22

Industrial Vision Systems, Real Time and Demanding Environment: a Working Case for Quality Control

J.C. Rodríguez-Rodríguez, A. Quesada-Arencibia and R. Moreno-Díaz jr
Institute for Cybernetics (IUCTC), Universidad de Las Palmas de Gran Canaria
Spain

1. Introduction

This chapter exposes an OCR (Optical Character Recognition) procedure able to work with very high speeds.
The architecture of the pattern recognition algorithm we present here includes certain concepts and results which are developed in previous publications [3,4]. We consider a production line of a beverage canning industry where cans with faulty imprinted use date or serial number have immediately to be discharged from the line.
The problem is well-known in the industrial scene. A code or a set of characters is registered on the surfaces (can bottoms) to very high speed. The registration can fail, can take place only partially, or can print wrong something. It is important to know with certainty what has happened. The most general solution is to read what has been printed immediately after print itself.
Surfaces are metallic (tinplate/aluminium) can bottoms for our particular case and the code denotes the limit of the use of the product (beer or similar).

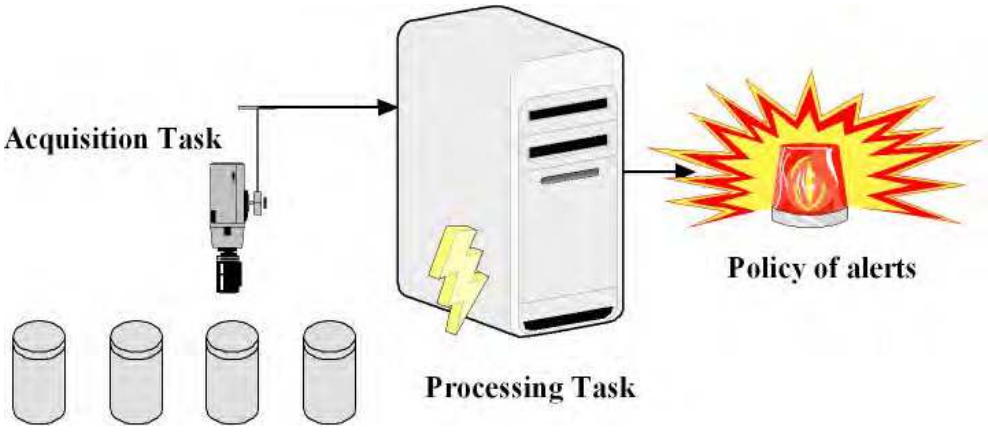


Figure 1. The scheme of the goal

Source: Vision Systems: Applications, ISBN 978-3-902613-01-1
Edited by: Goro Obinata and Ashish Dutta, pp. 608, I-Tech, Vienna, Austria, June 2007

Our goal is to build a capable application to process 120,000 cans per hour (35 cans per second). Nowadays, there is not application on the market which is able to manage this speed.
Therefore, our purpose has to do with an OCR that is confronted to two challenges:

1. Extraction of characters registered on a difficult surface.
2. Processing to very high speed.

Our keys to approach the problem have been:

1. Capable hardware
2. Domain restrictions
3. Intensive calibration
4. Parallel architecture
5. A progressive and aggressive reduction process of the interest area
6. Simple operations in integer arithmetic
7. Two independent tasks: Validation and Traineeship

Here is a brief explanation of these keys.

1. Capable hardware: The critical requirements are that the system is in fact invariant from its position during the text analysis of nearly 30.000 cans per minute in real time. The system has to be reliable and it relies on specific hardware. Thus, a high speed acquisition camera, an efficient acquisition board, a strong multiprocessing system and a considerable bandwidth for main memory load are the basic requirements.
2. Domain restrictions: A specialized environment which is very well known and restricted diminishes the number of contingencies to consider and allows therefore making presumptions easily. View section [1] and section [2].
3. Intensive calibration: There are two types of calibration of the system. The first type is to focus you on guaranteeing an enough quality of image for the treatment. It affects mainly physical parameters of the system. The second type has to do with the training of the system. The system should be trained with expected shapes for comparison.
4. Parallel architecture: Use intensive of multithread at several architecture layers in strong multiprocessing system.
5. A progressive and aggressive reduction process of the interest area: Reducing the input domain contributes to reduce the time of processing.
6. Simple operations in integer arithmetic: Sum, subtraction, multiplication of integers, and integer division are the absolutely dominant operations.
   In computers of general purpose, the integer arithmetic unit is faster than the float arithmetic unit. All procedures described in this chapter disregard float arithmetic for validation computation. The real essential operations are out side of the continuous cycle of validation.
7. Two independent tasks: Validation and Traineeship. Only The Validation Stage is described in this paper. However there is another stage in where the knowledge base is built.

2. Visual Scenario

We will work at an industrially dedicated environment; therefore the scene of events should be restricted. The reality will be mapped to two-dimensional matrixes with values between 0 - 255 representing different levels of grey. The possible states can be anticipated and listed: They are not too numerous neither very complex.

1. There can be nothing but cans and "background" in our field of vision.
2. The "background" has fixed and unchangeable visual characteristics [very dark colour].
3. There is NO background inside the border of a can.
4. Outside the border of a can there can only be background or other cans, but nothing else.

