World Academy of Science, Engineering and Technology
International Journal of Computer and Information Engineering
Vol:1, No:5, 2007

Performance Evaluation of Compression Algorithms for Developing and Testing Industrial Imaging Systems

Daniel F. Garcia, Julio Molleda, Francisco Gonzalez, and Ruben Usamentiaga

***Abstract*—**The development of many measurement and inspection systems of products based on real-time image processing can not be carried out totally in a laboratory due to the size or the temperature of the manufactured products. Those systems must be developed in successive phases. Firstly, the system is installed in the production line with only an operational service to acquire images of the products and other complementary signals. Next, a recording service of the image and signals must be developed and integrated in the system. Only after a large set of images of products is available, the development of the real-time image processing algorithms for measurement or inspection of the products can be accomplished under realistic conditions. Finally, the recording service is turned off or eliminated and the system operates only with the real-time services for the acquisition and processing of the images. This article presents a systematic performance evaluation of the image compression algorithms currently available to implement a real-time recording service. The results allow establishing a trade off between the reduction or compression of the image size and the CPU time required to get that compression level.

***Keywords*—**Lossless image compression, codec performance evaluation, grayscale codec comparison, real-time image recording.

I. Introduction

CURRENTLY, the utilization of machine vision systems to measure or control the quality of the manufactured products is more and more essential. In many cases, the operational conditions of the machine vision system in a real manufacturing line can be emulated in a laboratory with high accuracy. For example, this is the case of an inspection system for electronic circuit boards. But in other cases, the big size and the high temperature of the products to be inspected, as well as the manufacturing process of the products, do not allow the development and the tuning of a machine vision system in the laboratory. A typical example of this situation appears in the rolling of steel strips or other metals.

The Fig. 1 shows a simplified scheme of a system to measure the 3D shape of each steel strip processed in a rolling mill. The flatness of the steel strip can be calculated from the 3D shape. The flatness is one of those essential properties that define the final quality of the steel strip.

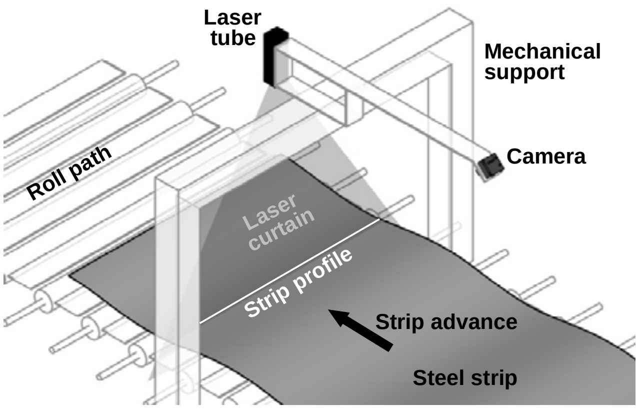

Fig. 1 Scheme of a 3D shape meter

The system uses a laser curtain projected perpendicularly to the steel strip. The intersection of the laser curtain with the strip draws the strip profile.

While the strip advances, a 2D camera is taking images of the strip profiles. An algorithm extracts the profile from the other elements of the image and calculates the height corresponding to each pixel, converting the strip profile in a height profile. The union of the successive height profiles allows the composition of a 3D image of the shape of the strip. In [1], additional details of this system can be found.

The entire development of a machine vision system like this, totally in a laboratory, is practically impossible for multiple reasons.

Firstly, the system can be used to inspect cold or hot rolled strips. The hot strips have a color, which varies from a dark gray, for not very hot strips (≈400ºC), to brilliant white, for

Manuscript received June 8, 2007. This work was partially supported by the Spanish Ministry for Education and Science with a grant for collaboration in research tasks.

