# i-Street: Detection, Identification, Augmentation of Street Plates in a Touristic Mobile Application

Stefano Messelodi[1], Carla Maria Modena[1(✉)], Lorenzo Porzi[1,2], and Paul Chippendale[1]

[1] FBK-irst, Via Sommarive 18, I-38123 Povo, Trento, Italy
{messelod,modena,porzi,chippendale}@fbk.eu
[2] University of Perugia, Perugia, Italy

**Abstract.** Smartphone technology with embedded cameras, sensors, and powerful computational resources have made mobile Augmented Reality possible. In this paper, we present i-Street, an Android touristic application whose aim is to detect, identify and read the street plates in a video flow and then to estimate relative pose in order to accurately augment them with virtual overlays. The system was successfully tested in the historical centre of Grenoble (France), proving to be robust to outdoor illumination conditions and to device pose variance. The average identification rate in realistic laboratory tests was about 82%, remaining cases were rejected with no false positives.

**Keywords:** Augmented reality · Mobile devices · Text in scene images

## 1 Introduction

Augmented Reality (AR) is a concept that is already present in everyday life: using mobile devices with a connection to the Internet allows us to almost instantly gain access to the collective global knowledge, thus outsourcing memory [1]. In this report, we present a technology to visually augment information into the real world through portable devices, in a pervasive AR paradigm, where external available information can be presented in a *user* rather than a *device* centric way. In particular, we present a novel feature of an Android-based Smartphone Application [2] developed for a touristic scenario. The user visiting the historic centre of a city, can receive pertinent historical facts about the street in which (s)he is walking down, enriched with familiar directional indicators that highlight relevant and local points of interest.

Understanding user location cannot be completely solved solely through positional context *e.g.* GPS (Global Positioning System), as the urban canyon effect degrades reception and accuracy, particularly in historic centres, where streets are often narrow. According to experiments [3], GPS performs very badly in such environments with a median error of 6.7 mt. and only 62% of all GPS points


V. Murino and E. Puppo (Eds.): ICIAP 2015, Part II, LNCS 9280, pp. 194–204, 2015.
DOI: 10.1007/978-3-319-23234-8_19

falling within 10 mt of the real path. To improve location accuracy and thus inject information into the scene in the appropriate place, computer vision can help to refine position accuracy through the detection of geo-located visual pointers. In our case we will utilise street plates, which commonly lay at street intersections. We can reinforce or correct location errors by scanning for and subsequently reading the text of such plates and then by cross-referencing text strings with the known location of streets, gleaned from sources like OpenStreetMap. However, Computer vision alone cannot solve completely the problem, as multiple instances of the same plate often adorn intersections. For this reason, we also combined e-compass data from the mobile device to reduce ambiguities. Thus, by elaborating images coming from a camera, reading from the compass and GPS, and combining this with maps, it is possible to gain better self localization and orientation than pure sensor readings alone. We have integrated this strategy into a working demonstrator in the historic centre of Grenoble (France), where a user can point the device towards a street plate and obtain historical information about that street, whilst simultaneously correcting GPS location error estimates, thus improving the guidance of the user through the AR tour.

In this contribution, we present the i-Street application, developed as a part of the 3rd year demonstrator for the the European project VENTURI [4]. The intention was to demonstrate the possibility to integrate information coming from different sensors and then add visual augmentation to the video flow. The paper is organized as follow. In Sect. 2, we provide an overview of the system. In Sect. 3, we briefly explain the method adopted to localize, extract and read the text on a street plate. Section 4 reports a method to use recognized text and a-priori knowledge to identify a plate. Section 5 describes how virtual arrows are rendered into the real scene. Section 6 presents challenging situations and examples in which the system fails to provide a good result. In Sect. 7 results obtained in the laboratory and in the real demo environment are presented, and Sect. 8 concludes the paper.

