Abstract— The criteria applied on the choice of the compression algorithm is generally based on distortion, compression rate, processing time and hardware limitation. On most cases, the energy expenditure is not considered, however, in some applications, the energy expenditure is a good parameter for choosing the algorithms (E.g. embedded systems that apply the use of batteries). Thus, this article proposes an analysis scheme for algorithm’s energy expenditure and a focused assessment based on the scope of image compression. The scheme will allow the choice of image compression algorithms by using the energy expenditure as a metric. The most common image-compression algorithms will be analysed and the results will be demonstrated through three-dimensional curves (rate-distortion-energy).

Keywords— Image-compression, Energy-measurement, Rate-distortion Theory, Satellite Communication, Green-communication.

I. INTRODUCTION

The increase on the world’s demand for energy resulted on the increase of the need for developing systems with lower energy consumption. The environmental impact of power generation from fossil fuel has been clearly aggravated. As mentioned in [1], reducing the CO₂ emission has been considered an important goal within the design of communication systems. Overall, the green information and communication technologies have the potential for helping in this matter.

The relevant subjects related to green communication are discussed in [1] and are mainly related to system design, network coding and smart grid. It should also be mentioned that in 2011, another article related to green communication was published and considered a description of ongoing international efforts intended to standardize the methods that accurately quantify the carbon abatement potential caused by information and communication technology (ICT) [2].

In short, the subjects behind the green communication consist in reducing the energy expenditure of communication systems or developing systems that can help improve the quality of the energy by using intelligent energy distribution systems (the smart grids [3]).

Within this context, it is possible to point out multiple researchers that have proposed strategies to reduce energy consumption in different parts of the communication systems: (1) antennas [4], (2) equalization [5], (3) synchronization [6], (4) modulation [7], source coding and others.

It is worth mentioning two articles related to source coding. In the work published in [8], a comparison was made between the energy expended on the information compression with the energy expended within the transmission of the original data (using cellphones). If the cost of energy compression for the gained compression on the left hand side is lower than the efficiency of the energy communication on the right hand side, the compression is defined as efficient. The second work, documented in [9] describes the trade-offs between the power consumption, compression and quality of video streaming in wireless environment. Both above mentioned articles are focused on reducing the battery consumption in mobile devices.

On the image compression field, it is worth mentioning a few devices that use batteries which vary from a simple digital camera to remote sensing satellites. In addition, a large number of image compression algorithms can be chosen to perform the compression task on this kind of devices. As discussed in [10], the problem within data compression is a subject studied in the rate-distortion theory approached in 1959 by Claude E. Shannon in [11]. In his article, Shannon demonstrated that for a class of distortion measures and sources of information, there is a R(D) function (depending on the distortion measure and information source) that returns the rate R to an admitted distortion D (in [11] is it also presented some generalization regarding other kind of information sources: ergodic, continuous, among others).

After the publication of Shannon’s article, other authors have published works regarding the same subject, many with different focuses and approaches. In 1968, Robert G. Gallager devoted an entire chapter of his book [12] to the source coding with fidelity criteria. Years later, Kiffer published an article intended review the problem by using a different approach and future trends point [13].
Finally, another important reference concerning the above mentioned is the Thomas Cover book [14]. In this book, Cover devoted a chapter to the rate-distortion theory by using a very interesting approach.

In the definition of the rate-distortion function, there are no constraints on the resources applied by the encoder [14]. There are no constraints on computational complexity or energy consumption. However, in some applications such as image transmission from a remote sensing satellite, it would be interesting to use a data compression solution which has a low energy demand. In [10], an analysis was presented regarding energy consumption of the image compressor recommended by the Consultative Committee for Space Data Systems (CCSDS), CCSDS 122.0-B-1, applying two different platforms. The first ran the algorithm by using only a CPU to compute the encode task, while the second was supported by a Graphics Processing Unit (GPU). Gains in number of encoded images were observed during the help of the GPU (24% more encoded images in some cases with the same amount of energy). An analysis including the time expenditure was made and gains around 29.1% were observed. This has proven that the compressors efficiency are platform dependent and that time and energy expenditure can have different magnitudes depending on the hardware used.

Another article around the energy saving subject was written by Fonseca and Queiroz [15]. This article demonstrated the energy consumption of the H.264 Codec (coder-decoder) running in a General Purpose Processor (GPP). The authors showed that 35% of the energy could be saved with a small impact on rate-distortion performance in the tested GPP.

