DIGITAL AUTOMATION OF H.E.B. RECEIPT SCANNING

Vivian Rogers

University of Texas at Austin, EE371Q Digital Image Processsing

ABSTRACT

Sharing an apartment with roommates can incentivize the pooling of groceries, of which the cost breakdown is not necessarily trivial. While the individual items may still require human input to determine their "ownership" on the receipt, much of the process may still be automated. This work proposes an automated scanning of an HEB receipt, such that it can be digested by a text parser and added up via interactive command-line interface, simplifying the process of reimbursement. The success of these scans motivates future works to further simplify the digestion of these receipts, such as the inclusion of automated voice commands or an android application to further streamline the process.

1. INTRODUCTION

Living with roommates, it is often easier to buy groceries as a group and split the bill later. Unfortunately, this process may take upwards of 10-20 minutes, and can involve a great deal of back-and-forth to determine how the costs are shared and counted up. Traditionally, this will be done with a pen and calculator (more recently excel sheet) to "debate" and mark the individual items for Michelle (M), Vivian (V), Angie (A), or Shared (S), tally up the costs, and send out Venmo requests to recompensate the buyer (Vivian) for purchasing the collection of objects. This is generally not desirable, as it gives the buyer (Vivian) a financial motive against going home and napping on the couch if they want to be recompensated. It is empirically known that public excursions, such as grocery shopping, can be rather tiring; no one wants to haggle over the ownership of HEB N.Y SHARP CHUNK, create a spreadsheet, and tally costs up after such an event.

This motivates us to consider what the ideal process flow may look like. In a perfect world, one could take images of the receipt, go line-by-line and denote the owner, and have the Venmo recompensation amounts deposited to one's account after a quick check. To avoid writing secure code to interface with a major financial platform, the scope of the project is reduced to simply calculate the totals for each person. The scope of this work is further reduced to strictly the image processing portion of the project.

2. METHODOLOGY AND STEP-BY-STEP RESULTS

2.1 Project overview

The breakdown of the project is as follows:

1) Images are taken on a smartphone, and uploaded to the home linux (arch) machine via KDE connect.
2) Raw images are taken in for pre-processing, which increases the success rate on the image stitcher.
3) Fragments of the receipt are then assembled together in openCV.
4) With the stitched receipt image, edges are identified and then contoured.
5) The image is transformed to roughly "flat", allowing for easier text recognition.
6) Post-processing is performed to increase the success rate of the OCR software.
7) Text is pulled from the image via pytesseract.

Primarily, openCV in python is used for image manipulation. Imutils' four_point_transform and pytesseract's image_to_string are both used as well, but are not the focus of the project. With that said, let us jump into the pre-processing of the images.

2.2 Pre-stitch processing

One notable feature about receipts is that they are printed on paper, and tend to be made using black ink so that stores may reduce their costs. This gives us a useful heuristic to differentiate a well-lit receipt from background, by considering the color of a pixel. Initially, the idea was to consider the grey scale image as an average, and compute the standard deviation of the pixel from grey, which could be subtracted from the image and turned to black.

$$\sigma=\sqrt{((R-Grey)^2+(G-Grey)^2+(B-Grey)^2)}$$

This turned out to strangely remove parts of the slightly-colored receipt, while leaving parts of the background intact such as glare. Quickly, a more effective metric was determined. Saturation values (and lightness values) are built into the HSV color model, and performed very well at filtering out the background. A blur is applied to this saturation channel to smooth out small peaks such as glare, or white patches like the wood grain on my desk. The lightness channel has an adaptive threshold applied to regularize the color over the document. Finally, these effects are added together and clamped between 1 and 255 for a valid uint8 grey image.

