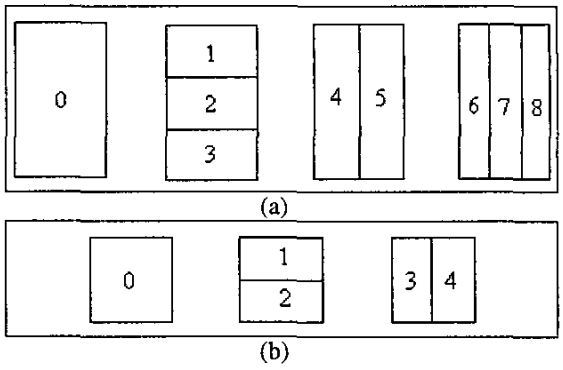


Figure 3: Bounding box sub-rectangles preclassification groups. (a) 1 and 3, (b) 2 and 5

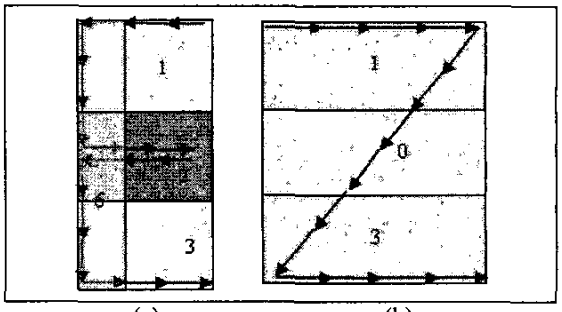


Figure 4: Region coded representation. (a) E=111666222222666333 and (b) Z=11110000003333

2.4. Finite State Machines

The recognition system consists of Finite State Machines (FSMs) corresponding to individual characters. Separated chain codes and calculated region codes are inputs to the FSM. The FSMs generating the minimum error identifies the recognized character. The weighted sum of the error from a finite state machine determines the final error for a character in the recognition process. Finite state machines for some characters are shown in Figure 5.

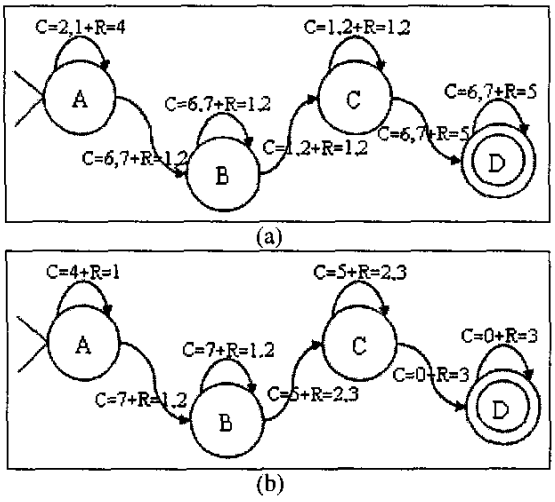


Figure 5: Finite State Machines, where C = Chain Code, R = Region Code. (a) M and (b) $\sum$

Here is an example showing how FSMs work in our system. When the 2322221777111766 6 chain code and the 4444444122115555 region code are applied as an input to M's machine, the first element of chain 2 and region 4 are correct values at the FSM's starting state. Therefore, FSM is started with error count 0. The second element of chain 3 generates an error and the error counter is set to 1. The FSM remains in the first state with other 2s of chain and 4s of region code values. It remains at the same state with the subsequent 1 since 1 and 2 are the inputs of the machine's first state for M. Input of chain code 7 and region code 1 makes the FSM go to the next state, and the subsequent chain codes of three 7s and the region codes of 1 and 2 make the machine remain there. Whenever the chain code becomes 1 and the region code becomes 2, the FSM moves to the third state. The machine stays in this state until the chain code is 7 and region code is 5, and this makes FSM go to the final state. The rest of the input data, chain 6 and region 5, makes the machine stay in the final state, and when the input is finished, the FSM terminates. For this input sequence, the machine's error for character M is 1. However, the other FSMs generate either greater or infinite error values for this input.

2.5. Steps of the Character Recognition Algorithm

The character recognition algorithm is given as pseudo-code in Figure 6. The overall operation does not take long in a PC. Because the character recognition process is an O(N) operation [3].

After characters are recognized, they are passed to LaTeX code generation procedure. Some character combinations are used for LaTeX keyword generation. Some characters have different effects in different environments. Supported LaTeX environments are array construction, citation, section, itemization, equation and normal text environment. Character combination and corresponding LaTeX codes are shown in Table 2.

```
- locate the reference lines
- calculate the bounding boxes
for each bounding box
  - separate pen-tip points inside it
  - construct chain codes
  - apply preliminary classification
  - calculate sub-rectangles
  for each character in preliminary
      classification group
    while character states and pen-tip
          points are not finished do
      if chain code equals to
         character state and
         point is in state rectangle
        - move to next state
      else
        - increase recognition error
        - move to next point
      end
    end
    if character states not finished
      - set error to infinity
  end
  - post process the results
  - find the best match
end
- generate LaTeX Code
```