Daniel F. Garcia is with the Department of Informatics, University of Oviedo, Campus de Viesques, 33204 Gijon, Spain (phone: +34-985-18-2066; fax: +34-985-18-1986; e-mail: dfgarcia@uniovi.es).
Julio Molleda is with the Department of Informatics, University of Oviedo, Campus de Viesques, 33204 Gijon, Spain (e-mail: jmolleda@uniovi.es).
Francisco Gonzalez is studying Informatics at the University of Oviedo, Campus de Viesques, 33204 Gijon, Spain (e-mail: UO80939@uniovi.es).
Rubén Usamentiaga is with the Department of Informatics, University of Oviedo, Campus de Viesques, 33204 Gijon, Spain (e-mail: rusamentiaga@uniovi.es).

very hot strips (≈1300ºC). The speed of the strip varies from 0 to 25 meters/second. The realistic reproduction of these conditions in the laboratory is totally impossible. The cold strips have a color, which varies from very light gray to very dark gray. In a same strip, the color of the center can be light gray while the color of the borders is dark gray. From time to time, big spots of oil and oxidized areas of very dark colors appear randomly on the surface of the strips. Furthermore, in rainy days, wet areas appear on the strips, which also change drastically the reflection of the light. The proper reproduction of these conditions in the laboratory is almost impossible.

Therefore, the development and optimization of the image processing algorithms can not be accomplished with videos taken in a laboratory, but with videos and signals taken in the place of installation of the system in the real manufacturing line.

If the system can not store the video (sequence of captured images), but it only stores the heights, the checking of the effect of a change of the image processing algorithm or a modification of its configuration parameters could only be accomplished comparing the results of the strips rolled before the change with the results of strips rolled after the change. But obviously, the comparison must be based on different strips.

If the system could be capable of storing video, the processing of the video could be emulated later in the laboratory many times, changing the image processing algorithm or changing the configuration parameters of a same algorithm.

Therefore, it is necessary to include a recording service of video and signal in these measurement and inspection systems, which allows to emulate all the operation of the system in the laboratory.

The main problem that arises when trying to implement this service is the storage size required for the videos. The real problem is that the system should store not just a few videos, but it must remain capturing videos for long periods of time. Once, we have a large set of videos, it is easy to select videos of strips with very different characteristics of color, width, etc. Furthermore, a large set of videos allows the utilization of techniques for tuning the parameters of the image processing algorithms, which requires many examples, like neural networks or genetic algorithms.

To obtain an optimal recording service, we have to evaluate which are the best video container and the best video codec for this type of services.

II. Analysis of Video Containers and Codecs to Storage Industrial Images

The first element to select in order to store a video jointly with other signals is a multimedia container. The second step is to select the most appropriate codec to compress the video stream. The selection of the codec must be done as a function of the container selected previously, because, although many codecs can be used with any container, other codecs can only be used with specific containers.

A multimedia container is a file which stores information of several types (video, audio, text, etc). The most common use of containers is to store movies or music, but they can also be used for less conventional tasks, like storing a stream of images jointly with a stream of a signal samples. To save storage space, the streams are compressed before storing them in a container and the streams must be decompressed before reusing them again. These tasks are carried out by a hardware/software component called coder-decoder. The compressed format of a stream generated by a coder-decoder is called simply a codec. The containers considered in this work are:

- AVI (Audio Video Interleave), [2].
- ASF (Advanced Streaming Format), [3].
- QuickTime, [4].
- MPEG-4 Part 14, [5] [6].
- OGG/OGM, [7].
- Matroska, [8].

Any of these containers can be used to develop the recording service. However, the AVI container can store most of the common codecs, while the other containers are more oriented to the storage of specific codecs which provide special functionalities to reproduce or edit movies. Besides, the AVI container is the most extended and easiest to program, so it is the most appropriate to develop the recording system proposed in this research work.

Next, we are to consider the selection of an appropriate video codec. Most of video codecs have been designed to compress movies. Their main objective consists in reducing the storage size of the frames eliminating the unnecessary information. For the movies, the unnecessary information consists of the visual details that can not be appreciated by the human eye, so that, its elimination can not be noted by the viewer of the video.

This type of video codecs is not useful to fulfill the objectives of a recording service, because the video will not be processed by the human eye, but by a computer system, to extract information from each image (frame) of the video. Therefore, any degradation (elimination) of information from the frames can lead to incorrect results when the frames are processed.

We have to use lossless codecs. They apply compression algorithms similar to the ones used in WinRAR or WinZIP programs to compress files. These algorithms reduce the size of the original file but also allow an exact reconstruction of the original information.

As expected, the compression algorithms with information loss have compression rates notably higher than the lossless algorithms. In fact, the final size of a video compressed with a codec that allows loss of information could be more and more reduced at the expense of reducing more and more the quality of the compressed video.