Motivated by the platform dependence of the compressors energy consumption, this article will show a scheme that analyses the compressors efficiency for different platforms. In this scheme, a simple external hardware was developed by using a micro-controlled board with a current sensor. Now, some three-dimensional curves were plotted and a discussion can be carried on around the rate-distortion-energy. Thus, this type of analysis can assist in the development of compression algorithms with a greened perspective (green data compression).

II. THE POWER-MEASURING SCHEME

The proposed scheme can be used for measuring the energy expenditure of a large number of algorithms. For the analysis performed in this work, the following grey-scale image compressors were used:

- CCSDS image compression algorithm: The image compression algorithm recommended by the CCSDS [16] for aerospace application. The C++ implementation available by Nebraska University was used.
- SPIHT CODETREE algorithm: The SPIHT algorithm with arithmetic codification and wavelet transform.
- SPIHT FASTCODE algorithm: The SPIHT algorithm with non-arithmetic codification and with wavelet transform.
- JPEG2000 algorithm: Available by the OPENJPEG group.

All listed implementations were compiled to be used on a Windows platform. In this sense, the simulation software was developed to run also in Windows. To avoid energy measuring of others process, the simulation tool was developed to run in “safe mode with network”.

The main idea consists in using two computers. The first one to manage the external measurement hardware and control the simulations (master) and the second to run the algorithms (slave). In fact, the second computer will be under analysis. In other words, the measuring process will evaluate the slave hardware running the compression tasks. This scheme was illustrated in Fig. 1.

![Fig. 1. Power-measuring schematic diagram.](image)

Both computers can communicate by using UDP (User Datagram Protocol) packets through an Ethernet interface. The measuring hardware can communicate to the master by using the serial communication. The slave’s software has all the compressors algorithms and the images files. In fact, the slave computer just performs the codification process, while the master computer communicates to the power measuring hardware to get the current consumption information. To facilitate the understanding of the simulator, a systematic is described as follows:
1. The slave computer is turned on in “safe mode with network” and the slave’s software is executed. This is the only thing required on the slave computer. The slave’s display can be turned off at this point.

2. The simulation parameters can be selected in the master’s software: compress algorithm, compression rate, number of images to encode, number of repetitions to be done (to calculate a mean value for more than one simulations) and other communication parameters.

3. On pressing “Start simulation” the master computer sends a message to slave computer and to the metering external hardware. The message to the slave computer contains the simulations parameters. For the measuring hardware, the master sends just a flag to start the measuring. Next, the master will await a message from the slave to know when the encoding tasks are completed.

4. When the master computer receives the slave’s message, it sends a message to the external hardware requiring the measurement results and saves it into a text file.

It is well known that the real power \((P)\), in Watts, for alternated circuit (AC) can be calculated by

\[
P = \frac{1}{2} V_{\text{peak}} I_{\text{peak}} \cos \theta \\
= V_{\text{rms}} I_{\text{rms}} \cos \theta \quad [W],
\]

Where: \(V_{\text{peak}}\) is the peak voltage in volts, \(I_{\text{peak}}\) is the peak current in amperes, \(V_{\text{rms}}\) is the root-mean-square voltage in volts, \(I_{\text{rms}}\) is the root-mean-square current in amperes and \(\theta\) is the phase angle between the current and voltage sine waves.

It is important to mention that we are particularly interested on the results in terms of energy and for this reason it is necessary to consider the encoding time. It is known that

\[
P_{\text{real}} = \frac{E}{t}.
\]

Where \(E\) is the energy in Joules and \(t\) is the time in seconds.

