An Embedded GPU Accelerated Hyperspectral Video Classification System in Real-Time

Jaime Sancho*, Manuel Villa*, Gemma Urbanos*, Marta Villanueva*, Pallab Sutradhar*,
Gonzalo Rosa*, Alberto Martin*, Guillermo Vazquez*, Miguel Chavarrias*, Ruben Salvador‡,
Alfonso Lagares†, Eduardo Juarez*, Cesar Sanz*
* Research Center on Software Technologies and Multimedia Systems. Universidad Politécnica de Madrid (UPM)
† Department of Neurosurgery, Hospital Universitario 12 de Octubre; Universidad Complutense de Madrid
‡ CentraleSupélec, CNRS, IETR, UMR 6164, France
{manuel.villa.romero, jaime.sancho, gemma.urbanos, marta.villanueva.torres, pallab.sutradhar, gonzalo.rosa.olmeda,
a.martinp, guillermo.vazquez.valle, miguel.chavarrias, eduardo.juarez, cesar.sanz}@upm.es
alfonso.lagares@salud.madrid.org, ruben.salvador@centralesupelec.fr

2021 XXVI Conference on Design of Circuits and Integrated Systems (DCIS) | 978-1-6654-2116-4/21/$31.00 ©2021 IEEE | DOI: 10.1109/DCIS53048.2021.9666171

***Abstract*—Hyperspectral imaging (HSI) has been adopted during the last years in different applications where material classification plays an important role. This favoured the improvement and development of new HS sensors, leading to HS snapshot cameras, where the sensor is able to acquire images in a real-time video fashion, reducing the spatial and spectral resolution for these cameras. However, although these cameras are able to acquire RAW HS images at high frames per second (FPS) rates, it is necessary to pre-process them to obtain an actual HS image and then obtain a classification map.**

This work addresses the development of an embedded CPU+GPU HS video classification system able to acquire and classify HS images in real-time video, i.e., at more than 25 FPS. Also, it includes a comparison where three different CPU+GPU embedded platforms are tested: NVIDIA Jetson Nano, NVIDIA Jetson TX1 and NVIDIA Jetson TX2. The results obtained show the feasibility of a HS video classification system using these embedded platforms.

***Index Terms*—HSI, real-time, video, classification, embedded**

I. Introduction and motivation

During the last years, hyperspectral imaging (HSI) has become an important technique in the field of material classification. This technology consists in the acquisition of images in multiple narrow bands of the wavelength spectrum, ranging from ultraviolet (UV) to infrared (IR), and allowing the spectral characterization of the pixels captured in the scene. This characterization, for each pixel, is called spectral signature, and it is where complex classification algorithms rely on to make founded decisions.

HSI has been tested in different domains, extending its use to numerous fields where the classification plays an important role. Although historically it was mainly used in remote sensing for agriculture, geography or surveillance [7], nowadays HSI is also employed in food industry [9], art restoration [11] or medical applications [5].

This fact caused new hyperspectral (HS) sensors that improved the former ones in different aspects. These sensors were usually of the type known as line-scan, i.e., a line of pixels able to acquire hundreds or thousands of spectral bands. As a drawback, the pixels line needs to be moved to scan the whole scene. This method was improved over the years by improving the spectral resolution or the wavelength range of the bands included in the sensor and the acquisition speed for this line. However, the intrinsic requirement for this type of sensors of moving the pixels line, entails two crucial drawbacks: (i) the camera needs to be integrated into a system where either the camera or the scanned scene is moving synchronously, and (ii) this movement along with the acquisition of HS images with a high spatial and spectral resolution prevents the use of these cameras for HS video.