Another important aspect to consider in the selection of a codec is the type of images that it can process: color images (RGB or YUV) or grayscale images. This work is basically focused on codecs specifically designed to operate with grayscale images, because they are the common images used in the real-time measurement and inspection systems.

From [9], a complete list of codecs capable of working with the AVI container can be retrieved, but only 8 codecs can compress the images without loss of information. Therefore, the codecs considered in this work are:

- ASLC – AlparySoft Lossless Codec, [10].
- FFV1 – FFMPEG Codec, [11].
- HFYU – Huffman Lossless Codec, [12].
- LZO1 – Lempel-Ziv-Oberhumer Codec, [13].
- LEAD – Lead MCMP-MJPEG Codec, [14].
- PIMJ – Pegasus Lossless JPEG, [15].
- ZLIB – Lossless Codec Libraries [16].
- MSZH – Lossless Codec Libraries [16].

The first four codecs, ASLC, FFV1, HFYU, and LZO1 can only work with color images, and therefore, their utilization with grayscale images, although possible, would be very inefficient.

The LEAD codec is an adaptation for video of the JPEG standard for static images. This codec compresses each frame individually without using inter-frame compression in order to avoid the loss of information. It has been designed to work with color images (RGB of 24 bits) and grayscale images (8, 12 and 16 bits).

The PIMJ codec allows the lossless compression of color (RGB of 24 bits) and grayscale (8 bits) images, using a predictor 1 Lossless JPEG technique.

The ZLIB codec is included in the Lossless Codec Library (LCL) which allows the compression and decompression of images and 3D animations. It can work with color images (RGB and YUV) and grayscale images. The compression level can be configured by the user, allowing the operation in two modes: HiSpeed (High speed and low compression) and HiCompress (High compression and low speed). This codec uses the Deflate technique, which combines Huffmann trees with LZ77 compression. The objective of those trees is to codify each item of the information to be compressed using a code of less than 15 bits. The LZ77 compression avoids writing several identical items consecutively, by writing only the first item and indicating the number of identical items that follow the first one.

The MSZH codec is also included in the Lossless Codec Library. It can work with color and grayscale images.

Before starting the performance evaluation work, we searched for the available documentation on video codec comparison. An extensive comparison of the performance of lossless codes can be found in [17], but all the evaluation has been carried out using RGB and YUV color spaces. There is no information about the performance compression of grayscale videos. General information can be retrieved from [18], but there is also a lack of comparison for grayscale videos. Finally, only a very brief comparison of codecs using grayscale videos can be found at the end of the presentation of Takamura [19]. The lack of information about the performance of lossless compression codecs when they operate with grayscale and binary images has motivated the evaluation work presented in the next sections of this paper.

III. Performance Evaluation Methodology

In order to develop the performance evaluation experiments we follow the well-known systematic procedure proposed by Raj Jain [20]. The first step consists in the selection of appropriate performance metrics. In the second step, the factors that could affect those performance metrics must be defined. Next a performance evaluation experiment must be carried out for each relevant combination of the values of the factors. All these considerations are summarized in the Fig. 2 in a graphical form.

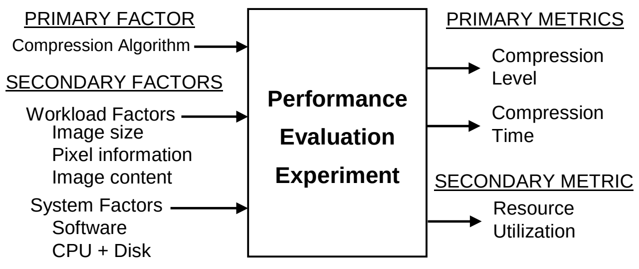

Fig. 2 Factors that affect the output performance metrics of performance evaluation experiments

The most important performance metric for the evaluation of the compression algorithms is the compression level obtained for a sequence of images. The compression level can be expressed by the relation between the size of the compressed images and the size of the correspondent original images. Therefore, the lower this relation is the higher compression is obtained from the algorithm. This metric is always between 0 and 1.