We are only interested in processing a single can within the frame of our camera. Therefore the physical acquisition platform will reinforce the presence of a single clean can in each capture. On the other hand, the hardware/software system will have to guarantee that no can will be lost during acquisition: all can processed by the installation will be processed too by our system.

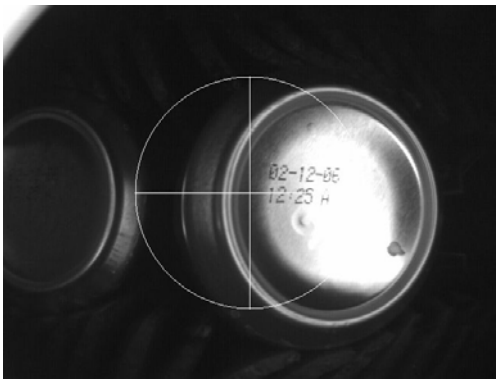


Figure 2. Typical acquisition from camera

3. System Preconditions

The code is associated to a single orientation and processing direction. In order to compare the expected code with the acquired code during validation both codes must have the same orientation and processing direction. The question is: How do we know the orientation and processing direction of the acquired code?

We have the following facts in our favour:

1. Once the print head and the acquisition source camera are fixed, orientation and processing direction are fixed for all cans.
2. The print head and the camera can be freely rotated if it is more convenient to our purposes. There are no other factors to consider except our own satisfaction.
3. Due to the previous fact, we can force to have an orientation and processing direction for the code. Therefore, these are known from the beginning before processing starts. It is not necessary to make a specific computation to get them.

As we will see soon, trying to have the orientation parallel to the natural vertical and horizontal axes will make the computation easier. This is what we are trying to get.

4. Elliptical Histograms Algorithm

The basis of algorithm of the elliptical histograms is analogous to the biological receptive field concept. The computation input field has a specialized shape that maximizes its

perceptiveness to certain stimuli. A purposeful task can be the simple stimulus presence detection.
The stimulus is the can. The task is to determine if a can is in the vision field or not. If so, estimate the position of its centre within the vision field.
The facts which support our procedure are:

1. The cans always are shown up at the vision field like can bottoms.
2. The can bottoms are always brighter than the background.
3. The can bottoms are nearly circular.
4. The can bottoms run a restricted and predictable trajectory inside the vision field.

The idea is to look for any measure that has maximum or minimum values according to fact that the can is present or absent. Arbitrary noise in the image should not provoke false positives or false negative easily. On the other hand, the measure should be computed fast.
The histogram is a classification and count of levels of grey steps. Provide the distribution of colour of a collection of points. It is a very fast operation because only it implies queries, comparisons and sums.
The fact 2) establishes that the presence/absence of a can modifies the distribution of color. Therefore, the fact 2) determines the sensibility to histograms.
We should look for an optimal collection of points that provides very different histograms according to the fact that the can is present or not. The collection of points should be efficient and avoiding unnecessary queries.

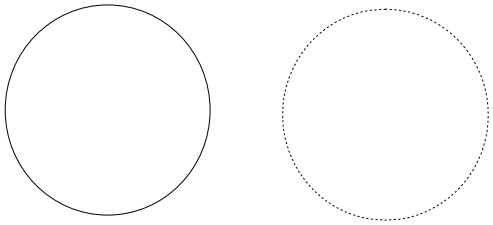


Figure 3. The receptive field shape and queries position

The fact 3) provides us what kind of shape should be looked for inside the image.
The circular shape is suitable like the collection of points of search. The receptive field is defined as a circle (i.e a circular line or circumference).
A sufficiently "bright" histogram for a particular receptive field can give only a hint that there is a can whose centre is also close to the centre of the histogram circle. So, it is an ON/OFF receptive field.
Besides, it is established in the section [2] that the unique objects that would invade the vision field are cans. Only the noise would be able to cause troubles. The shape circular is sufficient complex to distinguish between cans and shapes of luminous noise for the majority of cases.
The procedure would be:

1. A circle which has not been processed yet is chosen. If there is no not-processed circumference then it is declared that there are no cans in the image and we jump to step 5.
2. The histogram for that circular line is computed.
3. If the histogram is enough bright then it is declared that a can has been found and we jump to step 5.

4. Otherwise the circle is marked like processed and we return to step 1.
5. Status of the image declared!

Discussing the general idea, it gives us some hints to optimize making good use of the restrictions of the environment:

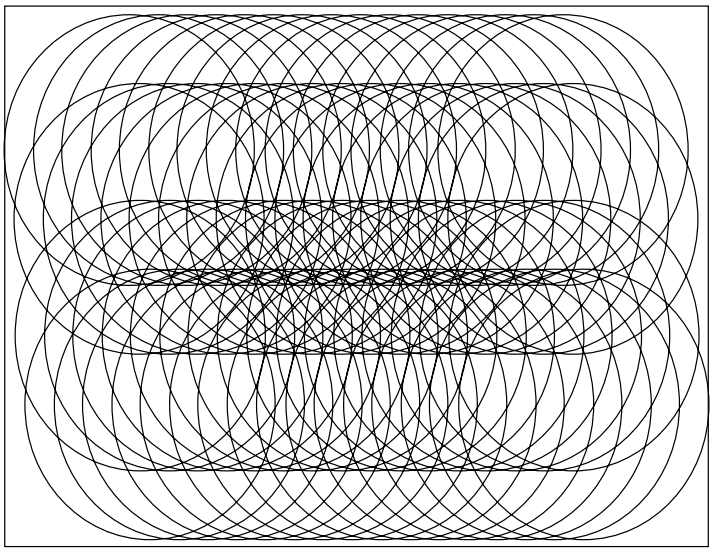


Figure 4. A can could be located at any position sweeping the image with the described receptive fields

