Abstract
This paper presents the analysis of Google’s Tesseract OCR for license plate recognition in Brazil. The performance results presented for Tesseract OCR will be compared to market grade OCR products known here as “A” and “B”. This is a necessary measure due to a confidentiality agreement with the company supporting this research. The use of OpenCV is also considered due to limitations inherent to Tesseract OCR.

Key words: OCR, computer vision, automatic license plate recognition.

Introduction
In Brazil, the identification of any vehicle is foreseen to be done using the National System for Automatic Vehicle Identification (or SINAIV). This is a RFID based tag which is intended to be embedded in the national vehicle fleet by default in the next years (deadline by November 2011) (Mello, 2008).

While SINAIV is not available, there is some motivation for non-invasive and cheap solutions for monitoring vehicles in public highways. This paper describes one of these approaches, inserted in the context of the project Intelligent System for Vehicle Classification and Identification (or SICIV). Although SICIV is intended as an axis counting system for automation of toll collection in highways, a module for vehicle identification was devised.

This module called Module for the Identification of License Plates (or MIPV) was intended as a short-term contribution while SINAIV is not fully operational and is not intended as a replacement.

Different from SINAIV, MIPV uses OCR based technology to identify the vehicle license plate and would cross-reference it with the government vehicle registry. The communication module for integration with government databases is yet to be done by means of a partnership with a law enforcement agency. Therefore, this paper will focus on the OCR module, mainly exploring the possibility of using an open source technology as Google’s Tesseract OCR.

Module for the Identification of License Plates – MIPV
First of all we need to consider the scenario where MIPV is inserted. As of today, MIPV is not a fully operational system. It might be classified as a research effort to create a module to support law enforcement agencies in Brazil while SINAIV is not ready. It is incorporated inside a bigger project (SICIV) and we present here the results of the analysis that must be considered in the specification of a final prototype.

The implementation of a system for license plate’s recognition, must consider the regulations for license plate design in Brazil. According to the National Traffic Department, resolution n. 231/07 (CONTRAN, 2007), the license plates must be printed using the font “Mandatory” (Figure 1) and formatted according to Figure 2.

![Figure 1](image1.png)  Mandatory font specification (CONTRAN, 2007).

![Figure 2](image2.png)  License plate specification in mm for cars, trucks etc (CONTRAN, 2007). Motorcycle and the like will not be considered in this paper.
In Figure 2, the upper strip represents the vehicle’s place of origin and the pattern of 3 letters and 4 numbers represent the vehicle identification. No two vehicles present the same ID (disregarding illegal practices like license plate cloning).

MIPV will consider the described conditions to operate. Considering the partnership with a law enforcement agency, it was possible to devise MIPV according to Figure 3.

MIPV is composed by four subsystems:
- **MIPV-PRE.** It is the pre-processor sub-system. It is responsible for acquiring the vehicle images and the area of interest (i.e. the license plate).
- **MIPV-OCR.** Represents the OCR sub-system. This module will obtain the license plate number.
- **MIPV-POS.** It is the post-processor sub-system. It processes the output of the MIPV-OCR according to the intrinsic characteristics of the OCR module. Depending on the chosen system, the post-processor might operate differently.
- **MIPV-COM.** The communication sub-system for integration with governmental databases.

**MIPV-PRE**

Depending on the OCR technology, the detection of the area of interest might be a feature or not. Therefore, considering the specification of the MIPV system in its first stages, it is recommended to define a general-purpose approach to deal with the need for this characteristic.

Intel’s OpenCV library includes most of today’s computer vision algorithms. It has Windows and Linux versions, which accounts for portability across platforms. Some promising tests were done in order to determine the area of interest containing the license plate. To accomplish this, the fastest approach was to modify the “Square Detector”-demo example, which is shipped with the OpenCV library.

The modified version of this program returns a sequence of squares detected on a image. This sequence is stored in a specified memory storage. The target image is then down-scaled and up-scaled to filter out the noise. To do so, the functions cvPyrDown and cvPyrUp were used.