## 2 i-Street Overview

The goal of i-Street is to fuse scenario knowledge, image analysis, and sensors data to identify street plates in a Smartphone's camera field of view and to supply augmented information to the user. In this scenario, computer vision algorithms have to cope with challenges such as harsh lighting conditions encountered in the outdoor environments, blurring of images due to motion of the device, uninteresting text presence, plate pose variation with respect to the device: Knowledge of the scenario and data coming from mobile sensors contribute to make the task of recognizing the correct street name feasible. Figure 1 depicts the main phases of the process along with the input role. The images coming from the phone camera are analyzed on the fly to detect candidate text strings that could potentially be present in the scene. Suspected text regions are segmented and filtered according to pre-defined formatting rules suggested by the real scenario. Each detected text line is de-skewed and its apparent deformation is corrected to facilitate the

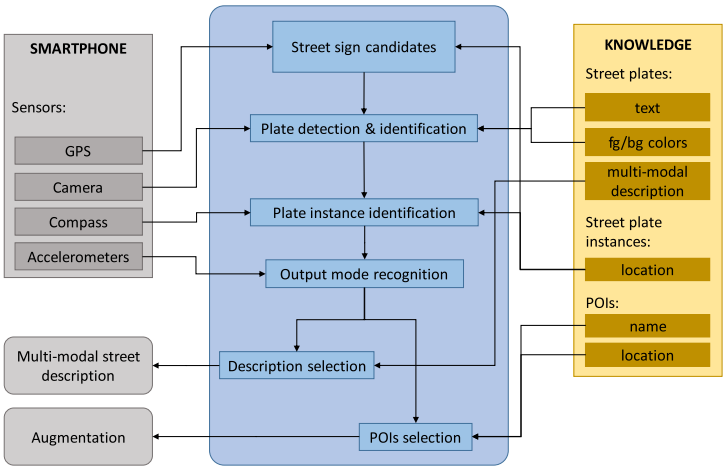


**Fig. 1.** i-Street architecture. The video frames are analysed taking into account specific domain knowledge and Smartphone's sensor data. Output is supplied to the user with different types of augmentation.

work of an Optical Character Recognition (OCR) module. The OCR outputs are compared to a dictionary of expected local street names (ranked according to the user's known location from GPS and a street map in the user's surroundings). A scoring function considers the spatial nearness of the lines and readability scores, thus enabling the system to determine the most probable street plate containing the read text. The live camera-stream is then augmented with information pertinent to the location. In particular, it is enriched with two virtual arrows that inform the user of some points of interest (POIs) in the right and left directions. These arrows are superimposed with the same apparent geometric deformation as the detected street plate to maintain realism. The i-Street application is articulated in the following main parts:

1. text detection and extraction from images and its recognition;
2. recognition of the plate and its location in the city;
3. augmentation of the image to supply information.

External knowledge (refer to Fig. 1) plays a fundamental role: the flexibility of the system to different scenarios is obtained by changing the content of the external knowledge. It helps the system to improve its robustness by reducing false plate detection and by driving the OCR. Furthermore it provides information for the augmentation task. Scenario knowledge is encoded inside a structured XML file that contains a list of the plates present in the city, and a list of points of interest, each one characterized by an unique identifier. The file stores, for each plate, the following data:

– code that describes roughly the pair (foreground, background) colours, *e.g.* white text on blue plate, or vice versa;

- printed text of each line that appears on the real plate;
- historical information about the location, possibly links to iconic pictures;
- number of plate instances (appearance of the same plate in the town). And for each instance:
  - approximate location (longitude, latitude);
  - identifiers of two particularly relevant POIs (optional).

Each POI is characterized by an identification number, its geo-coordinates (longitude, latitude), and its denomination to be shown on the virtual arrows. POIs can also be dynamically assigned, if none are prior associated. In this case, the two nearest (on the left and on the right) will be determined based on relative distance.
The on-board sensors providing input to the application are: *Camera* - frames to be elaborated; *GPS* - rough user location in the city map to restrict the set of street plate possibilities; *Compass* - estimate of the device direction for instance disambiguation and point of interest selection; *Accelerometer* - as a trigger to change the type of augmentation. A further operative requirement is the on-device storage of the language files needed for the OCR process. In this way, the system can operate completely off-line and in real-time.