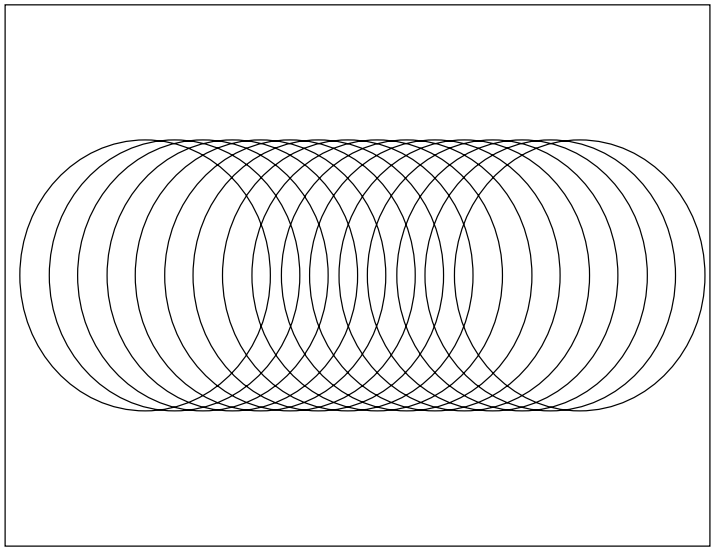


Figure 5. The nature of the chain of can filling imposes that cans move around in a restricted line so the search can be restricted to exploration of that axis of line

A new principle is:
Once the previous location of a circumference in a previous frame is known we can determine which is the next more-probable-position for relocating the can. The search stops immediately when the can is found. Therefore it will be tracked those circumferences that correspond to those positions in decreasing probability order.

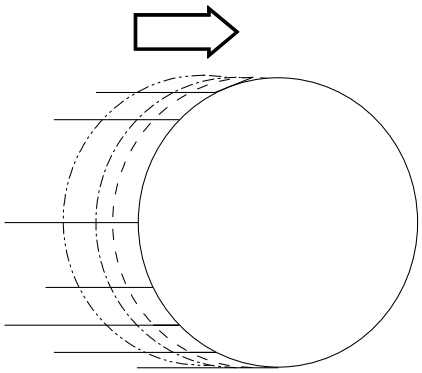


Figure 6. Processing the circles with greater probability of being active first can improve the speed

Why elliptical histograms? The frames produced from the camera are distorted due to the natural limitations of the quality of optics. This distortion does not affect of appreciable way the readability of the code. But the cans leave the circular shape as they get close to the boundaries of the image.

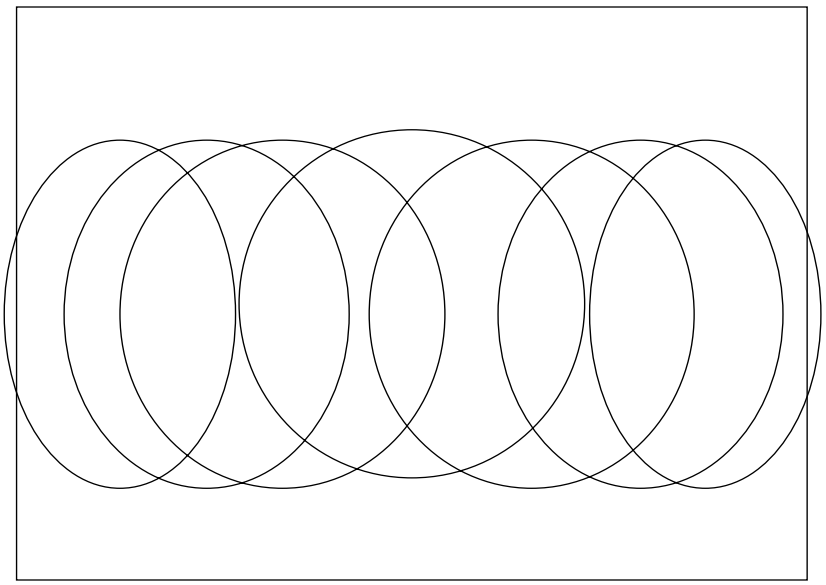


Figure 7. Sweeping with ellipses

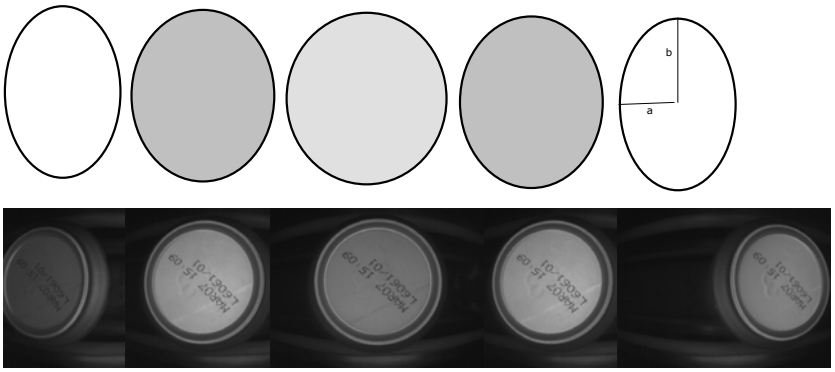


Figure 8. This gradual flattening is contemplated by the procedure turning the circumferences in ellipses. An additional advantage is that fewer points are processed

Can Counter

The can count is made using ellipse activation. There are three important facts:

1. The motion direction of the cans is previously well known (from left to right, or else, from right to left).
2. The ellipse activation order should correspond to that motion direction.
3. When a new can comes into scene, the ellipse activation order breaks the pattern of motion.
   The can counter counts the breakings of fact 3).

