EFDA–JET–PR(11)31

M. Portes de Albuquerque, M. P. de Albuquerque, G. Chacon,
E.L. de Faria, A. Murari and JET EFDA contributors

High Speed Image Processing Algorithms for Real Time Detection of MARFEs on JET

High Speed Image Processing Algorithms for Real Time Detection of MARFEs on JET

M. Portes de Albuquerque[1], M. P. de Albuquerque[1], G. Chacon[1],
E.L. de Faria[1], A. Murari[2] and JET EFDA contributors*

JET-EFDA, Culham Science Centre, OX14 3DB, Abingdon, UK

[1]*Centro Brasileiro de Pesquisas Físicas, CBPF/MCT, Rua Dr. Xavier Sigaud 150 Urca,*
Rio de Janeiro 22290180, Brazil.
[2]*Consorzio RFX-Associazione EURATOM ENEA per la Fusione, I-35127 Padova, Italy.*

** See annex of F. Romanelli et al, "Overview of JET Results",*
(23rd IAEA Fusion Energy Conference, Daejon, Republic of Korea (2010)).

Preprint of Paper to be submitted for publication in
IEEE Transactions on Plasma Science

ABSTRACT

This work presents a high-speed image processing algorithm applied to MARFEs detection on JET. The algorithm has been developed in C/C++ language using the OpenCV library for image processing, the LibSVM library for pattern classification and has been tested on a dedicated Intel 64 bits Linux computing platform. The code implemented achieves at the same time high accuracy and good run-time performance. The final version presents a correct detection rate of 93.3% and an average image processing rate of 650 frames per second. A complete analysis of each image processing module is presented to illustrate the overall characteristics and the performance of the algorithm.

1. INTRODUCTION

Cameras and digital image processing have become important tools in many types of scientific instrumentation. Several techniques for acquisition, segmentation and classification of patterns can be used for better understanding, characterization and control of physical phenomena. In some applications, the information processing chain has to be carefully designed aimed at the execution of quite sophisticated algorithms with very high precision and in real time.

This tendency of relying more on image measurements is particularly evident in Magnetic Confinement Nuclear Fusion (MCNF) [1], in which many more cameras, both visible and InfraRed (IR), have become routine diagnostics in the last years. An important example is the Joint European Torus, the largest fusion device in operation worldwide; in the last campaigns many experiments relied on IR and/or visible cameras and about 15 new cameras are being installed for the next experiments with the new metallic wall. One of the most challenging characteristics of cameras as scientific instruments is the large amount of data that they produce. Again in JET, for example, out of a global data base of more than 90 terabytes of data, about 50% are images [2, 3]. In MCNF cameras have become very important not only for the interpretation of the experiments but also for the feedback control of the discharges. Visible cameras are used to detect instabilities and transport of impurities into the plasma [4]. They have found very useful applications also in the diagnostics of pellets; small sphere of frozen hydrogen or deuterium injected into the plasmas for fuelling and control purposes.

In this paper we describe an image processing algorithm to process JET videos in high speed. We present also its main characteristics and discuss its performance in terms of accuracy and execution time. In this context, accuracy is the system error rate in identifying objects, whereas execution time is the measure of the system speed and of its constituent parts.

When developing algorithms for image processing and pattern recognition, it is usually necessary to establish the criteria for assessing performance. One of the main questions that arise in the evaluation of computer algorithms for data analysis is [5]: is there a data set for which the correct answers are known? In order to evaluate different image processing algorithms, a data base of videos has been built in JET. It comprises 22 videos with identical optical settings for a total of 4,236 image regions, consisting of: 718 (16.9%) MARFEs, 3,373 (79.7%) Non-MARFEs and 145 (3.4%) other patterns that have morphological characteristics at the border between the first two and could be classified as either. The algorithm proposed in this paper used this data base as a reference to image analysis. This image data base contains the following information for each region: all invariant moments, x and y coordinates of its barycenter, pattern classification, frame number and video name (more details are given in the following sections).

2. MARFE INSTABILITY

The Multifaceted Asymmetric Radiation From the Edge (MARFE) is an instability, which appears in tokamaks as a toroidal ring of increased radiation. It is typically located along the inner wall and can move from the upper part of the divertor and around the X-point up to the top of the machine [6]. The MARFE is a radiation instability, in which the plasma is locally cooled by radiation in a range of parameters such that this cooling leads to increased radiation and to further cooling. The main reason for this positive feedback resides in the increase of emission of radiation by impurities with decreasing temperature. Since MARFEs cause a significant increase in impurity radiation, they leave a quite clear signature in the frames taken by visible cameras. A sequence of frames of JET fast visible camera, showing the top down movement of a MARFE, is reported in Figure 1.

The results on MARFEs presented in this paper are based on the data recorded with the visible camera that is installed on JET, the wide angle fast visible camera. This camera is a Photron APX-RS which utilizes a CMOS detector [7]. The camera is installed on the same Cassegrain endoscope dedicated to IR thermography, Figure 2.

The camera has a maximum frame size of 1024×1024 pixels at 3,000 frames per second; however the frame size can be reduced allowing faster capture up to 250,000 frames per second. The PNG output image format of the camera has been used as the input to our code to develop, run and characterize the algorithm described in this paper. Since in our application the information to be retrieved resides in the patterns detected by the camera, it is sufficient to analyze the luminosity of the various pixels and no special calibration has to be considered. An algorithm, designed to detect MARFE in video frames, has to be able to distinguish them from other kinds of objects appearing in the field of view of the camera. In the case of MARFEs, unfortunately they tend to have a shape which can be confused with the one of ELMs (Edge Localized Modes), instabilities, which appears in the videos as a global sudden increase of luminosity. The locations of the MARFEs, close to the inner wall, and their particular movement from top to bottom are characteristics that have been exploited to identify them, as described in the next sections.

Since the fast visible camera grabs an image with a rate of 30,000 frames per second, the processing algorithm should be able to read a gray level image (224 width and 256 height), process it, recognize if a MARFE is present and return an alarm to the Main Control Unit within a maximum interval of 33μs. The treatment of all these images within this period is a considerable technological challenge in the field of real time image analysis. Any effective technical solution should strike the right balance between acceptable image processing rates and maintaining a high success rate of MARFE recognition.