## 3 Text in Scene Images

From a general point of view, text found in natural scenes can provide important contextual information. Several methods can be found in literature to cope with this problem [5,6]. We chose to use our previously proposed text detector [7,8], which provides good performance on low resolution images ($640 \times 480$), permits us to detect skewed lines, text that exhibits positive as well as negative contrast, is easily customizable using scenario information, and is computationally efficient, therefore suitable to be ported onto a mobile device.

Each frame coming from the phone's camera is processed in order to detect potential text lines in the scene. The algorithm outputs a list of binarized regions corresponding to candidate text lines. It works through the following steps:

**intensity normalization** - It is applied to compensate for light variations throughout the image. Normalization is achieved by the computation of the divisive local contrast of the intensity map. This operation improves image details and the local contrast in shadowed regions.

**double binarization** - For the detection of positive and reverse text, two thresholds are determined by taking into account the shape of the histogram of the intensity normalized map, computing the left and right deviation from the histogram's mode. The thresholds are then used to extract two binary maps that should contain, respectively, positive and negative contrasting elements.

**elementary objects** - The connected components of both bitmaps are analyzed by a cascade of filters to mark likely non-text components as non-interesting. The filtering criteria relate to area, elongation, convexity of the

connected components and delimitation value, *i.e.* the percentage of border pixels exhibiting a sufficiently high gradient in the colour image.

**aggregation into lines** - Survived elementary objects are recursively clustered according to proximity, alignment and size similarity. The alignment criterion consider the expected angular range from horizontal given by possible perspective distortions. Only clusters that satisfy certain characteristics are considered in order to discard possibly spurious lines. Lines whose aspect ratio (width/height of the minimum bounding rectangle) is below a threshold are discarded. The threshold is computed starting from the aspect ratio of the shortest string in the dictionary as typed in a sans-serif font. Furthermore, colours inside the line zone are tested, which should be concentrated on the pair (foreground, background) colours, provided by the knowledge.

**deformation correction** - Clusters that survive are geometrically normalized to facilitate the subsequent OCR process, by correcting the apparent deformation. The slant deformation due to perspective is corrected by searching the minimum trapezium around the text line region, whose parallel side corresponds approximately to the begin and the end of the string. The region is then stretched, using bilinear interpolation, in order to obtain a rectangular image of fixed height depicting a candidate text line.

Figure 2 provides an example of the candidate text lines extraction steps. By reducing the number of candidate lines, system processing time can be dramatically reduced as the OCR step is one of the most demanding in the whole chain. We utilise a free OCR engine released under the Apache License, called Tesseract [9,10], considered to be one of the most accurate open source OCR engines currently available. We apply Tesseract to each rectangular region with the single-line analysis option. The output is a string of characters accompanied by an index, in the range $[0-100]$, quantifying an average recognition confidence. We filter away lines with a low OCR confidence, as they are likely to be derived from noise. In Tab. 1, column one shows examples of images feed into Tesseract and column two the OCR output.

**Fig. 2.** Main steps of candidate text lines extraction: (from left to right) input frame, normalized contrast map, double binarization and filtering (retained components are shown in black), clustering into lines (blue, green, and red: three groups of components). Prudential criteria are used in filtering phases, in order to avoid the risk of discarding the text of interest.

## 4 Plate Identification

The recognition of plates is based on the OCR output, Knowledge, and GPS data. OCR output can be partial or fuzzy information: in fact, not all text lines are guaranteed to be detected, noise or out-of-plate text can also be captured and read, and text of interest can be read with errors. Strings obtained from the OCR step are therefore compared to the plate text dictionary using a string matching algorithm [11]. The dictionary is formed starting from the knowledge file, collecting expected text taking into account text/background colours. Only the OCRed lines that have a sufficient match with the dictionary are retained (Tab. 1, third column). Current geo-location, coming from the GPS, cross referenced to the city map with a tolerance radius can help to restrict the dictionary to local street names. The area has a radius of about 50 mt.

