Improved Number Plate Localisation Algorithm and its Efficient FPGA Implementation

Xiaojun Zhai, Faycal Bensaali and Soodamani Ramalingam
School of Engineering & Technology
University of Hertfordshire
Hatfield, UK
{x.zhai, f.bensaali, s.ramalingam}@herts.ac.uk

Abstract—Number plate localisation is a very important stage in an Automatic Number Plate Recognition (ANPR) system and is computationally intensive. This paper presents a low complexity with high detection rate number plate localisation algorithm based on morphological operations together with an efficient multiplierless architecture based on that algorithm. The proposed architecture has been successfully implemented and tested using a Mentor Graphics RC240 FPGA (Field Programmable Gate Arrays) development board equipped with a 4M-gate Xilinx Virtex-4 LX40. Two database sets sourced from the UK and Greece and including 1000 and 307 images respectively, both with a resolution of 640×480, have been used for testing. Results achieved have shown that the proposed system can process an image in 4.7 *ms* whilst achieving a 97.8% detection rate and consuming only 33% of the available area of the FPGA.

1. Introduction

Automatic Number Plate Recognition (ANPR) systems allow users to track, identify and monitor moving vehicles by automatically extracting their number plates. These systems are rapidly becoming used for a vast number of applications including, automatic congestion charge systems, access control, tracing of stolen cars, and identification of dangerous drivers [1]. The fundamental requirements of an ANPR system are image capture using an ANPR camera and processing of the captured image. The image processing part, which is a computationally intensive task, includes three stages: Number Plate Localisation (NPL), Character Segmentation, and Optical Character Recognition (OCR). NPL is the stage where the Number Plate (NP) is localised in the input image from the ANPR camera. The character segmentation stage is an important pre-processing step before applying OCR, where each character from the detected NP is segmented before recognition so that only useful information is

retained for recognition. In the last stage, optical character information will be converted into encoded text by pre-defined transformation models.

ANPR is a computationally intensive task and often has to be under real-time constraint, the common hardware choice for its implementation is often high performance workstations. However, the cost, compactness and power issues that come with these solutions motivate the search for other platforms. Recent improvements in low-power high-performance Field Programmable Gate Arrays (FPGAs) and Digital Signal Processors (DSPs) for image processing have motivated researchers to consider them as a low cost solution for accelerating such computationally intensive tasks [2]. Current ANPR systems generally use a separate camera and a stand-alone computer for processing. By optimising the ANPR algorithms to take specific advantage of technical features and innovations available within new FPGAs, such as low power consumption, development time, and vast on-chip resources, it will be possible to replace the 3GHz roadside computers with small in-camera dedicated platforms. In spite of this, costs associated with the computational resources required for complex algorithms together with limited memory have hindered the development of embedded vision platforms [2].

This paper presents a speed and area-efficient architecture based on a low complexity NPL algorithm suitable for FPGA implementation. The proposed algorithm is mainly based on open and close morphological operations. A MATLAB implementation of the proposed algorithm was used as a proof of concept prior to the hardware implementation, and the proposed architecture implemented and verified using the Mentor Graphics RC240 FPGA development board equipped with a 4M Gates Xilinx Virtex-4 LX40. For comparison purposes two different databases, including a public one, were used. The first one contained 1000 images with UK number plates while the second one , taken from an online database, contained 307 images with Greek number plates with a resolution of 640×480[3]. For the UK database, some images were collected by the authors with the rest provided by CitySync Ltd. [4] who are one of the leading UK providers of ANPR solutions. The images were grouped into six different sets based on different criteria such as distance and illumination conditions. These will be discussed later.

The remainder of this paper is organised as follows: related work is reviewed in Section 2 while section 3 describes the morphological based algorithm. The proposed NPL architecture is then described in Section 4, the MATLAB and analysis of the experimental results are in Section 5 while Section 6 is concerned with FPGA implementations and discussion of the experimental results. Section 7 then concludes the paper.

2. Related Work

To date, most ANPR research has been software based, the commonly used platform for this purpose often being high performance computer workstations located at the roadside connected to an ANPR camera. Recently researchers have started to consider the use of compact embedded hardware devices to replace these expensive computer workstations that can also be placed within an ANPR camera housing. However the issue of the necessary high levels of performance requirement set against limited memory resources associated with embedded hardware devices still needs to be addressed [2].

Table 1 summarises the most recent software based ANPR systems. Generally, NPL algorithms reported in previous research are mainly classified into three classes: edge detection based algorithms [5], [6], colour-based algorithms [7], [8] and texture-based algorithms [9], [10], [11]. The algorithms based upon combinations of edge statics and mathematical morphology showed good results. These methods try to find a rectangle shaped liked region from a vertical edge density map. A disadvantage of this method is that it cannot deal with the complex images, since they are too sensitive to unwanted edges. In spite of this, after combining with morphological operations that eliminate some unwanted edge information, the NPL rate is relatively high and fast [12]. Colour-based algorithms for NPL use prior knowledge of NP (e.g. plate background and text colour) and for this reason are country specific. Image transformations (e.g. Gabor filter and wavelet transform) are major tools for texture analysis – and are normally computationally expensive and slow for images with large resolution.

Table 1. Existing Software Based ANPR Systems

ANPR System	Character Set	System Part	Image Type	NPL Rate (%)	NPL Speed (ms)
[6]	China	NPL	Greyscale	99.6	100
[7]	Taiwan	Whole ANPR system	Colour	97.9	N/A
[8]	China	NPL	Colour	95.1	400
[10]	Korea	NPL	Colour	92.7	1280
[11]	Taiwan	NPL	Greyscale	97.3	180

Recent improvements in the computing power of FPGAs and DSPs have motivated researchers to consider them as an alternative solution to implement ANPR systems. These devices can be used as a low-cost System-on-chip solution that allows the FPGA or DSP-based processing unit to be placed within an ANPR camera housing to create 'intelligent cameras'– namely cameras that record and process images for sending back to a server. Table 2 summarises the most recent FPGA and DSP based ANPR systems. A variety of algorithms have been used is these systems. This includes AdaBoost, Support Vector Machine (SVM) [2], [13], Gabor filter [14], morphological operation [15] and background modelling and pixels classification [16]. These algorithms are either computationally expensive (i.e. high execution time) or they have a low detection rate. The only work that involves the use of only one FPGA to implement the first stage of an ANPR system is the work presented in [16]. Results show that it has the fastest processing speed to locate NP with a relatively low detection rate compared to other existing work.

