READING STREET SIGNS USING A GENERIC STRUCTURED OBJECT DETECTION AND SIGNATURE RECOGNITION APPROACH

Sobhan Naderi Parizi, Alireza Tavakoli Targhi, Omid Aghazadeh and Jan-Olof Eklundh
Computational Vision and Active Perception Laboratory
Royal Institute of Technology (KTH)
SE-100 44, Stockholm, Sweden

Keywords: Structural object detection, Text detection, Text segmentation, Text recognition.

Abstract: In the paper we address the applied problem of detecting and recognizing street name plates in urban images by a generic approach to structural object detection and recognition. A structured object is detected using a boosting approach and false positives are filtered using a specific method called the texture transform. In a second step the subregion containing the key information, here the text, is segmented out. Text is in this case characterized as texture and a texton based technique is applied. Finally the texts are recognized by using Dynamic Time Warping on signatures created from the identified regions. The recognition method is general and only requires text in some form, e.g. a list of printed words, but no image models of the plates for learning. Therefore, it can be shown to scale to rather large data sets. Moreover, due to its generality it applies to other cases, such as logo and sign recognition. On the other hand the critical part of the method lies in the detection step. Here it relied on knowledge about the appearance of street signs. However, the boosting approach also applies to other cases as long as the target region is structured in some way. The particular scenario considered deals with urban navigation and map indexing by mobile users, e.g. when the images are acquired by a mobile phone.

1 INTRODUCTION

Recognition of an outdoor location from images taken from an indefinitely wide variety of scenes full of different objects is not a straightforward problem. It is even more challenging if we want to search for specific elements of these scenes such as signs, logos or specific buildings. One way of approaching the problem could be to recognize street name plates or other landmarks in the scene. That would allow us to localize our position, index into a map and to recognize the scene, whether or not we had access to additional information from other knowledge sources including GPS (Global Positioning System) which is not always reliable or available in urban areas. In the application behind this work the images come from cell phones or other mobile devices. Then some general contextual information that limits the scope of the problem may be available. Positioning information can be sent by cell phones to the service provider transceiver stations (BTSs). Result of this information is a rough estimate of location of the cell phone user as shown in Figure 16. We address the problem of recognizing street name plates in urban scenes by a rather general approach of structural object detection/recognition. More precisely we define the problem as consisting of three steps: the problem of detecting a specific class of objects about which you have some a priori knowledge, *the object detection problem*, the problem of finding an area which holds some representative information, in this case areas containing text, *the (within object) segmentation problem* and finally, the problem of recognizing the information in this area, *the recognition problem*. In the paper we will introduce a fast, accurate and general purpose framework for dealing with the three mentioned steps, i.e. detection and recognition of parts holding some representative information, in this case text plates. It is assumed that there is a signature that uniquely identifies the output. The way we have formulated the approach it has many other applications as well, although we here focus on finding and interpreting street name plates in urban scenes. In order to have a full overview of what is done in our detection and recognition scenario, you

Naderi Parizi S., Tavakoli Targhi A., Aghazadeh O. and Eklundh J. (2009).
READING STREET SIGNS USING A GENERIC STRUCTURED OBJECT DETECTION AND SIGNATURE RECOGNITION APPROACH.
In *Proceedings of the Fourth International Conference on Computer Vision Theory and Applications*, pages 346-355
DOI: 10.5220/0001797703460355

can have a look at Figure 3.

Throughout this paper we experiment with our methods a new database of street plate objects taken from central part of *Graz* city in *Austria* (mostly around *Herrengasse* street)[1] . To have a sense of the location where the images are taken from you can have a look at Figure 1.

Figure 1: Streets included in the database are marked as red in the map of *Graz* city. The long tilt street is *Herrengasse*.[2]

This database consists of two sets of images. The first set contains 86 outdoor images of street plates in various view points, scales and light conditions taken by an ordinary cell phone camera with 1280 $\times$ 960 pixels of resolution. The second set is comprised of 120 images of similar scenes but taken by a different camera (2448 $\times$ 3264 pixel images). For each single plate, there are 9 images with three different scales and three different viewpoints. Variation of scales and orientations of the target objects within the database are showed in Figure 2.

In the remainder of the paper we first review some related work. Section 2 goes through the detection and recognition phase. A detailed discussion on methods investigated for localization of the target objects is found in Section 2.1. Filtering false positives by post-processing is described in 2.1.1. In particular we explain a specific technique, the LU-transform in detail there. Then the methods used for segmentation of textual regions of interest are presented in Section 2.1.3. The last step of our framework, recognition of the text images, is explained in Section 2.2. In Section 3 we present a system where our solutions come in handy. We finally make some conclusions in Section 4.

[1]This is a new database created by us and will be available for public after publication of this paper.