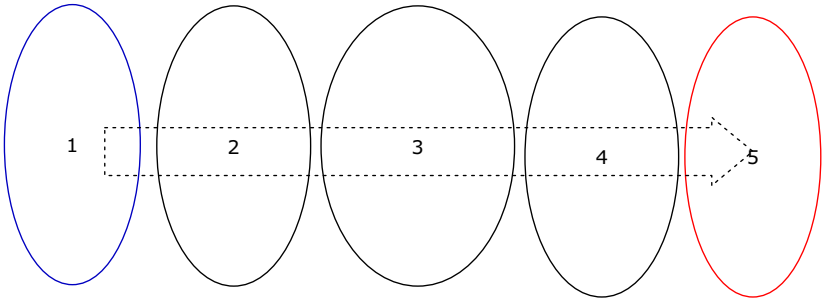


Figure 9. Natural order of ellipse activations

The normal can movement follows the motion sequence [1, 2, 3, 4, 5].
A mistake in the motion sequence (For example, [1, 2, 3, 5, 4]) means that a new can has come in the vision field of the camera completely.

5. Thickening

The pre-process of thickening improves the general quality of the image with a cheaply computer cost. It simplifies superfluous details of the image without harming the readability of the code.

Its main advantage is that it provides a code with less fragmented and more consistent characters.
Its main disadvantage is that characters can stick erroneously among themselves.
The procedure involves applying a limited convolution the winning ellipse of the algorithm of elliptical histograms.

Figure 10. Image after thickening and image before thickening

a	b	c
d	e	f
g	h	i

$$= \text{Min}(a,b,c,d,e,f,g,h,i)$$

Figure 11. This is the convolution mask. Thence there are no complex implicated operations

6. Segmentation by flooding

The reduction of the number of queries is the reason that justifies the application of this technique. A query is the reading/writing of a pixel of the actual image.
It is possible to go through every pixel of the image checking if they are part of the code. A systematic tour will give us the best results. However, it's very important to avoid unnecessary operations/queries. The Flooding Techniques are one of the possible solutions to use.
The mechanism of the flooding technique main goal (also known as Pixel Progressive Addition Method) is to obtain which is result as the systematic tour, but with less operations/queries.
It's use should be success full because it depends on two easily contrastable principles:

1. The majority of queries will give a negative result concerning their belonging to the code.
2. The queries results with positive code are those positions which have only a short-distance from each other positions.

The flooding as well as the segregation of points that are part of the code from those points that are not, groups the located points of code in useful sets. The pixels that form the main semantic object scene (the code) can be grouped in sub-semantic objects (characters). Therefore, this is the decisive step of transforming pixels into abstract entities (characters of code).

Here is the procedure:

1. Select a set of flooding seeds. The flooding seeds are driven queries with a higher probability of finding a code pixel. The selections of seeds are based on some heuristic knowledge.

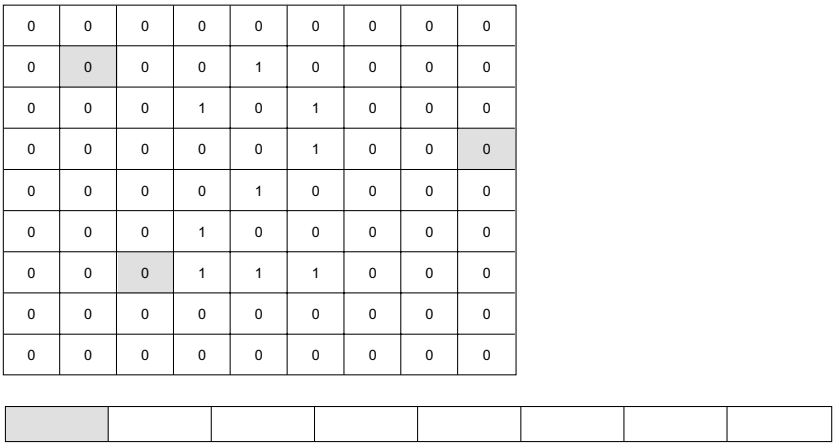


Figure 13. Randomly seeds without ink

2. Choose an not-yet-discarded seed, if there is no one available go to exit. If the seed has not yet been marked as visited, do it and go to step 3. Otherwise discard it and repeat step 2.
3. If there is no ink in the seed, declare it as sterile and discard it .Return to step 2. Otherwise go to step 4.