**Table 1.** Tesseract input/output and string matching with the known scenario words. The first line is not considered as a text plate string, as it doesn't match with any dictionary word, the two other are retained along with their matching score.

| Candidate text line after rectification (OCR input) | OCR output (confidence) | Dictionary comparison and matching score |
|---|---|---|
|  | `ECOIE el COHIGE BAYAIB` (74) | *no match* |
|  | `PLICE` (65) | PLACE - 80% |
|  | `du TEMPLE` (80) | du TEMPLE - 100% |

Text lines belonging to the same plate must exhibit the same contrast, similar height, be spatially near (with respect to their height) and placed one under the other. Using this knowledge, the detected lines are aggregated into potential plate regions. A scoring function based on spatial nearness of the candidate lines, contrast consistency, matching scores and frequency distribution of the words in the dictionary, allows us to determine the most probable street plate containing the read text. Figure 3, on the left, illustrates an example of plate identification. A string read as `PLACE`, though it exhibits a perfect match between the OCRed text and a dictionary word will give a small contribution to the plate score because the string is common to many plates, while the word 'TEMPLE' provides a greater contribution for the identification of the correct plate even when read with an OCR error, for example as `TEMPII`.

Note that the described elaboration refers to each single frame. The integration of results through subsequent frames, managed by a finite state machine, greatly improves plate identification, particularly in critical cases, avoiding hasty identification decisions.

## 5 Image Augmentation

The image that the user sees through the portable device's display is augmented with two AR arrows clamped to the real street plate. They direct the user to points of interest in the right and the left directions. To maintain realism, the arrows must be superimposed in a scene compliant manner. This is achieved through an estimation of the apparent dimension and geometric deformation of existing scene text.

The best recognized line of the plate is taken as a basis to determine the first arrow anchor point. Let $H$ be the height, in pixels, of the text in its middle point. The anchor is placed on the vertical axis through the middle point of the line at a distance that depends on $H$ and on its ordered position in the real plate. The interline space, as well as the space between the last line and the plate border, are considered to be similar to $H$. The height of the AR arrow is proportional to $H$, *e.g.* $2H$. The second arrow anchor point is vertically positioned under the first one, at a distance of $H/2$. The minimum enveloping trapezium of the best recognized plate line is used to roughly estimate a vanishing point for the arrow's slope and its perspective deformation. This method provides a nice render under the working assumption that device roll is not far from zero. Figure 3, on the right, illustrates schematically the arrows rendering inside a frame. The rendering of the arrows is continuously adapted to match the dimensions and slope of the text in subsequent frames, thus matching user movements.

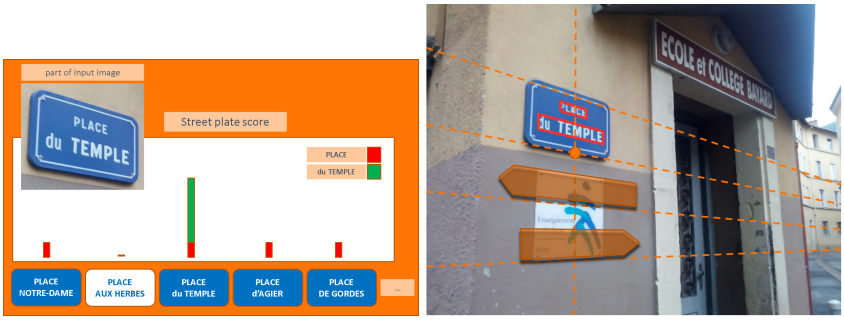


**Fig. 3.** On the left: example of plate scoring for plate identification. Each detected plate line contributes to assign a score to each possible plate. The contribution depends on the matching score (*e.g.* 80%) and the number of different plates containing the line (*e.g.* 'PLACE' appears in 4 plates - with the same contrast - while 'du TEMPLE' appears only in the plate at hand). On the right: A schematic representation to define the position of the AR-rows under the recognized plate and to estimate their deformation for their rendering inside the frame.