[2]Screen shot from Google map search.

1.1 Related Work

There has been a large amount of research on object detection, text segmentation (detection) and text recognition. We need a method which can provide a solution to these three tasks simultaneously and we want it to be accurate, fast, general and insensitive to illumination variations and affine transformations. We also want the method to perform well without any specific assumptions about the font, size and any other characteristics of the text.

Although character based text recognition has attracted considerable attention in research, there are cases that character recognition is not easy or feasible e.g. at high levels of noise, low resolution pictures (commercial OCR software usually works on scanned data). There has been some work on word-level text recognition that have shown promising results (Adamek et al., 2007) (Ishidera et al., 2002) (Ataer and Duygulu, 2007). Due to the limitations of character based methods and since we have a limited vocabulary, we based our method on such an approach.

The work most similar to ours is found in the solutions to License Plate Recognition (LPR) problem (Shapiro et al., 2006) (Shapiro and Gluhchev, 2004) (Yan et al., 2001) which also addresses the three subproblems together. There are special constraints in the LPR problem which do not hold in our problem e.g. we have a larger variation of affine transformations (scale and rotation) for street plates (Figure 2). In addition, our approach deals with the detection and recognition problems in a more general way which in principle allows it to detect and recognize any specific object, logo or sign.

There have also been successful solutions to the text detection problem (Ye et al., 2007) (Kim et al., 2003) (Merino and Mirmehdi, 2007) which do not provide any solution to the recognition part. The approach to the text detection problem presented in (Merino and Mirmehdi, 2007) is very similar to ours. However, their method does not recognize the detected text. Some other published text detection methods require the text regions to have very specific characteristics to work well (Kim et al., 2003) e.g. sharp edges, good resolution, constraints on where the text is located etc. Therefore, they have limited use, such as detection of subtitles or rendered text on still images and video sequences and thus, are not applicable in general outdoor scenarios. Some successful solutions to the recognition problem in contexts related to ours have been presented that either assume the detection phase a solved problem (Ataer and Duygulu, 2007) (Ganapathi and Lourde, 2006) or use specific constraints e.g. high resolution (Ataer

(a) (b) (c) (d)

Figure 2: The database contains a wide variety of outdoor images each of which contains a street plate inside and comes with 3 different viewpoints {*-45, 0, 45*} and 3 scales {*large, medium, small*}. So, for each street plate we have 9 images in the database. Range of viewpoint and scale variation is represented by these four pictures.

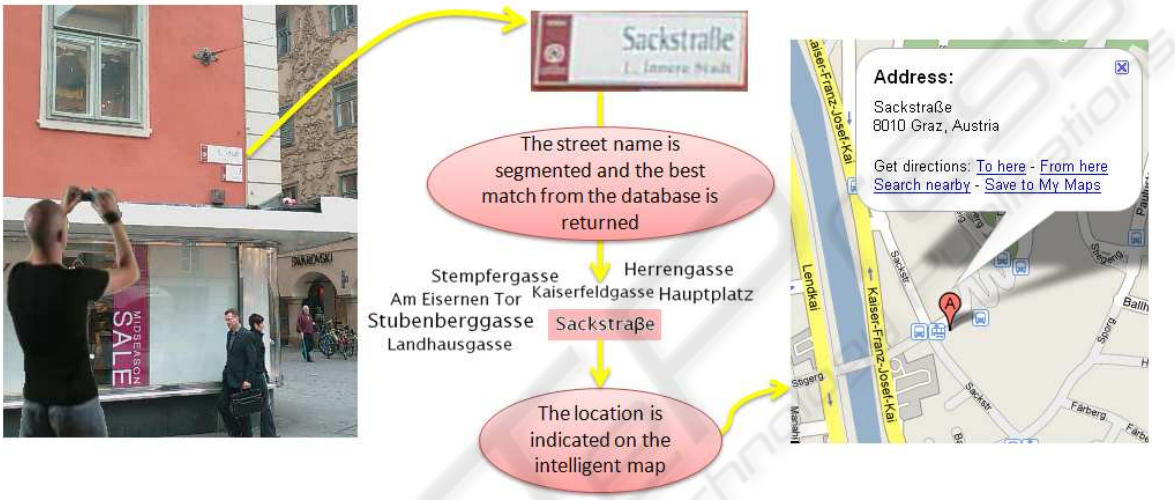


Figure 3: User takes an image from an urban place using his/her mobile phone and sends it to a server. Some target objects within the image are detected (the street plate) such that location of the person can be identified by recognizing some uniquely discriminant sign within the detected area of interest (the street name).

and Duygulu, 2007) and limited affine transformations of text (Ataer and Duygulu, 2007) (Ganapathi and Lourde, 2006). Hence, these methods still need to be combined to detection methods to be able to perform well in more general cases.

