Survey on Progressive Image Compression and Transmission and its application in Underwater Intervention Missions

Eduardo Moscoso, Raul Marin Prades, Jorge Sales, Pedro J. Sanz
Departamento de Ingeniería y Ciencia de Computadores
Av. Sos Baynat, s/n, 12071 Castellón de la Plana (España)
Universitat Jaume I
{rubino, rmarin, salesj, sanzp}@uji.es

Abstract

This work provides a review of compression techniques that can be applied in Autonomous Underwater Vehicles for Intervention, such as the one designed for projects RAUVI, FP7 TRIDENT and TRITON. The review focuses on Progressive Image Compression and Transmission, presenting a comparative analysis of further work that can be applied in this context.

Palabras clave: Autonomous Manipulation, Underwater Application, Image Compression, Image Transmission.

1 Introduction

Robotic applications, and particularly Autonomous Underwater Vehicles for Intervention (AUV-I) use to focus on image input to control the system in real-time and send the information to the operator, which can normally interact with the system and adjust the task execution in a supervised manner [1]. This kind of control has been experimented in the FP7 TRIDENT Project, to perform autonomous visually guided grasping in the sea [2, 3]. Other related projects show the vision as the main input for underwater interventions, such as ALIVE [4], SAUVIM [5], and PANDORA [6]. Moreover, within the TRITON project (www.irs.uji.es/triton), the underwater vehicle performs a station docking to an underwater panel where the manipulator has to perform an intervention on a valve and a connector (see Figs. 1, 2, and 3). The arm performs the action in three steps: first of all it generates a 3D view of the panel by using a stereo camera and a laser (PCL, [7]) then, using the data from the Point Cloud, a Grasping Determination algorithm calculates the stable grasps and finally, the grasp execution is performed, always having the possibility for the operator to provide more information to the system such as alternative grasps.

Besides this, communications is a crucial subsystem in any robotic application, specially the ones that permit the user to interact remotely with the system. For that, image compression and transmission is necessary in order to send the required information at the lowest time-delay and without compromising the network and the whole system.

In this article we are going to present several progressive image compression and transmission techniques and analyse the way they can be applied to the TRITON project, considering different ways to connect the robot to the user, such as radio frequency (433/866MHZ), Sonar, and Ethernet connection through the umbilical. In fact, depending on the real scenario an Ethernet connection should be considered at a first glance. For cooperative applications where more than one robot must be included, the user will need to interact with the system at lower rates, such as radio frequency (few meters) and sonar.

As long as possible, the transmission of real images from the vehicle to the user must be considered, as they provide more information to the operator in order to adjust the autonomous behaviors or abort the operation in case of risk. Of course, the sonar solution will need to send only a very few relevant data of the intervention scenario, such as grasping points. For example, as the system provides a laser scanner for calculating the point cloud of the environment, depending on the available bandwidth the transmission of the images will be avoided, sending just the represen-
Figure 2: Autonomous Intervention in controlled indoor underwater environment: inserting and extracting a connector (hot stab)

Figure 3: Image captured while underwater intervention in sea conditions

tative data of the point cloud.

Moreover, it is necessary to clarify that this article presents compression techniques can be used for underwater communications. At the moment of writing a compression image server is already implemented and the researchers are working to adapt it to the robot platform. Comparative results of the compression techniques in underwater intervention missions are expected for the next september 2014, on the AUV-I platform at IRS Lab.

2 Image Compression Techniques

The objective of image compression is to reduce its entropy in order to store or transmit it in a more efficient manner. We can also clearly distinguish between lossless and lossy compression. In lossless compression, the decompressed image will be exactly the same as the original image while, in lossy compression, the decompressed image will be an approximation of the original image.

Digital images usually have 3 color components which means that what we perceive as one color image is (without loss of generality), in fact, composed of a luminance channel (black and white version of the color image) and 2 color difference channels (which can usually be subsampled without much loss - see HVS below).

Progressive image compression is such that it is trivial and very inexpensive in terms of processing power (there is no need to decompress and recompress the image) to supply an image which is either resolution or quality progressive, meaning that the image data can be truncated at any point and we would still get a lower resolution or quality version of the original image (in this sense, lossless streams can become lossy by simple truncation). In the case of color images, we could also prepare the image in such a way that a monochrome version of it could be obtained with the same progressive characteristics as before.

As opposed to video compression, image compression does not address the compression of the high temporal correlation between adjacent frames in a video sequence, i.e., in the context of a video sequence, image compression is also known as intra-frame or key-frame compression. In fact, the reason why video compression actually compresses so well is that the highest gains come from an adequate compression of the temporal information using preceding and/or succeeding frames to actually predict the current frame and code the error between the current and the predicted frames. This prediction is usually done with the help of motion estimation and compensation and is usually quite a computation-intensive task (most video compressors are highly asymmetric, where the compressor is usually slower than the decompressor). The resulting compressed frames are called inter-frames or delta-frames.