With the aim of characterizing the time evolution of MARFEs as seen by JET fast visible camera, Figure 3 indicates the MARFE duration in ms (and the equivalent total number of frames) obtained from the aforementioned data base. The average time duration of a MARFE is 43.15 ± 14.67 frames (1.438 ± 0.489ms) with a maximum time of 65 frames (2.17ms) and a minimum of 3 frames (0.1ms).

3. IMAGE PROCESSING MODULES

The image processing system is divided into a series of modules, each aimed at a specific treatment, as shown in Figure 4 [8]. The Time Analysis and Performance Analyzer modules are dedicated to the measurement of the code execution time and accuracy respectively and will be detailed in section IV. We describe in the sequel the implementation of the image processing chain.

Seven modules have been developed in C/C++ language for Linux O.S. Two main libraries have been used to assemble the entire image processing system: OpenCV and LibSVM. OpenCV stands for Open Source Computer Vision and is a library of programming functions for real time computer vision and image processing [9, 10]. LIBSVM is an integrated software package for pattern classification and provides a simple interface that can be linked with our own code. The main features of LibSVM are efficient multi-class classification and cross validation for model selection, implemented in C++ [11].

3.1. OPEN IMAGE MODULE

This module is responsible for opening the image stored on disk and checking it for size compatibility (224×256 pixels), necessary for the successive treatment. With the goal of testing the complete chain, a total of 12,000 images have been stored and processed. In order to build a concise test platform in the Linux O.S., we allocated a portion of the RAM memory as a mounted file system. Normally, files and directories are stored on hard disk drives, which provide a lot of space at slow data transfer rates (between 80MB/s−250MB/s). Ram disks are virtual file systems (unlike Hard Disk Drivers which are hardware) that live completely inside the computer's RAM. They offer significantly higher data transfer rates (1300MB/s−3200MB/s) at the cost of volatility and space (limited by the amount of RAM memory installed on the system, including swap space).

All images of the fast visible camera have been copied to a ram disk temporarily to test the present program. However, even if stored in ram disk memory, the processing for opening the image file is executed as if it is accessing a normal file system, making the algorithm spend much of the time performing I/O equivalent routines.

After the opening and reading procedure, a memory allocation function is executed using the OpenCV structure to encode the image data for later successive treatment. This memory allocation deserves special attention when executing the main loop of the processing chain. It is worth mentioning that the final objective is to run the program in continuous mode for several frames and it is essential to keep control of this run time allocated memory.

3.2. BACKGROUND IMAGE ESTIMATION AND BINARIZATION MODULE

One of the main problems in the identification of the MARFE candidates, is the presence of many other events that present the same optical characteristics, such as flashes (probably caused by ELMs), high radiation of the plasma in the divertor or from the poloidal limiters. For this reason, it is not easy to define a general approach that allows identifying and isolating clearly the MARFEs. The proposed approach is therefore based on a first preprocessing stage, consisting of background subtraction; in this way it is possible to identify moving objects in the image. Background subtraction techniques are in general based on the comparison of each frame with a reference or background model. Pixels in the current frame that are different from the background are considered to be moving objects (foreground). Obviously, a successive processing stage is needed in order to localize and track the objects of interest. In the present work, the choice of the background is a very important task and really influences the overall performance of the system. The approach presented here is based on the modeling of the background as a Gaussian distribution. In our approach, an averaging filter [12] is used in order to obtain the background image. This method evaluates the background image B(x,y) as the mean of the previous N frames.

The simplest approach for image segmentation is to check whether the pixels in the current image I(x,y) are significantly different from the corresponding pixel of the estimated background image B(x,y). A threshold T is commonly used to obtain a binary image, as shown in Equation 1:

$$\left| I(x, y) - B(x, y) \right| > T \tag{1}$$

We evaluate the overall performance of the system by considering the correct classification and the number of binary regions analyzed as function of N, as show in figure 5.

Figure 6 shows the optimal threshold level determination of 13 in order to achieve the maximum number of correct classifications.

In order to remove noise or artifacts introduced by the background subtraction phase, a control on the area of the identified objects is applied. It has been found that regions smaller than 70 pixels or bigger than 25% of the image can be safely rejected, since they do not correspond to MARFEs [4].

3.3. FEATURE DETECTION MODULE

This module aims at extracting region characteristics in binary images. The main features are the object barycenter, the region area and the Hu set of invariant moments. In the algorithm presented in this paper, the nonlinear centralized Hu moments have not been used. Although they can increase the overall accuracy of the classification module by a small factor, they also influences the final processing time. Hu moments have been used in other techniques, as described in [4], and could be considered for further refinements of the approach described in this paper.

To build this module, we use the OpenCV library: cvBlobsLib. It performs binary images connected component labeling with two basic functionalities: i) extract 8-connected regions in binary or grayscale images (referred as blobs) and ii) filter the obtained blobs to get their region features. This algorithm is detailed in [13].

3.4. CLASSIFICATION MODULE

The main goal of the classification module is to determine whether a shape belongs to the classes:

MARFE, Non-MARFE or Other. We used the Support Vector Machine Classification (SVM) technique to classify the binary region among these three possible classes. SVMs are a set of related supervised learning methods that analyze data and recognize patterns, used for classification and regression analysis. The optimal separation function is defined as the one with the highest margins with respect to the closest points which are called support vectors.

In SVMs systems there are two stages. The first is dedicated to the training procedure which aims at model definition that contains the information of the support vectors [14]. The second is the pattern recognition itself, which consists of presenting to the classifier its characteristics obtained by the previous modules of the image processing sequence. In this work we use a total of 2,523 samples to define the model (training) and 1,713 for characterizing their performances. From these two set of samples we tested 11,018 different SVM models. The classifier performance for the identification of the three classes was 96.15% (MARFEs, non-MARFEs and Others). When focusing only on MARFE the final success rate was 94.14%. A polynomial kernel function of degree 6 was the best boundary separator function for this analysis.