On the other hand, the achieved compression level is obtained consuming CPU time. In general, a high compression level also requires a high CPU time. The execution time of the compression algorithm, equivalent to the consumed CPU time, can be used as a metric indicative of the computational cost of the compression. Generally, it is considered as a secondary metric in non real-time systems, but it is also an essential metric when the compression algorithms must operate in real-time.

Under real-time operational conditions, an appropriate trade off between the two metrics is of primary importance. Therefore a system designer must consider the desired compression level and the CPU time required to compress the images at the same time.

These two metrics are the main outputs of any performance evaluation experiment. However, it is also interesting to analyze secondary metrics, like the resource consumption (%CPU and %Disk) obtained with each compression algorithm, in order to know their compression efficiency.

There are several factors that affect these metrics. Evidently, the primary factor that affects the metrics is the compression algorithm used. There are also other secondary factors that can also affect the output performance metrics. These factors can be classified in system factors and workload factors. The main system factors are the implementation of the algorithm (software factor) and the CPU used to execute the algorithm (hardware factor). The main workload factors are related to characteristics of the processed images, such as their resolution (width x height in pixels), the information contained in each pixel (1 bit for black and white images or 1 byte for 256 grayscale images), and the content of the images.

The experimental design must be focused on evaluating the impact of the primary factor on the two performance metrics selected. Therefore, the values of the secondary factors must be fixed to one or two usual values and these values must not be modified at any time during the evaluation experiments.

In relation with the secondary workload factors, the dimension of the images is 640x260 pixels. For the pixel information two values were used: images of 256 gray levels and binary images, in which the levels can only take the values 0 and 255. The content of the images depends on the gray levels considered: Fig. 3 shows the typical content of the grayscale images used in the performance evaluation experiments and Fig. 6 shows the correspondent binarization of that content.

The secondary system factors include the software implementation of the compression algorithms and the hardware used to execute them. The implementations of the algorithms were obtained from the web references indicated previously and they were integrated in a service that acquires, compresses and writes the images to an AVI container. The service operates on the Windows XP operating system and was executed on a Pentium D, 3.0 GHz. The hard disk used to store the signals is a Serial ATA II, 7200 rpm.

The recording frame rate used in the experiments depends on the pixel information: 10 frames/second for grayscale images and 100 frames/second for binary images.

The high-resolution performance counter provided by the computer hardware was always used to measure the times taken by the recording service to record an image. Its resolution is under 1 microsecond, which allows measuring the execution times with enough accuracy.

IV. Experimental Results

All the experimental results obtained have been classified as a function of the pixel content, that is, there is a set of results for grayscale images and other set for binary images. They are explained in the next two independent subsections.

A. Experimental Results with Grayscale Images

Fig. 3 shows an example of a typical grayscale image containing a laser line, which draws the profile of a steel strip. Under the line, the reflection of light on a roll, over which the strip moves, can be appreciated.

This type of scene is very common in the measurement and inspection systems of continuous or very long products, which are manufactured in many industries (steel, textile, paper, etc.) Therefore, the videos used in the evaluation experiments are representative of the videos that must be processed by this type of industrial imaging systems.

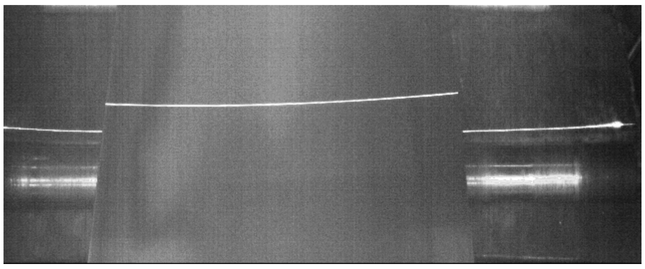

Fig. 3 Example of grayscale image used in the performance evaluation experiments

Fig. 4 shows the results obtained from compression algorithms operating with grayscale images. It is important to analyze the results in relation to the absence of compression, defined by a black circle in Fig. 4. This circle indicates that the recording of the frames without compression takes an average time of 1.23 milliseconds and the reduction of the image size is 1, indicating this, the absence of reduction.