With this, it is possible to find the energy expended by the slave computer during a period. However, we are interested in measuring the energy expended by the image compression process. The \(E\) will compute the total energy used by the computer: to stay turned on, to run the operational system, to run the simulation software and others background processes (lateral processes). Based on this, it is possible to consider that

\[
E_{\text{enc}} = E_{\text{full}} - E_{\text{basic}}
\]

Where \(E_{\text{enc}}\) is the energy expended by the compressors, \(E_{\text{full}}\) is the total energy expended by the slave computer and the \(E_{\text{basic}}\) is the energy expended by the slave because of the lateral processes. Isolating the \(E\) in (3) and using (4), we have

\[
E_{\text{enc}} = P_f t - P_b t ,
\]

Where \(P_f\) and \(P_b\) are the total power used by the slave computer and the power used for the lateral processes respectively. Next, through (1), the energy expended by the compression processes are

\[
E_{\text{enc}} = V_f I_f \cos \theta_f t_f - V_b I_b \cos \theta_b t_b .
\]

It is possible to consider \(V_f = V_b\) and \(\cos \theta_f = \cos \theta_b\) because both are variables of the same computer. In other words, the voltage and the cos \(\theta\) are independent of the computer tasks and processes.

Due to the lateral processes which are running during all time used by the compression tasks, we can consider \(t_f = t_b\). Now (6) can be written

\[
E_{\text{enc}} = V_{\text{rms}} \cos \theta t \left(I_f - I_b\right) \quad [J].
\]

Finally, using the external measuring hardware, it is possible to find \(I_f\) and \(I_b\). With this, it is achievable to find the energy expended by the compressors for the processing platform analysed (slave). In the Section IV a set of simulations will be described and some results will be shown.

### III. The Power-Measuring Hardware

The measuring hardware was developed to represent a low cost. In fact, the Arduino micro-controlled boards present a good solution to achieve the objective described in this article. The Arduino is a microcontroller board for electronic prototyping with a free system development kit (SDK). The main purpose of this platform is to be applied on the development of independent or interactive objects to be connected to a host computer or manager. The Arduino is presented in many types of applications such as robotics, security, art, among others. More information about the Arduino platform can be found in [17].

On the market, there are several types of extensions (called Shields) that complement the Arduino board. It is possible to find a shield that has a current sensor type of ACS712. The current sensor ACS712, provides low cost and precise solutions for AC or DC current sensing in industrial, commercial and communications system.
Typical applications include motor control, load detection and management, and others. The device consists in a precise, low-offset, linear Hall circuit with a copper conduction path located near the surface of the die.

In short terms, the external hardware used in this work was composed by an Arduino UNO R3 board and one shield ACS712 (ACS712ELCTR-05B-T current sensor to measure at most 5A). With this, the Arduino UNO board will inform the master computer the mean current used by the slave. The connection between the Arduino board and the current sensor is trivial, like showed by Fig. 2.

```cpp
void loop()
{
  Serial.println(analogRead(sensorPin));
}
```

This is a very simple code and can be easily improved if needed. The master and slave’s algorithms will not be described within this article.

IV. METHODOLOGY AND RESULTS

At this point, the scheme described in section II will be used in a set of simulations with the following parameters:

- Compression rates: 0.25, 0.5, 0.75, 1, 1.25, 1.5 and 1.75 bits-per-pixels (bpp).
- Input images: 100 images of 4096² pixels from the satellite CBERS-2B (the third satellite of the program China-Brazil Earth Resources Satellite).
- Slave computer specification: LG Electronics R490-G.BR52P1 (x64-based PC), 1 processor (Intel64 Family 6 Model 37 Stepping 2 GenuineIntel 2128 Mhz) running the Microsoft Windows 8.1 Pro in “safe mode with network”.
- \( \cos \theta \): The power supply used to maintain the computer working is the “SADP-65KB D” Laptop AC Adapter (Delta Electronics). The efficiency informed by the producer is 0.84. So, we can use \( \cos \theta = 0.84 \) to translate the power lost on the AC-DC conversion including the power factor.

Note that due to the slave being a laptop, the battery was removed to perform the simulations.

Firstly, it is required to find the \( I_b \). To achieve this objective, the master starts a simulation with the slave turned on but not performing the compression processes (meaning, not simulation at this time). In other words, the slave software was just awaiting the master’s command to start the encode process. After measuring for one hour, the result was the rms \( I_b = 384.7 \) mA. Now (7) can be written using the power supply efficiency and the local rms voltage.

\[
E_{enc} = 127 \times 0.84 \times t \left( I_f - I_b \right) \quad \text{(8)}
\]

\[
= 106.68 \times t \left( I_f - 0.3847 \right) [\text{J}] \quad \text{(9)}
\]

Next, the master starts the simulation for all above mentioned compressors. Some values of time (in seconds) and current (in Amperes) can be found by looking into Tables I and II respectively.
Finally, similar to what was shown in [10], it is possible to plot the rate-distortion-energy curves for each compressor. The results are plotted in Fig. 3, where \( R \) is the rate in bits-per-pixel, \( D \) is the PSNR (Peak-Signal-to-Noise-Ratio) and Energy is the \( E_{\text{enc}} \) in Joules used to compress the 100 images (meaning 40 simulations for each point).