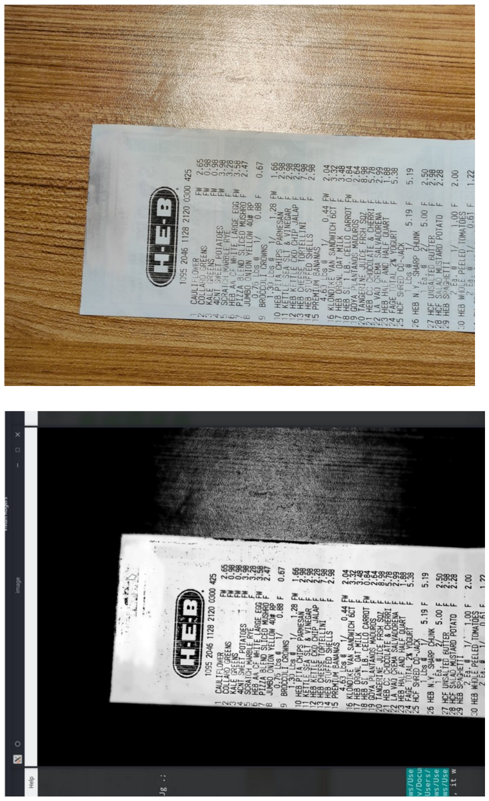


Fig. 1. The pre-processing includes blur, color subtraction, adaptive thresholding and clamping.

To be transparent, the top of the receipt got somewhat greasy from constant handling for this project, and was edited in GIMP to remove the color channel as it was subtracting from the document. This is not expected to happen in common practice. Rather, perhaps latex gloves may become part of the post-grocery recompensation workflow.

2.3 Stitching

One method partially implemented for image stitching was SIFT, where one could identify keypoints and descriptors of the image and determine the correct orientation of the images it is given. This is documented in `stitchImg`. Quickly, I discovered that openCV had a built-in image stitcher, which performed the task generally well. The python wrapper for the C++ openCV code is rather opaque with few options, so little is to be said here.

One option, however, is to switch between “panorama” and “scans” mode. Scan mode tended to poorly assemble the document, and may require more pre-processing tinkering to get working. This may also be because of the document trying to match the glare in different regions, and implies that this should be done on visually flat, non-reflective colored surfaces in the future. Panorama mode induced distortions, however the document tended to come out well overall, so I moved forward with cv2’s stitcher method. Finally, a small black border was added so that the receipt could not run off the page; a rectange with 4 corners can be detected no matter what.

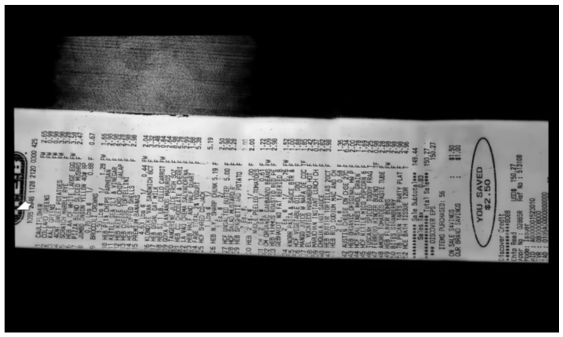


Fig. 2. The stitched image is shown, with an additional border around it to assist the edge detection software.

2.4 Post-stitch processing and contouring

The standard process for edge detection is employed here. First, the stitched image is downscaled greatly, as the many-thousand by many-thousand stitched image is slow to process, and is also blurred by the downscaling. An additional gaussian blur is applied to the downscaled image to filter out higher spatial frequencies, such as the wood grain's reflection. After this, openCV's built-in Canny edge detection was employed, which uses a first-derivative filter and thresholding. With the downscaled image and a small dilation to close any stray edges, a thick border was clearly detected around the receipt.

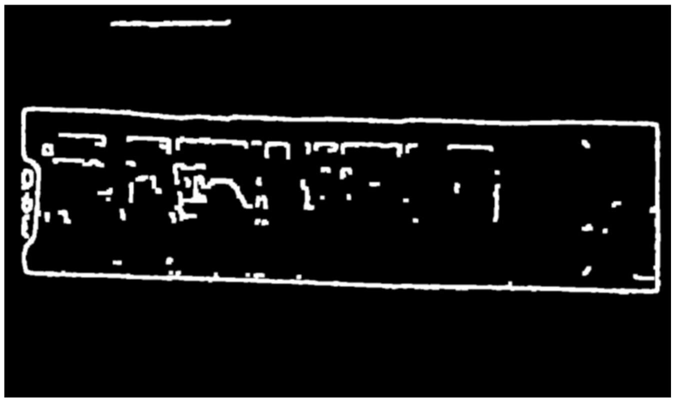
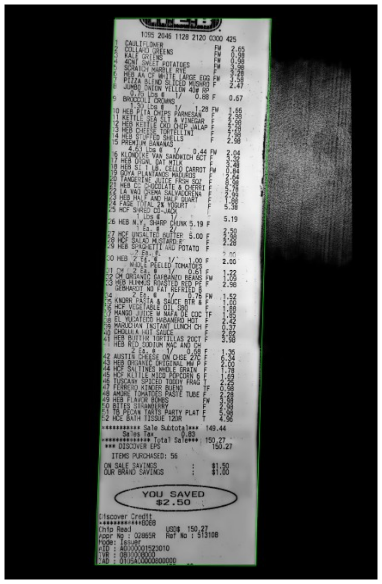