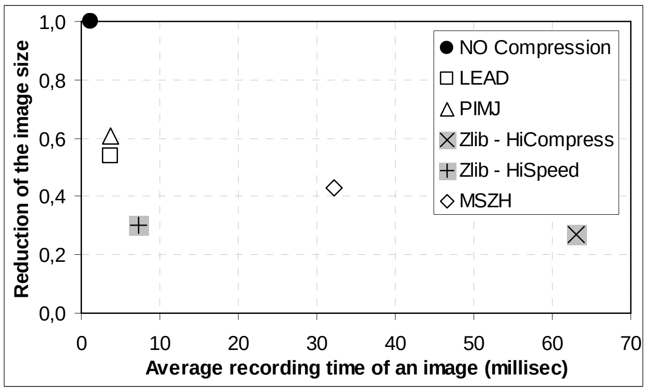

Fig. 4 Performance of compression algorithms with grayscale images

The LEAD and PIMJ algorithms behave in a similar manner. They can almost reduce the size of the images to half of their original size, but the recording time is multiplied by three. A better compression can be obtained with the ZLIB algorithm configured to operate in the HiSpeed mode. It can reduce the sizes of the images to a third of the original size at the expense of multiplying the recording time by six.

Finally the MSZH and Zlib-HiCompress algorithms can not operate at recording frame rates over 100 fps. The MSZH algorithm provides less compression and requires a notably longer time than the Zlib-HiSpeed algorithm, as can be appreciated in Fig. 4. The Zlib algorithm, configured in HiSpeed mode, reaches a reduction factor of 0.30, only a little bit higher than the factor obtained when it is configured in

HiCompress mode, of 0.27. However, Zlib-HiSpeed is about 8 times faster than Zlib-HiCompress and reaches nearly the same compression. Therefore, considering the relation between the compression and the execution time, the MSZH and Zlib-HiCompress algorithms can not be considered appropriate to implement an efficient real-time recording service.

Next, we compare the efficiency of the compression algorithms using the available computational resources. The Table I shows the percentage of CPU and Disk utilization as a function of the compression algorithm used in the recording service when a period of 100 milliseconds is used to store the grayscale images.

TABLE I
UTILIZATION OF DEVICES WITH GRAYSCALE IMAGES

Compression Algorithm	% CPU Utilization	% Disk Utilization
NONE	2.94	3.54
LEAD	2.99	1.88
PIMJ	3.09	2.19
Zlib-HiCompress	63.05	0.74
Zlib-HiSpeed	3.56	1.17
MSZH	32.78	2.12

For a better comparison of the resources required by the algorithms to compress and store the images, the CPU utilization measured for each compression algorithm is divided by the CPU utilization measured when no compression is used. The same procedure is used with Disk utilization. The relative utilizations of CPU and Disk are represented in Fig. 5, where the point (1,1) represents the lack of compression, and it is the reference point for comparing all the compression algorithms.

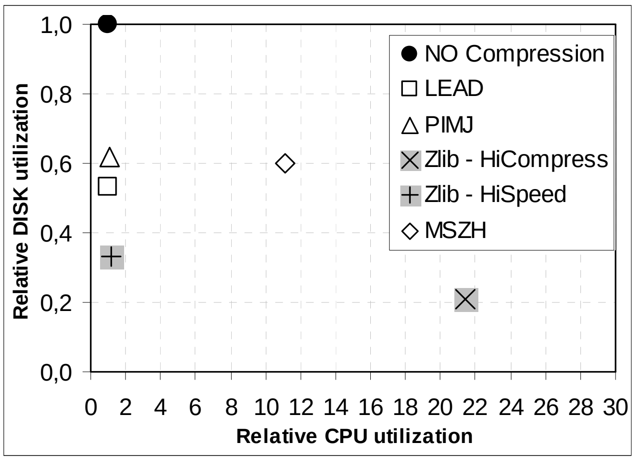

Fig. 5 Relative CPU and Disk utilizations for grayscale images

Fig. 5 shows that the relative disk utilization decreases proportionally to the reduction of the image size (shown in Fig. 4). However, the relative CPU utilization does not increase with the algorithms LEAD, PIMJ and Zlib-HiSpeed. The only explanation for this behavior is that the increment of CPU utilization due to the execution of the compression algorithm is compensated with the decrement of CPU utilization required to store the compressed images in the disk. The Zlib-HiSpeed algorithm shows the best performance. It gets the maximum possible compression without increasing the CPU utilization noticeably.