In [15], a desired operational region on the three-dimensional plot was the lower convex hull (LCH). Herein, it is different because the algorithms have a strict behaviour. To understand what this means, we can analyse the hatched regions of the plans \( (R,D) \) and \( (E,D) \).

These regions are the area formed between two lines of projection. We can call these lines Upper Limit Line (ULL) and Lower Limit Line (LLL). The LLL is formed by the \( (R,D,E) \) points which minimize the energy expenditure (\( LLL = \min_{(R,D,E)} \)) and the ULL by the \( (R,D,E) \) points that minimize the distortion, maximizing the PSNR (\( ULL = \min_{D} \))

Looking into Fig. 3 it is possible to see that, for almost all rates, the algorithm that minimizes the energy consumption maximizes the distortion. The exception is for the rates between 0.25 bpp and about 0.7 bpp. In this region, one algorithm satisfies both conditions: SPIHT codetree. For the other rates, the minimum distortion is achieved by the SPIHT codetree and the minimum energy by the CCSDS algorithm.

In fact, all the \( (R, D, E) \) points that have the projection inside both hatched plans represent a viable option. In other words, the region of interest to operate has both areas dashed like a projection. Overall, the JPEG2000 algorithm is an option from the rate 0.7 bpp. This algorithm can be used like a medium between energy and distortion.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>CCSDS</td>
<td>0.25</td>
<td>311.81</td>
<td>330.33</td>
<td>310.74</td>
<td>7.079</td>
</tr>
<tr>
<td>CCSDS</td>
<td>0.75</td>
<td>408.36</td>
<td>408.90</td>
<td>407.88</td>
<td>0.069</td>
</tr>
<tr>
<td>CCSDS</td>
<td>1.25</td>
<td>481.36</td>
<td>482.01</td>
<td>480.70</td>
<td>0.124</td>
</tr>
<tr>
<td>CCSDS</td>
<td>1.75</td>
<td>549.67</td>
<td>550.31</td>
<td>549.05</td>
<td>0.086</td>
</tr>
<tr>
<td>SPIHT CTREE</td>
<td>0.25</td>
<td>155.50</td>
<td>156.09</td>
<td>154.75</td>
<td>0.063</td>
</tr>
<tr>
<td>SPIHT CTREE</td>
<td>0.75</td>
<td>657.80</td>
<td>659.05</td>
<td>656.50</td>
<td>0.330</td>
</tr>
<tr>
<td>SPIHT CTREE</td>
<td>1.25</td>
<td>1399.9</td>
<td>1402.5</td>
<td>1397.6</td>
<td>1.008</td>
</tr>
<tr>
<td>SPIHT CTREE</td>
<td>1.75</td>
<td>4845.5</td>
<td>4850.1</td>
<td>4839.6</td>
<td>5.256</td>
</tr>
<tr>
<td>SPIHT FCODE</td>
<td>0.25</td>
<td>528.82</td>
<td>529.56</td>
<td>527.26</td>
<td>0.258</td>
</tr>
<tr>
<td>SPIHT FCODE</td>
<td>0.75</td>
<td>1379.4</td>
<td>1380.7</td>
<td>1378.0</td>
<td>0.544</td>
</tr>
<tr>
<td>SPIHT FCODE</td>
<td>1.25</td>
<td>2474.5</td>
<td>2476.4</td>
<td>2472.9</td>
<td>1.106</td>
</tr>
<tr>
<td>SPIHT FCODE</td>
<td>1.75</td>
<td>4307.7</td>
<td>4311.9</td>
<td>4304.2</td>
<td>4.509</td>
</tr>
<tr>
<td>JPEG2000</td>
<td>0.25</td>
<td>837.74</td>
<td>838.94</td>
<td>836.91</td>
<td>0.394</td>
</tr>
<tr>
<td>JPEG2000</td>
<td>0.75</td>
<td>856.77</td>
<td>857.81</td>
<td>855.88</td>
<td>0.423</td>
</tr>
<tr>
<td>JPEG2000</td>
<td>1.6</td>
<td>872.55</td>
