Word Hunter

Yang Zhao
yzhao3@stanford.edu

Shuo Liu
shuol@stanford.edu

Lingren Zhang
lz7@stanford.edu

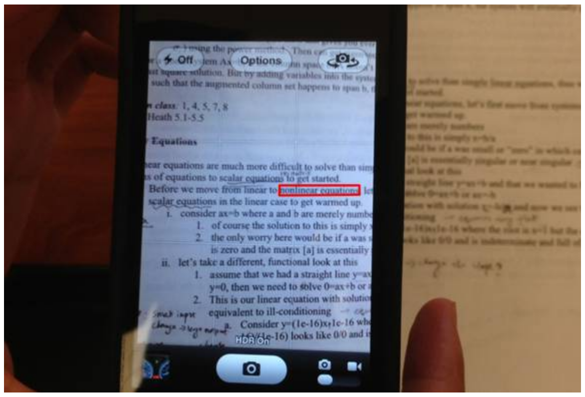


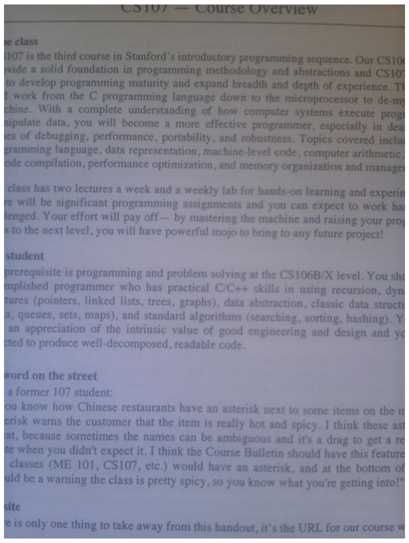
Fig. 1. Word Hunter Demo

***Abstract*—For this project we propose and implement a reading tool on Andriod platform that can be used to identify keywords on paper-based media. After receiving an image of the media, we first pre-process the image (binarization and de-skew), and then use OCR (in particular, Tesseract OCR engine) to recognize the text and find the keywords, at last we highlight the keywords by circling it with a red box.**

I. Introduction

Have you ever read a long paper-based article and find yourself unable to locate keywords? With the advancement of digital media, sometimes we take basic word search for granted. But the world is still full of media printed on paper and it would make our lives much simpler if we can automate this basic reading tool for good old fashioned books. We propose a mobile application that will be able to find a word that the user specified through a phones viewfinder. As soon as the phone detects the word it will highlight it, saving the user many minutes of looking for the word him/herself.

For example, we want to search for nonlinear equation in this page of paper. We only need to use our smart phone to scan over the paper. Whenever the word nonlinear equation appears on the phone screen, it will be immediately circled in red.



CS107 — Course Overview

e class
107 is the third course in Stanford's introductory programming sequence. Our CS10
vide a solid foundation in programming methodology and abstractions and CS107
to develop programming maturity and expand breadth and depth of experience. Th
l work from the C programming language down to the microprocessor to de-my
chine. With a complete understanding of how computer systems execute prog
nipulate data, you will become a more effective programmer, especially in dea
es of debugging, performance, portability, and robustness. Topics covered inclu
gramming language, data representation, machine-level code, computer arithmetic,
ode compilation, performance optimization, and memory organization and manage

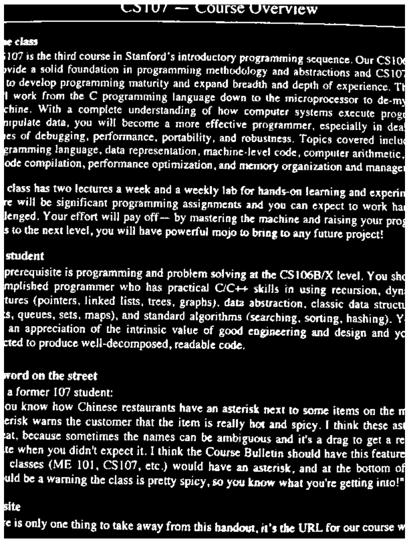
class has two lectures a week and a weekly lab for hands-on learning and experin
re will be significant programming assignments and you can expect to work ha
lenged. Your effort will pay off— by mastering the machine and raising your prog
s to the next level, you will have powerful mojo to bring to any future project!

student
prerequisite is programming and problem solving at the CS106B/X level. You sho
mplished programmer who has practical C/C++ skills in using recursion, dyn
tures (pointers, linked lists, trees, graphs), data abstraction, classic data structu
s, queues, sets, maps), and standard algorithms (searching, sorting, hashing). Y
an appreciation of the intrinsic value of good engineering and design and yo
cted to produce well-decomposed, readable code.

word on the street
a former 107 student:
ou know how Chinese restaurants have an asterisk next to some items on the m
erisk warns the customer that the item is really hot and spicy. I think these ast
at, because sometimes the names can be ambiguous and it's a drag to get a re
te when you didn't expect it. I think the Course Bulletin should have this feature
classes (ME 101, CS107, etc.) would have an asterisk, and at the bottom of
uld be a warning the class is pretty spicy, so you know what you're getting into!"

site
e is only one thing to take away from this handout, it's the URL for our course w