These inconveniences motivated the development of a new type of HS sensors that does not need to be moved, the so-called snapshot. These sensors are similar to the ones found in conventional RGB cameras, except for filter employed to acquire different spectral bands. HS snapshot cameras have a mosaic filter which includes up to 5×5 bands arranged in a squared pattern, hence featuring a reduced number of spectral bands and compromising the spatial resolution. This fact is due to the fact that the complete sensor is not capturing pixels for every spectral band, as it depends on the filter pattern. Despite the reduction in spatial and spectral resolution, these cameras allow the possibility of capturing HS video at more than 25 frames per second (FPS), defined as real-time video in this application. Therefore, HS snapshot cameras are key to support the development of HS video classification.

However, although raw image acquisition is fast, processing HS images entails a high computational load that needs to be handled carefully, typically with the help of hardware accelerators [16]. In this work, a complete system composed of a HS snapshot camera and a CPU+GPU embedded platform able to (1) capture HS, (2) pre-process and (3) classify raw images in real-time video is presented. In this work, the case of study consists in the classification of brain tumors during resection operations [13] using a previously analysed and tested algorithm chain. For system testing, different CPU+GPU embedded platforms have been employed, finding differences,

This work was supported by the Regional Government of Madrid (Spain) through NEMESIS-3D-CM project (Y2018/BIO-4826). Also, we acknowledge NVIDIA for donating Jetson TX2.

advantages and disadvantages and identifying potential uses in different scenarios. The results of this comparison are presented. Besides, the feasibility for HS video classification tasks of the different platforms is demonstrated. The main contributions of this paper are the following:

1) The detailed description of an embedded HS video classification system stage by stage.
2) The analysis of different embedded platform-based HS video classification system.
3) The recommendation of using these platforms in specific scenarios.

The rest of the paper is structured as follows: Section II introduces the related works found in the state-of-the-art (SoA). Section III and Section IV detail the main concepts involved in HS camera acquisition and the key classification algorithms employed in this work. Section V describes the implemented system. Then, in Section VI, experiments and materials are described, leading to the Section VII where the results are presented and analyzed. Finally, Section VIII presents the main conclusions of the work.

II. Related work

During the last years, multiple works addressed the task of using HS snapshot cameras within a classification processing chain. Al-Sarayreh et al. [2] proposed a 3D-CNN architecture to classify red meat using HS images captured from a snapshot camera. This work compares three different classification algorithms PLS-DA, SVM and their proposal 3D-CNN, evidencing their feasibility with a classification accuracy around 81%, 91% and 96%, respectively. As an important remark, the authors reported that spatial-spectral algorithms obtain the best results given the limited spectral resolution of the snapshot cameras. In terms of computational time, their tests showed that PLS-DA, SVM and the proposed 3D-CNN, need 13, 38 and 61 seconds, respectively, to process one HS frame. Hence, the system is far from achieving real-time video. In other work, Steinbrener et al. [14] introduced a CNN able to classify fruits and vegetables with an accuracy above 92% using an snapshot camera with 16 spectral bands. This work also demonstrates that HS images with a limited number of spectral bands are useful for classification tasks. However, the authors did not aimed at accelerating the algorithm, only focusing on the classification results.

Other works have aimed to combine HS cameras with real-time classification. For example, Hohmann et al. [8] introduced an endoscopy in-vivo HS video classification system where the sensor does not include any filter; the HS images are obtained varying the wavelength of the light source and capturing the different bands synchronously. With this system, the authors are able to generate 6 spectral bands in about 0.45 seconds that are pre- and post-processed in around 1.5 seconds more. The results show a classification accuracy around 50% and 60% for different classifiers, in applications for cancer detection and prevention in the stomach.

To the best of authors' knowledge, there have not been other works addressing the HS snapshot video classification task, relying normally on high resolution HS linescan cameras, where the acquisition time is far from real-time and the computational load is higher due to the higher spectral and spatial resolution [6]. These works are compared in Table I.

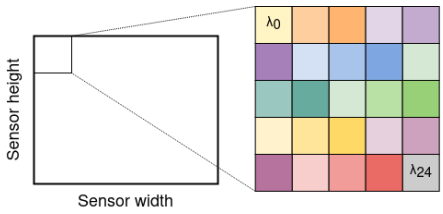

Fig. 1. Snapshot camera filter pattern.