Table 2 Existing FPGA and DSP-based ANPR systems

ANPR System	Character Set	System Part	Image Type	Hardware Platform	NPL Rate (%)	NPL Speed (ms)
[3]	Australia	NPL	Colour	TI C6414 DSP and Altera FPGA	96	141.62
[13]	Australia	Whole ANPR system	Greyscale	TI C64 DSP and PC	N/A	<50
[14]	Turkey	Whole ANPR system	Greyscale	FPGA Virtex IV and Video board	91.70	N/A
[15]	US	Whole ANPR system	Greyscale	FPGA Virtex II Pro and PC	N/A	N/A
[16]	Japan	NPL	Greyscale	FPGA Virtex II pro	87	9.25

Generally, the software based ANPR systems have a higher detection rate compared to hardware based systems, however the processing time of the former is higher than the latter. The design and implementation of hardware based ANPR systems is limited by the hardware architecture and the available resources. Efficient methods and techniques to implement ANPR algorithms should be considered to map them on the chosen hardware.

The work presented in this paper focuses on developing a low complexity and relatively stable NPL algorithm suitable for a single FPGA implementation. The innovations introduced in this paper can be summarised as follows: 1) a novel NP feature extraction and enhancing method based on two morphological *open* operations and an image subtraction operation. 2) a novel efficient FPGA architecture and its implementation.

3. Number Plate Localisation Algorithm

A NP image is normally recorded as a pattern with high variations of contrast. This feature is used to locate the plate and have been found to be relatively robust to changes in lighting conditions and view orientation. Most of the previous work based on morphological operations have used edge detection to extract the edge information around the NP region followed by morphological operations as a fusion tool to connect the pixels together in that region. After that a Connected Component Analysis (CCA) labelling algorithm is used for the NP region selection. However, the edge detectors are based on matrix multiplication and the entire image needs to be scanned, which increases the computing cost of the algorithm. Therefore, in this paper, we are using a morphological *open* operation and image subtraction to replace the edge detection operator, which reduces the computation complexity whilst maintaining a satisfactory detection rate.

The proposed algorithm is mainly based on two *open* and a *close* morphological operations, the first *open* morphological operation is used to extract the features of the NP, the second *open* operation is used to remove noise, the *close* operation is then used to fuse the pixels in the NP region together.

The proposed algorithm consists of two major stages:

1. Morphological operations for extracting plate features;
2. Selection of candidate regions.

Fig. 1 shows a block diagram of the proposed NPL system.

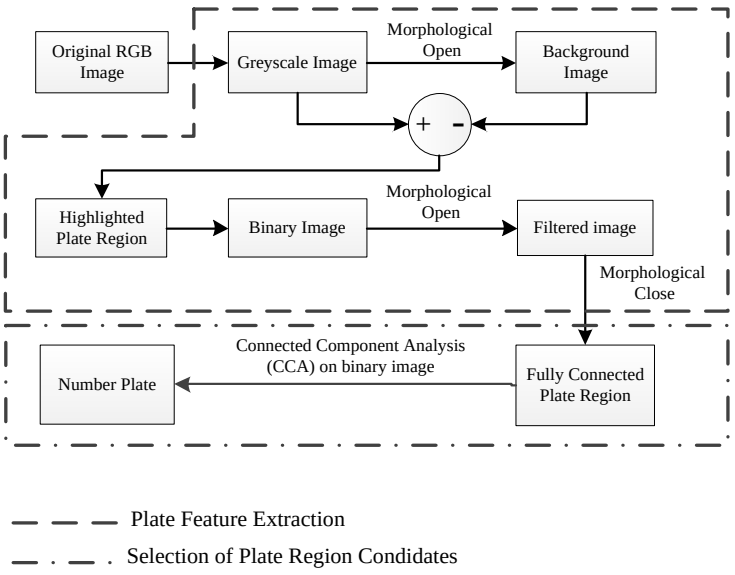


Fig. 1. Proposed NPL system.

3.1 Plate Feature Extraction

The proposed algorithm mainly utilises three morphological operations to minimise the pixels of the non-plate region and to enhance those of the plate region. The original RGB image is first converted into a greyscale image, which will be used as an input to the following block where the first morphological *open* operation will be used.

The morphological *open* operation is an *erosion* followed by a *dilation* and the opposite operation (i.e. *close* operation) is a *dilation* followed by an *erosion*. The shape of the morphological operations is based on a suitable structuring shape employed as a probe called the Structuring Element (SE) [17]. *Open* I_O and *close* I_C operations can be performed as shown in (1) and (2) respectively where I denotes a greyscale input image, $\oplus$ denotes a *dilation* operation and $\odot$ denotes an *erosion* operation:

$$I_O=(I \odot \mathrm{SE}) \oplus \mathrm{SE} \tag{1}$$

$$I_C=(I \oplus \mathrm{SE}) \odot \mathrm{SE} \tag{2}$$

When applying the morphological *open* operation on a greyscale image, pixels will be 'averaged' in the area of SE. When applying it on a binary image, pixels will be erased if the SE area is not fully filled by pixels with value '1'.

UK NPs normally consists of black characters on a white background. This feature causes the pixel values to be highly variant in the NP region. On the contrary, the margin area of the NP region normally consists of a constant colour, in particular the

windscreen glass and engine hood. Therefore, if applying a morphological *open* operation with enough large SE on greyscale car image, characters can then be removed from the NP region while the remaining features of the rest of image will be kept. By performing a subtraction between the original greyscale image and the resulting image after the *open* operation image, the output will be a highlighted plate region image.