To the best of our knowledge there is no method that can provide an efficient solution to the three mentioned tasks simultaneously in the full generality and scale of our problem.

2 METHODOLOGY AND FRAMEWORK

2.1 Detection Phase

Adaboost (Freund and Schapire, 1995) has been proven to be fast and accurate for structural object detection problems. It is an algorithm for constructing a strong classifier as a linear combination of simple features, called *weak classifiers*. For general object detection *Haar features* have been used as weak classifiers(Papageorgiou et al., 1998). These features can be defined as the difference of the sum of pixels of areas inside windows, which can be at any position in the original image and have different scales, see Figure 4 for examples. Viola and Jones (Viola and Jones, 2001) introduced *integral images* as a fast method to calculate the difference of the sum of pixels.

To detect street plates in outdoor images we use adaboost in a similar way as in (Viola and Jones, 2001) where it was used for detecting faces. To train the adaboost we need positive and negative samples. As positive samples we consider rectangular regions of the image containing only a plate. To get them we annotate all the images in the database. Later we will use half of the images in the database for training the adaboost and the rest for test, but we need to annotate all images to evaluate how detected windows intersect true plate regions. The annotation consists of the coordinates of the four corners of the polygonal plate

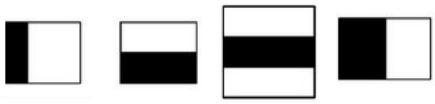
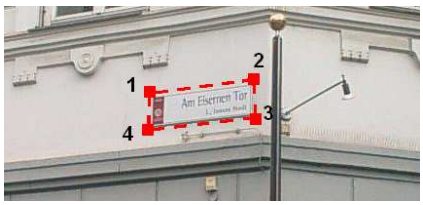


Figure 4: Representative of Haar feature samples.



(a)

(b)

Figure 5: Database annotation: (a) shows how images are annotated. (b) several samples of segmented plate regions from images with different views and scales. We use these segmented regions as positive samples for training.

region, as is shown in Figure 5.a.

As illustrated in figure 5.b many plate regions are not rectangular because of choice of viewpoints. Therefore, we apply a set of affine transformations to the whole image to approximately fit the plates to an axis parallel rectangular region. These transformations are different in the sense of scale and angle. We cover 5 different scales and 7 different angles to make our detector invariant to viewpoint and scale. Obviously, this may cause a loss of some part of the plate region border or bring in a few pixels from the background. Anyway we use these rectangular plate regions as positive samples in training the adaboost. To obtain negative samples a fixed window is slid over each image in the training database without overlapping. We exclude those windows that intersect plate regions. In our experiments we have used approximately 100 positive and 7000 negative samples for initial training of the adaboost. We normalize all samples by re-scaling them into a fixed size window 20×70 and also normalize the pixel intensities by converting the values to the interval [0, 1]. Now we can apply this adaboost on the test images to detect candidate plate regions in the entire image. Figure 6 shows an example of the initial adaboost result. As is illustrated in the figure, we indeed detect plate regions, but at the same time many non-plate regions are labeled as plates (false positives).

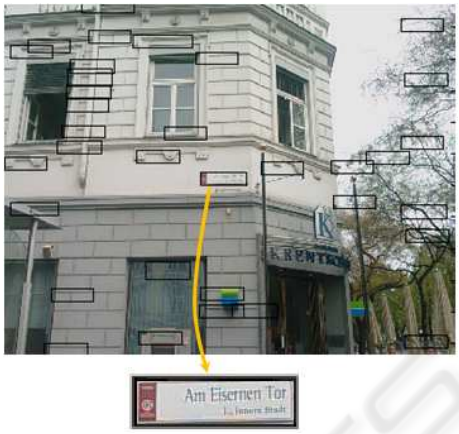


Figure 6: The initial boosting has lots of false positives. We use these detected windows to enrich our training database.

Improvement by Generating More Training Data. To decrease the number of false positives we increase the number of training samples. This idea is not new and has been applied for texture classification and object detection (Laptev, 2006) (Tavakoli et al., 2008). In previous work it is shown that the adding more training data by generating new images increases detection performance. Here to add new positive and negative samples to the training data, we apply the initial adaboost on the training images instead of the testing images. We then analyze the detected windows as follows. If they have more than 80% overlap with a plate region we add them to positive samples and and if they have less than 30% overlap they are added to the negative samples. The overlap is checked using the annotation information. The result of this sample generation procedure is adding 10000 negative and 500 positive samples to our training set. Therefore, the final boosting is trained with this enriched set of training samples. To try the final adaboost to each test image of the database, we first apply a set of affine transformations by resampling the entire images at varying scales and angles. Then we perform the adaboost on the set of transformed images. By applying these transformations the adaboost will detect plates at different slopes and scales.