TABLE I
Related works comparison

	Al-Sarayreh et al. [2]	Steinbrener et al. [14]	Hohmann et al. [8]	Proposed method
FPS	0.02	-	0.51	30
Camera type	Snapshot 25 bands	Snapshot 16 bands	Variable λ light source	Snapshot 25 bands
Real-time	No	No	No	Yes
Accuracy	96 %	92 %	50 - 60 %	64 %

III. Background: HS snapshot camera

To understand this work, the description of HS snapshot cameras is of paramount importance, as raw images obtained from these cameras need to be pre-processed to obtain HS images. These cameras employ rectangular CMOS sensors along with a HSI squared-pattern mosaic filter added at wafer-level. In this work, the camera filter employed features a 5×5 squared pattern, meaning that the 25 pixels included in these squares present a filter in a different wavelength. Consequently, 25 bands are captured while the spatial resolution is reduced in 5 pixels per dimension. This is depicted in Figure 1.

For this reason, the raw HS image is rearranged to generate a HS image. The result is denominated HS cube, as the HS image consists of three dimensions: x and y for the spatial dimensions of the image and z for the spectral bands. To obtain a pre-processed HS cube, a white-black calibration procedure on the reconstructed cube is applied. This process is employed to convert the raw sensor acquisition values into reflectance ones, expressed as percentages. In this way, the capture can be compared to other HS images in different light conditions. To do so, it is necessary to acquire *white* and *black* reference images, i.e. two images that provide, under the current light conditions, the maximum and minimum values of every pixel. This process will be detailed in Section V-A1. As a final step, a spectral correction is performed. In this procedure, every band is corrected considering the main and secondary lobes for every filter band in the sensor [15].

IV. Background: SVM

A support vector machine (SVM) [4] is a machine learning (ML) algorithm that finds an optimal hyperplane to separate two classes, maximizing the margin between them. The SVM training process aims to maximize the previous margin, given

a ground truth data-set. This margin is defined as the distance between the separating hyperplane and the training samples closest to the hyperplane, called the support vectors. There are two strategies to use this classifier in a multiclass problem: one-vs-all (OvA) and one-vs-one (OvO), the one used in this implementation.

The SVM implementation in this work employs a linear kernel, where the function in charge to compute the distance with the hyperplanes is the one presented in Equation 1. On the equation, w refers to the normal vector of the hyperplane of each class. x is the hyperspectral pixel that is going to be classified. Finally, b is the bias factor of the linear kernel. This process is repeated across all spatial pixels as denoted by sub-indices i and j.

$$f_{ij}(x) = w_{ij} * x + b_{ij} \tag{1}$$

V. Algorithm and Acceleration

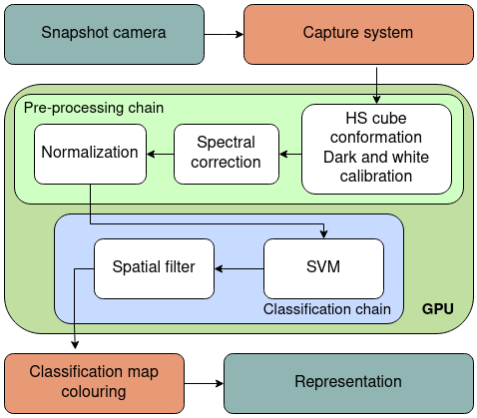

Fig. 2. System pipeline.

The proposed pipeline for the HSI video classifier is shown in Figure 2. The block diagram represents where each part of the code is executed and the followed sequence. Orange nodes refer to CPU code, while nodes with green background represent execution in the GPU. Within the green GPU block, each of the white nodes represent a GPU kernel.

The process starts with the snapshot camera, which needs a certain exposure time to take the photo. The camera is working in an open-loop; A capture is taken as soon as the camera is ready. The reading process, along with a crop on the image are implemented on the CPU, referred in Figure 2 as *Capture system*. The reason for this crop is that not all the camera sensor is effective, finding inactive pixels in borders.