The function cvPyrDown performs downsampling step of Gaussian pyramid decomposition. First it convolves the source image with the specified filter and then downsamples the image by rejecting even rows and columns. The function cvPyrUp performs up-sampling step of Gaussian pyramid decomposition. First it upsamples the source image by injecting even zero rows and columns and then convolves the result with the specified filter multiplied by 4 for interpolation. Therefore, the destination image is four times larger than the source image (Spain, 2010).

Squares are identified in every colour plane of the image. To do so, several threshold levels are tested. In this context Canny is used instead of a zero threshold level. A user adjusted upper threshold is taken and the lower value is 0. This forces edges merging. The result is dilated to remove potential holes between edge segments.

Each contour is mapped, tested and approximated with accuracy proportional to the contour perimeter. Square contours should have 4 vertices after approximation to a relatively large area (to filter out noisy contours) and be convex.

---

**Figure 3** Schematics for the MIPV sub-system.
Here, an absolute value of an area is used because the area may be positive or negative – in accordance with the contour orientation. Then, we find the minimum angle between joint edges (maximum of cosine). If the cosines of all angles are small (all angles are ~90 degree) then quadrangle vertices to the resultant sequence are written. This procedure is made for all the contours of the analysed image and returned.

This algorithm extracts all the rectangular elements from the selected image and highlights them to the user. The result of this procedure is shown in Figure 4. Considering Figure 4a, 4b, 4c and 4d, it is possible to verify that the license plate is highlighted. The problem for now is that other rectangular areas are mistakenly detected as well. The creation of mechanisms to filter out the undesirable rectangular regions allowing the best use of the MIPV-OCR is one of the foreseen objectives for this project.

**MIPV-OCR**

The OCR module (MIPV-OCR) can be seen as an “engine” for the recognition of license plates. This role can be played by market grade applications like “A” and “B”, or based on open source software like Google’s Tesseract OCR. The remainder of this section presents the error analysis of these tools aiming at the verification of their potential as a character recognition module inside MIPV. The case study consisted in the analysis of images collected by the supporting company in 2007. The database is composed by approximately 2,300 images that present various adverse conditions to the identification of the license plate.

The following samples were chosen for a preliminary analysis of the OCR software:

- Data collected on 28/03/2007. The images were sampled in an internal environment of the supporting organization where license plates were simulated with A4 sheets in order to mimic the actual size. This sample contains 19 images and its objective is to prove the tool potential in a controlled environment;
- Data collected on 05/04/2007. There were devised tests in an open environment to verify the robustness of the character recognition software. The images consider the variance of brightness and potential reflexes generated by sunlight in the license plate. In this context, the detection was performed in a test vehicle, aiming at a more controlled environment. This sample contains 89 images.
- Data collected on 10/04/2007. There were inserted new variables, considering the recognition of other vehicles besides the test vehicle considered in the last sample. This was an important step considering the variation of plates regarding format and year of production. The conservation state was also an important factor. This sample contains 223 images.
- Data collected on 17/04/2007. This batch of images considers a 24 hour sampling in a partner of the supporting company. Therefore, it was impossible to keep the environment controlled and we could get closer to a real application environment. This scenario presents problems regarding the brightness of the environment, mainly at night and on some hours of the day, where the sunlight made impossible the plate recognition.

![Figure 4](image.png) Example of rectangular area detection using OpenCV.
Character precision

To perform a precision estimation, the method chosen was the Jack knife estimator (Rice et al., 1995). This method was adapted to analyse the present scenario where the data set considers only license plates. This test was applied at The Fourth Annual Test of OCR Accuracy and is a recognized form of measuring the performance of OCR systems.

According to Rice et al. (1995), there are other ways for measuring the deviation between the text generated by an OCR and the original text. The chosen approach measures a more fundamental measurement, which consisting on the effort performed by a human reviewer to correct the wrongly recognized characters supplied by the OCR.

