

String Extraction From Color Airline Coupon Image Using Statistical Approach

Yi Li, Zhiyan Wang, Haizan Zeng
School of Computer Science

South China University of Technology, Guangzhou, 510640, P.R.China
cs_liyi@hotmail.com wangzhiy@ieee.org cs_haizanzeng@hotmail.com

Abstract

A novel technique is presented in this paper to extract strings in color images of both Business Settlement Plan (BSP) and non-BSP airline coupon. The essential concept is to remove non-text pixels from complex coupon images, rather than extract strings directly. First we transfer color images from RGB to HSV space, which is approximate uniformed, and then remove the black component of images using the property of HSV space. A statistical approach called Principal Components Analysis (PCA) is applied to extract strings by removing the background decorative pattern based on priori environment. Finally, a method to validate and improve performance is present.

1. Introduction

There're millions of airline coupons needed to be processed by the airline companies everyday. The staff of airline company settlement center needs to record the contents in airline coupon, i.e. carrier, airport code, flight number, etc. Therefore, they need an automatic application to process coupons to accelerate their working accuracy and productivity. We developed a system called ATOS (Airline Tickets Operating System) to satisfy this requirement and the research in this paper is part of ATOS.

The first problem we encounter in the ATOS application is that coupon images are difficult to analysis using general algorithms because layouts of the coupon are different and unique for different airline company. Other problems include (1) strings touch with form lines frequently, (2) strings may be smudgy and (3) color distribution of content strings may be very closed to that of background color. In our case, background of Business Settlement Plan (BSP) coupon comprises light red and light green pixels, and that of the non-BSP coupon comprises light gray pixels. (Fig 1)

Basically, printings in coupon can be classified into two kinds: (i) pre-printed blank coupon table with background and (ii) filled-in computer printed strings, which are printed by the ink layers on the back of the upper coupon in certain format. Therefore, strings could be shifted out of

the table and images can't be processed by the graph-text separation algorithms in the analysis of drawings or maps [1].



Fig 1. A Typical BSP Coupon

2. Related Works

In past years, few researchers focus on the airline coupon because of low requirement. As the business requirement becomes stronger, more researchers are involved in this field recently. J.Mao's research [2] is one of the excellent examples, but it is based on gray level and the algorithm can only handle the BSP coupon.

To overcome the disadvantage of gray scale images and with development of the high-speed color scanner, we can use color information to extract specific color out of the background, which works as a pre-process phase to enhance input of Mao's algorithm.

We present a novel technique to extract strings in both BSP and non-BSP color airline coupon images. It is based on statistical theory, and the process algorithm is robust and adaptive. The result of spatial and temporal is suitable for high-speed computing.

3. Color Space Comparison and Selection

Although source images are stored in RGB color space, it is not easy to process them because RGB color space is not perceptually uniform [3,4]. We should choose a uniform color space to process computer vision problems. There're many color models to satisfy our need. In our research, we compare CIE L*a*b, YCbCr and HSV and finally we choose HSV color space.

In YCbCr color model, Y stands for the gray scale according to definition, which is the same with traditional

This paper is supported by Guangdong Provincial National Science Funding, Project B6-109-497

gray images. Therefore we only focus on the CbCr sub-plane. We analyze Cb-Cr plane and can't find dominant and similar features between different coupon types. CIE L*a*b color space is another standard to specify color [5], and a comparison of CIE L*a*b and HSV color space [3] shows that the HSV is slightly better than CIE L*a*b color model in computer vision field.

HSV decouples the intensity information from the color [4,6], which means that we can remove the black component (i.e. form lines) easily and the coupon can be adjusted to luminance invariant. We can analyze this uniform color space to extract corresponding strings intuitively.

In the HSV color model, if the V-value is fixed, the H-S pair is in a polar coordinate, so we use the orthotropic representation by converting it to an orthotropic coordinate after we get the result of (H,S,V) by the algorithm RGB_To_HSV() in reference [7].