The following part of the pipeline is the actual process, where the pre-processing and the classification are implemented. Then, the labels obtained from the classification chain are converted to a classification map, assigning one colour per label. Finally, the classification map is displayed.

The implementation of the different kernels is explained in the following subsections, that are split in two main parts: the pre-process chain and the classification chain. These stages are also represented in Figure 2.

A. Pre-processing chain

1) Cube conformation, white and dark calibration: The process of cube conformation is explained using the two-pixel and 25 bands of a HS image represented in Figure 3. The RAW image obtained from the camera is depicted on the upper part of the Figure, where the left darker pixels belong to the first HS pixel (and its 25 bands) and the right lighter pixels belong to the second HS pixel (and its 25 bands). The bottom part of the figure refers to the output of the kernel, the hypercube, in band interleaved pixel (BIP) format [3]. Here, every HS pixel along with its bands are placed together, i.e., first, the first HS pixel with its 25 bands and then, the second HS pixel with its 25 bands. These HS pixels have been padded with zeroes to generate 32 bands (7 are dummy) in order to make the spectral information fit to the warp size of the GPU hardware employed (32 threads).

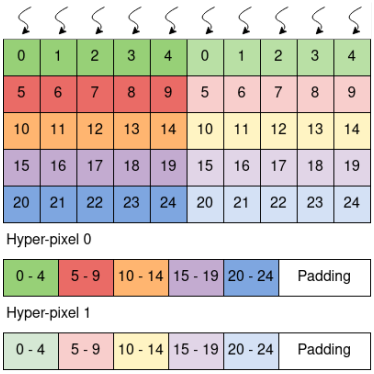

Fig. 3. Cube conformation process. Darker colours refer to a spectral pixel and lighter colours to another pixel.

The scheme employed to launch the threads of the kernel is also represented in Figure 3; one thread per spectral pixel, i.e., a thread per every pixel and band in the HS image.

Another computation implemented in this kernel is the white and black balance (Equation 2). In this computation, the difference between the original image Img and the dark reference D_{ref} is first obtained and then divided by the white reference W_{ref} and the dark reference. This is computed before rearranging the RAW image, using the same grid.

$$Img' = (Img - D_{ref})/(W_{ref} - D_{ref}) \tag{2}$$

2) Spectral correction: A multiplication between the hypercube and the correction matrix provided by the manufacturer is needed to linearly correct the sensor spectral information (Equation 3).

$$P'_i = \sum_{\lambda=0}^{N} (P_{i\lambda} * C_{i\lambda}) \tag{3}$$

Where P'_i is a HS pixel, C is the correction matrix, and λ refers to a specific band. As the hypercube is arranged in BIP and padded to 32 bands, a warp reduction is employed to improve the GPU utilization as shown in Figure 4.

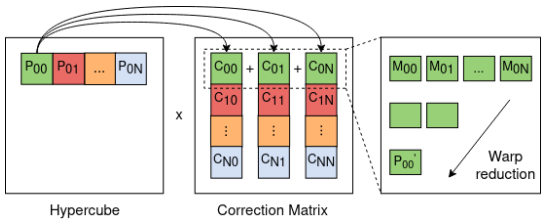

Fig. 4. Spectral correction process. Represents the multiplication between the hypercube and the correction matrix.

Using a grid where every pixel and band is handled by a thread, the first step is to store in a register the corresponding band of a pixel, for every pixel. Then, using warp intrinsic operations [10], a sum reduction is performed where each value of the warp is multiplied by the corresponding position of the correction matrix.