Figure 7: Most of remaining false positives are quite similar to the plates and we can not expect adaboost to distinguish it.

Even after applying the final boosting, we still have some false positives. However, many of these false positives are structurally similar to the target objects and are not supposed to be removed in the cur-

rent boosting stage (Figure 7). In the next section you will see how these false positives can be removed by applying a filtering method.

2.1.1 Filtering False Positives

The remaining false positives have similar structure, so not surprisingly they also have similar Haar feature responses. Therefore, a new feature is required to discriminate the true positives that contain a text region from the false positives. Apart from shape features texture information reflects small scale structures such as in regions containing text. There is an abundance of texture descriptors in the literature, for example various linear filters, wavelets, co-occurrence matrices, energy measures from the Fourier transform, Markov random fields, local binary patterns, and texton histograms. Some of these could be used in our case. However, several of them, for instance the filters, respond to brightness edges which is not suitable in outdoor images. We applied also the fast computed descriptors based on LBP (Ojala et al., 2002) to our plate detection problem but it turned out that lots of false positives remained after filtering with that method. We therefore went for yet another method.

LU-transform and Filtering. In recent work Tavakoli Targhi et. al. proposed a fast and simple texture descriptor, called the Eigen-transform (Tavakoli et al., 2006). The texture descriptor is derived from image matrix decompositions. It has a number of properties which are desirable for bottom up processing in real-world applications. It captures small-scale structure in terms of roughness or smoothness of the image patch and unlike most other texture descriptors, it does not generate spurious responses around brightness edges. Also it is not sensitive to changes in brightness. It is fast to compute and provides a compact representation which is easy to store and perform calculations on. Finally almost no parameters need to be tuned.

The basic idea of Eigen-transfrom is to compute the singular values or eigenvalues of matrices representing the local neighborhoods of a pixel and form a descriptor as the average of the smallest of their absolute values. This yields a one-dimensional descriptor which fires in ”rough” areas of the image. The descriptor is computed for all pixels or on a sub-sampled regular grid. In (Tavakoli et al., 2006) it is shown that similar descriptors can be computed by any form of triangulation of the image patch matrix, such as by the LU-transform, which is rather fast. Mirmehdi and his co-workers (Merino and Mirmehdi, 2007) used this texture descriptor to track text regions in outdoor images. Their results demonstrated that the method efficiently can detect text in real time applications and motivated us to use it to filter the false-positive windows.

In filtering we use the LU-transform to detect textured areas in the candidate regions under the assumption that these contain text. Figure 8 illustrates the output from a plate region. We see that the text regions pop out from the background as desired. Hence, we apply the LU-transform on all the (positive and negative) samples of the training set. Then we train a new adaboost on the transformed samples. We use the same methodology as we used before to train adaboost on transformed windows, that is we extract Haar features from transformed samples instead of the original ones. We call this the adaboost-filter. We also add this step to our testing. More specifically, for every test input image we apply the adaboost to discriminate between true and false positives. To filter the false positives we then perform the LU-transform on each detected window and finally apply the adaboost-filter. As a result of the filtering step we have less than two false positives on average for each test image. As we will see below these false positives will in general automatically be filtered in the recognition phase.

Figure 8: Result of LU-transform on a plate region.