When using the whole image processing chain, the best result for the classification module was a SVM model based on a polynomial kernel of degree 10. The performance results are detailed in section IV dedicated to the Performance Analysis. The classification module was implemented using the LibSVM library in C++ [11 - LibSVM WebSite]. Figure 8 presents the SVM boundaries of this kernel for 2,523 samples of the image data base, represented here by the barycenter coordinates.

4. PERFORMANCE ANALYSIS

A) COMPUTER PLATFORM FOR PERFORMANCE EVALUATION

We have used Intel C++ Compiler (icc) which is an integral part of C and C++ compilers from Intel Corporation available for GNU/Linux [15]. The icc supports compilation for Intel 32 and 64 bits processors. The OpenCV library version 2.2 has been used to build the main image processing functions. The LibsSVM library version 3.1 (04/2011) has been used as a pattern classification tool. The computer test platform is a Linux cluster node with the following characteristics: Supermicro, Super-Server, 1U rackmount, 2 Motherboard (Mobo) with 8GB of RAM memory each, with 2x Intel CPU Xeon E5430 HarperTown Quad-Core 2666.431MHz (8 cores/Mobo), 6MB of cache memory, 1TB of HD SATAII, O.S.: Unix-Like x86_64 / Ubuntu 10.10, icc version 12.0.3 20110309 and kernel: 2.6.35-22-SMP, x84_64. All results presented in this paper have been obtained with this 64 bits computing platform.

B) ACCURACY

Achieving a high performance image processing system is the principal goal, mainly due to the high image acquisition rates involved in this work. The accuracy of the entire system has been determined using the Confusion Matrix method. A Confusion Matrix contains information about current and predicted classifications, and is commonly used in supervised learning systems. As described in section I, we rely on an image data base previously classified with the expected result for each class (Non-MARFE, MARFE and Other).

The final goal of the Performance Analyzer module is the construction of the Confusion Matrix for various algorithms of image processing and pattern classification. We have tested 10,368 different parameters for the algorithm based on the bests 16 SVM kernels encountered by the Classification module analysis (kernels of polynomial degree 6, 8, 10 and 12 for the SVM polynomial function have been evaluated in these tests). The image processing algorithm has been tested varying the following parts: the number (N) of previous frames for background image estimation; the segmentation types (e.g. thresholding techniques: fixed, Otsu and the one described in section III) for image binarization and the processing of the binary images with morphological techniques of eroding and dilating (for various steps and orders).

In order to check the image processing results with the previously described data base, two aspects of the classical Confusion Matrix must be checked. Some results of the image processing algorithm can lead to regions blobs that are not present in the image data base (x_{DB}), or the image data base can contain regions that have not been found by the processing algorithm (x_{IP}). All results have been finally stored in an Expanded Confusion Matrix, as shown in Figure 9. The values 0, 1 and 2 represent the classes Non-MARFE, MARFE and Others respectively. Column and row 3 indicate results that have not been found in the image data base and by the image processing algorithm respectively. It is worth mentioning that the values in the diagonal of the Confusion Matrix, in Figure 9, (marked in green) show the TN (True Negative) and TP (True Positive) samples for the entire image processing system.

One advantage of the Confusion Matrix is the multiple analyses allowed. In this article, we present only the performance of the overall classification of the processing system and those dedicated to MARFE recognition. The image processing algorithm has found and processed 1,409 regions, resulting in a performance of 93.3% correctly classified, 3.0% of false positives and 3.7% of false negatives, as show in Table 1 and Figure 10.

If we consider only the MARFE samples presented in the test set of the data base (290), we have a total of 80.5% regions correctly detected, 14.4% misclassified and 5.1% not found in the data base. These regions have not been recovered in the data base mainly due to the image treatment used. As expected, the processing may result in fragmented regions that make the classifying task much more difficult.

C) TIME ANALYSIS

Measuring time in computer systems require special attention. In order to benchmark a program, several measurements of elapsed time can be used. In Linux O.S. there exist different time measurements: walltime (time perceived by the user), CPU time (time that the process was actually executed by the processor, excluding time spent by the O.S.), user time (CPU time excluding time spent during system operations triggered by the process, e.g., I/O) and system time (CPU time containing only system operations). In this paper all time measurements refer to walltime, which is an estimate of the total time consumed by the whole system. For more information about measuring computing time in Linux O.S, refer to [16].

In the implementation of the Time Analysis module, we used the Time Stamp Counter technique.

The Time Stamp Counter (TSC) is a 64-bit register present on all ×86 processors. TSC is an excellent high-resolution (12.1 ±0.5ns in this computing platform), low-overhead way of getting CPU timing information. Function RDTSC - Read Time Stamp Counter - returns the number of clock cycles, since the CPU is powered up or reset. Since the computing platform has eight cores, which are automatically allocated by the O.S., from the point of view of the overall performance, it is as if the system consisted of a single and exclusive processor for code execution. We have placed a series of start and stop probes between each module, in order to analyze their execution time performance. In Figure 11, the overall analysis for each module is presented.

Modules Feature Extraction (TFHu), Opening Image (TOp) and Background Image Subtraction (TSAv) together represent 99% of the total run time. The average run time was 1.546 ± 0.12ms with an image processing average frame rate of 650.29 ± 47.92 images per second.

Figure 12 shows the Feature Extraction module execution time separately and the Total execution time for the video KL8_70052V4 (400 frames). This module spends on average 0.675 ± 0.25ms to process each frame. The main reason is due to the scanning of the binary image for regions of white pixels, and then the determination of their sizes and barycenter coordinates. Any effort to eliminate image regions, which are not to be considered for MARFE detection, can improve the overall processing time. We implemented a secondary code based on an image mask, in order to reduce the total processing time. With function cvAnd, we calculate a per-element bit-wise logical multiplication of the binary image with a binary mask. With this operation, the total number of images processed per second is increased by about 5%, c.f. Table 2.