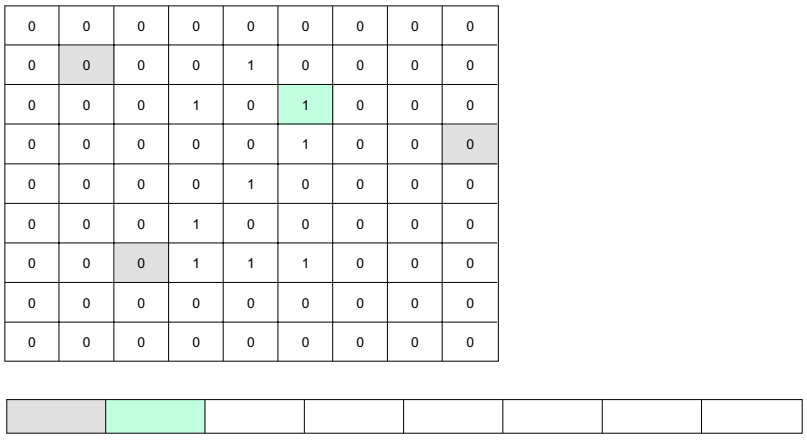


Figure 14. A seed with luck!

4. Open up a flooding point. The seed is declared as the flooding starting point.
5. Look up the immediate proximity neighbours marking all performed queries like visited neighbours. Go to step 6.

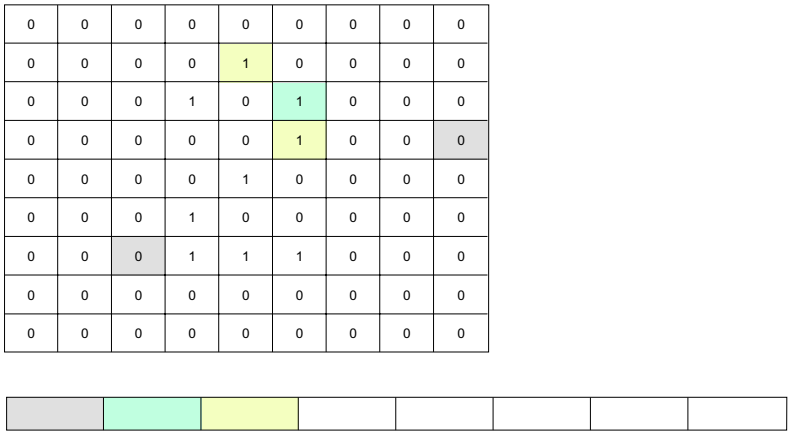


Figure 15. Inspecting neighbours

6. Declare all neighbour marked with ink as a flooding point. Return to step 5. If all neighbours with ink have been processed, go to step 7.

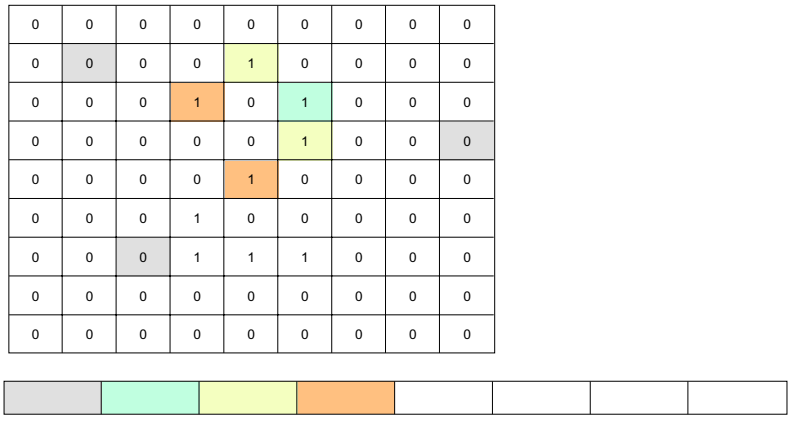


Figure 16. Inspecting neighbours of neighbours

7. Close the current flooding. Discard the current flooding if:
   - It has too many collected points (too big to be a character, so it is just noise).
   - it is too small to be a character (it is noise too) .

   Go to step 2.

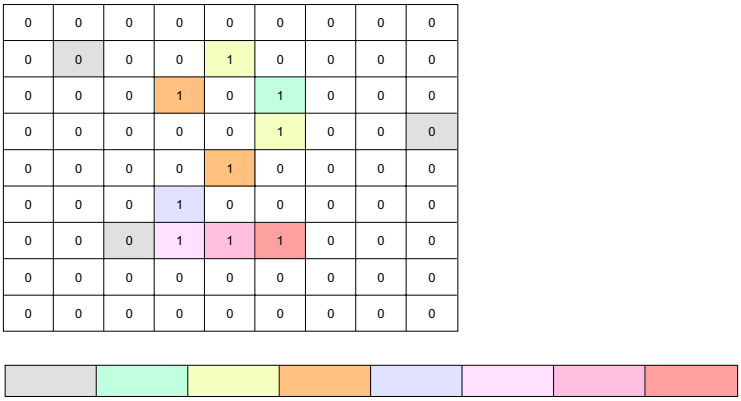


Figure 17. The flooding task has concluded

8. Flooding Completed

7. Grouping

The main goal of Grouping is to distribute the flooded zones got in the previous stage between the real characters of the code. It is composed of three steps.

7.1 Division in Bands

This technique tries to separate the lines of printed code. Besides this, you can get additional useful information for noise suppression. The requirement is that the orientation of the expected code has to be known previously. This condition be comes true because it was established in the section [2].