Figure 6: The character recognition algorithm

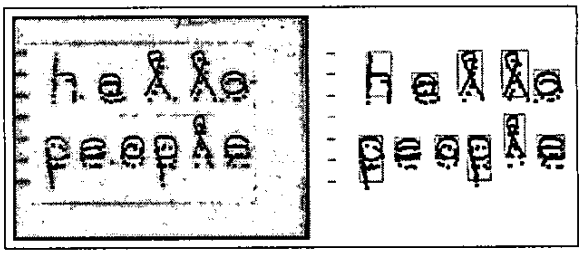
3. EXPERIMENTS

In our experiments we use an ordinary board marker having black color and we attached a blue band near its tip for detection of the pen tip while characters are drawn. We write characters on white A4 paper or a white board, which is assumed to have stripes on it. Consecutive frames are captured using a USB CCD camera that can capture 6.3 frames per second with 320 x 240 pixels. A PC with AMD Athlon 1400 processor with 256 Mbytes of memory is used during recognition for all processing.

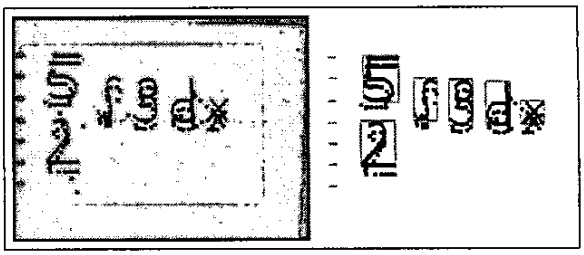
To test the system, we used at least 20 samples for each character and a total of 2170 characters. The system requires 15 image frames per character. The system handles 10 characters per minute. This rate can be improved by using a high resolution camera with a better frame grabber. After conducting experiments, we reached a 90% recognition rate at a writing speed of about 10 characters per minute. The main recognition

Characters	LaTeX code
<a	begin{array}
, in array mode	&
\ in array mode	\\
>a	end{array}
<c	\cite{
>c	}
<e	begin{equation}
>e	end{equation}
<i	begin{itemize} \item
\ in itemize mode	\item
>i	end{itemize}
<s	\section{
>s	}

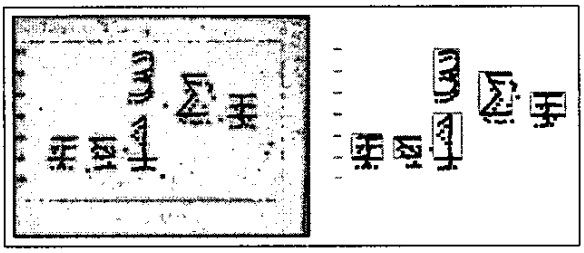
Table 2: Character groups and LaTeX codes



hello people



$\int\limits_{2}^{5}{3dx}$



$\sum\limits_{i=1}^{3}{i}$

Figure 7: Examples and corresponding LaTeX codes.

errors were due to the inaccurate writing habits and ambiguity among similar shaped characters. Most of the confusion was between character pairs such as "e" and "c", "5" and "S", and "u" and "v". This could be avoided by using a dictionary to look-up for possible character compositions. The presence of contextual knowledge will help to eliminate the ambiguity.

In Figure 7, some examples characters and their LaTeX codes are given. In this figure, there are 2 columns: (a) this is the last frame captured during character are written. The pen movement points are shown with blue color. (b) bounding boxes and pen trace points falling within each bounding box.

4. CONCLUSION

In this paper, we present a computer vision based text and equation editor for LaTeX. The user writes text and equations on paper and a camera attached to a computer records actions of the user.

The results show a 90% correct recognition rate. The main recognition errors are due to the inaccurate writing habits and ambiguity among similar shaped characters. This could be avoided by using a word dictionary to look-up for possible character compositions or using a pop-up menu for confused characters. The presence of contextual knowledge will help to eliminate the ambiguity.

The recognition accuracy can be increased using a high rate frame grabber, acquiring 25-30 frames per second. The USB camera can provide 6-7 frames per second, which reduces the accuracy of the chain codes. The reason that we use a USB camera in our system is to realize a low-cost system with standard equipment.

5. REFERENCES

[1] H. Bunke, T. von Siebenthal, T. Yamasaki and M. Schenkel, "Online handwriting data acquisition using a video camera", In *Proc. of Int. Conf. on Document Analysis and Recognition*, pp. 573-576, 1999.

[2] X. Li and D. Y. Yeung, "On-line handwritten alphanumeric character recognition using dominant points in strokes", *Pattern Recognition*, Vol. 30, No. 1, pp. 31-44, 1997.

[3] Ö. F. Özer, O. Özün, C. Ö. Tüzel, V. Atalay and A. E. Çetin, "Vision-Based Single-Stroke Character Recognition for Wearable Computing", *IEEE Intelligent Systems*, Vol. 16, pp. 33-37, 2001.

[4] S. Smithies, K. Novins and J. Arvo, "A handwriting-based equation editor.", In *Proc. of Graphic Interface*, pp. 84-91, 1999.

[5] X. Tang and F. Lin, "Video-based Handwritten Character Recognition", *Proc. of IEEE ICASSP*, 2002.

[6] M. Wienecke, G. A. Fink and G. Sagerer, "Video-based on-line handwriting recognition", In *Proc. of Int. Conf. on Document Analysis and Recognition*, pp. 226-230, 2001.