The size of SE is decided based on the gap between two neighbouring characters on the NP. Due to the variant distances between the car and the camera, the size range of NPs ($a \times b$) in the used databases is between 18×160 (pixels) and 60×300 (pixels). Let $d_{h\max}$ (pixels) and $d_{v\min}$ (pixels) denote the maximum distance between the two neighbouring characters on the horizontal and the minimum distance between the character and boundary of the NP respectively. They both depend on the size of the NP and the shape of neighbouring characters (See Fig. 2).

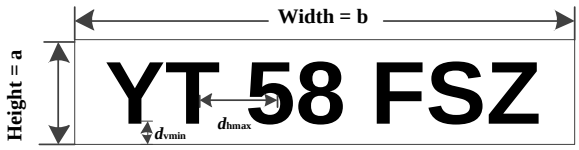


Fig. 2. An NP example.

On the other hand, the size of SE ($S_1 \times S_2$) is determined by $d_{h\max}$ and $d_{v\min}$, where $1 \le S_1 \le d_{v\min}$, $d_{h\max} \le S_2 \le d_{h\max} + \Delta$, Δ as a variable that is calculated based on experiment results.

For the UK database, images were randomly taken from different real-world environments with variant NP sizes (note that details of this database are given in Section 5). Based on the above description on how the SE size is selected and tests performed using the UK database and then validated with the on-line public Greek database, the size of the used SE for the *open* operation was set to 3×30. Fig. 3 shows the used 'rectangle' shaped SE.

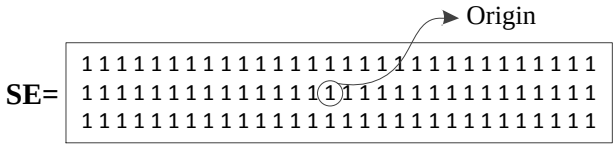


Fig. 3. A 'rectangle' shaped SE with size = 3×30.

This 3×30 SE has an origin pixel point, which is the centre of the whole SE. The origin point is mainly used for marking the SE's location when the morphological operation is performed.

A morphological *open* operation with this 3×30 SE is performed on an original greyscale image, which will generate a background image (Non-NP region). The background image is subtracted from the original greyscale image and the result of this operation is a highlighted NP region. Fig. 4 illustrates this process.

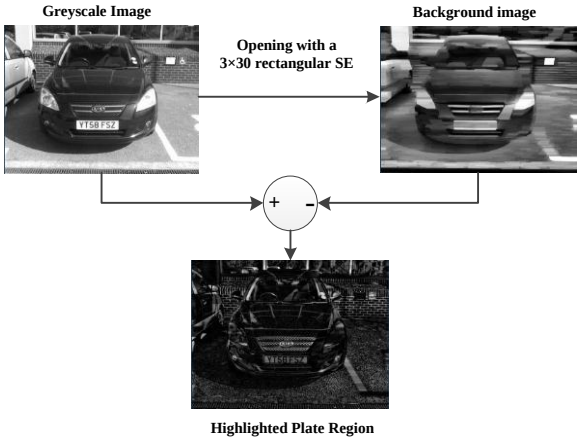


Fig. 4. The process for highlighting the plate region.

In order to further eliminate Non-NP regions, the highlighted plate region image is binarised. Let $g_{\min}$ and $g_{\max}$ denote the minimum and maximum pixel value of the highlighted NP region in the database respectively, the best threshold T_b should adaptively change from $g_{\min}$ to $g_{\max}$ when different images are applied, however, this process requires extra memory to store the entire image, before analysing the best T_b for each input image. For FPGA implementation, this slows down the processing speed and increases hardware usage. Therefore, the proposed algorithm uses a fixed threshold T_f to replace T_b. if $T_f \le g_{\min}$, all the highlighted NP regions should be kept after image binarisation. For the used databases $g_{\min} \ge 60$, therefore the value of the fixed threshold T_f is 60. Although the fixed threshold can benefit hardware implementation, a lower threshold value will increase the noise level. To overcome this problem, an extra morphological *open* operation is used to remove the noise. Fig. 5(b) shows the result after noise removal.

Fig. 5. The process of image binarisation and enhancement.

For the process shown in Fig. 5 ‘diamond’ and ‘rectangle’ shaped SEs are used for the last morphological *open* and close operations. The two SEs are shown in Fig. 6.

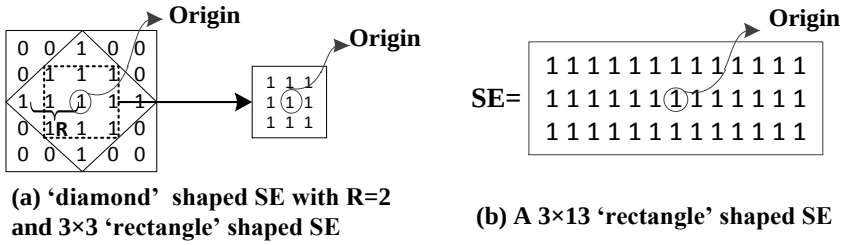


Fig. 6. The ‘diamond’ shaped and ‘rectangle’ shaped SEs.

Fig. 6 (a) shows a ‘diamond’ shaped SE, where the matrix has a radius R=2 and all 1’s are inside the ‘diamond’. When this special structure is used during the *open* operation on a binary image only diamond-shaped regions filled by 1s will be kept. This operation is very useful in erasing net-shaped and narrow lines surrounding the plate area. This SE can efficiently erase most of the unwanted information, as can be seen in Fig. 5 (b). However, in order to reduce hardware usage, the ‘diamond’ shaped SE has been replaced by a 3×3 ‘rectangle’ shaped SE for hardware implementation. As can be seen from Fig. 6 (a) the difference between the ‘diamond’ shaped SE and the ‘rectangle’ one is that the first has an extra four corners. Although the *open* operation can effectively remove noise, some pixels in the NP region can also be eliminated. Therefore, the system needs an extra operation to fully fill the plate region to connect the pixels. A morphological *close* operation is used for this purpose. Fig. 6 (b) shows a ‘rectangle’ shaped SE for this *close* operation, where the matrix has 3×13 ‘rectangle’ shaped 1s. Any non 1 pixels in this rectangle region will be changed to 1, which means all the parts in this region will be fused together. As can be seen from Fig. 5(c), the plate region can clearly be identified as it is a group of connected pixels which can be easily extracted using some known geometrical conditions (e.g. Width / Height ratio).