3) Normalization: The normalization used on the system consists of computing the mean value of the energy of each spectral band, then the image is divided by this value, as can expressed in Equation 4.

$$E_i = \frac{1}{N}\sqrt{\sum_{\lambda=0}^{N} P_{i\lambda}^2} \longrightarrow P_i^n = P_i/E_i \tag{4}$$

Using a grid where every pixel and band is handled by a thread, the first step is to load and square the corresponding value of band to carry out the normalization. As the band number equals the warp size, the average of the values is obtained performing a sum warp reduction. The result is stored in the first thread of the warp, in which the square root is computed . This result is copied on the rest of the warp threads, taking advantage of the intrinsic warps operations. Finally, each thread divides its corresponding band by the energy mean of the HS pixel bands.

B. Classification chain

1) SVM: SVM classification has been divided into two parts: The first one computes the distance of each HS pixel to the support-vectors of the model (kernel_score). The second one calculates the probability that each HS pixel belongs to each class (kernel_estimate).

For an optimal implementation of an SVM classifier over a GPU, the HS image needs to be arranged as band sequential (BSQ). Hence a transformation between the pre-processing chain and the classification chain has to be done. To transform the hypercube from BIP (pre-processing) to BSQ (SVM), the original BIP matrix is transposed.

Kernel_score performs a matrix-vector multiplication as reflected on Equation 1. To do so, a bi-dimensional grid of threads is launched; threads of x dimension are used to calculate a dot product within every HS pixel and a support vector. This is repeated for each support vector, i.e., for every pair of classes (an OvO classifier), using the threads of y dimension. When the products are done, the factor b_{ij} is added.

Once the distance to every support vector is computed, the probability of belonging to a class is calculated in the kernel_estimate. The kernel_estimate uses the sigmoid function (Equation 5) to estimate the probability of each class and decoupling the probabilities between them [12].

$$f(x) = \frac{1}{1+e^{-x}} \tag{5}$$

2) Spatial filter: Due to the low spatial resolution of snapshot cameras and the granularity of SVM classifications, a spatial filter [1] is proposed to smooth the classification maps obtained with the system. This behaviour is implemented in filter and max_class kernel.

The filter kernel groups the classification pixel coming from the SVM in a 3×3 window and move this window over the whole map. Then when the groups are done, the max_class kernel is in charge of applying a majority voting in each group and selecting the class with the higher average probability.

Based on the SVM probabilities for each class, the filter averages the probabilities for every pixel in the window, obtaining an average probability for every class. Then, the class with the highest probability is chosen as the pixel class. In Figure 5 the functionality of the filter is depicted: as can be seen in the highlighted window, pixel with highest class probabilities are represented with different colors. As the highest average probability class is green, that pixel is set as class green.

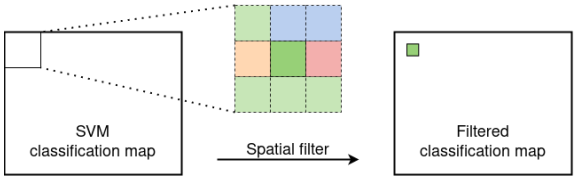

Fig. 5. Spatial filter.

VI. Test Material

A. HS snapshot camera

The HS snapshot camera (XIMEA, MQ022HG-IM-SM5X5-NIR) employed in this work featuring a spatial resolution of 409×217 with 25 spectral bands, ranging from 600 to 975 nm and with a full width at half maximum (FWHM) of approximately 16 nm. This camera can perform up to 170 raw FPS, depending on the exposition time settings.

B. Embedded CPU+GPU platform

In this work, three different CPU+GPU embedded platforms from NVIDIA are tested. They are characterized in Table II. The reason of comparing these platforms are their widely use in embedded CPU+GPU ecosystems, with the benefit of the CUDA programming language compatibility and support.

C. HS sequences and ground-truths

A set of video sequences acquired during resections of real brain tumours have been used to validate the designed experiments. The main characteristics of these video sequences are the following: Video sequences feature a spatial resolution

TABLE II
COMPARISON OF EMBEDDED CPU+GPU PLATFORMS.