Fig. 3. Re-upscaled and dilated image of the gradient-based edge detection Canny routine.

From here, cv2's findContours function will trace these edges and create a list of the contours in the image. Fortunately, we have a couple methods to determine which contour is of interest in our image. We know that 1) the main contour will have a fixed, closed area, and it will likely be the largest in the document, and 2) an approximation of the contour will be approximately a distorted rectangle, or at least have 4 corners. Both methods should work, however a a contour approximation has to be performed regardless to determine the corners for warping, so we proceed with 2).

The contours of the image are looped through, and we search for the approximation with 4 corners. Having successfully found the correct contour, the software can draw this back onto the original frame for us to verify.



```
1095 2046 1128 2120 0300 425
1  CAULIFLOWER
2  COLLARD GREENS              FW   2.65
3  KALE GREENS                 FW   0.98
4  4CNT SWEET POTATOES         FW   0.98
5  SCRATCH MARBLE RYE          FW   3.98
6  HEB AA CF WHITE LARGE EGG   F    3.28
7  PIZZA BLEND SLICED MUSHRO   FW   3.58
8  JUMBO ONION YELLOW 40# RP   F    2.47
    0.76 Lbs @   1/    0.88 F       0.67
9  BROCCOLI CROWNS
    1.30 Lbs @   1/    1.28 FW      1.66
10 HEB PITA CHIPS PARMESAN     F    2.98
11 KETTLE SEA SLT & VINEGAR    F    2.98
12 HEB KETTLE CKD CHIP JALAP   F    2.28
13 HEB CHEESE TORTELLINI       F    7.98
14 HEB STUFFED SHELLS          F    2.98
15 PREMIUM BANANAS
    4.63 Lbs @   1/    0.44 FW      2.04
16 KLONDIKE VAN SANDWICH 6CT   F    3.32
17 HEB ORGNL OAT MILK          F    3.48
18 HEB SI 1 LB. CELLO CARROT   FW   0.84
19 GOYA PLANTANOS MADUROS      F    2.64
20 TANGERINE JUICE FRSH SQZ    F    8.98
21 HEB CC CHOCOLATE & CHERRI   F    5.78
22 LA VAQ CREMA SALVADORENA    F    2.99
23 HEB HALF AND HALF QUART     F    1.88
24 FAGE TOTAL 2% YOGURT        F    5.38
25 HCF SHRED CO-JACK
    1 Lbs @   1/                    5.19
26 HEB N.Y. SHARP CHUNK 5.19 F
    1 Ea. @   2/                    2.50
27 HCF UNSALTED BUTTER  5.00 F      2.98
28 HCF SALAD MUSTARD           F    2.28
29 HEB SPAGHETTI  ARD POTATO   F
    2 Ea. @                         2.00
30 HEB   2 Ea. @   1/   1.00 F      2.00
    WHOLE PEELED TOMATOES
31 CM   2 Ea. @   1/    0.61 F      1.22
32 CM ORGANIC GARBANZO BEANS   FW   1.09
33 HEB HUMMUS ROASTED RED PE   F    2.98
   GEBHARDT NO FAT REFRIED B
34   2 Ea. @   1/    0.76 FW        1.52
35 KNORR PASTA & SAUCE BTR &   F    1.00
36 HCF VEGETABLE OIL SBO       F    1.88
37 MANGO JUICE W NAFA DE COC   TF   1.85
38 EL YUCATECO HABANERO HOT    F    2.42
39 MARUCHAN INSTANT LUNCH CH   F    0.37
40 CHOLULA HOT SAUCE           F    2.82
41 HEB BUTTER TORTILLAS 20CT   F    3.98
   HEB RED SODIUM MAC AND CH
    2 Ea. @   1/    0.68 F          1.36
42 AUSTIN CHEESE ON CHSE 27C   F    6.34
43 HEB ORGANIC ORIGINAL MW P   F    2.00
44 HCF SALTINES WHOLE GRAIN    F    1.78
45 HCF KETTLE MICO POPCORN 6   F    1.69
46 TUSCANY SPICED TODDY FRAG   T    2.25
47 FERRERO KINDER BUENO        TF   0.98
48 AMORE TOMATOES PASTE TUBE   F    2.28
49 HEB FLAVOR BOMBS            FW   3.98
50 BITES STRAWBERRY            F    2.98
51 TB PECAN TARTS PARTY PLAT   F    5.98
52 HCE BATH TISSUE 12DR        T    4.96
*********** Sale Subtotal***   149.44
     Sales Tax         0.83
************** Total Sale***   150.27
 *** DISCOVER EPS                150.27

  ITEMS PURCHASED: 56

 ON SALE SAVINGS        :     $1.50
 OUR BRAND SAVINGS      :     $1.00

           YOU SAVED
             $2.50

Discover Credit
**************8068
Chip Read           USD$  150.27
Appr No : 02865R    Ref No : 513108
Mode: Issuer
AID : A0000001523010
TVR : 0800008000
IAD : 0105A0000000800000
```