The text to be depicted on the virtual arrows comes from the Knowledge. Points of interest can be forced to the most interesting, depending on the opinion

of the file maker, or calculated on the fly as the nearest with respect to location and orientation. A walking time to reach the POI is also calculated, based on relative distances, and added as further information on the virtual arrows. Because occurrences of identical plates can be present on both sides of a street, and they are indistinguishable simply using GPS, a POI direction is assigned to match user orientation using e-compass data as a disambiguation criterion.

Other important information that can be given to the user is a textual description of the historical significance of street, presented in a user's spoken language. It is shown in an info box only when the device is orientated parallel to the ground plane. The phone pose is estimated from data coming from the device's accelerometer that acts as a trigger to switch the type of augmentation.

## 6 Requirements and Criticalities

As the system is mainly vision based, the user must be situated within a distance sweet spot from a street plate to obtain a robust detection, similar to those depicted in the frames shown in Fig. 4. Low resolution images result in poor detection, but an analysis of subsequent frames can provide correct output as distance or motion blur decreases, albeit with a low OCR confidence value. In general, frontal images perform better as the distortion-free nature results in an improved character appearance, while very angular views give rise to apparently touching characters. If the device roll is far from zero (more than ten degrees) the method adopted for the rectification of the detected text works poorly, therefore it often produces a low confidence OCR output; furthermore, in this case, the virtual arrows rendering may appear inconsistent.


**Fig. 4.** Examples of maximum distance of the street plate in the camera field of view.

Real world plates sometimes present intrinsic difficulties due to physical damage, defacement or dirt. Examples can be seen in Fig. 5. The first plate is damaged: the lack of blue enamel gives rise to white bullets, with the result that only the third line is correctly detected and segmented. In the second one, graffiti is present and the blue plate is degraded by rust. Text segmentation is not perfect,


**Fig. 5.** Examples of challenging cases for text segmentation and OCR interpretation. In the case on the left, only one line is detected. In the second example, the upper line is noisily segmented and OCRed as `HUE MVP`. In the case on the right, a poor characters segmentation leads to a poor OCR results (the upper line is read as `NUL`).

giving rise to OCR errors, particularly on the first line. The third example is a challenging case, mainly for text segmentation (and therefore for the OCR), as the character font exhibits extremely thin stroke.

## 7 Lab Tests and Real Usage

i-Street has been tested and tuned on a database of 134 high resolution pictures of 39 plates and 14 video clips collected in the historical centre of Grenoble. The photos were used as static input for the text detection algorithms during the development phases, before the porting and integration into an Android application. The videos were used to estimate the operative distance range.

For lab testing of the application, the pictures were displayed on the workstation screen and aimed at by the Smartphone that was moved to various distances and angles in front of the screen, whilst analyzing the correspondent output during the continuous elaboration of the video flow. The text lines detected and recognized with sufficient OCR confidence were emphasized with coloured outlines to check their positions. We observed that on and off false lines arose, but the clustering of those belonging to the street plate was correct. The plate recognition at frame level occasionally produces a false alarm. In these cases, the finite state machine, avoiding hasty identification decisions, improves the system performance at the cost of a small delay for new initialisation.

We performed various test sessions aiming at 134 city scenes using a low cost Smartphone (MIZ Z1, Android 4.2), considerably different from that used for development and final project demonstration (Sony Xperia Z, Android 4.4). The system proved to be very robust: between 109 and 112 correct identifications were obtained (average identification rate is 82%). In our tests no false alarm was produced (18% average rejection rate). A particular plate, characterized by

slightly bluish text on blue background, violating the white on blue assumption, proved to be particularly challenging. One third of the miss detections was occurrences of this plate. Occurrences of "Rue Brocherie" are occasionally not recognized because of the presence in the database of "Impasse Brocherie" too. Therefore, they have equal scores if the first word is not captured, causing a rejection. Finally, two other plates were often not recognized due to their narrow text font. We emphasize that in lab tests, GPS information was not utilizable, and the dictionary comprised all the words present on the 39 street plates. Tests revealed that some plates were recognised immediately, whilst others required more time (up to 2 seconds). This was due to detection uncertainty in one or more of the process phases, which forced the system to wait until successively high confidence detections were forthcoming.