Table 2 shows the complete processing time for all 26 videos used in the time performance analysis.

CONCLUSIONS AND FURTHER DEVELOPMENTS

In this work we present an image processing algorithm that enables the detection of MARFEs at high speeds rates. This technique achieves an average of 650 frames per second with a correct detection rate of 93.3% taking into account all image processing modules. The processing in a real environment still depends on the system settings of image acquisition rate using the high performance camera. Since the procedure for transferring images from camera to computer is not yet defined, we studied also the execution time without considering the Open Image Module. The image processing system without the Open Image Module can treat images with an average rate of 1,050 frames per second (1,120 frames per second if the logical multiplication by an image mask technique is implemented).

In this work, the developed algorithm has been used to process all frames offline, which is equivalent to process all KL8 images sequentially every 33μs. With the current image processing rate, we would reach an analysis of one frame every 46 images in a real time environment. However, the average duration of a MARFE on the data base is 43.15 ± 14.67 frames (or 1.4383 ± 0.489ms). This would lead us to analyze only one image within a period of a MARFE, greatly reducing the chance of detection and classification. Considering the current performance of the classifier (i.e. more than 8 hits in 10, on average), a good compromise would be a real-time analysis of at least ten frames within a MARFE period. This would lead us to a significant acquisition rate of 7,575 frames per second (i.e. period of 4×33µs). Several technological propositions have been made in order to achieve these high processing rates. Among them we can mention the implementation of these codes in electronic devices using Digital Signal Processors (DSPs) and the development of algorithms using parallel programming techniques.

Although we obtained a good rate of correct classification, it is important to improve the image processing system, in order to increase the detection of binaries regions presented to the classification module. This should be done with new image analysis techniques in all the processing chain modules. Finally, we have two comments about the proposed algorithm: i) the repeatability of the process and ii) the variations in test reliability with sample size. With regard to repeatability, we can say that the robustness of the algorithm has been reached mainly thanks to the Background Image Subtraction (SAv) and Feature Extraction (FHu) modules. The SAv technique increased the classification performance to the current values and FHu module control the rejection of small particles that can be derived from variations in lighting conditions or noise present in the images. As far as the variation in test reliability with sample size is concerned, the results of the algorithm performance are strongly related to the available Image Data Base. We processed 4,236 samples from the current JET data base. A new characterization test should be considered using a larger Image Data Base that is being built in JET.

ACKNOWLEDGMENTS

The authors thank the support from the National Council for Scientific and Technological Development (CNPq) and the Brazilian Research and Projects Financing (FINEP) of the Brazilian Ministry of Science and Technology. This work has been carried out under the European Fusion Development Agreement under the Brazil - EURATOM collaboration agreement on fusion research, coordinated by the Brazilian Fusion Network and by the Brazilian Nuclear Energy Commission. The views and opinions expressed herein do not necessarily reflect those of the European Commission.

REFERENCES

[1]. J. Wesson, “Tokamaks”, Clarendon Press, Oxford, 3rd. edition, 2004.

[2]. R. Layne et al, “Fusion Engineering and Design”, vol. **60**, Issue 3, June, pp.333-339, 2002.

[3]. R. Layne et al, “Fusion Engineering and Design”, vol. **85**, Issues 3-4, July, pp.403-409, 2010.

[4]. A. Murari, M. Camplani, B. Cannas, D. Mazon, F. Delaunay, P. Usai, J.F. Delmond, “Algorithms for the automatic identification of MARFEs and UFOs in JET database of visible camera videos”, IEEE Transactions on Plasma Science, vol. **38**, Issue 12, pp.3409-3418, December, 2010.

[5]. N.A. Thacker, A.F. Clark, J.L. Barron, J. Ross Beveridge, P. Courtney, W.R. Crum, V. Ramesh, and C. Clark, “Performance characterization in computer vision: A guide to best practices”. Computer Vision and Image Understanding, vol. **109**, Issue 3, pp.305-334, March, 2008.

[6]. H.B. Lipschultz, B. LaBombard, E.S. Marmar et al., “MARFE: An Edge Plasma Phenomenon”, Nuclear Fusion, vol. **24**, Issue 8, pp. 977-988, 1984.

[7]. J.A. Alonso et al., "Fusion and Plasma Physics" (Warsaw, Poland), In Proceedings of the 34th EPS Conference on Control, 2007.

[8]. Y. Lucas, A. Domingues, D. Driouchi, and S. Treuillet; "Design of experiments for performance evaluation and parameter tuning of a road image processing chain". EURASIP Journal of Applied Signal Processing, pp.212-212, January, 2006.

[9]. G. Bradski and A. Kaehler, "Learning OpenCV: Computer Vision with the OpenCV Library", 978-0596516130, O'Reilly Media, 2008.

[10]. OpenCV (Open Source Computer Vision) web site: http://opencv.willowgarage.com/wiki/ (2011).

[11]. Chih-Chung Chang and Chih-Jen Lin, LIBSVM library for support vector machines, 2001. Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm

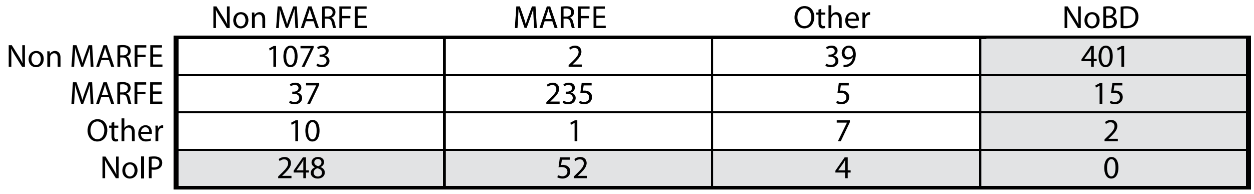
[12]. R. Cucchiara, M. Piccardi, and A. Prati, "Detecting moving objects, ghosts, and shadows in video streams," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 25, Issue:10, pp. 1337-1342, October, 2003.

