Word Hunter

Yang Zhao
yzhao3@stanford.edu

Shuo Liu
shuol@stanford.edu

Lingren Zhang
lz7@stanford.edu

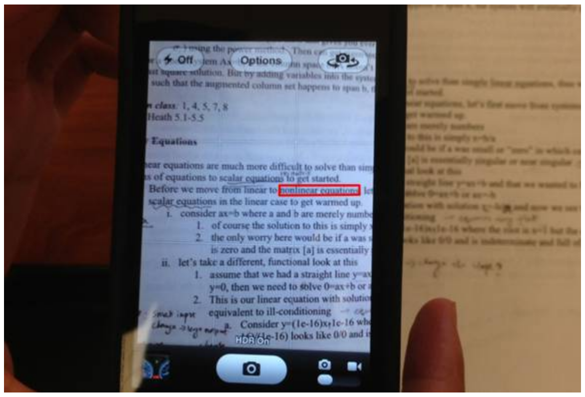
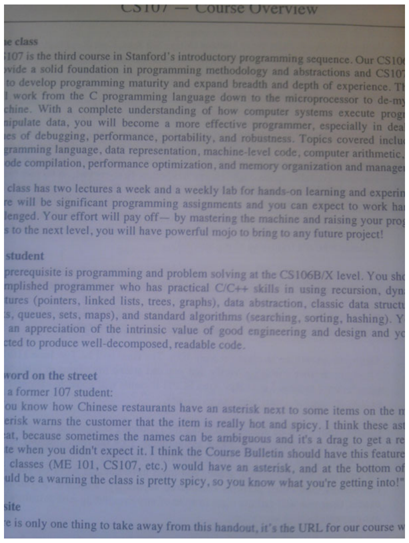


Fig. 1. Word Hunter Demo



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. T
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 experi
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 pro
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 sh
mplished programmer who has practical C/C++ skills in using recursion, dyn
tures (pointers, linked lists, trees, graphs), data abstraction, classic data struct
s, queues, sets, maps), and standard algorithms (searching, sorting, hashing). Y
an appreciation of the intrinsic value of good engineering and design and y
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
erisk warns the customer that the item is really hot and spicy. I think these as
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

***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.

II. Preprocessing

A. Binarization

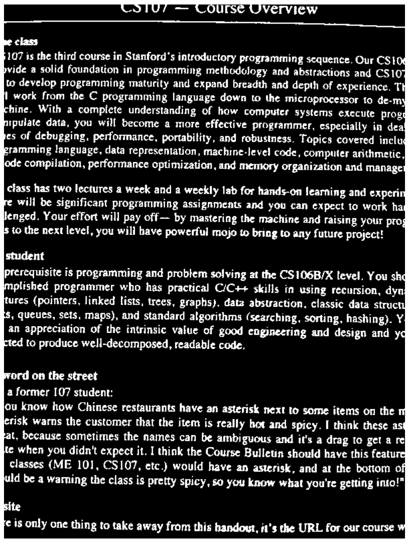
The first step is to apply locally adaptive thresholding algorithm in order to separate text from background in a grayscale image (which can be derived from RGB). The reason we choose locally adaptive thresholding instead of global thresholding is that the lighting/brightness of the image is not uniform which will cause global thresholding to perform poorly in the extreme bright/dark regions.

The idea of locally adaptive thresholding is divide the image into blocks/windows. For each block, use grayscale variance to determine whether the block is uniform. If the block is non-uniform (high variance), apply Otsu's method/global thresholding to the block; if the block is uniform (low variance), classify the entire block as all black or all white based on the mean grayscale value. The reason not to apply Otsu's method to every block is because some blocks maybe entirely background with a number of noise pixels, Ostu's method will keep the noise pixels while classifying the entire block as background will eliminate noise.

We applied OpenCV function adaptiveThreshold for binarization. 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.

B. De-skew

Hough transform (OpenCV function HoughLinesP) is used to detect (text) lines in the image. 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


CS107 — Course Overview

e class

107 is the third course in Stanford's introductory programming sequence. Our CS10
ovide 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 deal
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.

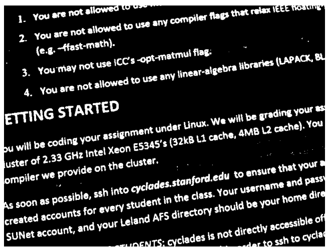
word on the street

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 as
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


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

UDENTS: cyclades is not directly accessible of
rder to ssh to cycla

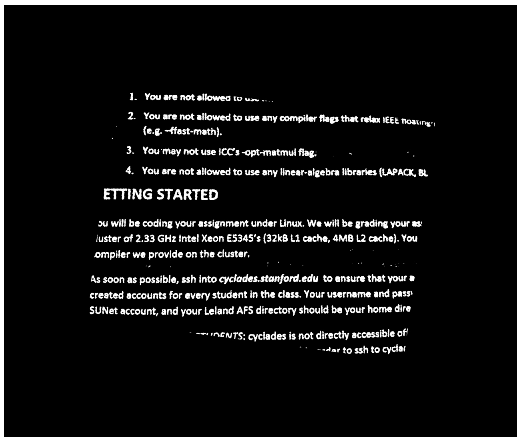
Fig. 4. Binarized Image Before Deskew

image before and after de-skew. Notice that Figure 5 is slightly larger because of padding.

III. Word Recognition

A. Segmentation By Letter

The first step of word recognition is to segment the image into rectangles each containing a letter. Because most letters consists of one connected component in the image, we can draw a contour (OpenCV function findContours and drawContours) around each letter and then find a bounding box of each contour (OpenCV function boundingRect). See Figure 6 for an example of each letter surrounded by a rectangular bounding box.


1. You are not allowed to us
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

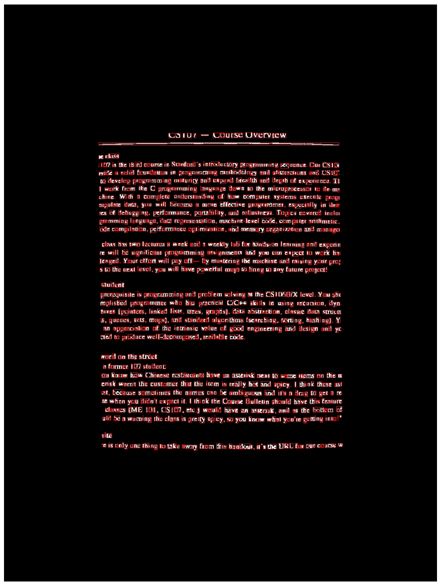
ETTING STARTED

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

UDENTS: cyclades is not directly accessible of
rder to ssh to cycla

Fig. 5. Binarized And Deskewed


CS107 — Course Overview

e class

107 is the third course in Stanford's introductory programming sequence. Our CS10
ovide 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 deal
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 as
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. 6. Segmentation By Letter

Some letters (for instance, lower case "i", "j") consists of two connected components, and hence will have two bounding boxes, but it doesn't affect the algorithm as they will get combined in the next step as we combine letter bounding boxes into word bounding boxes.

B. Segmentation By Word

In this step, we need to combine neighbouring letter bounding boxes into a word bounding box, but there is no existing OpenCV functions that performs this task. We implemented the three functions below in C++:

1) Find Character Size: We implemented function findCharSize to estimate the average height and width of each letter (or its bounding box).

2) Find Neighbour: We implemented isNeighbour to determine where two letters are next to each other in the same word.

C. Tesseract

findEditDistance

IV. Server Client Connection

V. Conclusion

Acknowledgment

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