The purpose of this technique is to divide the image in bands. A band is a longitudinal section of an image which is parallel to the code orientation axis. This technique tries to contain each of the different processed code lines in bands. Therefore, a band can be empty (or filled with noise) or contains a code line. Bands are described by their boundaries.

Ideally, the band boundaries are straight lines. However, a band boundary can not cut flooding zones. So it has to surround the flooding zones in an optimal way. Band boundaries have their starting points separated at a certain distance from each other, and should not cross between them.

The speciality of the technique consists of taking advantage of the interlineations code space. It is hope that the interlineations code space could be easily perceived by a band boundary.

The band boundaries are generated by the following procedure:

1. Assume an origin: The beginning of the boundary, and the direction of movement.
2. Check whether the end of the image has been reached or if there has been a cross between boundaries (a band boundary can never cross another boundary). If the check is positive, exit.
3. Apply a collision direction test with a flooded zone.
   3.1. If the test is negative, advance a step in a straight line (direction of movement)

3.2. If the test is positive we surrounded the flooded zone, choosing the direction of movement with less resistance. If there are several options of movement with equal resistance, we choose one randomly.

4. Return to step 2.

7.2 Splitting Flooded Zones

The use of the term "super-zone" is used to designate an oversized flooded zone. The oversize is exposed according to a criteria related the maximal size of the expected character. A super-zone indicates an anomaly. Here is the list of probable anomalies whose symptom causes the final identification of a super-zone:

1. Noise.
2. Two or more characters have merged among themselves in an error.
3. One or more characters have been merged to noise.
4. A severely deformed character.

In those cases the super-zone should be decomposed in sub-zones. Decomposition of super-zone is made by applying the same flooding algorithm described in chapter [5] but the input field is much more restrictive. The new input field will be the same image but with an inferior level of thickening.

The thickening defragment the image hiding their details [4]. Generally this is fine because the excess of details penalizes the general processing. Therefore, ignoring them normally is advantageous. Unfortunately some details that are inadvertently suppressed are critical at times. It is possible to recover the lost details making use of images with inferior levels of thickening.

Here is the procedure:

1. Only the points of coordinates that constitute the located super-zone will be processed. The grey levels will be recovered from a less-thickened image.
2. The segmentation by flooding will be applied like if is described in section [5] (Segmentation by Flooding). As a result, one or more flooded zones will be created.
3. The points of the super-zone which are not yet assigned by the previous step will be associated to the new zones according to some criteria like nearest neighbourhood.
4. If the criterion of step 3 is unable to reassign some points to a new specific flooded zone, these points will not be assigned evermore. The point will be labelled as background.
5. This set of new flooded zones (composed by one or more zones) will finally replace the previous super-zone.
6. If any of these new flooded zones comes out to be a new super-zone, it will be labelled as noise. So the procedure will not be applied recursively.

7.3 Merging flooded zones

In general two flooded zones can be merged if they overlap them self.

Let us define the rules of the overlay algorithm:

1. Our domain will be discrete because the flooded zones that are defined as finite set of point with coordinates in a discrete space. The point coordinates of the flooded zones are expressed like Cartesian pair in the plane.

2. The overlapping dimension of a flooded zone is equal to the greatest distance between the projected points of the flooded zone on the perpendicular axis of the overlapped axis.
3. The traced segment between the borderline points of the overlap dimension is denoted as overlap segment of flooded zone.
4. Two flooded zones are overlapped in relation to a suitable axis of overlay when it is possible to draw at least one parallel line to this axis that cuts its two overlap segments.
5. The overlay property is applied equally to all implicated flooded zones.
6. The overlay among flooded zones from different bands is always zero.
7. A flooded zone can overlap with multiple flooded zones.
8. The number of not-coincidental parallel lines of overlap is the dimensions of the overlay between two flooded zones. Even though the dimension of the overlay is equal for the two flooded zones, the degree of overlay of each flooded zone is the quotient among the fore mentioned dimension and the dimension of its overlapped axis.
9. A flooded zone is included by another one that is overlapped by the first, if all its points are intersected by the overlapped rays. Two or more flooded zones can include themselves mutually.
10. The distance among the overlap segment midpoints should not exceed a certain distance in order that the overlay will be accepted.

Here is the procedure description:

1. A table of overlay with the flooded zones is built.
   The overlay table is a collection of all degree of overlay between the existent flooded zones. It is calculated confronting all the zones among themselves applying the described rules.

The table construction require to establish the overlay axis perpendicular to the code orientation axis.

	Zone I	**Zone II**	**Zone III**	**Zone IV**	**…**
Zone I	**0 %**				
Zone II		**0 %**			
Zone III			**0%**		
Zone IV				**0%**	
…					

Table 1. Scheme of overlay table

The intersection of row II and column III supply the degree of overlay of the zone II on zone III.

2. The merging couples are recovered from table in strictly decreasing order of degree of overlay.
3. A merging is valid if the maximum size of the merged zone does not exceed the *maximum size for a character.* And the two most distant points of the merged zone do not exceed the limit *of size for a character.*

Once a successful merging was made the overlay table is recalculated for this new zone. On the other hand, the two zones that have been merged are suppressed.

The procedure is iterated. If no new fusions are possible, the procedure stops.