B. *Experimental Results with Binary Images*

Fig. 6 shows the typical binary images provided by the frame grabber card, when hardware binarization is activated.

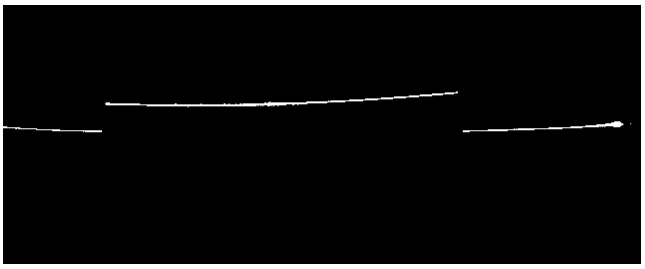

Fig. 6 Example of binary image used in the performance evaluation experiments

Fig. 7 shows the results obtained from compression algorithms operating with binary images. With these images, the recording of each frame without compression takes an average time of one millisecond. All algorithms reduce the size of the images very much, with small increments of the recording time.

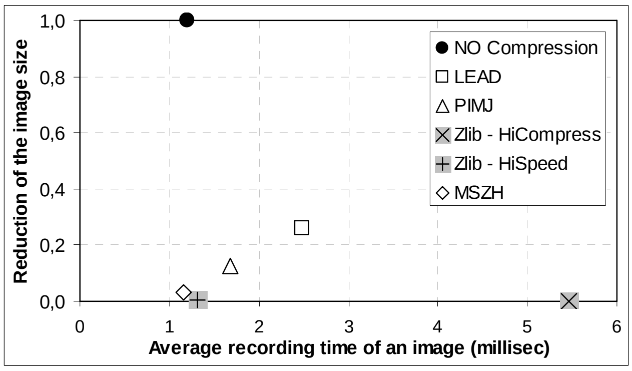

Fig. 7 Performance of compression algorithms with binary images

A surprisingly good performance is obtained for the Zlib algorithm. The reduction of the image size is very high, 0.002 when it operates in HiCompress mode and 0.005 when it operates in HiSpeed mode. However, when it operates in HiSpeed mode the increment of recording time is negligible.

The other three compression algorithms can be easily ordered by performance, because MSZH reduces the image size very much using a short recording time and LEAD reduces the image size very little taking a longer recording time. PIMJ shows an intermediate behavior between the other two algorithms.

Now, we also compare the efficiency of the compression algorithms using the computational resources consumed by them. Table II shows the percentage of CPU and Disk utilization as a function of the compression algorithm used in the recording service when a period of 10 milliseconds is used to acquire and store the binary images.

TABLE II
UTILIZATION OF DEVICES WITH BINARY IMAGES

Compression Algorithm	% CPU Utilization	% Disk Utilization
NONE	3.62	17.95
LEAD	4.21	5.95
PIMJ	3.80	3.68
Zlib-HiCompress	13.71	0.76
Zlib-HiSpeed	3.45	0.32
MSZH	3.21	1.32

The relative utilization of CPU and Disk are shown in Fig. 8, in which the point (1,1) represents the absence of compression, and it is the reference point for comparing all the compression algorithms.

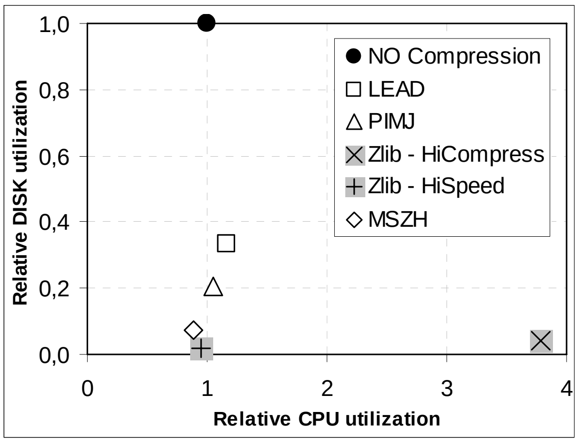

Fig. 8 Relative CPU and Disk utilizations for binary images

Fig. 8 shows that the relative disk utilization decreases proportionally to the reduction of the image size (shown in Fig. 7). However, the relative CPU utilization does not increase with the algorithms LEAD, PIMJ, Zlib-HiSpeed and MSZH. The justification for this behavior is that the increment of CPU utilization, due to the execution of the compression algorithm, is compensated with the decrement of CPU utilization required to store the compressed images in the disk. The Zlib-HiSpeed algorithm shows the best performance. It gets the maximum compression possible without increasing the CPU utilization.