3.2 Selection of Candidates Plate Region

The output image from the previous stage consists of a set of groups of connected pixels. A labelling algorithm CCA is used to mark these pixels. In the proposed work, the CCA uses a ‘4-connectivity’ method, and labels them using different numbers. Once all the groups of pixels have been determined, each pixel is labelled based on the group it belongs to. Therefore, a set of potential candidates can be selected from the image using the known geometrical conditions, which mainly consist of the width, height and ratio of the plate region. Let P denote the extracted plate region with the size $H \times W$, the first criterion is the ratio R between the

height and width of *P* (i.e. *R* = *W*/*H*). The second criterion is the range of *H* and *W*. The third criterion is the area of *P*. Ranges for *H*, *W* and *R* were selected to be relatively large enough to cover most of the possible sizes of the plate region in the databases. Basically, there are two selection conditions (Condition 1 and Condition 2) used for this purpose. For both conditions, the width, height, area and ratio of the NP are considered. Condition 1 is stricter than Condition 2 where some of the candidates may not meet Condition 1 but can meet Condition 2. The maximum and minimum coordinates of the rectangular plate regions that pass one of the conditions are returned. Normally, the strictest condition (i.e. Condition 1) is perfectly suited for selecting candidates from good condition images (e.g. daytime and clear images); while Condition 2 can be used for selecting candidates from bad quality images (e.g. far view, blur and complex background images). Fig. 7 shows a block diagram that illustrates the selection process.

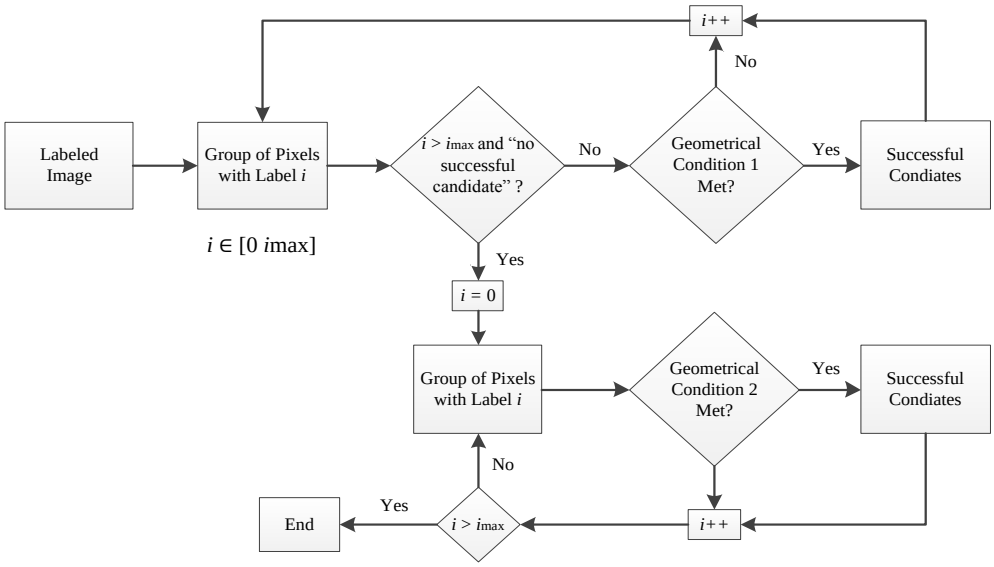


Fig. 7. Flowchart of selection process.

The final Number plate will be extracted from original greyscale image. Fig. 8 shows the selected Number plate.

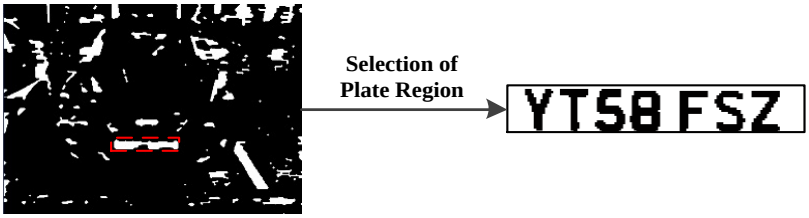


Fig. 8. Selection of Number plate.

4. Proposed Number Plate Localisation Architecture

Morphological operations based architecture consists mainly of an image reader, three morphological operations and CCA. Therefore, this architecture can be designed using the following modules:

- Memory Reader Module;
- Converter Module;
- Morphological Operations Module; and
- CCA Module.

The structure of the proposed architecture is shown in Fig. 9.

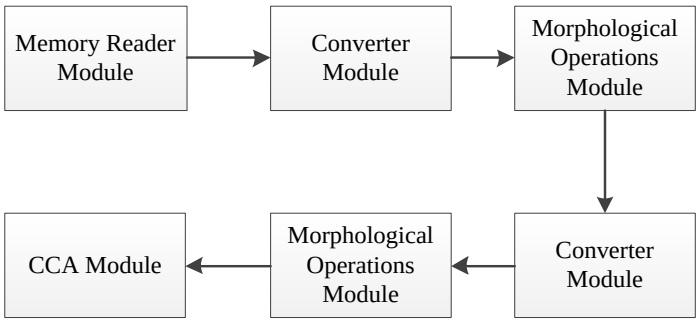


Fig. 9. Morphological operations based system.

4.1 Memory Reader and Converter Module

The first module in the proposed architecture is the memory reader and converter. The memory reader part of the module is used to read the RGB values for each pixel from the original RGB image which has a size of 640×480 and to assign a position coordinate. Fig. 10 shows a block diagram of the memory reader.