Specifically, this will be computed by the minimum number of edit operations due to a wrong recognition in license plates to conform the result generated by the OCR. In Rice et al. (1995), this measurement refers to the number of errors performed by the OCR. In this paper, the measure expresses a percentage based on the total number of characters of each license plate, aiming at a character identification precision. This measure (hereby referenced as character precision) can be expressed as:

\[
\text{Character Precision} = \frac{\text{#characters} - \text{#errors}}{\text{#characters}}
\]

Confidence intervals will be calculated to measure the character precision. These intervals were computed using the statistical technique known as Jack Knife method (Cochran, 1977).

The jack knifing method is used in statistic inference to estimate the bias of the standard error in an statistical inference, when a random sample is used. The basic idea behind this estimator is to re-compute the statistic estimative leaving one sample out in every sub-sample created from the original sample. From this new subset, a statistical measurement for the bias and the variance is calculated.

The Jack knife estimator considers that a given simple random sample (SRS) \(X_1, X_2, \ldots, X_n\) chosen from a population for a given unknown \(\theta\) can be expressed as:

1. Step 1: determine the statistic to be re-computed \(\theta_{(i)}\).
2. Step 2: compute \(\theta_{(i)} = \frac{1}{n} \sum \theta_{(i)}\). This is the mean of the character precision recomputed.
3. Step 3: compute \(\sigma^2_{\text{Jack}} = \frac{(n - 1)^2}{n \cdot \text{var}\left(\theta_{(i)}\right)}\)
4. Step 4: The confidence interval is given by \(\theta_{(i)} \pm t^* \cdot \sigma_{\text{Jack}}\), where \(t^*\) is the index that maximizes the T-distribution with \((n-1)\) degrees of freedom.

Using this technique, we assume that the characters in a license plate are independent, but we don’t consider the characters in the set of plates independent. An OCR system that behaves consistently with the sample reflects a short confidence interval, while an OCR with a broader interval indicates a considerable variation. Comparing the performance of two systems, the non-overlapping of the intervals indicates the existence of a meaningful statistical difference between them.

For the following tests, it was used a character precision of 95% (0.95 to calculate \(t^*\)) for an OCR in a particular sample. Therefore, the character precision of the system will be inside this threshold.

**OCR “A”**

OCR “A” is a broad spectrum tool, which can be used to identify license plates of still vehicles or with reduced speed. The foreseen applications include the access control and registry of vehicles in parking lots, surveillance posts, highway toll collection cabins, borderline verification, etc.

According to the specification supplied by the vendor, OCR “A” achieves a typical rate of success of 85% to 95% and does have a component to enable the easy integration in multiple platforms. The tests were performed on SUSE Linux, although with some modifications, this software can be executed in any flavour of Linux.

After the analysis of the sample described at section “Character Precision”, the obtained results are as follows:

- Data collected on 28/03/2007. The analysis of the confidence interval reveals a range of 51% to 54%.
- Data collected on 05/04/2007. The analysis of this sample reveals a confidence interval in the range of 24% to 25%.
- Data collected on 10/04/2007. The confidence interval for this sample is in the range of 14%. Conditions regarding the conservation of the license plates, variations of brightness and complicated angles were some of the highlighted factors. Besides, it can be noted that this software does not deal with partial plates (i.e. plates with any kind of occlusion) and motorcycle like vehicles.
- Data collected on 17/04/2007. The confidence interval is in the range of 25% to 26%. In this scenario, partial license plates and motorcycle license plates were not correctly identified.

**OCR “B”**

OCR “B” is a Dynamic Linked Library (dll) for the development of automatic recognition systems based on digital image processing. It performs the automatic selection of the interest area (i.e. the portion of image containing the license plate), separates the characters and recognizes each one, returning their ASCII code. This turns OCR “B” into a good candidate for the recognition engine to be used on MIPV. The only drawback is that this software runs only on Windows platforms.

Regarding its performance, it was empirically verified (and statistically validated) that its performance is far better
than OCR “A” because it processed correctly images in adverse conditions of brightness and conservation state. These images, usually were not correctly identified by the OCR “A”.