V. CONCLUSION

In this work, we have presented the impossibility of developing complex image processing systems totally in the laboratory. We have proposed to develop firstly the image acquisition service, and then include an efficient image recording service that allows the development of the image processing services of the global system in a laboratory.

The key aspect of the design of the recording service is the proper selection of the codec to compress the images. The performance evaluation experiments show that the Zlib-HiSpeed algorithm is the best suited to develop the recording service for the two types of images considered: grayscale and binary.

The advantages of using an efficient image recording service to develop and tuning the image processing algorithms systems are clear: the great number and the high representativeness of the images that compose the training set of the image processing algorithms.

Finally, just to remark that the performance comparison of video codecs based on grayscale and binary images presented in this work is the only comparative information currently available for the designers of industrial imaging systems.

REFERENCES

[1] C. Lopez, D.F. Garcia, R. Usamentiaga and J.A. Gonzalez, “Real-time system for flatness inspection of steel strips,” in Proc. 17th Int. Symp. on Electronic Imaging: Machine Vision Applications in Industrial Inspection XIII, San Jose, CA, 2005. SPIE Vol.5679, pp.228-238.
[2] Microsoft, “Specification of the AVI container,” Available: http://windowssdk.msdn.microsoft.com/en-us/library/ms779636.aspx
[3] Microsoft, “Specification of the ASF container,” Available: http://go.microsoft.com/fwlink/?LinkId=31334
[4] Apple, “Documentation of the QuickTime container,” Available: http://developer.apple.com/documentation/QuickTime/QTFF/index.html
[5] Chiariglione, “MPG-4 file formats white paper,” Available: http://www.chiariglione.org/mpeg/technologies/mp04-ff/
[6] ISO, “Standard ISO/IEC 14496-14 MP4 file format,” Available: http://www.iso.ch/iso/en/prods-services/
[7] Xiph, “Documentation of the OGG container,” Available: http://www.xiph.org/ogg/
[8] Matroska, “Specification of the Matroska container,” Available: http://dl.matroska.org/downloads/libmatroska/
[9] Fourcc, “Video codec and pixel format definitions,” Available: http://www.fourcc.org
[10] AlparySOFT, “AlparySoft lossless video codec,” Available: http://www.alparysoft.com/products.php?id=8&item=35
[11] SourceForge, “DirectShow and VFW FFMPEG codec,” Available: http://sourceforge.net/projects/ffdshow
[12] Berkeley, “Description of the Huffman codec,” Available: http://neuron2.net/www.math.berkeley.edu/benrg/huffyuv.html
[13] Oberhumer, “Libraries of the LZO codec,” Available: http://www.oberhumer.com/opensource/lzo/
[14] LeadCodes, “Description of the MCMP-MJPEG codec,” Available: http://www.leadcodecs.com/codecs/
[15] PegasusImaging, “The Pegasus lossless JPEG codec,” Available: http://www.pegasusimaging.com/pvlosslessfeatures.htm
[16] Geocities, “Libraries of ZLIB and MSZH lossless codecs,” Available: http://www.geocities.co.jp/Playtown-Denei/2837/prg/LCL223.ZIP
[17] D. Vatolin, I. Seleznev, M. Smirnov, “Lossless Video Codecs Comparison’2007,” Technical Report of the Graphics & Media Lab (Video Group) of Moscow State University (MSU). Available: http://www.compression.ru/video/codec_comparison/index_en.html
[18] CompressionLinks, “Lossless Video Codecs Area,” Available: http://www.compression-links.info/Lossless_Video_Codecs
[19] S. Takamura, “Lossless Video Coding,” Lecture of the Course EE398B on Image Communication at Stanford University. Available: http://www.stanford.edu/class/ee398b/handouts/lectures/LosslessVideoCoding.pdf
[20] R. Jain, *The Art of Computer Systems Performance Analysis: Techniques for Experimental Design, Measurement, Simulation and Modeling.* New York: John-Wiley & Sons, 1991.