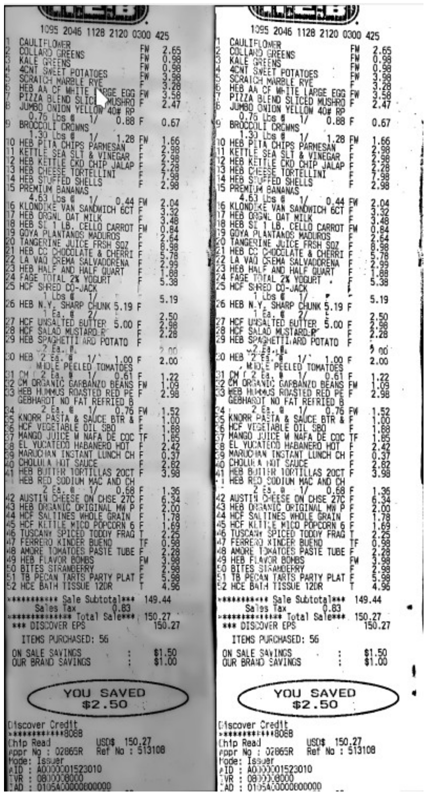
Fig. 4. The outermost contour of the document is outlined in green for the user's verification purposes.

2.5 Warp to upright

After this, the approximated outline can be decomposed by its corners and fed into imutil's four_point_transform. This works in two stages, a constant offset, and a linear transform, though the details of the implementation remain somewhat opaque to me. It can be said that each corner is a vector y, such that y = Ax + b, A is a matrix, and b is a constant offset. four_point_transform determines the mathematical content of this transform for all corners, and spits out what should be a perfectly flat, unwarped image. Of course, this is not exactly the case, as the stitching routine is imperfect and likely requires further pre-processing.

2.6 Final post-processing

While the individual sub-images had well-scaled adaptive thresholds, the stitched sub-images had no means by which to compare inter-subimage threshold scales, and produced slight discontinuities when affixed to one another. To remedy this, a final adaptive threshold with a very large spread is performed on the warped receipt. This makes the text very readable, though it produces some noise in the white areas of the image. To remedy this, a CLOSE operation is performed with a small kernel on the image, which will DILATE and ERODE the white regions of the paper, destroying the noise and then re-expanding the text so that it remains mostly readable to the OCR program.