<td>873.62</td>
<td>871.53</td>
<td>0.243</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>CCSDS</td>
<td>0.25</td>
<td>0.5030</td>
<td>0.5148</td>
<td>0.4783</td>
<td>0.892</td>
</tr>
<tr>
<td>CCSDS</td>
<td>0.75</td>
<td>0.4891</td>
<td>0.5083</td>
<td>0.4715</td>
<td>1.24</td>
</tr>
<tr>
<td>CCSDS</td>
<td>1.25</td>
<td>0.4853</td>
<td>0.5033</td>
<td>0.4675</td>
<td>0.804</td>
</tr>
<tr>
<td>CCSDS</td>
<td>1.75</td>
<td>0.4957</td>
<td>0.5176</td>
<td>0.4711</td>
<td>1.13</td>
</tr>
<tr>
<td>SPIHT CTREE</td>
<td>0.25</td>
<td>0.4536</td>
<td>0.4642</td>
<td>0.4431</td>
<td>0.227</td>
</tr>
<tr>
<td>SPIHT CTREE</td>
<td>0.75</td>
<td>0.4503</td>
<td>0.4752</td>
<td>0.4350</td>
<td>1.12</td>
</tr>
<tr>
<td>SPIHT CTREE</td>
<td>1.25</td>
<td>0.4528</td>
<td>0.4764</td>
<td>0.4370</td>
<td>1.04</td>
</tr>
<tr>
<td>SPIHT CTREE</td>
<td>1.75</td>
<td>0.4545</td>
<td>0.4982</td>
<td>0.4046</td>
<td>4.95</td>
</tr>
<tr>
<td>SPIHT FCODE</td>
<td>0.25</td>
<td>0.4925</td>
<td>0.5068</td>
<td>0.4547</td>
<td>1.75</td>
</tr>
<tr>
<td>SPIHT FCODE</td>
<td>0.75</td>
<td>0.4856</td>
<td>0.5133</td>
<td>0.4270</td>
<td>4.65</td>
</tr>
<tr>
<td>SPIHT FCODE</td>
<td>1.25</td>
<td>0.4860</td>
<td>0.5101</td>
<td>0.4426</td>
<td>3.27</td>
</tr>
<tr>
<td>SPIHT FCODE</td>
<td>1.75</td>
<td>0.4568</td>
<td>0.4956</td>
<td>0.4223</td>
<td>6.79</td>
</tr>
<tr>
<td>JPEG2000</td>
<td>0.25</td>
<td>0.4868</td>
<td>0.5103</td>
<td>0.4499</td>
<td>4.56</td>
</tr>
<tr>
<td>JPEG2000</td>
<td>0.8</td>
<td>0.4765</td>
<td>0.4988</td>
<td>0.4424</td>
<td>4.28</td>
</tr>
<tr>
<td>JPEG2000</td>
<td>1.6</td>
<td>0.4976</td>
<td>0.5147</td>
<td>0.4639</td>
<td>2.64</td>
</tr>
</tbody>
</table>

Finally, the SPIHT fastcode can be used from 1.65 bpp, nevertheless, it is not a good option since it reduces just a little bit of power consumption at the cost of increasing the distortion considerably.
V. CONCLUSIONS

This article describes a low-cost tool to evaluate the energy consumption of the compressors used in remote sensing systems. This tool can help some researchers make the choice of an image compressor for systems that have power constraint or reduce the energy consumption of a massive compression system.

In the case of the GPP analysed, it is possible to develop a compression system with multiple algorithms. Thereby, the user can choose from three compressors for rates above 0.7 bpp. Selecting the CCSDS algorithm in the case of minimum energy consumption, the JPEG2000 to operate with moderate energy expenditure and the SPIHT codetree for minimal distortion.

Considering the large difference in energy consumption of the analysed algorithms and the lack of a theoretical limit in terms of energy (E.g.: E(D) or E(R)), it is possible to believe in the creation of image compression algorithms that save more energy and present a quite acceptable rate-distortion curve.

ACKNOWLEDGMENT

This work was supported by CNPq-Brazil (grant 143355/2011-2) and FINEP-Brazil.

REFERENCES