4. Extract Strings by Background Removal using Principal Components Analysis

There're two kinds of pixels we need to sample to train and test our algorithm according to the classification of printings in Section 1: (i) text pixels and (ii) background pixels. Basically, background pixels are used for training and text pixels are used for result validation. However, even if we mass select text pixels in coupon images in our database carefully, we can't reflect its true distribution, which is mainly caused by the reason that they're printed by printers of various qualities and various situations.

To overcome the problem of sampling text pixels, we can exert a technique as follows. We select coupons whose reverse side is covered by ink paste and choose some seals whose shapes are known. Place these coupons onto white papers with ink layers downward, and press seals in the upper side of selected coupons by using different pressure. Therefore, we can retrieve their images in white papers and we can sample the text strings in a simulated method.

Note that the distribution of these text samples varies much and therefore they can't be used for training. However, we can still use them in test phase to validate whether most text pixels remain in images with most background pixels removed.

Therefore, we conclude two sampling principles as follows:

Sampling Principles:

1. Background Pixel Sampling: To sample the background, we must select and copy the background without the color of ink layers because that may poison the performance manually.
2. String Pixel Sampling: Use the technique in previous three chapters to sample text bitmaps of different qualities and shapes.

According to these two principles, we analyze the color distribution of coupons in H-S-V space (Fig 2). In our experiment, we find that the pixels of background decorative pattern are located in a pretty uniform space, while string pixels in other locations, however, it's not easy to take them apart because some of them are so close, especially the light red background and the red strings.

We can first remove the black component of each pixel by determining the V-value (i.e. we apply Otsu algorithm to binarize V values in light-plane, and retrieve the pixels whose values are larger than thresholding), which the form border and form lines are removed, and now the problem is how to extract strings from similar background. There're two approaches to perform this task: (1) extract strings directly and (2) remove the background.

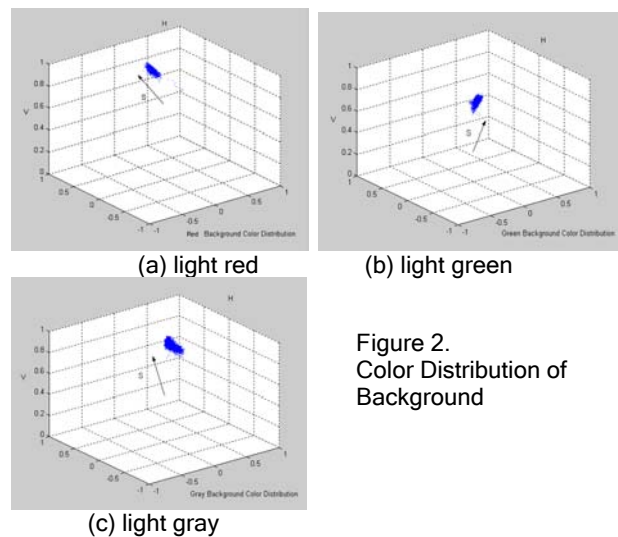


Figure 2.
Color Distribution of
Background

The first approach is intuitive but can't work properly in our case due to the reason we state before. On the contrary, we can sample the background decorative patterns because they're printed in limited factories and have a universal standard (BSP coupon) or company standard (non-BSP coupon). Therefore we can remove the reverse part of strings – the standardized, constant and regularly background pattern. In order to remove them effectively and robustly, we compare some pattern recognition algorithms in the following analysis.

Many pattern recognition methods are tested to segment the string out of the background. Basically, Linear Discrimination Analysis (LDA) works but since it just uses a plane to segment the space, the result can't be improved and adaptive to satisfy our need. Secondly, we use the multi-LDA, and the process phase become complicated and it is also non-adaptive. Third, cluster methods are widely used in color segmentation [8], in our research, we use some unsupervised cluster methods, like k-mean and hierarchical cluster, but the result is not reliable since most

of the cluster methods don't have any previous knowledge of the image.

To unsupervised learn from the priori environment, we can use the statistical method. Self-organized learning is one of unsupervised learning technique. The proposal of self-organized learning is to discover significant features during computing of statistical values. It's adaptive because it (1) can learn from the priori statistical environment, and (2) can be easily re-computed without user intervention, when environment changes.