[13] Fu Chang, Chun-Jen Chen, Chi-Jen Lu, A linear-time component-labeling algorithm using contour tracing technique, Computer Vision and Image Understanding, vol. **93**, Issue 2, pp.206-220, February, 2004.

[14]. B. E. Boser, I. Guyon, and V. Vapnik, "A training algorithm for optimal margin classifiers", In Proceedings of the Fifth Annual Workshop on Computational Learning Theory, pp. 144-152. ACM Press, 1992.

[15]. Intel® C++ Composer XE 2011 for Linux; http://software.intel.com/en-us/articles/intel-ccomposer-xe-documentation/#lin

[16]. J. Corbet, A. Rubini and G. Kroah-Hartman, Linux Device Drivers, pp.183-196, 3rd Edition, O'Reilly Media, 2005.



	Non MARFE	MARFE	Other	NoBD
Non MARFE	1073	2	39	401
MARFE	37	235	5	15
Other	10	1	7	2
NoIP	248	52	4	0

Table 1: Expanded Confusion Matrix for the Test set and for 1,409 regions detected.

	JET Video	Total Images processed	Mean Execution Time (ms)	Frames / s (without Mask)	Frames / s (without Mask)
1.	KL8_70029V2	400	1.596	626.56	645.34
2.	KL8_70032V6	400	1.461	684.61	714.66
3.	KL8_70033V3	400	1.424	702.03	724.44
4.	KL8_70053V3	400	1.654	604.63	648.86
5.	KL8_70053V4	400	1.524	656.28	679.31
6.	KL8_70053V6	400	1.477	677.03	702.79
7.	KL8_70054V3	400	1.569	637.32	667.92
8.	KL8_70054V5	400	1.482	674.65	701.61
9.	KL8_70055V2	400	1.815	550.98	584.01
10.	KL8_70055V3	400	1.537	650.41	671.86
11.	KL8_70055V4	400	1.499	667.12	693.70
12.	KL8_70056V4	400	1.478	676.47	701.65
13.	KL8_70056V5	400	1.630	613.33	652.53
14.	KL8_70097V2	2000	1.844	542.21	580.62
15.	KL8_70050V1	400	1.456	686.97	714.80
16.	KL8_70050V2	400	1.502	665.76	693.87
17.	KL8_70050V3	400	1.438	695.21	716.56
18.	KL8_70050V4	400	1.494	669.47	701.84
19.	KL8_70050V5	400	1.466	681.90	701.28
20.	KL8_70050V6	400	1.474	678.32	701.90
21.	KL8_70052V1	400	1.493	669.93	698.38
22.	KL8_70052V2	400	1.436	696.27	720.30
23.	KL8_70052V3	400	1.455	687.39	717.39
24.	KL8_70052V4	400	1.466	682.10	706.17
25.	KL8_70052V5	400	1.816	550.63	598.14
26.	KL8_70052V6	400	1.724	579.99	613.49

Table 2: Algorithm mean Execution Time and image processing frame rate for 26 JET videos.

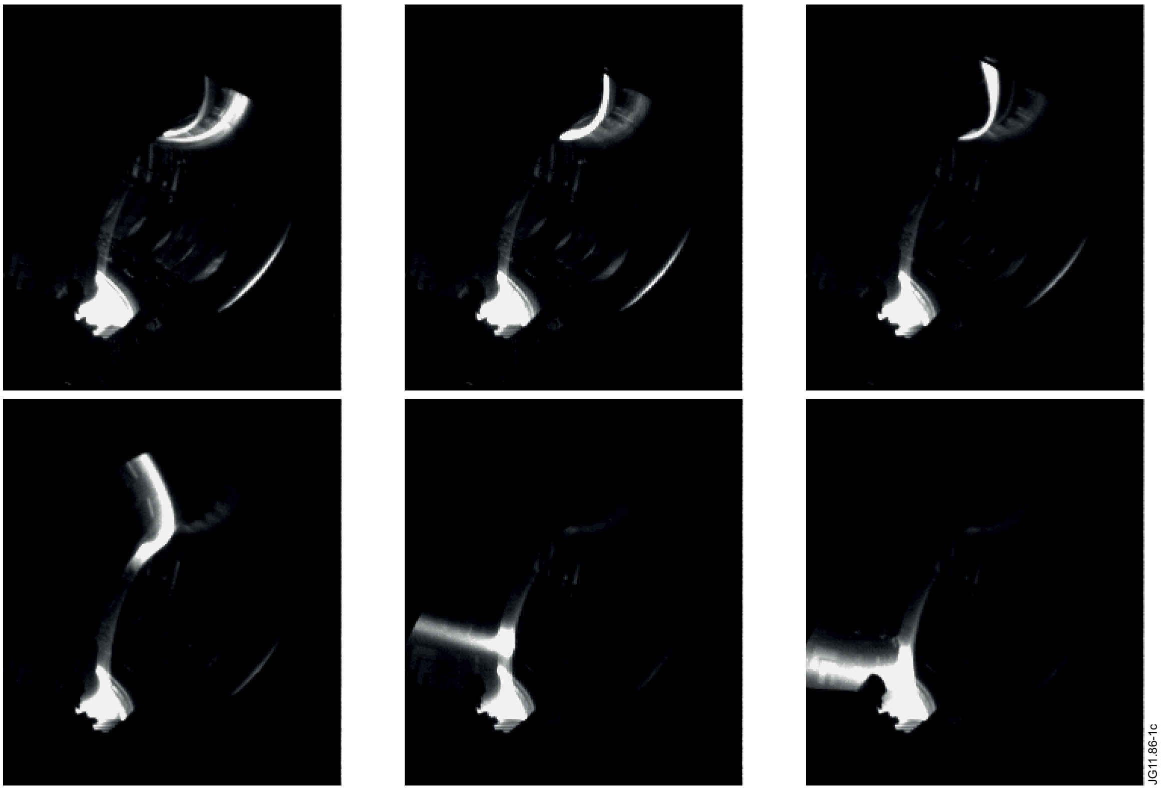


Figure 1: From left to right, top to bottom, a typical sequence of a MARFE as captured by JET fast visible amera. The ring of increased radiation moves up and down along the inner wall of the vacuum vessel.