Platform	Jetson Nano	Jetson TX1	Jetson TX2
CPU	Cortex-A57 4 cores	Cortex-A57 4 cores	Cortex-A57 4 cores + Denver2 2 cores
GPU	128-core Maxwell	256-core Maxwell	256-core Pascal
Memory	4GB 64-bit	4GB 64-bit	8GB 128-bit
Max. Power	5 Watts	10 Watts	15 Watts
Size	69.6 x 45 mm	87 x 50 mm	87 x 50 mm
Weight	61 g	88 g	85 g

of 409×217, a spectral resolution from 600 to 975 nm (25 bands), and 200 frames per sequence with an exposure time of 70 ms.

The database at our disposal has a total of seven video sequences in which the ground truth needed to perform the SVM training has been obtained with the help of neurosurgeons.

To classify the video sequences, seven different models in which one image is excluded have been trained. The description of the experiment is shown in Figure 6.

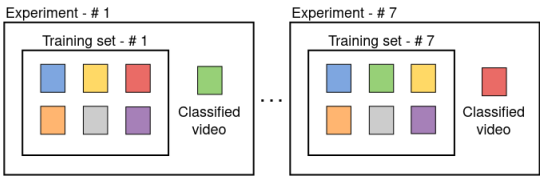

Fig. 6. Classification experiment.

VII. RESULTS AND DISCUSSION

The system has been tested using the previously mentioned material: a HS brain cancer classification chain and three different platforms. Classification results with the given models show an average accuracy of 64.2 % for the sequences tested. This result contrasts with higher classification results found in the SoA, mainly caused by the spectral reduction of the HS snapshot camera and the lack of extensive HS databases for the application. The platforms considered show differences of root mean squared error equal to 0.014 % in average accuracy, taking Jetson Nano as reference. This error is mainly due to different floating-point numbers rounding effects in different platforms and the intrinsic camera noise. Therefore, it is considered that these platforms obtain almost equal functionality results.

Time results for the three different platforms, are summarized in Figure 7. In this image, the time for each stage and platform is represented within the maximum slot of time for this application, i.e., the camera exposure time. This is considered the maximum time slot for real-time processing, as the process is done while the camera is acquiring the following frame. The acquisition-processing process is implemented concurrently with a delay of 1 frame. As can be seen, these three platforms achieve smaller processing times than the maximum time slot (23-23-35 ms vs 70 ms), meaning that the system will always be limited by the acquisition, rather than by the processing process. Without an acquisition time limitation, i.e., in applications where the exposure time could be lower, the system would allow the acquisition and classification of more than 28 and 40 classified FPS, for Jetson Nano and Jetson TX1/TX2, respectively. This would happen when conditions of light were enough to ensure the correct acquisition using low exposure times; using sun light or strong enough light sources.

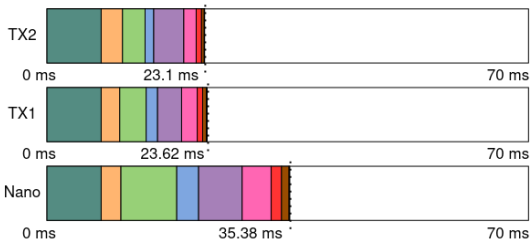

Fig. 7. Slot of time between captures. Each region refers to a different kernel: copy camera-memory (green), HS cube conformation (orange), spectral correction (green), normalization (blue), BIP_to_BSQ (purple), kernel_score (pink), kernel_estimate (red), filter (brown) and max_class (black).

In addition, these results show that the difference between Jetson TX1 and Jetson TX2 is negligible, whilst Jetson Nano achieves near half the performance obtained. This result also reveals that the process is completely limited by the computations performed, rather than the memory; the main difference between Jetson TX1 and TX2 is the memory width bus (64-bit and 128-bit), featuring the same number of processing cores with a slight increase of the GPU frequency clock in the case of Jetson TX2. This explains the slight difference in processing time for both platforms. In the case of Jetson Nano, although the memory width bus is the same as Jetson TX1, the number of processing units is halved compared to Jetson TX1 and Jetson TX2. For this reason, the processing time is near the double for Jetson Nano, taking into account that the copy from the camera to the memory (in green in Figure 7) is constant for the three platforms.