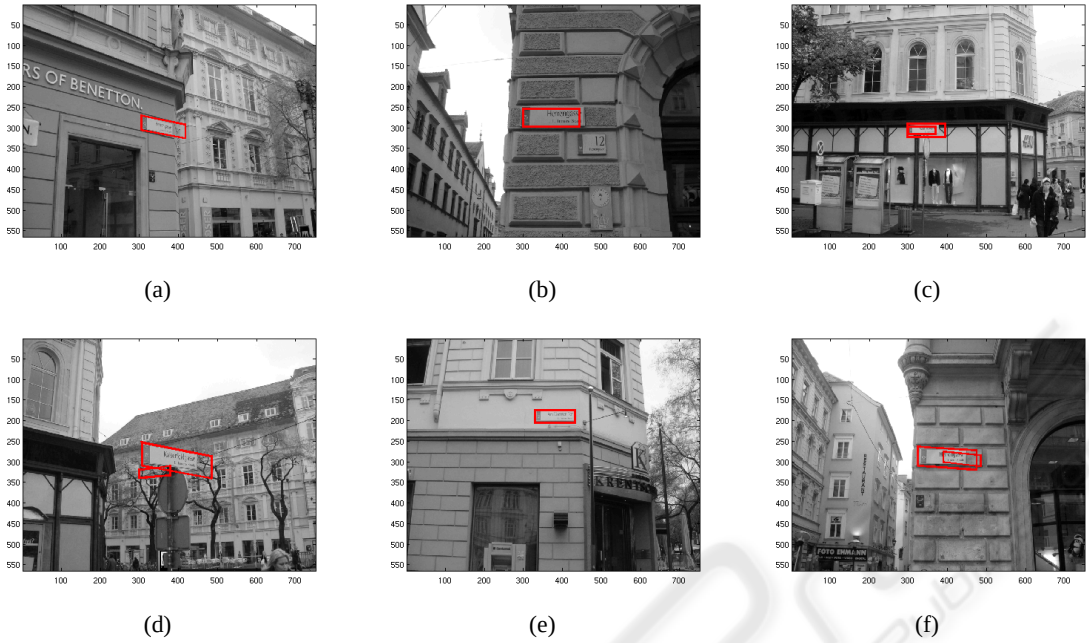
2.1.2 Final Decision

As we mentioned before, to make our method affine invariant, each input image is transformed by different affine functions like rotation and scale. Then we apply the detection separately on each of these transformed images. Therefore we have several detection windows of each plate. So we need to merge the overlapping detected windows which are close enough together. To do this, we use a threshold on the minimum of the four euclidian distances between corresponding corners of the windows. This threshold can be between 100 to 300 pixels.

To evaluate the detection accuracy we define a *confidence* value for each detected window. Number of windows merged together is considered as the confident of each merged region. Also the overlap of the detected windows is computed by (1).

$$overlap = \frac{S_{ground-truth} \bigcap S_{detected-window}}{S_{ground-truth} \bigcup S_{detected-window}} \quad (1)$$

Where $S_{ground-truth}$ is the area of the whole plate in the ground-truth and $S_{detected-window}$ is the area of the detected window. The overall detection rate is 92% with 40% overlap and confidence value 2. In



(a) (b) (c)

(d) (e) (f)

Figure 9: Some sample images after filtering. As you see, even after filtering, there may remain some false positives which are almost guaranteed to be removed in recognition phase.

average the detection phase returns two candidate region at the end, one of which is always the plate. The remaining false-positive will be rejected in the recognition phase since no text match it.

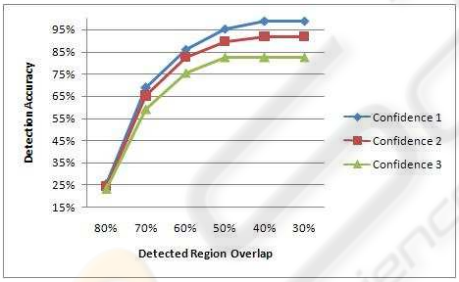


Figure 10: Detection rate for required overlap and confidence. Different curves show the detection results for different confidence values.

Figure 9 shows how the final detection procedure performs after making the final decision.

2.1.3 Text Segmentation

In the previous section we explained how we detect plates. For recognizing the street name we need to extract the text information from the plate region. This is not a straightforward task because candidate windows resulting from the detection phase don't fit exactly to the borders of the plates. One of the main reasons is perspective effects which cause detection windows to be wider (or narrower) than the actual plate when pictures are taken from different viewpoints. Therefore, there is a need to have a more accurate segmentation method to extract the text regions inside a roughly detected plate region. Our text segmentation strategy is based on a pixel-wise classification and segmentation method called *single histogram* (Schroff et al., 2006) which use histograms of *visual words* (or *textons*) as feature vectors. Histograms of visual words (Varma and Zisserman, 2003) have been effective in tasks such as image classification and object class recognition. The single-histogram approach represents each object class by a single histogram from visual words unlike the common way in which each object class is represented by a set of histograms. Classification is achieved by k-nearest neighbor search over the exemplars. This method perform simultaneous pixelwise segmentations and recognition of image regions. We here consider the single-histogram approach as a two class problem namely the text regions containing *street names* and the rest of plate region (*background* of the street name).To train the classifier we generate training data by manually segmenting all street names in the training images by drawing a bounding box around the street name region. Figure 11.a illustrates the manual segmentation.

Then we extract the feature vectors densely at each pixel location which here are raw 5×5 gray-

(a) (b) (c)

Figure 11: Segmentation scheme: (a). Original street plate with annotation bounding box, (b) probability map, (c) pixel-wise segmentation.

scale intensity patches. Thus, the dimensionality of the feature vectors is 25. During training, the vocabulary of visual words is built by clustering (performed by K-means) the feature vectors extracted from the training street name regions. We associate each pixel in the training images with the closest visual word (cluster center). Finally, we compute histograms of visual words for each of the training regions and combine them together to produce a single model for text regions corresponding to street names.