Fig. 2. Before Binarization



CS107 — Course Overview

e class
107 is the third course in Stanford's introductory programming sequence. Our CS10
vide a solid foundation in programming methodology and abstractions and CS107
to develop programming maturity and expand breadth and depth of experience. Th
l work from the C programming language down to the microprocessor to de-my
chine. With a complete understanding of how computer systems execute prog
nipulate data, you will become a more effective programmer, especially in dea
es of debugging, performance, portability, and robustness. Topics covered inclu
gramming language, data representation, machine-level code, computer arithmetic,
ode compilation, performance optimization, and memory organization and manage

class has two lectures a week and a weekly lab for hands-on learning and experin
re will be significant programming assignments and you can expect to work ha
lenged. Your effort will pay off— by mastering the machine and raising your prog
s to the next level, you will have powerful mojo to bring to any future project!

student
prerequisite is programming and problem solving at the CS106B/X level. You sho
mplished programmer who has practical C/C++ skills in using recursion, dyn
tures (pointers, linked lists, trees, graphs), data abstraction, classic data structu
s, queues, sets, maps), and standard algorithms (searching, sorting, hashing). Y
an appreciation of the intrinsic value of good engineering and design and yo
cted to produce well-decomposed, readable code.

word on the street
a former 107 student:
ou know how Chinese restaurants have an asterisk next to some items on the m
erisk warns the customer that the item is really hot and spicy. I think these ast
at, because sometimes the names can be ambiguous and it's a drag to get a re
te when you didn't expect it. I think the Course Bulletin should have this feature
classes (ME 101, CS107, etc.) would have an asterisk, and at the bottom of
uld be a warning the class is pretty spicy, so you know what you're getting into!"

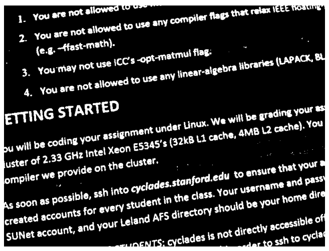
site
e is only one thing to take away from this handout, it's the URL for our course w

Fig. 3. After Binarization

II. Preprocessing

A. Binarization

The first step is to apply an adaptive thresholding algorithm in order to separate text from background in a grayscale image (which can be derived from RGB). The reason we choose adaptive thresholding instead of global thresholding is that the lighting/brightness of the image is not uniform which will cause global thresholding to perform poorly in the extreme bright/dark regions. We also observed that a blockSize of 41 yields the best thresholding result. See Figure 2 and Figure 3 for an example of image before and after binarization.


1. You are not allowed to use
2. You are not allowed to use any compiler flags that relax IEEE floating
   (e.g. –ffast-math).
3. You may not use ICC's -opt-matmul flag.
4. You are not allowed to use any linear-algebra libraries (LAPACK, BL

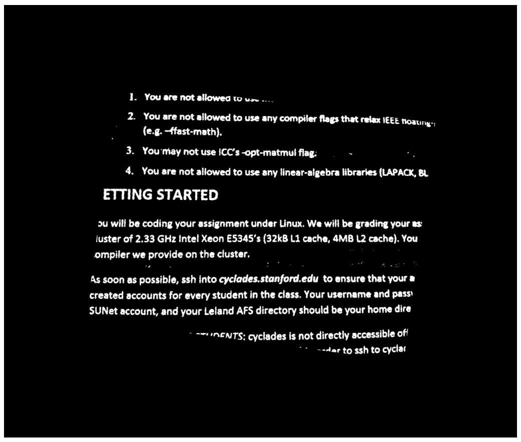
ETTING STARTED

ou will be coding your assignment under Linux. We will be grading your as
luster of 2.33 GHz Intel Xeon E5345's (32kB L1 cache, 4MB L2 cache). You
ompiler we provide on the cluster.

As soon as possible, ssh into *cyclades.stanford.edu* to ensure that your a
created accounts for every student in the class. Your username and pass
SUNet account, and your Leland AFS directory should be your home dire

DENTS: cyclades is not directly accessible of
der to ssh to cycla

Fig. 4. Binarized Image Before Deskew


1. You are not allowed to use
2. You are not allowed to use any compiler flags that relax IEEE floating
   (e.g. –ffast-math).
3. You may not use ICC's -opt-matmul flag.
4. You are not allowed to use any linear-algebra libraries (LAPACK, BL

ETTING STARTED

ou will be coding your assignment under Linux. We will be grading your as
luster of 2.33 GHz Intel Xeon E5345's (32kB L1 cache, 4MB L2 cache). You
ompiler we provide on the cluster.

As soon as possible, ssh into *cyclades.stanford.edu* to ensure that your a
created accounts for every student in the class. Your username and pass
SUNet account, and your Leland AFS directory should be your home dire

DENTS: cyclades is not directly accessible of
der to ssh to cycla

Fig. 5. Binarized And Deskewed

B. De-skew

Hough transform is used to detect (text) lines in the image, we use OpenCV function HoughLinesP. Then we rotate binarized image by the mean of the rotation angles calculated from each text line, we call OpenCV function getRotationMatrix2D to do it. We also need to make sure not to cut off corners in the rotation process, therefore we need to pad the binarized image before rotating. See Figure 4 and Figure 5 for an example of image before and after de-skew. Notice that Figure 5 is slightly larger because of padding.

C. Layout analysis

Find CharSize, Find Neighbour, FindDist.

III. Word Recognition

IV. Conclusion

Acknowledgment

We would like to thank David Chen and Sam Tsai for their generous and responsive help on the project.