The scenario that fits the statistical methods is the environment whose features are statistical. In our case, this requirement is satisfied because (1) the background patterns are mostly uniform, (2) the background color clusters in a stable location of color-space. Therefore we can compute background statistical values in the image samples, and extract the main features of these values to represent the original data.

Principal Components Analysis (PCA) is based on unsupervised self-organized theory [9], it use the previous statistical knowledge to form a feature space. The feature space represents the original data set with minor feature ignored. We can transfer pixels in data space to feature space by using major features to represent the original data, which also reduce the dimension of essence of data.

We compute the principal components of the background, and calculate the percentage of the total variance in the observations explained by each eigenvector. The result is:

$$\begin{cases} ObsVar_{RED}=[94.831\%,4.2704\%,0.8918\%]^T \\ ObsVar_{GREEN}=[69.59\%,24.26\%,6.15\%]^T \\ ObsVar_{RED}=[79.566\%,11.626\%,8.8.84]^T \end{cases} \quad (1)$$

We can find that the first two principal components roughly took more than 91% of the total variability. We can conclude that we can count the first 2 principal components as main features of the background pixels.

As a conclusion, we can sample the pixels of the background decorative pattern, and then compute their statistical values, finally we can remove the similar pixels in the image and leave the strings.

5. Extraction Algorithm and Performance Improvement

There are three phases in our algorithm. First we compute the statistical data by background bitmap samples. Second, we apply these statistical values to process the airline coupon and extract strings from background. Finally we present our method to test and improve the performance.

5.1. Unsupervised Learning Phase

Suppose a background sample set contains k bitmaps whose size is the same with r by s , and p is the dimension of each pixel element, which is 3 in our case. We scan each bitmap from left to right and from top to bottom, connect two bitmaps nose to tell. Therefore, we can store it in a vector V with the length $r \times s \times k$, and each element of V is a p dimensions element to store H-S-V values.

Compute the mean value M of the V , and let each element in V minus M to create a zero mean vector called X . In mathematical formula, that is:

$$X=[V_1-M, V_2-M, \dots, V_{r \times s \times k}-M]^T \quad (2)$$

Compute the covariant of the matrix X using Singular Vector Decomposition (SVD) method, and we can get a 3 by 3 covariant matrix Σ . Calculate eigenvector and eigenvalue of Σ and we can get 3 eigenvectors $PC[i], i \in [1,2,3]$ and their corresponding eigenvalues EV . We sort the principal components $PC[i], i \in [1,2,3]$ based on eigenvalue EV in descending order, and get the sorted eigenvector $PC[1], PC[2], PC[3]$. Determine if the first two leading eigenvalues took more than 85% variability of the whole and if answer is true, $[PC[1], PC[2], PC[3]]^T$ is the transform matrix from HSV data space to feature space, and first two principal components are the main features.

5.2. Process Phase

Assume each input pixel is $P=[r, g, b]^T$, convert P to HSV color space using RGB_To_HSV() in [7], and then transform from HSV data space to feature space based on $[PC[1], PC[2], PC[3]]^T$ in the first phase, and set the third value of feature coordinate to zero. Finally we transfer the result from feature space to data space. In mathematical formulas, that is:

Let

$$P1 = P - M \quad (3)$$

$$B = PC \times P1 \quad (4)$$

Set $B[3] = 0$

Let

$$P2 = PC^{-1} \times B \quad (5)$$

$$Dist = || P1 - P2 || \quad (6)$$

Calculate the Euclid distant from $P1$ to $P2$, set it to white color if result of Equation (6) is larger than a pre-specific threshold called *Threshold*. The threshold may vary for each type of coupon and in the next phase we'll adjust it by the performance.

5.3. Performance Improvement Phase:

To improve the performance of the algorithm, we can adjust *Threshold* by the following technique. This is also the fundamental algorithm to proof our experimental

result. Note that we compute different statistical classifiers for different color sample training sets, but for simplification and without losing generality, we only demonstrate how to improve performance of only one specific color class.