During testing an input plate region image is converted into its corresponding texton map by labeling each pixel. Then pixelwise classification is obtained by a sliding window technique. A window of dimension 9×9 is slid across the plate region to generate a histogram of visual words for each position in the plate region. The center pixel is then assigned a value which is a distance between the histogram at the pixel and single histogram of text region model. Since we here have only one object class, we build a probability map which represents the distance from each pixel in the plate region to the model. This is illustrated in Figure 11.b. To end up with an accurate and coherent text segment, we merge those pixels which have high probability values in the map. This is done by Connected Component (CC) extraction methods. We apply an algorithm that has been used in (Leon et al., 2005) for text detection where the Connected Component are reconstructed at multiple gray scales. We similarly define multiple thresholds and apply them on the result of the single-histogram classification. To remove the small text region which often contains some information about the city area, we simply use a threshold. An example of the final segmentation result after finding the Connected Component is given in Figure 11.c. We finally fit the segmented text region in a bounding box that will be used for recognition which will be explained in the next section.

2.2 Recognition Phase

The detection phase provides us with a set of candidate regions for which we know that they in over 98% of the cases contain a target plate with at least 40% of overlap (see Figure 10). Hence, we are given a number of cropped regions that likely contains text representing a street name. In order to recognize the text in such an image (region) one can either apply a character recognition approach or a direct matching technique. The latter approach of course requires that the texts that could appear in the images are known beforehand, which we indeed do in our case. Since our images of the plates often are noisy and low quality it is difficult to find individual characters in them. Therefore, we preferred to use image matching approaches. Such methods can be based on extracted features. In particular SIFT-features are considered to be powerful. We applied such techniques, but although we could achieve acceptable accuracy in that way the method turned out to be extremely sensitive to parameter selection, noise and other changes in the test images. As a consequence we instead exploited Dynamic Time Warping (DTW), (Sakoe and Chiba, 1990), which is a technique that has been used with success in speech recognition problems.
This method works by simultaneously modifying the 1D test signal by compressing and extending intervals on which it is defined and measuring its similarity to a reference model signal. The total matching cost is defined by combination of the cost of the interval modification with a measure of similarity between the obtained test signal and the reference.

DTW as used in speech recognition works on 1D signals, while here we have images. To adapt our problem to DTW, we extract vertically projected features from the image and consider them as a 1D signature along the text length (Shanker and Rajagopalan, 2007) (Rath and Manmatha, 2003). Experimentally we found that the sum of pixel intensities along the columns of the image were sufficiently informative. Upper or lower contours of each column can be useful as well and gave even more discriminative feature vectors. In fact there are even more sophisticated projected features such as HOG descriptors (Dalal and Triggs, 2005), but the three features mentioned in our experiments sufficed for acquiring perfect matching accuracy. Figure 12 shows the sum of intensities and upper contour features for a sample text image. It is worth noting that we also needed to perform certain normalization and preprocessing tasks, but we don't go into detail about these rather straightforward tasks.

The matching results in our application were generally very good. However, we often also have a rough estimate of our location in the map. For instance it can be obtained from the mobile phone. Based on (Ratti et al., 2007), there is one transceiver antenna every 100-300 meters in urban places, though this distance increases to several kilometers in rural areas. Therefore, the number of possible street names in practice is very limited - only a few tens. To investigate the matching performance we there-

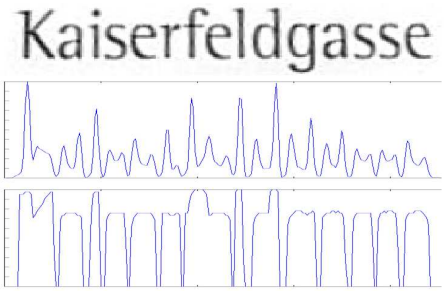


Figure 12: Topmost is a sample text image; in the middle, sum of intensity values are showed; and at the bottom you see upper contour of the text.

(a) Misclassified samples: *Hauptplatz*, *Murgasse*, *Sackstraβe* respectively

(b) High quality sample images: *Hans-Sachs-Gasse*, *Sporgasse* respectively

(c) Hard samples: *Am Eisernen Tor*, *Jungferngasse*, *Hauptplatz* respectively

Figure 13: For 1000 extra text models, as showed in Figure 14, recognition accuracy descends to 96.51% which means three false samples. (a) shows those three samples. (b) represents some high quality samples which can probably be recognized by OCR methods as well. (c) contains some hard samples that are recognized correctly by our method but are evidently very hard to be recognized by character recognition based methods.

fore also tested the approach on larger sets. It turned out that with up to a hundred real street names we could not observe any considerable decrease in accuracy. An experiment performed on one hundred real street names around Herrengasse street in Graz city resulted in three misclassifications. To provide an insight into the three incorrectly classified samples we depict them in Figure 13.a.