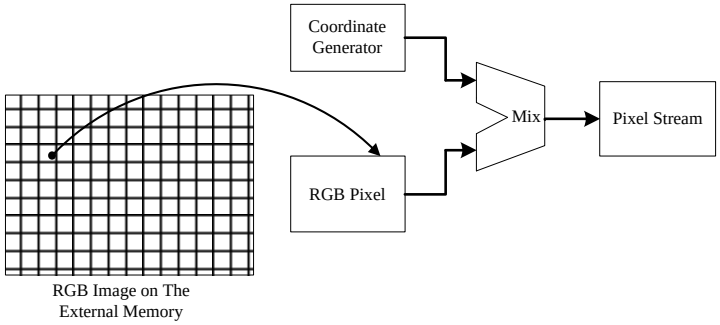


Fig. 10. Block diagram of memory reader.

The converter part of the module is used for the standard RGB (24 bits) to greyscale conversion (8 bits) using (3):

$$Y = \frac{R \times 77 + G \times 155 + B \times 29}{256} \tag{3}$$

This module is also used for the greyscale to binary conversion using a fixed threshold T_f out of 255 (i.e. T_f = 60), which means all values less than T_f will be treated as '0' and values larger or equal to T_f will be treated as '1'.

4.2 Morphological Operations Module

The morphological operations module consists of the morphological *open* and the morphological *close* sub-modules. According to the equation (1) and (2), the morphological *open* operation and the morphological *close* operation can be divided into two sub-filters respectively, i.e. the morphological *dilation* and the morphological *erosion* sub-filters, where the order in each case decides whether the morphological operation is *open* or *close*. The greyscale *dilation* calculates the maximum pixel value in a specific SE. On the contrary, the greyscale *erosion* calculates the minimum value in a specific SE.

The proposed algorithm uses 3×30 rectangle shaped SE, however, for efficient hardware implementation where parallelism can be exploited, this rectangular shaped SE has been decomposed into two small rectangle SEs with the sizes 1×30 and 3×1. Fig.11 shows the block diagram of a proposed pipelined *dilation* filter.

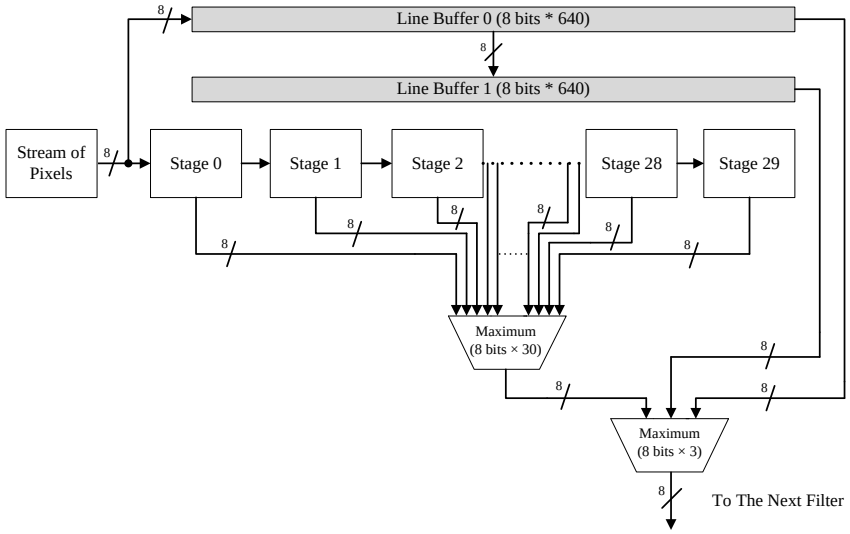


Fig. 11. The block diagram of a pipelined *dilation* filter.

First of all, the value of current input pixel is simultaneously passed into the internal buffers “Stage 0” and “Line Buffer 0” then after every clock cycle it is passed to the next stage until it reaches “Stage 29” and then the maximum pixel value of the current 30 pixels in the 30 stages is calculated. In the meantime, the values of the pixels from two consecutive lines of the greyscale image (i.e. 640 pixels per line) are stored in the two line buffers in order to calculate the maximum value from three consecutive pixels from the same column. The first origin of SE (1×30) is the fifteenth pixel of the first line, so the first coordinate of output should be kept consistent with the coordinate of the fifteenth pixel instead of the coordinate of the current input pixel.

The structure of the *erosion* filter is similar to the *dilate* filter. The only difference is that the minimum value of the pixels is calculated instead of the maximum one. Fig.12 shows the block diagram of a pipelined *erosion* filter.

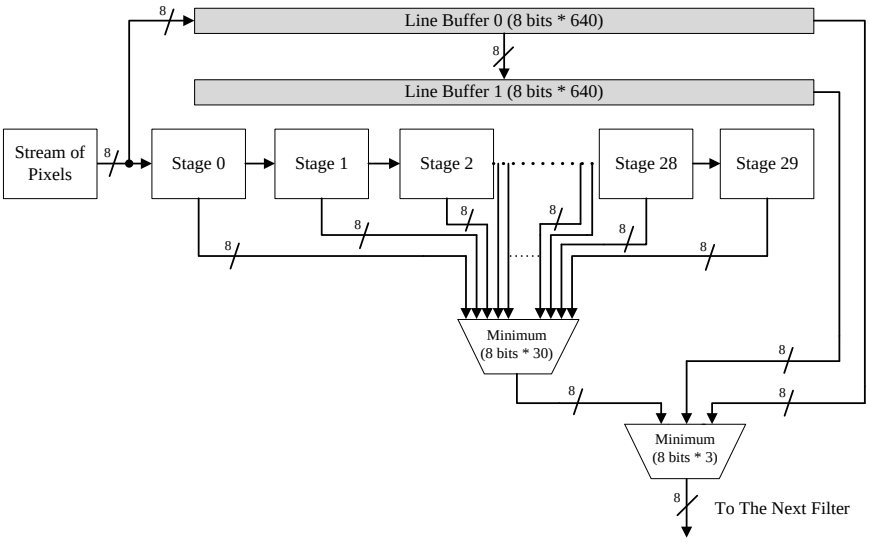


Fig. 12. The block diagram of the pipelined *erosion* filter.