The i-Street application was integrated into a multi-aspect AR demonstrator that featured in a live demo in the city of Grenoble in November 2014 providing excellent results (Fig. 6). In the final version, when the camera is pointed towards a street plate, the system detects and recognizes text on the plate, giving the user visual feedback, namely two AR brown arrows showing POIs and walking times, and an icon advising the user to turn the device parallel to the ground to see more detailed street information. Text to speech was also available.


**Fig. 6.** i-Street app working in a real scenario. Here the text localization has been emphasized with green outlines for demonstrative purposes only.

## 8 Conclusions

In this paper, we have presented i-Street, an Android touristic application. Its goal is to supply augmented information to the user by aligning real and virtual objects in a *user* rather than a *device* centric way. Computer vision algorithms enable the system by detecting and reading the plate text in the video flow and by estimating plate pose and size with respect to the device. Knowledge of the scenario and data coming from mobile sensors makes the task of recognizing the correct location feasible. The live camera-stream is then augmented with information pertinent to the location. In particular, it is enriched with two virtual arrows that appear clamped under the real street plate, informing the user of some points of interest in the neighbourhood.

The system, successfully tested in the historical centre of Grenoble, showed to be robust to outdoor illumination conditions and motion blur. Moreover, it can be easily extended to other scenarios.

**Acknowledgments.** This research was funded by the European 7th Framework Program, under grant VENTURI (FP7-288238).

## References

1. Huang, Z., Hui, P., Peylo, C., Chatzopoulos, D.: Mobile augmented reality survey: a bottom-up approach. Technical Report arXiv:1309.4413, HKUST, Hong Kong University of Science and Technology (2013)
2. API Guide. https://developer.android.com/guide
3. Schipperijn, J., Kerr, J., Duncan, S., Madsen, T., Demant Klinker, C., Troelsen, J.: Dynamic Accuracy of GPS Receivers for Use in Health Research: A Novel Method to Assess GPS Accuracy in Real-World Settings. Frontiers in Public Health 2(21) (2014)
4. VENTURI - ImmersiVe ENhancemenT of User-woRld Interactions: EC FP7-ICTProject. http://tev.fbk.eu/projects/venturi (2011–2014)
5. Jung, K., Kim, K.I., Jain, A.K.: Text Information Extraction in Images and Video: a Survey. Pattern Recognition **37**(5), 977–997 (2004)
6. Ye, Q., Doermann, D.: Text Detection and Recognition in Imagery: A Survey. IEEE Transaction on Pattern Analysis and Machine Intelligence (2015)
7. Messelodi, S., Modena, C.M.: Automatic Identification and Skew Estimation of Text Lines in Real Scene Images. Pattern Recognition **32**, 791–810 (1999)
8. Messelodi, S., Modena, C.M.: Scene Text Recognition and Tracking to Identify Athletes in Sport Videos. Multimedia Tools and Applications, Special Issue on Automated Information Extraction in Media Production **63**(2), 521–545 (2013)
9. Smith, R.: An Overview of the Tesseract OCR Engine. In 9th International Conference on Document Analysis and Recognition, Curitiba, Brazil, pp. 629–633 (2007)
10. Lee, D-S., Smith, R.: Improving Book OCR by Adaptive Language and Image Models. In: 10th IAPR International Workshop on Document Analysis Systems, pp. 115–119 (2012)
11. Myers, E.: An O(ND) Difference Algorithm and its Variations. Algorithmica **1**(2), 251–266 (1986)