We note that the samples are of low quality and only with difficulty readable to a human. A simulated experiment performed on a dictionary of 5000 English words showed that the matching accuracy remained high also on such a large database, see Figure 14.

The figure shows that for 1000 extra models, the accuracy is 96.51% which means that only the three samples were matched incorrectly. Therefore, we conclude that the critical parts of the problem are found in the detection and text segmentation steps.
A strong point of this method is that it has no need

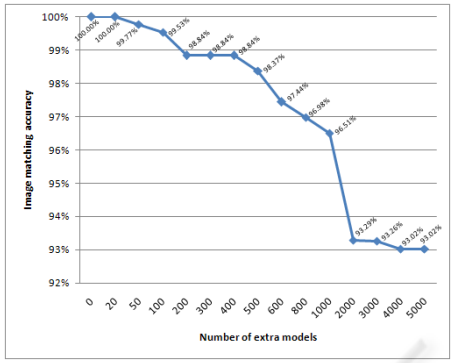


Figure 14: Text image matching accuracy over increasing number of reference models.

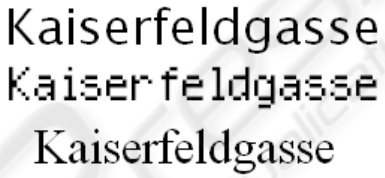


Figure 15: Automatically generated street name models. As we did not have the exact font used in our plates, different similar fonts were tested but it has no sensible effect on recognition results.

for reference models from real images. The only information used is a database containing a list of the street names in text format. By inputting this list to an image editor we artificially generate our text models, see Figure 15 for an example.

It turns out that the method to a great extent is invariant to the selection of the fonts used to generate text models. Furthermore, the method is extremely fast and needs only $m \times n \times d$ operations where m, n, d respectively stand for the length of test text, the length of reference text, and the number of projected features. In all this method has applications beyond the one studied here. It should be added that remaining false positives (if any) generally are rejected in the recognition stage. Again we want to stress that our experiments show that the recognition phase works almost perfectly if segmentation of the text areas is appropriate.

3 APPLICATION AND SYSTEM USAGE

The methods we utilized for both detection and recognition part are useful in different types of applications and can be used as a stand alone module in place recognition and urban area visit aid scenarios. For most visitors of a new city it is of value to be able

to find their path through streets and also find desired markets, hotels and restaurants around them. Google has developed a mobile friendly version of its map which can be downloaded for an increasing number of cities world wide. Installing a map on your mobile phone you would be able to track your paths. However, these maps need somehow well formatted input hints about the location to start the search to provide you with the required information.
For most cell phone users is not convenient to enter name of the target location with the keypad. This problem becomes much more sophisticated when you need to locate yourself by looking at the street plates nearby. Yet more complicated is the case where you are in a country with an unfamiliar alphabet like Chinese or Arabic. An effective way of doing this, is to take a photo of the plate and extract the information from the image as shown in this paper. Since we are using image matching for recognition part, our approach can be used for any plate or sign with different alphabet and characters.

Recently Google has released a software named *My Location*[3] which can estimate position of a cell phone user by just a simple packet of data transferred between the cell phone and connection service towers around (Figure 16). This estimate, though goes far rough (up to 5000 meters) in unsettled places, depends on concentration of cell phone towers around. In urban areas density of the towers is high enough to assure an accuracy while GPS information is not always available nor accurate, especially in high density aria. Integration of our plate detection and recognition system with available functions like *My Location* will result in a quite user friendly and applicable device. The overall processing time of our system is less than 10 seconds per image (independent of picture size).

Figure 16: As an estimate, Google draws a circle on the map of user's cell phone, highlighting where the user probably is.

4 CONCLUSIONS

In the paper we have presented a method for detecting and recognizing street name plates with applications to urban navigation and map indexing by mobile users, e.g. when the images are acquired by a mobile phone. We use a generic approach to structural object detection and recognition. A structured object is detected using a boosting approach and false positives are filtered using a specific method called the texture transform. In a second step the subregion containing the key information, here the text, is segmented out. Text is in this case characterized as texture and a texton based technique is applied. Finally the texts are recognized by using Dynamic Time Warping on signatures created from the identified regions. The recognition method is general and only requires text in some form, e.g. a list of printed words, but no image models of the plates for learning. Therefore, it can be shown to scale to rather large data sets. Moreover, due to its generality it can be applied to other problems, such as logo and sign recognition. The detection step relies on knowledge about the appearance of street signs. However, the boosting approach also can be applied to other cases as long as the target region is structured in some way.
In the experiments we obtain over 91% overall detection and recognition accuracy on 100 test images containing street plates from the city of Graz. The results show that the detection phase is the most crucial and also time consuming part. The overall speed has not yet been a major concern and presently the whole system requires slightly less than 10 second per image, of which segmentation and recognition takes less than one second. We used a desktop PC with 3.2 GHz CUP and 2 GB memory. The detection, as the first step, is of major importance and needs to be as accurate and robust as possible. So, continued efforts are on speeding-up this part without loss of accuracy. The recognition method we utilized turned out to be accurate enough, even if we have 1000 reference models. However in practice we can in our application limit this number to about 50 by estimating the location of the mobile device. This would give an overall recognition rate of almost 100 percent.