We sample specific color text and its corresponding background's pixels using the method in "Sampling Principles". These two sets are called $SzSampleSet$ with $SzSetSize$ elements and $BGSampleSet$ with $BGSetSize$ elements, respectively. Bitmap size of each set is $r \times s$ for simplification. Initialize the pre-specific threshold value as $initThreshold$.

Train PCA classifier with k elements in $BGSampleSet$, apply this classifier to $SzSampleSet$ and remaining elements in $BGSampleSet$ with $Threshold = initThreshold$. Note that pixels in each result image are those pixels that aren't removed, we calculate the ratio of remaining pixels to whole bitmap size, therefore, we get two arrays called the $Sz_REMAIN[j]$, $i \in [1..SzSetSize]$ and $BG_REMAIN[j]$, $j \in [1..BGSetSize-k]$, respectively

Let bitmap's area as the weight, we can calculate the weighted rate of remaining pixels:

$$Weighted_SZREMAIN = \frac{(\sum_{i=1}^{SzSetSize} SZ_REMAIN_{(i)})}{SzSetSize} \quad (7)$$

$$Weighted_BGREMAIN = \frac{(\sum_{j=1}^{BGSetSize-k} BG_REMAIN_{(j)})}{(BGSetSize-k)} \quad (8)$$

and let

$$Rate = \frac{1-Weighted_SZREMAIN}{Weighted_BGREMAIN} \quad (9)$$

Note that samples and the pre-computed statistical value aren't changed during each improvement phase, so the $Rate$ is the function of only one variable – $Threshold$. Thus we can adjust this value from $initThreshold$ to the best value by finding the maximal $Rate$ in reasonable temporal and spatial cost.

6. Experiment and Result

Let n be the number of total pixel in a coupon image and m be the dimension of each element. We can figure out the time complexity based on n . In the learning phase, the time complexity will vary because of the difference methods to compute the Principal Components. In one of the widely used methods, the time complexity of computing mean vector is $O(n \times m)$, of computing eigenvector is $O(n \times m^2)$ and of sorting is $O(m \log m)$. In out case, the m is equal to 3, so the total complexity is still

$O(n)$. In the processing phase, the time cost will be in the mean vector computing, transformation from the original HSV color space to the orthotropic coordinate based on the Principal Components and vice versa. The time complexity of computing mean vector is $O(n \times m)$, of computing first transformation is $O(n \times m^2)$ and of computing second transformation is $O(n \times m^2)$, so the total complexity is still $O(n \times m^2)$. In out case, the m is equal to 3, so the total complexity is $O(n)$.

According to the definitions in Section 5, we set $r = 30$, $s = 30$ and $k = 50$ in our experiment, which means 50 sampling bitmaps whose size is 30×30 . We select totally other 250 bitmaps, including 130 red-green BSP coupons and others 120 non-BSP coupon samples, in our coupon database to test and compute statistical data.

As to the accuracy, we use the method in "Performance Improvement Phase" to proof. We first sample the background pixels, and process the bitmap using our algorithm, then calculate the remaining pixels. A table of the tested data is showed as following:

Table 1. Rate of Removed Background Pixels

Background Type	Statistical Parameters	Min (%)	Max (%)	Avg (%)
BSP red background	light red background	75	96.3	93
BSP green background	light green background	99.7	99.99	99.97
Non-BSP background	light gray background	99.47	99.96	99.88

The result shows that we can remove the background in most cases with little noise left. The average rate is over 90% is acceptable.