After the analysis of the collected images, the results obtained revealed the following:

- Data collected on 28/03/2007. The analysis of this sample revealed a confidence interval in the range of 83% to 85%.
- Data collected on 05/04/2007. The analysis of these data presented a confidence interval of approximately 100%. Besides the excellent results, we must pinpoint that the variation regarding the analysed license plates was non-existent (only the test vehicle’s plate was analysed) and the positions were almost the same on all the population.
- Data collected on 10/04/2007. The confidence interval of the sample was in the range of 69% to 70%. This software deals with the partial license plate identification, although the in-depth analysis of this feature was not focused here. A careful analysis of this feature must be done in the future. This software also does not consider the identification of motorcycle license plates and the like.
- Data collected on 17/04/2007. The confidence interval is in the range of 75% to 76%.

**Google’s Tesseract OCR**

Google’s Tesseract OCR was developed at HP Labs in 1985 (Rice *et al.*, 1995). It was one of the top 3 OCR engines considering the precision analysis created by the University of Nevada, Las Vegas (UNLV). Between 1995 and 2006, there wasn’t meaningful advances, but it still is one of the most accurate (open source) OCR engines available today.

Tesseract OCR demands that the image is in 8-bit gray scale and gives as output a text file with the text detected. The input file is a non-compressed tiff file, or using the libtiff library, it is possible to read compressed images. In this case study, non-compressed images were used. The supported platforms are Linux (Ubuntu), Windows and Mac OS X (x86 and PPC), what makes this system strong candidate for the implementation of MIPV-OCR.

In Figure 5, there is an execution example of Tesseract OCR in a Debian 4.0 system. Considering the source code is available and the required libraries for its compilation are in the system, it is possible to compile and install the software in any Linux distribution.

Tesseract OCR was not envisioned for the current scenario of detection, so some guidelines have to be created:

1. The area of interest must be cropped manually and the resulting image converted to 8-bit gray scale (this procedure will be automated considering the use of MIPV-PRE).
2. Detections are considered only if the system identifies at least the clusters of characters foreseen for Brazilian license plates (3 letters and 4 numbers). Although this condition was bent a little in various cases, it is necessary to identify visually that the system is in fact parsing the correct area of the image, and not the upper strip of the plate.
3. Incorrect identifications are accounted as errors in the calculation of the character precision. Identifications resulting in 7 or more errors are considered as an incorrect license plate recognition (even if the identification was correctly done inside the flow of characters).
4. Shell scripts were created using tools like sed and awk to process automatically the output generated by Tesseract OCR. The pre-processing script tess.sh performs the recognition of the license plate using tesseract in all of the tiff images in a sample, generating as output a text file. This file is processed using sed in order to filter non-printable and special characters different from letters (uppercase) and numbers. For each tiff file, is generated an (empty) output file of the form “900-1-xxxxxx.tif-TESS-DKB937__log”, where “900-1-xxxxxx.tif” represents the name of the tiff file, “TESS-DKB937__” tesseract’s output. The tag “TESS” is used solely to simplify the visual identification of the recognition process, which in this example is “DKB937__”. The characters “__” represent the blank spaces detected. The post-processing script print.sh parses the generated files and automatically extracts the license plate ID. In the example above, the pattern extracted is “DKB937__”. This simplifies the insertion of the data in a spreadsheet software to perform the confidence interval calculations using the Jack knife estimator.

![Figure 5](image-url) Example of detection with the test file eurotext.tif (a) which is included in the tesseract-2.03.tar.gz package and the result supplied (b).
Considering the above-mentioned conventions, results were obtained for Tesseract OCR versions 1.04b, 2.00, 2.01 e 2.03, considering the samples used with OCR “A” and OCR “B”.

The results obtained were:
- Data collected on 28/03/2007. The analysis of the sample revealed a confidence interval of a 100% (for all versions of Tesseract OCR). Considering Tesseract is not an OCR adapted to identify license plates in a pre-defined format, it also identified partial license plates in this simulation (Figure 6).