In the proposed architecture, there are three different SEs used for the three morphological operations (i.e. ‘rectangle’ shaped SEs: 3×3, 3×13, 3×30) which can be easily implemented using the block diagrams shown in Fig. 11 and Fig. 12 by simply changing the number of stages (i.e. if the size of SE is 3×3, it requires three stages). The ‘diamond’ shaped SE has been replaced by the ‘rectangle’ shaped SE (3×3) in order to use the same block diagrams shown in Fig. 11 and 12 which reduces the hardware complexity.

4.3 CCA Module

The CCA module is used to mark and select a candidate plate region from the entire binary image. Generally, the pixels of the input pixel stream are divided into several groups or blobs by the CCA module. The grouping is based on the pixels' connectivity. Fig. 13 demonstrates this procedure.

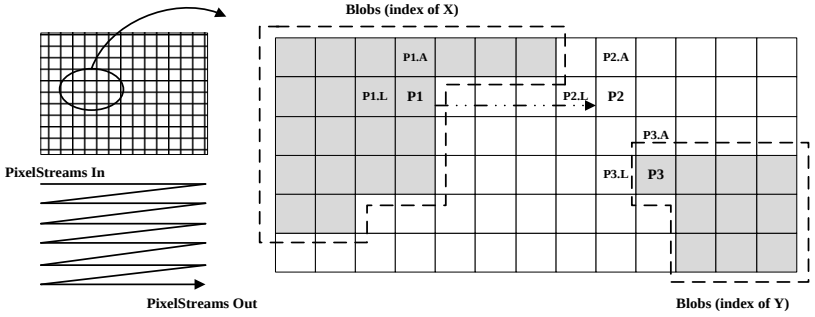


Fig. 13. The block diagram of CCA.

The grouping is performed as follows. The binary stream is scanned from left to right starting from the top line. For instance, a comparison between the current pixel "P1" from Fig. 13, its upper pixel "P1A" and left pixel "P1L", which have already been grouped, is performed. All pixels with value '0' will be assigned to one group with an index '0'. If the value of "P1" is '1' and the indexes of its neighbours are the same and not '0' then "P1" will be assigned the same index as its neighbours. If the indexes of the two neighbours are different and not '0', then the indexes of this pixel and its upper neighbour "P1A" will be the same as its left neighbour (i.e. "P1L"). If the indexes of the two neighbours are different and one of them is '0', then the index of this pixel will be the non-zero index of its neighbour. If the pixel value is '1' but the indexes of its neighbours are both '0', the index of a new group will be assigned to this pixel. Finally, the coordinates of each rectangular shaped group are recorded for the selection of candidates.

Once the whole image is scanned, the selection of a candidate region is performed using the selection process shown in Fig. 7 which is mainly based on the geometrical relationship of the NP region.

5. MATLAB Implementation and Results

The proposed algorithm was first tested in a MATLAB environment using a database of 1000 images containing UK NPs and verified using an on-line public database of 307 images containing Greek NPs. The resolution of all used images is 640×480. The UK number plate database consists of six different sample sets and the on-line Greek database consists of three different sample

sets, which are taken from natural scenes obtained in various illumination conditions and different distances between the camera and vehicles. The three sample sets from the Greek database are similar to the first three sample sets from the UK database. Therefore, for the purpose of performance testing of the proposed algorithm, all samples sets from both databases were grouped into six sets. The first three sample sets are:

- Sample Set 1 - day time colour: this set contains 631 images from the UK NP database and 136 from the Greek one. The NP regions in this sample set are clear and normal size.
- Sample Set 2 - day time close view: this set contains 70 images from the UK NP database and 122 from the Greek one. The size of the NP regions in this sample set is large and the images contain less complex background environment information.
- Sample Set 3 - day time with shadows: this set contains 68 images from the UK NP database and 49 from the Greek one. The NP regions and the backgrounds contain shadows.

The remaining three sample sets are:

- Sample Set 4 - day time moving vehicles: this set only contains 140 moving vehicle images from the UK NP database.
- Sample Set 5 - day time far view: this set contains 75 images from the UK NP database. The size of the NP regions in this sample set is small and the images contain more complex background environment information.
- Sample Set 6 - night time infrared: this set contains 17 images from the UK NP database, which are taken from an infrared camera at night time.

Table 3 shows images from each sample set and the size range of the NP, where the lowest and highest height/width of NP are 18/160 and 60/300 respectively in the databases. Therefore, the expected H, W and R values should fall in the following regions: $18 < H < 30$, $60 < W < 300$ and $2 < R < 9$.

Table 3. The Samples of Used Database

	Sample set 1 (Day time colour)	Sample set 2 (Day time close view)	Sample set 3 (Day time with shadows)	Sample set 4 (Day time moving vehicles)	Sample set 5 (Day time far view)	Sample set 6 (Night time infrared)
NP sizes	30×160 up to 40×220	42×230 up to 60×300	26×120 up to 42×230	30×160 up to 38×200	18×160 up to 30×160	30×160 up to 38×200
UK						
Greek				N/A	N/A	N/A

Table 4 shows the MATLAB implementation results in terms of NPL rate using all sample sets.

Table 4. Successful NPL Rate by Sample Sets (MATLAB Implementation Results)

Database	Sample set 1	Sample set 2	Sample set 3	Sample set 4	Sample set 5	Sample set 6	Overall
UK database	619/631 (98.1%)	69/70 (98.6%)	66/68 (97.1%)	135/139 (97.1%)	73/75 (97.3%)	17/17 (100%)	979/1000 (97.9%)
Greek database	133/136 (97.8%)	120/122 (98.3%)	48/49 (97.9%)	N/A	N/A	N/A	301/307 (98.0%)

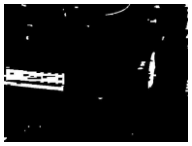
The proposed algorithm has an overall 97.9% NPL rate when tested using the UK images and 98.0% when using the Greek images.