In order to better compare these embedded platforms in the system, four figures of merit (FoM) have been defined: FoM_{weight} and FoM_{size} consider the physical features of the board, FoM_{fps} measures the maximum number of attainable FPS and FoM_{power} considers the maximum power consumption. Given the case of study in this work, classification of brain tumors in the surgeon room, the maximum of these FoMs are set to the following: $FoM_{weight} = 1kg$, $FoM_{size} = 100cm^2$, $FoM_{fps} = 60$ and $FoM_{power} = 12W$. They are set taking into account the requirements within the surgeon room and thinking on its usage along with other medical instruments as medical microscopy (Zeiss Tivato 700), where the weight and size are limited by the physical surface of the device and the power is limited by the electrical characteristics of the device. The maximum number of FPS is set to 60 given the requirement of an interactive video application. These FoMs for the three platforms are summarized in Figure 8.

As can be seen, Jetson Nano is the clear winner for all the FoMs except for FoM_{fps}, meaning that it is the smallest

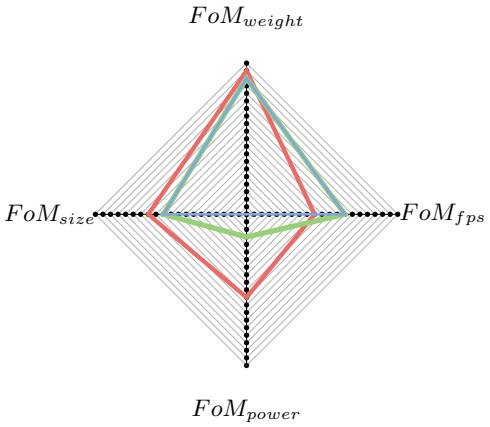

Fig. 8. Cross-platform comparison for the most significant parameters - Jetson Nano in red, Jetson TX1 in green and Jetson TX2 in blue.

and lowest-consuming platform among the comparison. As expected, the number of FPS obtained is smaller than the other platforms. However, this number is high enough to classify HS video in almost 30 FPS. Although Jetson TX1 and Jetson TX2 are similar in the FPS achieved, more than 48, their differences arise mainly concerning the power consumption, where Jetson TX1 is slightly better.

VIII. Concluding remarks and future lines

This work presents a novel embedded HS video classification system composed of a HS snapshot camera and three different CPU+GPU embedded platforms. The results obtained in this work show the feasibility of a HS video classification with 25 spectral bands and a resolution of 409×207 pixels. The results presented show that the classification obtained is almost the same for the three tested platforms, being the classification accuracy dependent on the SVM model and the application.

Time results show the viability of the three platforms under consideration, Jetson Nano, Jetson TX1 and Jetson TX2, to achieve a real-time video classification. In the case study of this work, the exposure time needed in the HS snapshot camera, 70 ms, limits the system, as the video acquisition only can achieve 14 FPS. In contrast, the classification processing time can be masked within that temporal window. However, if the illumination conditions were different and the exposure time would not be limiting, the system would reach a maximum FPS number depending on the classification processing time. This time is around 28 FPS for Jetson Nano and 40 FPS for Jetson TX1 and Jetson TX2.

For the comparison of these platforms, 4 different FoMs were defined, showing that Jetson Nano is smaller and consumes less energy than Jetson TX1/TX2, but only would achieve 28 FPS if the exposure time were not limiting. In this regard, Jetson TX1/TX2 would achieve up to 40 FPS, being considered a better option Jetson TX1 for its more minor use of energy. For these reasons, Jetson Nano is considered the best choice as an embedded system, as the penalty of having fewer FPS is not considered essential; the real-time video constraint is achieved, and the exposure time would be the limiting factor for almost every actual application.

The future lines of this work would address the test of different classification chains, analyzing the feasibility of HS classification video systems employing different classification algorithms to improve the accuracy results. These new algorithms would entail a higher compute capability needed, hence needing a further acceleration. This acceleration is expected by reducing the operation precision from 32-bits floats to 16-bits floats, or even to 8-bits integer.