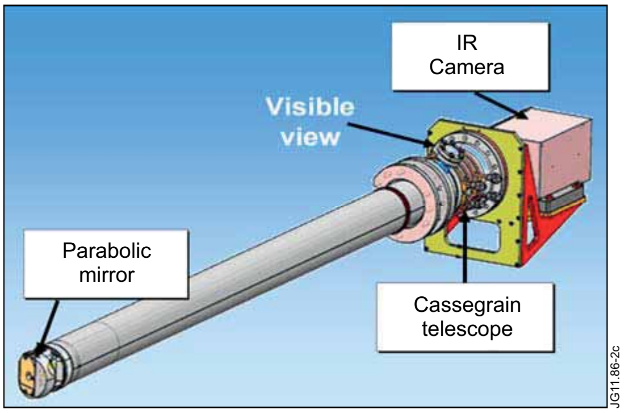


Figure 2: The endoscope supporting IR and visible cameras.

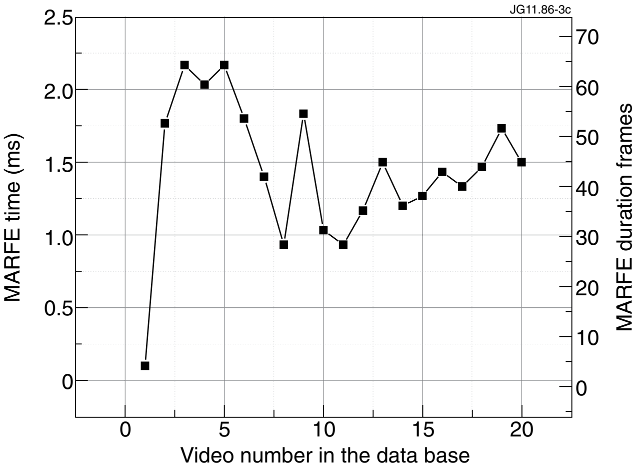


Figure 3: MARFE duration in milliseconds and in total number of acquired frames.

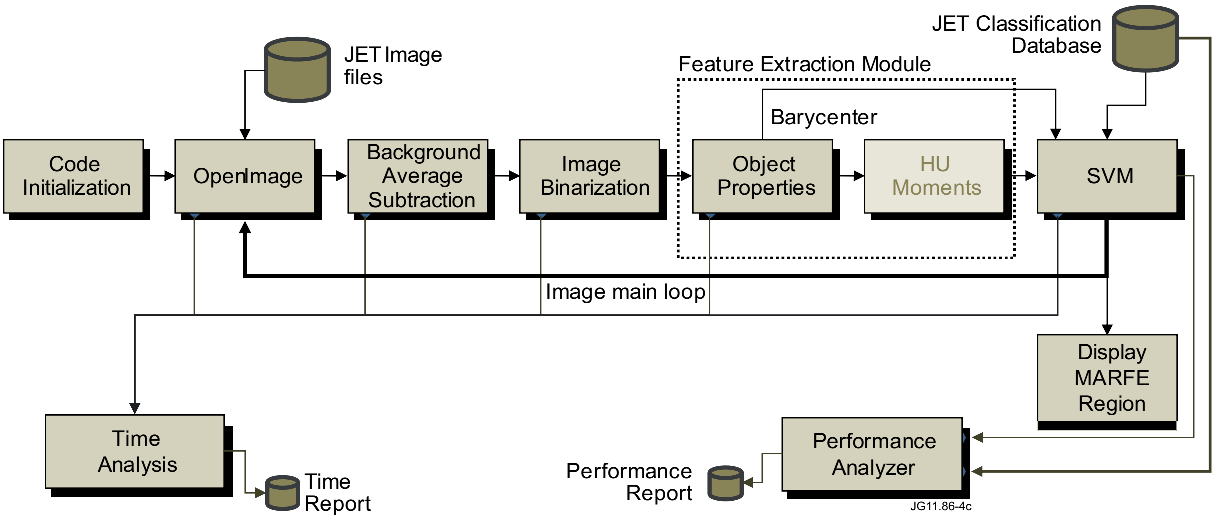
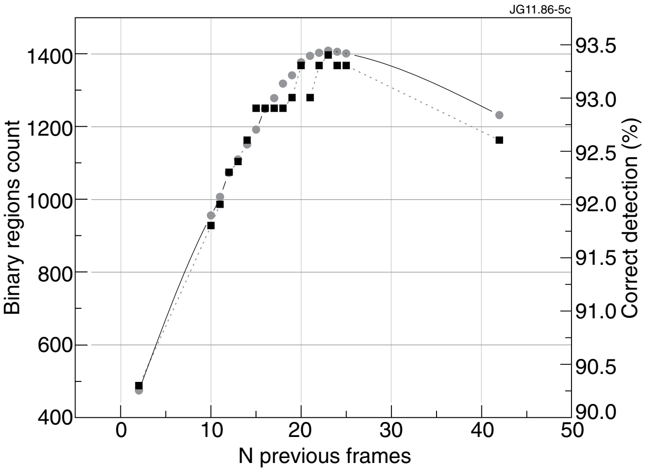


Figure 4: Image Processing Modules. The Main-Loop is responsible for the central part of the image treatement.