The NPL rate is high for sample sets 1, 2 and 6 compared to sample sets 3, 4 and 5, which is due to the fact that the scenes in the

latter sample sets contain more complex background environments. Generally, the proposed algorithm shows a similar NPL result and a relatively stable performance for both databases.

Although the two geometrical conditions have effectively improved the NPL rate, some images still cannot be handled successfully. Generally, there are two main failed data image sets (see Table 5): (1) Images with more than one successful candidate including the NP itself. (2) Images with no successful candidate. The main reasons for the first set are environment background and NP selection conditions. Since various illumination conditions and the range of NP size is very large (18×160 to 60×300), two selection conditions cannot fully cover all NPs. In some cases, the false candidates in some images are very similar in size to the true candidates in other images in the database which cannot be excluded. In order to overcome this problem, a validation process should be added before character segmentation for cases where there is more than one successful candidate. For the second set, there are no successful candidates due to the length of the distance between the camera and the car which results in very small NP images and an increase in the background noises. In this situation the NP feature cannot be extracted properly by the proposed morphological operations.

Table 5. Failed Images in Both Database (MATLAB Implementation)

	(1)			(2)		
Original Image						
Image before CCA						
Detected NP				No successful candidate	No successful candidate	

6. FPGA Implementation and Results

The proposed architecture for NPL has been simulated in PAL Virtual Platform (PALSim) [18]. After simulation, the architecture has been successfully implemented and verified using the Mentor Graphics RC240 FPGA development board equipped with a 4M-gate Xilinx Virtex-4 LX40 [19]. Handel-C and PixelStreams, which is a library that can be used for rapid development of video image streaming applications, have been used for the hardware description of the proposed architecture [20].

The original RGB image is first stored in an external memory on the RC240 board. The external memory data width is 32 bits, which means every pixel value (24 bits) can be saved on a single memory location. In Fig. 10 each RGB pixel is combined with its corresponding position coordinate and synchronisation information and then sent to the filter blocks previously outlined in Fig.1 running in parallel. Every clock cycle one data pixel is passed from one block to the next.

Both UK and Greek databases have been used for testing and validating the FPGA implementation. The results show a similar performance compared to the software implementation in terms of NPL rate where the entire overall rate is 97.8%. Table 6 shows the FPGA implementation results when using all sample sets.

Table 6. Successful NPL Rate by Sample Sets (FPGA Implementation Results)

Database	Sample set 1	Sample set 2	Sample set 3	Sample set 4	Sample set 5	Sample set 6	Overall
UK database	618/631 (97.9%)	69/70 (98.6%)	66/68 (97.1%)	134/139 (96.4%)	73/75 (97.3%)	17/17 (100%)	977/1000 (97.7%)
Greek database	133/136 (97.8%)	120/122 (98.3%)	48/49 (97.9%)	N/A	N/A	N/A	301/307 (98.0%)

6.1 Hardware Usage, Running Frequency and Power Consumption

Due to the low complexity of the proposed algorithm, the proposed architecture requires only 33% of the on-chip FPGA resources. Table 7 summarises the required on-chip resources.

Table 7. Usage of FPGA on-chip Resources

	Used	Available	Utilisation
Occupied Slices	6,195	18,432	33%
LUTs	8,871	36,864	24%
Block Rams	18	96	18%

The 33% on-chip resource usage leaves 67% to be used for implementing the next stage of an ANPR system (i.e. NP Segmentation and OCR). The maximum running frequency is 86 MHz and the number of clock cycles needed for one image to be processed is 401247. The execution time for processing one frame can be roughly calculated using the following equation:

$$T = \frac{C}{f} \tag{4}$$

Where T is the execution time in ms; C is the number of clock cycles needed for one image; and f is the maximum running frequency.

Based on Equation (4), the proposed architecture can process one image and produce a result in 4.7 *ms*. This means that the proposed architecture satisfies the minimum requirement for real-time processing. The result achieved in terms of maximum running frequency and area used for implementing this important part of an ANPR system shows that there is enough room to implement the whole ANPR system on one FPGA.

The consumption power of the designed circuit has also been analysed using Xilinx XPower [21], and the results obtained are shown in Table 8.

Table 8. Estimation of Power Consumption

Name of Power	Value of Power (mW)
Total Quiescent Power	416
Total Dynamic Power	212
Total Power	628

The total power consumption of FPGAs consists of quiescent and dynamic components. Table 9 shows that the total power consumption of the proposed architecture is 628 mW which is lower than the power consumption of a typical PC if it is used as the processing unit in an ANPR system.

6.2 Comparison with Existing Work

A comparison of the experimental computational speed and NPL rate with existing PC, DSP and FPGA based implementations of NPL is shown in Table 9.

Table 9. Performance Comparison

	Platform	Processor Clock Speed (MHz)	Image Resolution (pixels)	NPL Time (ms)	NPL Rate (%)
Proposed System on FPGA	FPGA Virtex-4	86	640×480	4.7	97.8%
Proposed System on PC	PC	2300	640×480	143	97.9%
[6]	PC	1700	768×534	100	99.6%
[2]	DSP C6414 and FPGA	600	352×288	141.62	96%
[16]	FPGA Virtex II	72.062	256×256	9.25	87%

The proposed system outperforms existing ones as it shows a higher NPL rate and faster NPL speed with higher resolution compared to the databases used in systems [2] and [16] on the table. Although the testing databases used for the three methods are different, the proposed system has been tested and verified using a large local database and an on-line public database and shows stable results. However, it should also be noted that the databases used in [2] and [16] are not available as they are not public databases.

However based on the results published, when compared to system [6], the proposed work has a faster NPL speed but slightly lower NPL rate. This is due to the fact that fixed measures of distance and angle, based on prior knowledge, have been used for the algorithm used in [6], which is based on edge detection and morphological operations. This prior knowledge boosts the results to a

high level of accuracy which is not the case for the proposed algorithm which uses images taken from different distances and angles more reflective of real life recordings.