References

[1] Energy and performance modeling of NVIDIA Jetson TX1 embedded GPU in HS image classification tasks for cancer detection using Machine Learning, school = ETSIS_Telecomunicacion, author = Jaime Sancho Aragón, year = 2018, keywords = Imágen hiperespectral; Medicina de precisión, url = http://oa.upm.es/54379/, (July)

[2] Al-Sarayreh, M., et. al.: Deep spectral-spatial features of snapshot HS images for red-meat classification. In: 2018 International Conference on Image and Vision Computing New Zealand (IVCNZ). pp. 1–6 (2018). https://doi.org/10.1109/IVCNZ.2018.8634783

[3] Aragón, J.S., et. al.: Characterizing HS Data Layouts: Performance and Energy Efficiency in Embedded GPUs for PCA-based Dimensionality Reduction. In: 2019 XXXIV Conference on Design of Circuits and Integrated Systems (DCIS) (2019). https://doi.org/10.1109/DCIS201949030.2019.8959835

[4] Braun, A.C., et. al.: Support vector machines, import vector machines and relevance vector machines for hyperspectral classification — a comparison. In: 2011 3rd Workshop on HS Image and Signal Processing: Evolution in Remote Sensing (WHISPERS). pp. 1–4 (2011). https://doi.org/10.1109/WHISPERS.2011.6080861

[5] Fabelo, H., et. al.: HELICoiD project: a new use of hyperspectral imaging for brain cancer detection in real-time during neurosurgical operations (2016). https://doi.org/10.1117/12.2223075

[6] Fabelo, H., et. al.: Spatio-spectral classification of hyperspectral images for brain cancer detection during surgical operations. PLOS ONE **13**(3), 1–27 (03 2018). https://doi.org/10.1371/journal.pone.0193721

[7] Fauvel, M., et. al.: Advances in spectral-spatial classification of HS images. Proceedings of the IEEE **101**(3), 652–675 (2013)

[8] Hohmann, M.e.a.: In-vivo multispectral video endoscopy towards in-vivo HS video endoscopy. Journal of Biophotonics **10**(4), 553–564 (2017). https://doi.org/10.1002/jbio.201600021

[9] Liu, D., et. al.: Recent advances in wavelength selection techniques for hyperspectral image processing in the food industry. Food Bioprocess Tech **7**(2), 307–323 (2014)

[10] NVIDIA: Warp-level Primitives (Online resource), accessed 20-5-2021 https://developer.nvidia.com/blog/using-cuda-warp-level-primitives/

[11] Peng, J., et. al.: Mining painted cultural relic patterns based on principal component images selection and image fusion of hyperspectral images. J Cult Herit (2018). https://doi.org/10.1016/j.culher.2018.09.008

[12] Ramírez, S.S.: Implementación de un acelerador en gpu empotrada de una máquina de vectores soporte (svm) para detección de tumores cerebrales (January 2019)

[13] Ruiz, L., et. al.: Multiclass brain tumor classification using HS imaging and supervised machine learning. In: 2020 XXXV Conference on Design of Circuits and Integrated Systems (DCIS). pp. 1–6 (2020). https://doi.org/10.1109/DCIS51330.2020.9268650

[14] Steinbrener, J., et. al.: HS fruit and vegetable classification using convolutional neural networks. Computers and Electronics in Agriculture **162**, 364–372 (2019). https://doi.org/10.1016/j.compag.2019.04.019

[15] XIMEA: HS imaging data correction (Online resource), accessed 20-5-2021 https://www.ximea.com/support/attachments/5981/SpectroNet-2016-03-Ximea-V02.pdf

[16] Yusuf, A., Alawneh, S.: A survey of gpu implementations for HS image classification in remote sensing. Canadian Journal of Remote Sensing (2019). https://doi.org/10.1080/07038992.2018.1559725