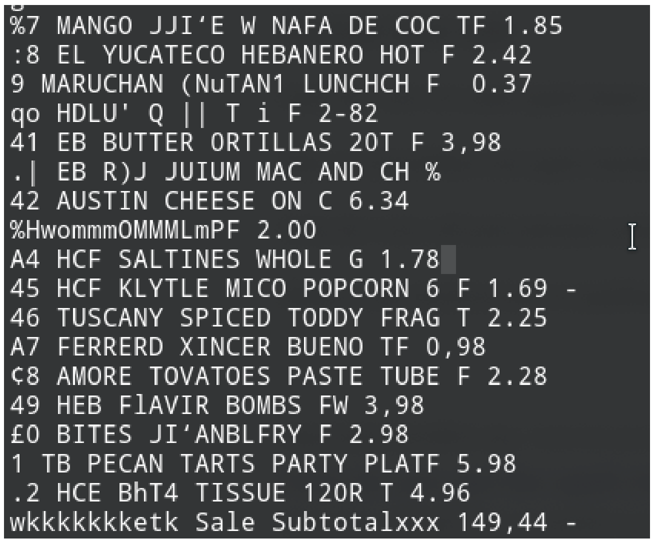
```
1095 2046 1128 2120 0300 425
1  CAULIFLOWER
2  COLLARD GREENS                FW   2.65
3  KALE GREENS                   FW   0.98
4  4CNT SWEET POTATOES           FW   0.98
5  SCRATCH MARBLE RYE            FW   3.98
6  HEB AA CF WHITE LARGE EGG     F    3.28
7  PIZZA BLEND SLICED MUSHRO     FW   3.58
8  JUMBO ONION YELLOW 40# RP     F    2.47
     0.76 Lbs @    1/     0.88
9  BROCCOLI CROWNS               F    0.67
     1.30 Lbs @    1/     1.28
10 HEB PITA CHIPS PARMESAN       FW   1.66
11 KETTLE SEA SLT & VINEGAR      F    2.98
12 HEB KETTLE CKD CHIP JALAP     F    2.98
13 HEB CHEESE TORTELLINI         F    2.28
14 HEB STUFFED SHELLS            F    7.98
15 PREMIUM BANANAS               F    2.98
     4.63 Lbs @    1/     0.44
16 KLONDIKE VAN SANDWICH 6CT     FW   2.04
17 HEB ORGNL OAT MILK            F    3.32
18 HEB SI 1 LB. CELLO CARROT     F    3.48
19 GOYA PLANTANOS MADUROS        FW   0.84
20 TANGERINE JUICE FRSH 59Z      F    2.64
21 HEB CC CHOCOLATE & CHERRI     F    8.98
22 LA VAQ CREMA SALVADORENA      F    5.78
23 HEB HALF AND HALF QUART       F    2.99
24 FAGE TOTAL 2% YOGURT          F    1.88
25 HCF SHRED CO-JACK             F    5.38
     1 Lbs @    1/
26 HEB N.Y. SHARP CHUNK 5.19     F    5.19
     1 Ea. @    2/
27 HCF UNSALTED BUTTER  5.00     F    2.50
28 HCF SALAD MUSTARD             F    2.98
29 HEB SPAGHETTI ARD POTATO      F    2.28
     2 Ea. @                          2.00
30 HEB  2 Ea. @    1/     1.00   F    2.00
   WHOLE PEELED TOMATOES
31 CM  2 Ea. @     1/     0.61   F    1.22
32 CM ORGANIC GARBANZO BEANS     FW   1.09
33 HEB HUMMUS ROASTED RED PE     F    2.98
   GEBHARDT NO FAT REFRIED B
34   2 Ea. @    1/     0.76      FW   1.52
35 KNORR PASTA & SAUCE BTR &     F    1.00
36 HCF VEGETABLE OIL 58O         F    1.88
37 MANGO JUICE W NAFA DE COC     TF   1.85
38 EL YUCATECO HABANERO HOT      F    2.42
39 MARUCHAN INSTANT LUNCH CH     F    0.37
40 CHOLULA HOT SAUCE             F    2.82
41 HEB BUTTER TORTILLAS 20CT     F    3.98
41 HEB RED SODIUM MAC AND CH
     2 Ea. @    1/     0.68      F    1.36
42 AUSTIN CHEESE ON CHSE 27C     F    6.34
43 HEB ORGANIC ORIGINAL MW P     F    2.00
44 HCF SALTINES WHOLE GRAIN      F    1.78
45 HCF KETTLE MICO POPCORN 6     F    1.69
46 TUSCANY SPICED TODDY FRAG      T    2.25
47 FERRERO KINDER BUENO          TF   0.98
48 AMORE TOMATOES PASTE TUBE     F    2.28
49 HEB FLAVOR BOMBS              FW   3.98
50 BITES STRAWBERRY              F    2.98
51 TB PECAN TARTS PARTY PLAT     F    5.98
52 HCE BATH TISSUE 12DR          T    4.96
*********** Sale Subtotal***    149.44
       Sales Tax        0.83
************** Total Sale***    150.27
*** DISCOVER EPS                150.27

  ITEMS PURCHASED: 56

ON SALE SAVINGS         :       $1.50
OUR BRAND SAVINGS       :       $1.00

        YOU SAVED
          $2.50

Discover Credit
************8068
Chip Read             USD$  150.27
Appr No : 02865R      Ref No : 513108
Mode: Issuer
AID : A0000001523010
TVR : 0800008000
IAD : 0105A0000000000000
```