The proposed method in [2] uses a DSP for NPL implementation and a FPGA for buffering video frames between a video input processor and the DSP. Although the DSP frequency is 600MHz, the processing time for one image is higher than the one for the proposed system. This is due to the fact that the proposed architectures is fully parallelised and requires less clock cycles which significantly increases the NPL speed.

By comparing the results of the PC and FPGA-based implementations of the proposed algorithm, it can clearly be seen that the latter outperforms the former with a 30-time speed-up with close accuracy; therefore, the proposed FPGA-based system can be used as a viable solution to replace software based solutions where cost, size and energy consumption will be reduced.

7. Conclusion

An ANPR system can be divided into three main image processing stages: NPL, NP segmentation and character recognition. All three stages are computationally intensive tasks. Recently, FPGAs have become a viable solution for performing computationally intensive tasks. Owing to the importance and the use of ANPR systems in law enforcement, an efficient NP localisation algorithm has been proposed in this paper for FPGA implementation. The algorithm is based on morphological operations and is multiplier/divider-free and requires only 33% of the available on-chip resources of a Virtex-4 FPGA. Parallel building blocks have been used for the FPGA implementation and the whole system runs with a maximum frequency of 86 MHz and is capable of processing one 640×480 image in 4.7 *ms* with a localisation rate of 97.8%.

The 33% resource usage of the FPGA in implementing NP localisation leaves 67% of the FPGA area free for the remaining parts of an ANPR system (i.e. NP Segmentation and character recognition). This allows the entire ANPR system to be implemented on a single FPGA that can be placed within an ANPR camera housing to create a stand-alone unit thus drastically improving energy efficiency whilst removing the need for the installation and cabling costs associated with bulky PCs situated in expensive, cooled, waterproof roadside cabinets.

References

[1] Juan, J. and Xu, J.: 'Research of overall program on highway toll collection system', Proc. Int. Conf. Information Science and Technology, March, 2011, pp. 1218-1221

[2] Arth, C., Leistner, C. and Bischof, H.: 'TRIcam: an embedded platform for remote traffic surveillance', Proc. IEEE Conf. Computer Vision and Pattern Recognition, 2006, pp. 125-125

[3] Anagnostopoulos C. N. E., Anagnostopoulos I. E., Psoroulas I. D., Loumos V., and Kayafas E.: 'License plate recognition from still images and video sequences: A survey', IEEE Trans. Intell. Transp. Syst. 9(3), 2008, pp. 377–391

[4] CitySync Limited, http://www.citysync.co.uk/ , accessed Jan 2012.

[5] Vargas M., Toral S.L., Barrero F., Cortés F.: 'A License Plate Extraction Algorithm Based on Edge Statistics and Region Growing', Lecture Notes in Computer Science, 2009, 5716/2009, pp. 317-326.

[6] Bai, H. and Liu, C.: 'A hybrid license plate extraction method based on edge statistics and morphology', Proc. 17th Int. Conf. Pattern Recognition, 2004, 2, pp. 831-834

[7] Chang, S., Chen, L., Chung, Y. and Chen, S.: 'Automatic license plate recognition', IEEE Trans. Intell. Transp. Syst., 2004, 5, pp. 42-53

[8] Wang, F., Man, L., Wang, B., Xiao, Y., Pan, W. and Lu, X.: 'Fuzzy-based algorithm for color recognition of license plates', Journal of Pattern Recognition Letters, 2008, 29, (7), pp.1007-1020

[9] Kim, K. I., Jung, K., Park, S. and Kim, H.: 'Support vector machines for texture classification', IEEE Trans. Pattern Anal. Mach. Intell.,2002, 24, pp. 1542-1550

[10] Kim, K. I., Jung, K. and Kim, J. H.: 'Color texture-based object detection: an application to license plate localization', Patten Recognition with Support Vector Machines, Lecture Notes on Computer Science, 2002, 2388/2002, pp. 321-335

[11] Wang, Y., Lin, W. and Horng, S.: 'A sliding window technique for efficient license plate localization based on discrete wavelet transform', Expert Systems with Applications, Apr. 2011, 38, pp. 3142-3146

[12] Anagnostopoulos C., Alexandropoulos T., Loumos V. and Kayafas E.: 'Intelligent traffic management through MPEG-7 vehicle flow surveillance', Proc. IEEE Int. Symp. on Modern Computing, 2008, 9, pp. 377–391

[13] Clemens, A., Florian, L. and Horst, B.: 'real-time license plate recognition on an embedded DSP-platform', Proc. IEEE Computer Vision and Pattern Recognition Conf., 2007, pp. 1-8.

[14] Cancer, H., Gecin, H. S. and Alkar, A. Z.: 'Efficient embedded neural-network based license plate recognition system', IEEE Trans. Veh. Technol., 2008, 57, pp. 2675-2683

[15] Bellas, N., Chai, S. M., Dwyer, M. and Linzmeiser, D.: 'FPGA implementation of a licence plate recognition SoC using automatically generated streaming accelerators', Proc. 20th Int. Conf. Parallel & Distributed Processing Symposium, April, 2006, pp. 8

[16] Kanamori, T., Amano, H., Arai, M., Konno, D., Nanba, T. and Ajioka, Y.: 'Implementation and evaluation of a high speed license plate recognition system on an FPGA', Proc. 7th IEEE Int. Conf. Computer and Information Technology, 2007, pp. 567-572

[17] Shih, F. and Wu, Y. 'Decomposition of arbitrary gray-scale morphological structuring elements', Pattern Recognition, Dec. 2005, 38, pp. 2323-2332

[18] PAL User Manual, Mentor Graphics Corporation. Jan. 2010.

[19] RC240 Datasheet, Mentor Graphics Corporation, Jan. 2010.

[20] PixelStreams User Manual, Mentor Graphics Corporation, Jan. 2010.

[21] Xpower Tutorial: FPGA Design, Xilinx, July 2002.