[3]www.google.com/gmm

ACKNOWLEDGEMENTS

The reported work has been performed within the EU-IST project MOBVIS, FP6-511051. This support is gratefully acknowledged.

REFERENCES

Adamek, T., OConnor, N., and Smeaton, A. (2007). Word matching using single closed contours for indexing handwritten historical documents. In *International Journal on Document Analysis and Recognition*.

Ataer, E. and Duygulu, P. (2007). Matching ottoman words: an image retrieval approach to historical document indexing. In *Proc. ACM international conference on Image and video retrieval*.

Dalal, N. and Triggs, B. (2005). Histograms of oriented gradients for human detection. In *Proc. Computer Vision and Pattern Recognition*.

Freund, Y. and Schapire, R. (1995). A decision-theoretic generalization of on-line learning and an application to boosting. In *Proc. European Conference on Computational Learning Theory*.

Ganapathi, T. and Lourde, R. (2006). Thresholding and character recognition from a digital raster image. In *Proc. International Confrence on System of Systems Engineering*.

Ishidera, E., Lucas, S., and Downton, A. (2002). Likelihood word image generation model for word recognition. In *Proc. International Conference on Pattern Recognition*.

Kim, K., Jung, K., and Kim, J. (2003). Texture-based approach for text detection in images using support vector machines and continuously adaptive mean shift algorithm. In *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

Laptev, I. (2006). Improvements of object detection using boosted histograms. In *Proc. British Machine Vision Conference*.

Leon, M., Mallo, S., and Gasull, A. (2005). A tree structured-based caption text detection approach. In *In Fifth IASTED VIIP*.

Merino, C. and Mirmehdi, M. (2007). A framework towards realtime detection and tracking of text. In *International Workshop on Camera-Based Document Analysis and Recognition*.

Ojala, T., Pietikinen, M., and Menp, T. (2002). Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. In *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

Papageorgiou, C., Oren, M., and Poggio, T. (1998). A general framework for object detection. In *Proc. International Conference on Computer Vision*.

Rath, T. and Manmatha, R. (2003). Word image matching using dynamic time warping. In *Proc. Computer Vision and Pattern Recognition confrence*.

Ratti, C., Sevtsuk, A., Huang, S., and Pailer, R. (2007). Mobile landscapes: Graz in real time. In *Location Based Services and TeleCartography*.

Sakoe, H. and Chiba, S. (1990). Dynamic programming algorithm optimization for spoken word recognition. In *Readings in speech recognition*.

Schroff, F., Criminisi, A., and Zisserman, A. (2006). Single-histogram class models for image segmentation. In *Computer Vision, Graphics and Image Processing*.

Shanker, A. and Rajagopalan, A. (2007). Off-line signature verification using dtw. In *Pattern Recognition Letters*.

Shapiro, V. and Gluhchev, G. (2004). Multinational license plate recognition system: Segmentation and classification. In *Proc. International Conference on Pattern Recognition*.

Shapiro, V., Gluhchev, G., and Dimov, D. (2006). Towards a multinational car license plate recognition system. In *Machine Vision and Applications*.

Tavakoli, A., Bjrkman, M., Hayman, E., and Eklundh, J. (2006). Real-time texture detection using the lu-transform. In *In Workshop on Computation Intensive Methods for Computer Vision*.

Tavakoli, A., GeuseBroek, J., and Zisserman, A. (2008). Texture classification with minimal training images. In *Proc. International Conference on Pattern Recognition*.

Varma, M. and Zisserman, A. (2003). Texture classification: Are filter banks necessary? In *Proc. Computer Vision and Pattern Recognition confrence*.

Viola, P. and Jones, M. (2001). Rapid object detection using a boosted cascade of simple features. In *Proc. Accepted Confrence on Computer Vision and Pattern Recognition*.

Yan, D., Hongqing, M., Jilin, L., and Langang, L. (2001). A high performance license plate recognition system based on the web technique. In *Proc. Intelligent Transportation Systems confrence*.

Ye, Q., Jiao, J., Huang, J., and Yu, H. (2007). Text detection and restoration in natural scene images. In *Journal of Visual Communication and Image Representation*.