Fig. 5. Left: the pre-postprocessing image is shown. Right: The warped receipt after adaptive thresholding and the CLOSE operation.

To the naked eye, the image at right is rather simple to read, but still somewhat warped. Ultimately, it is up to the OCR routine to decipher this.

2.7 OCR using PyTesseract

Google's Tesseract 4.1 software uses deep convolutional networks for image detection, with a pre-trained package for the english language on the pacman repository. While care could be put into segmenting the different regions in the receipt (item #, name, checkout information, etc), the default PyTesseract text recognition wrapper performed surprisingly well for this, with the slanted text and occasional noise, returning each line of the receipt in a well-delineated manner. Despite this mild success, the OCR struggled with the character recognition on this warped, poorly-joined scan.



```
%7 MANGO JJI'E W NAFA DE COC TF 1.85
:8 EL YUCATECO HEBANERO HOT F 2.42
9 MARUCHAN (NuTAN1 LUNCHCH F  0.37
qo HDLU' Q || T i F 2-82
41 EB BUTTER ORTILLAS 20T F 3,98
.| EB R)J JUIUM MAC AND CH %
42 AUSTIN CHEESE ON C 6.34
%HwommmOMMMLmPF 2.00
A4 HCF SALTINES WHOLE G 1.78
45 HCF KLYTLE MICO POPCORN 6 F 1.69 -
46 TUSCANY SPICED TODDY FRAG T 2.25
A7 FERRERD XINCER BUENO TF 0,98
¢8 AMORE TOVATOES PASTE TUBE F 2.28
49 HEB FlAVIR BOMBS FW 3,98
£0 BITES JI'ANBLFRY F 2.98
1 TB PECAN TARTS PARTY PLATF 5.98
.2 HCE BhT4 TISSUE 120R T 4.96
wkkkkkkketk Sale Subtotalxxx 149,44 -
```

Fig. 6. A screenshot of the zshell terminal shows the OCR's best guess at the content of our HEB shopping trip.

One can see that the OCR performed particularly poorly on the item numbers in the leftmost column of characters, given that the machine printed a line up the length of the cheap document. Despite this, it identified the prices reasonably well, and the items are nearly recognizable.

3. DISCUSSION

For a well-thought-out text parsing script, along with the option for human intervention to interrupt and correct a broken price, this project appears to be a mild success.

While it is unfortunate that the accuracy of the OCR seems to be low, given the bad data it is being fed, this could likely be corrected with more reflection on the stitching method and pre-processing. Perhaps the partially-implemented SIFT method could be finished, or replaced with SURF to join the areas of the document. Another approach would be to regularize the threshold scale on each of the sub-images before stitching them together.

Naturally, this needs more testing with future receipts, in other lighting environments and backgrounds so that our apartment can find the optimal workflow for parsing these receipts and relaxing after a busy night out on the supermarket.

This paper motivates future works, such as making the software into something that is not actively unpleasant to use. KDE connect can execute remote commands and mount a phone’s file system for example, which could be good to automatically pull the images after their creation, instead of digging through filepaths via command-line-interface.