8. Validation

The steps of validation are:

1. Labels of Character
   It is possible to find the coordinates of the flooded codes in terms of the expected codes because it is known that the orientation and processing direction of the printing signs. We will label each flooded zone with the pair: (Line index, position of the character inside the line). In this way (1, 4) means second line (Zero is the first), fourth character.
2. Retrieval of expected character
   It is possible to recover the family of morphologies of the expected character of our base of morphologies of those characters learned by means of the described pair.

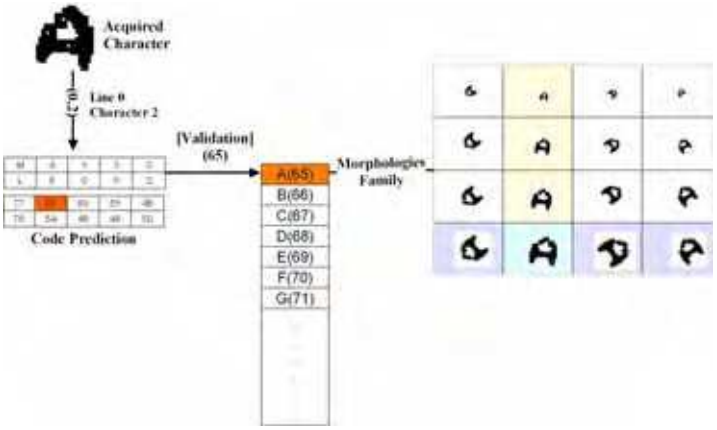


Figure 18. The steps for morphologies recovering

3. Comparison between expected characters and acquired characters
   First calculate the correlations of the acquired character against each one of the recovered morphologies. It is stops when finds a sufficiently seemed morphology.

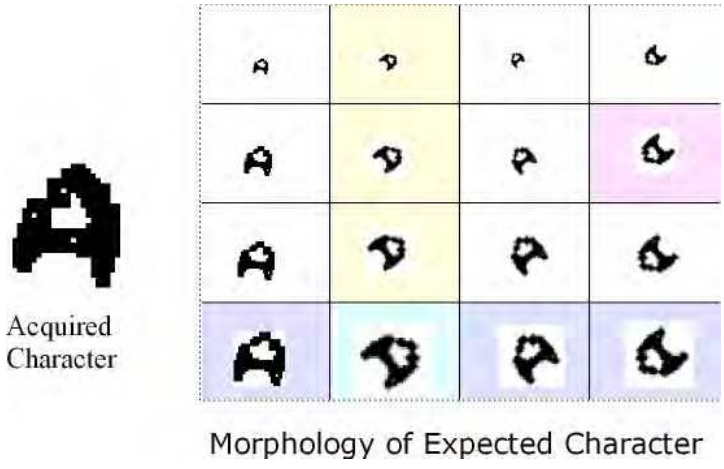


Figure.18. The strongly-hoped inputs for matching

4. Merging with the next character if an error
   If not coincidences/similarities occurs then it is applied a merging among the acquired character with the following should be made. Perhaps code dispersion has happened. Repeat steps 3-4 with the new acquired character. If it is not possible we must go to the next obtained character but we should persist with the expected actual character.
5. Policy of alerts
   If one or more expected characters are NOT found then they are NOT valid and we will declare NOT VALID. The policy of alerts, not described in this document, will make a decision how to act.

You must note: Templates are morphologic representations of the characters. The templates are processed and stored like rectangular bitmaps. The one bit marks presence of ink and the zero bit marks absence ink. A character has a set of associated morphologic templates called morphologies family.

9. Conclusions and future work

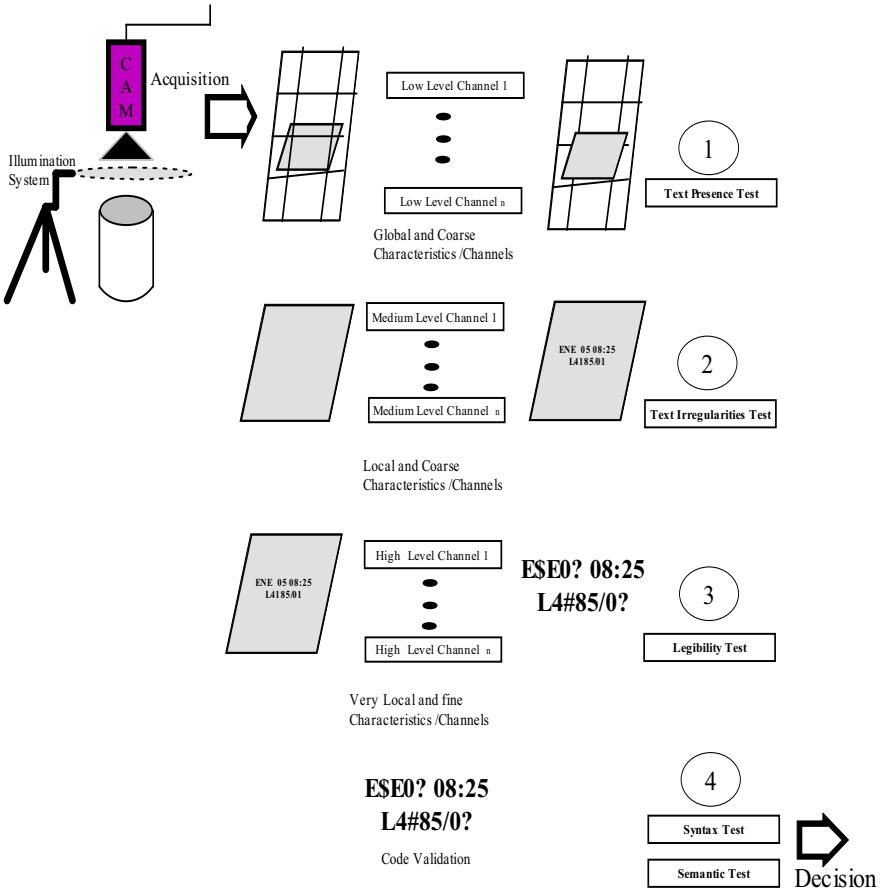
In this paper we studied a series of algorithms based on simple operations performed with fast arithmetic. The restriction on the used operations is due to the stress-producing requests of speed. Unfortunately, the obtained results prove that the series of algorithms mentioned IS NOT SUFFICIENT for the requirements.