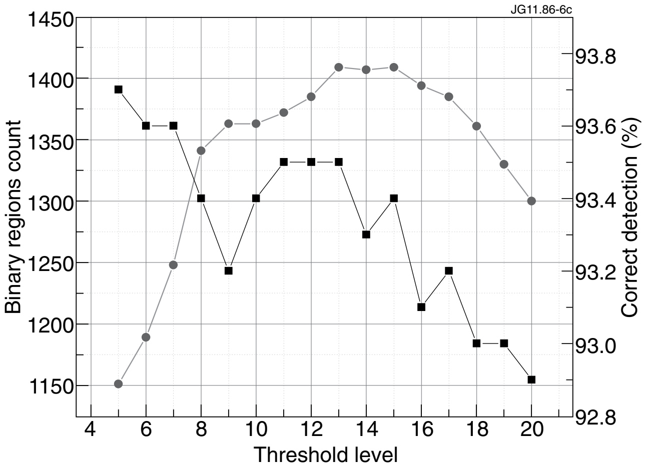
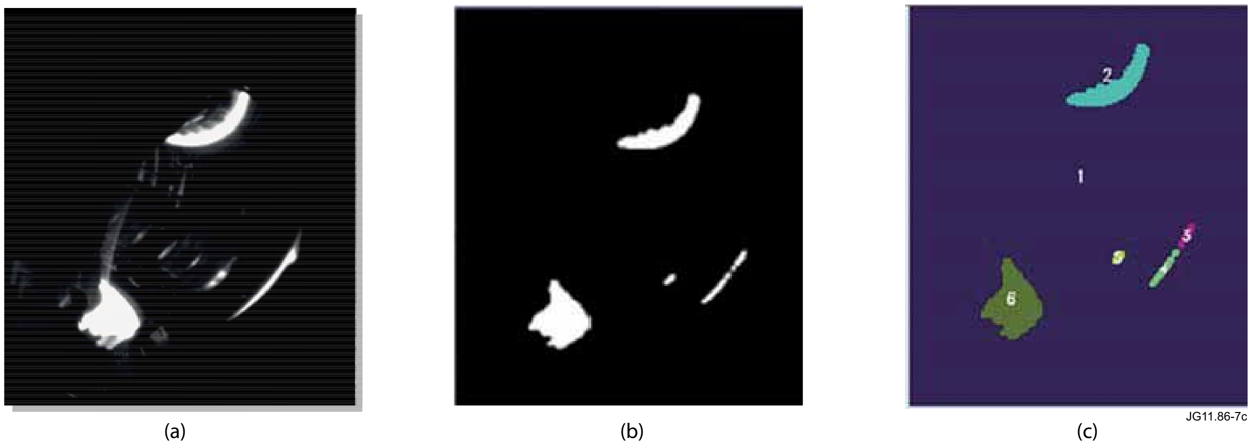


Figure 5: Background image estimation - Binary regions processed (circle) and correctly detected (square) as function of N previous frames. Nopt is 23 for 1409 binary regions processed and for 93.3% of corrected regions identified.

Figure 6: Optimal threshold determination - Binary regions processed (circle) and correctly detected (square). Topt is 13 for 1409 binary regions processed and 93.3% of corrected regions identifications.



	Hu1	Hu2	Hu3	Hu4	Hu5	Hu6	Hu7	Area	X_Center	Y_Center	Classe	Frame	number	Video
1.	–	–	–	–	–	–	–	–	–	–	–	41	–	70050V1
2.	0.818	1.971	3.786	5.954	–	–	–	615	129.50	72.50	0	41	1	70050V1
3.	1.170	2.681	6.812	7.363	14.452	8.727	–	51	181.00	161.50	0	41	2	70050V1
4.	0.604	1.308	6.451	7.763	15.146	9.480	15.298	91	166.50	179.50	0	41	3	70050V1
5.	1.600	4.162	8.998	10.857	20.787	12.963	–	31	136.50	173.50	0	41	4	70050V1
6.	1.723	7.145	7.154	9.610	18.913	13.485	18.079	947	66.50	196.50	0	41	5	70050V1

(d)

Figure 7: (a) Original image; (b) Binary image; (c) Labelled image and (d) Data structure to store blobs features, frame number and JET Video information used for data perfomance analysis comparison. The solid dark region (1), which correspond to the segmented background, is eliminated by the algorithm.

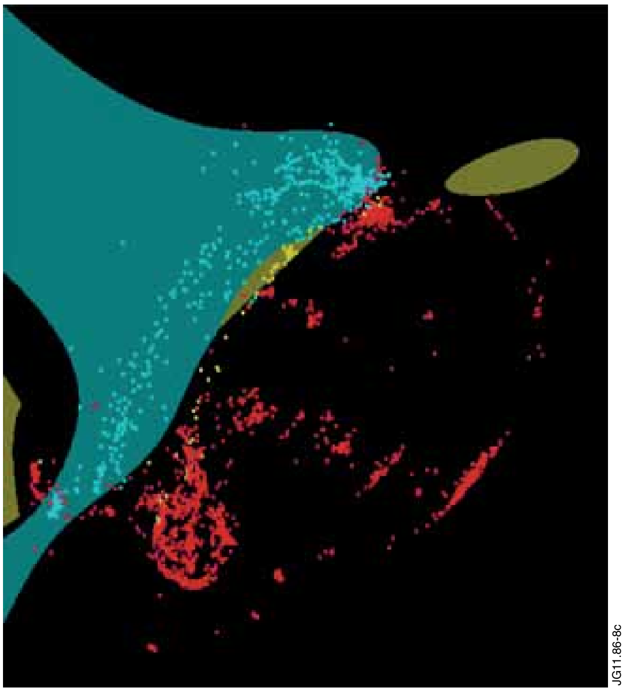
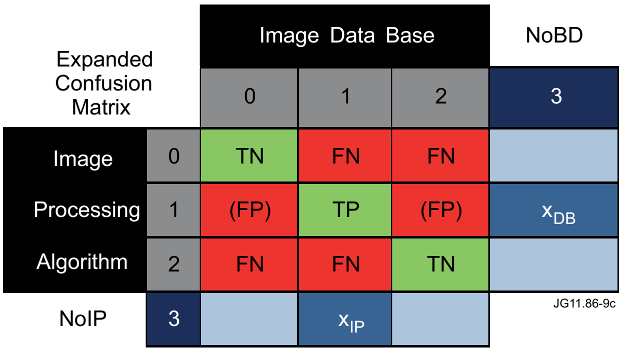


Figure 8: SVM function. Each point represents the barycentre coordinates obtained in the image data base; Blue: MARFEs regions; Red: Non-MARFE class; Yellow: Other class. The SVM kernel is presented for a polynomial of degree 10.