We also sample the string color using a known image by "Sampling Principles 2" and process the bitmap using our algorithm, then calculate the remaining pixels. A set of tested data are showed as following:

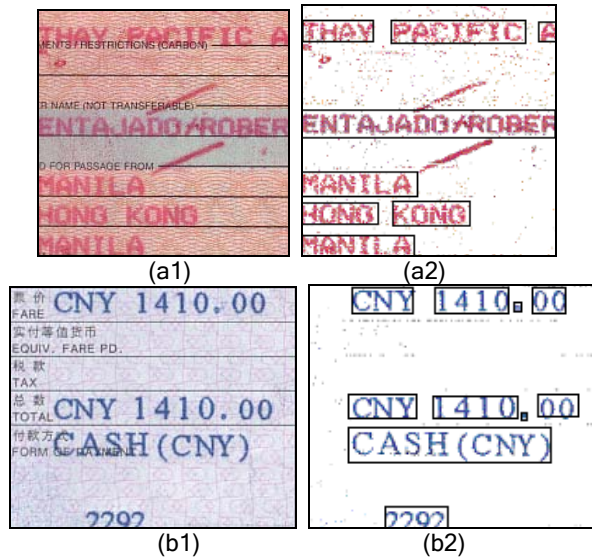
Table 2. Rate of the Remaining String Pixels

Coupon Type	Statistical Parameters	Min (%)	Max (%)	Avg (%)
BSP red background	light red background	1.68	19.07	6.36
BSP green background	light green background	0.11	24.66	3.46
Non-BSP background	light gray background	0.074	14.29	4.61

By judging coupons' background color, we can apply corresponding statistical data and threshold to extract the strings and then use connected component technique to

reduce noises and segment the strings. Result shows that most strings pixels still exist after the process. The average rate lower than 7% is acceptable. Fig 3 shows an example of the performance.

Note that in the example figure, strings touching with form lines are separated without losing information, blurry strings and the strings whose color is similar to the background can be extracted clearly.



(a1)BSP coupon, before process
 (a2)BSP coupon, after process
 (b1)non-BSP coupon, before process
 (b2)non-BSP coupon, after process

Fig 3. An Experimental Result

7. Conclusion

This paper presents a novel technique to extract strings from color airline coupon images. It is based on statistical theory, and the process algorithm is robust and adaptive. The result of spatial and temporal is suitable for high-speed computing.

As a conclusion, the algorithm is efficient because (1) the learning phase is faster than other approaches like neural networks and (2) the processing phase is fast because it just traverse the whole image only once. The algorithm is effective because most background pixels are removed while most string pixels still exist, and the strings in various situations can be extracted clearly without losing the information. The algorithm is also adaptive and robust for it can be re-computed if the environment is changed and it won't need the intervention of the user.

Reference:

[1] Peter Tofani, Rangachar Kasturi, Segmentation of Text From Color Map Images. 14th International Conference on Pattern Recognition, August, 1998

[2] J. Mao, R. Lorie and K. Mohiuddin, A System for Automatically Reading IATA Flight Coupons 4th International Conference Document Analysis and Recognition (ICDAR '97) Page(s) 153-157

[3] Paschos, G., Perceptually uniform color spaces for color texture analysis: an empirical evaluation, IEEE Transactions on Image Processing, Volume: 10 Issue: 6, June 2001 Page(s): 932 -937

[4] G. Wyszecki and W. S. Stiles, Color Science, Concepts and Methods, Quantitative Data and Formulae, 2nd ed. New York: John Wiley & sons, 2000.

[5] Vrhel, M.J.; Trussell, H.J, Problems in publishing accurate color in IEEE journals, IEEE Transactions on Image Processing, Volume: 11 Issue: 4, April 2002, Page(s): 373 -376

[6] H.Palus, D.Bereska, The Comparison Between Transformations from RGB Color Space to HIS Color Space, Used for Object-Recognition, Proc. Int. Conf. Image Processing and its Applications (IPA'95), Edinburgh, July 1995, Page(s) 825-827

[7] Foley, V.Dam, Computer Graphics, Principles and Practice, 2nd Edition, Page 579-599, Addison Wesley, 1998

[8] Chi Zhang and Patrick Wang. A New Method of Color Image Segmentation Based on Intensity and Hue Clustering, 15th International Conference on Pattern Recognition, September 2000

[9] Simon Haykin, Neural Networks – A Comprehensive Foundation (2nd Edition), Page 392-440, Prentice-Hall Inc, 1998