The problem results in that the processing time for each can is variable and sometimes it is excessive. This is due to the fact that each can can offer different challenges of process complexity. For example cans with a lot of noise generate more flooded zones, and the number of flooded zones enlarges the load of the process. Therefore, some cans are validated more slowly than other cans.

It is not possible to delay the processing of every can. There can be no queues of processes of cans. It is important to manage of that the maximum time of processing per can does not exceed the safety time.[i.e. The time interval between the begin of recording a can and the begin of recording the next can].

The immediate future work will be implement multithreading versions of the series of algorithms. On the other hand, the algorithms could run simultaneously. This is possible if consecutive beers are processed at same time.

In the following scheme, a biologically inspired architecture for this application is presented. The basic ideas of multichanneling in the visual tract are present. On the input image, a multiprocess task is first triggered to extract the area of interest where first text is to be located. Thus, a second multichannel analysis analysis the possible singularities in the text. The final validation consists of determining the coherence and plausibility of text syntactically and semantically. All these processes are independent and are separately operated. Thus, the labels {1}, {2}, {3} and {4} denote different stages within the same visual tract.



10. References

Alemán-Flores, M., Leibovic, K.N., Moreno Díaz jr, R.: A computacional Model for Visual Size, Location and Movement, Springer Lectura Notes in *Computer Science*, Vol 1333. Springer-Verlag. Berlin Heidelberg New York (1997) 406-419

Quesada-Arencibia, A., Moreno-Díaz jr, R., Alemán-Flores, M., Leibovic, K.N: Two Parallel Channel CAST Vision System for Motion Analysis. Springer Lecture Notes in *Computer Science*, Vol. 2178. Springer-Verlag. Heidelberg New York (2001) 316-327

Quesada-Arencibia, A.: Un Sistema Bioinspirado de Análisis y Seguimiento Visual de Movimiento. *Doctoral Dissertation. PhD Thesys*. Universidad de Las Palmas de Gran Canaria (2001)

J.C. Rodríguez Rodríguez, A.Quesada-Arencibia, R.Moreno-Díaz jr, and K.N. Leibovic: *On Parallel Channel Modelling of Retinal Processes* Vol 2809 Springer-Verlag. Berlin Heidelberg New York (2003) 471-481

Leibovic, K.N., *Science of Vision*, Springer Verlag, New York, 1990.

Vision Systems: Applications

Edited by Goro Obinata and Ashish Dutta

ISBN 978-3-902613-01-1

Hard cover, 608 pages

Publisher I-Tech Education and Publishing

Published online 01, June, 2007

Published in print edition June, 2007

Computer Vision is the most important key in developing autonomous navigation systems for interaction with the environment. It also leads us to marvel at the functioning of our own vision system. In this book we have collected the latest applications of vision research from around the world. It contains both the conventional research areas like mobile robot navigation and map building, and more recent applications such as, micro vision, etc.The fist seven chapters contain the newer applications of vision like micro vision, grasping using vision, behavior based perception, inspection of railways and humanitarian demining. The later chapters deal with applications of vision in mobile robot navigation, camera calibration, object detection in vision search, map building, etc.

How to reference

In order to correctly reference this scholarly work, feel free to copy and paste the following:

J.C. Rodriguez-Rodriguez, A. Quesada-Arencibia and R. Moreno-Diaz jr (2007). Industrial Vision Systems, Real Time and Demanding Environment: a Working Case for Quality Control, Vision Systems: Applications, Goro Obinata and Ashish Dutta (Ed.), ISBN: 978-3-902613-01-1, InTech, Available from: http://www.intechopen.com/books/vision_systems_applications/industrial_vision_systems__real_time_and_demanding_environment__a_working_case_for_quality_control

INTECH
open science | open minds

InTech Europe
University Campus STeP Ri
Slavka Krautzeka 83/A
51000 Rijeka, Croatia
Phone: +385 (51) 770 447
Fax: +385 (51) 686 166
www.intechopen.com

InTech China
Unit 405, Office Block, Hotel Equatorial Shanghai
No.65, Yan An Road (West), Shanghai, 200040, China
中国上海市延安西路65号上海国际贵都大饭店办公楼405单元
Phone: +86-21-62489820
Fax: +86-21-62489821