![Figure 6](image1.png)

**Figure 6** Partial license plate correctly identified as “DKB937__” in all versions of Tesseract OCR.

- Data collected on 05/04/2007. The analysis of this sample revealed a confidence interval of 29% a 30% for version 1.04b, 10% to 11% for version 2.00, 10% to 11% for version 2.01 and 12% to 13% for version 2.03. A curious aspect is that the recent versions of Tesseract OCR present a worse identification rate when compared to the older version (1.04b) for this scenario. An identification result is presented on Figure 7 for this sample.

![Figure 7](image2.png)

**Figure 7** License plate identified as “JFX9Q55__” by version 1.04b, “EJ5B__” for version 2.00, “EJ5B__” for version 2.01 and “EJ5H__” for version 2.03.

- Data collected on 10/04/2007. The confidence interval for this sample is about 6% for version 1.04b, 2% to 3% for version 2.00, 2% for version 2.01 and 2% for version 2.03. Observe on Figure 8 that although the recognition algorithm identifies the characters almost correctly, the character precision is low considering the insertion of non-existent elements in the identification. In this case, the character precision is calculated as (7-1*-4**)/7. The value marked with (*) is obtained from the non-identification of the numerical 6 in the license plate. The value marked with (**) is obtained by removing all the non-existent and non-necessary characters incorrectly detected (“LIY” and “A”). In a scenario like this, the challenge is to apply methods, which allow the filtering of the extra “noise” in the ID, what makes the correct detection of the license plate almost impossible.

![Figure 8](image3.png)

**Figure 8** License plate identified as “FHF_DXB68634__” by version 1.04b, “LIY_DXB863A__” for version 2.00, “LIY_DXB863A__” for version 2.01 and “LIY_DXB863A__” for version 2.03.

- Data collected on 17/04/2007. The confidence interval for this sample is about 5% for version 1.04b, 2% for version 2.00, 2% for version 2.01 and 2% for version 2.03. Observe a situation in Figure 9 where the algorithm used on version 1.04b presents a performance far superior to the recent versions of Tesseract OCR.

![Figure 9](image4.png)

**Figure 9** License plate identified as “_AKN62444__” by version 1.04b, “_” for version 2.00, “_” for version 2.01 and “_” for version 2.03.

**Comparative Results Revisited**

The confidence interval of the tools analysed in this section where gathered on the chart presented in Figure 10 (a to d).

Considering the market grade tools “A” and “B”, the accuracy method used points to the ineffectiveness of OCR “A” in a consistent manner throughout the case study. OCR “B” had the most solid results in all of the case studies, indoors and outdoors. Tesseract OCR presented the best results in the simulated environment (i.e. the simulation using paper sheets) and the worst result in the outdoor scenarios.

The discrepancies regarding the confidence interval in the simulation and in the outdoor scenario on Tesseract’s results may suggest a need for improvement on the infrastructure used to collect the original images.

Considering the limitations regarding the data acquisition, the quality of the images can treated by the use of a more intense post-processing mechanism (software based) in order to improve the detection rate. The use of OpenCV is seen as a good alternative considering the simple interface with the programmer. We already tested OpenCV to deal with the delimitation of the area of interest and it proved itself as an interesting solution, although we have to improve the algorithm.

Other alternative to improve the detection of Tesseract OCR would be the use of high resolution cameras with automatic adjust for brightness, contrast and hue (hardware based). We intend to redo the tests described in this article in the near future to verify the performance of Tesseract OCR considering the proposals described here.
Conclusion
This paper intended to present an open source solution in the context of the SICIV project. To do so, Tesseract OCR was analysed and compared to other market grade tools. The conclusion we could devise considering the analysis presented is that Tesseract OCR has the potential to be applied in this scenario. This can be seen considering the results for the simulated license plates (Figure 6). The main difference of this scenario and the real one was the quality of the images and the controlled characteristic of the environment.

Acknowledgements
This work acknowledges the invaluable importance of the FAPESP sponsorship and VisionBR$^4$ support.

The authors would like to thank and acknowledge the technical support given by José Luís Segatto Júnior, Valdinei Luís Belini and João Guilherme França.

Notes
2. This partnership is currently being analyzed.
3. Available at http://sourceforge.net/projects/opencvlibrary/.

References