Expanded Confusion Matrix		Image Data Base			NoBD
		0	1	2	3
Image	0	TN	FN	FN	
Processing	1	(FP)	TP	(FP)	x_{DB}
Algorithm	2	FN	FN	TN	
NoIP	3		x_{IP}		

Figure 9: Expanded Confusion Matrix. The values 0, 1 and 2 represent Non-MARFE, MARFE and Others elements respectively. TP : True Positives, TN: True Negatives, FP: False Positives and FN: False Negatives. Column and row 3 indicate those results that were not found in the image data base (xDB) and by the image processing algorithm (xIP) respectively.

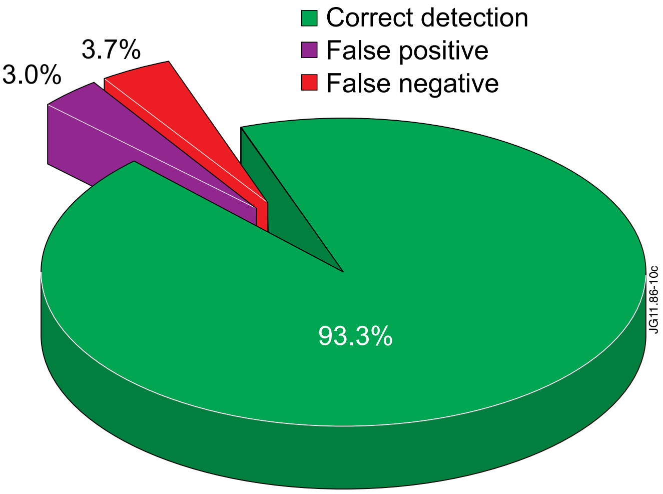


Figure 10: Percentages of correct detections, false positives and false negatives of the optimized version of the MARFEs detection algorithm.

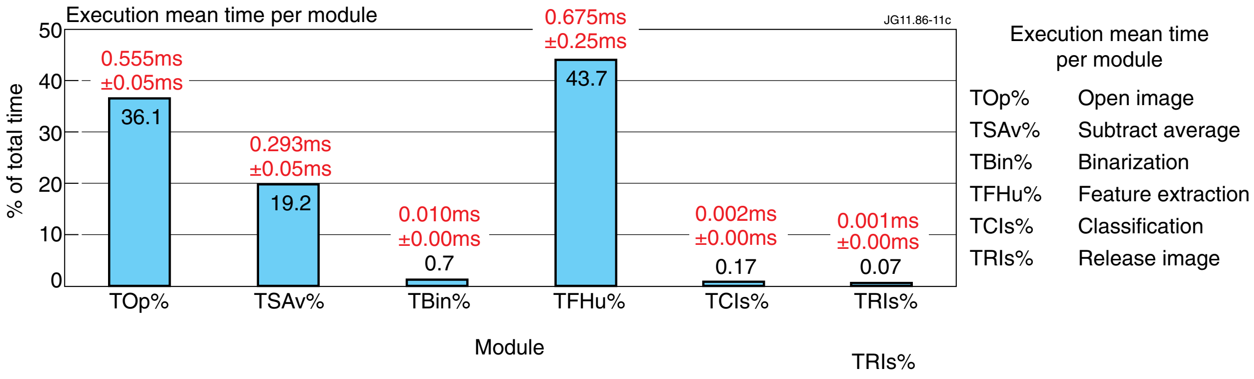


Figure 11: Percentages of Total Time spent in each image processing module for 12,000 frames for 26 JET Videos.

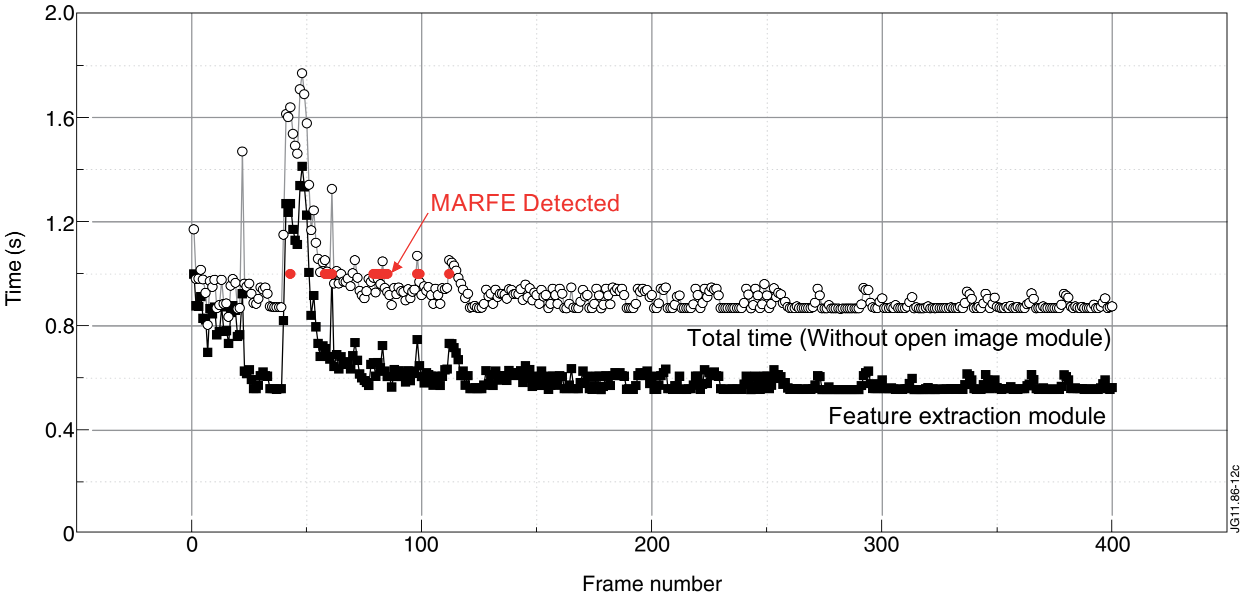


Figure 12: Total processing time for the Feature Extraction module and for the overall algorithm. Video sequence: KL870052V4.