In order for the decoder to be able to properly synchronize in case of a frame error or loss, or to be able to have some level of random access to the compressed stream, or to simply better adapt to changes in the image sequence, the compressor periodically inserts intra-frames in the compressed stream, as seen on figure 4.

It is worth noting that most video compression schemes allow for forward (P-frames) and backward (B-frames) predicted frames. Needless to say, B-frames increase the latency and the amount of buffering needed in exchange for better compression ratios.

It is important to note that, even though the final objective of image compression is reduction in size, many other factors come into play when discussing actual algorithms. For example, if the reduction
Figure 4: Group Of Pictures (GOP)

is size is such that the time taken for compression and transmission is smaller than transmitting the uncompressed image, we may say that the objective of compression is reduction of the transmission time of the image! Other factors such as latency or delay play a very important role in the design of an image and/or video compression algorithm, as well as random access to past frames, among others.

We will leave the important subject of video compression to another opportunity and focus in image compression for the rest of this document but it is important to note that there can also be progressive video compression which can share the same quality and resolution scalability as image compression and even temporal scalability in some cases.

With the ever increasing size and quality of captured images, even good quality compressed images are quite large and its transmission will most likely take most of the available bandwidth. Depending on the application, we may not need to send an image with full resolution, quality, and color information in order to fulfill our needs. Also, it is usually not practical to simply encode the image multiple times, one for storing and others for each transmission of a desired (resolution, quality, color) image. In the case of retrieval of stored images, it is usually not acceptable to decompress the high quality stored image and recompress it to the desired (resolution, quality, color) image for transmission.

It must be noted that, while image compression quality is usually measured objectively by means of MSE (Mean Square Error), there is a lot of subjectivity in it. It is quite usual to use the HVS (Human Visual System) [8] model to compensate for the human eye characteristics while compressing an image. This will most likely result in a lower quality image in the objective sense being perceived as higher quality image by a human observer.

For example, for the monitoring of security cameras in real time, it is not necessary to have neither high resolution nor high quality but, when searching for details on an image we need all the resolution, quality, and color we can get! Also, while browsing an image sequence database to find a frame that represents a certain moment when an event occurs, there is no need for high quality and resolution and, when this frame is found, it can be retrieved with full quality, resolution, and color so that it can be analyzed in all its available detail. Such a scheme would allow for much faster searches, specially if these searches are done remotely, by minimizing the bandwidth and pro-
Therefore, compression in general and image compression in particular is a very application specific task, with many available tradeoffs and many different algorithms that try to maximize (or minimize) some design criteria. Most image compression algorithms are lossy algorithms designed with the sole purpose of minimizing the resulting size of the image with minimal regard to its execution latency and, in most cases, the whole data is necessary in order to be able to decompress it.

Fortunately, there is a class of image compression algorithms that possess many desirable properties simultaneously and which can also be implemented very efficiently, with complexity similar to the JPEG standard in uniprocessors [9], and with less latency in modern multi-core processors by taking advantage of algorithm parallelism. These algorithms possess the following properties:

- lossless/lossy
- quality scalability
- resolution scalability
- color channel scalability
- random access
- ROI (Region Of Interest)

among others (some algorithms may combine many of the above attributes simultaneously).

Scalable compression usually takes advantage of multiresolution signal decomposition, which is natural for wavelets but can also be used with DCT [10] or other block transforms by simply rearranging its coefficients. The coefficients are sent either in bitplane order across all frequencies (resolution) or frequency order across all bit-planes (quality), or a combination thereof, achieving the desired scalability. During compression special markers can be inserted to separate the color components and also to introduce further blocking within the frequency bands to achieve the desired color scalability and random access, respectively. Also, bitplane shifting can be used to introduce region of interest (ROI), as long as its positions and shapes are transmitted as side information to the decoder.

A few of the known progressive algorithms are EZW (Embedded Zerotree Wavelet) [11], SPIHT (Set Partitioning In Hierarchical Trees) [12], SPECK (Set Partitioning Embedded bloCK) [13], EBCOT (Embedded Block Coding with Optimized Truncation) [14] which is the algorithm used for the JPEG2000 standard [9], among others. It should be noted that there are many extensions and variations for all these algorithms and most of them don’t have an optimized implementation readily available. This means that an efficient implementation, preferably parallel, should be developed so that it can be deployed in a real-world scenario.

For real-time communications a cleverly designed protocol could be used which uses simultaneous channels and receiver feedback so that the transmitter could send low resolution low quality images on the primary channel and, based on the receiver feedback timing, could send additional information on other channels based on available bandwidth. As long as there is a possibility of prioritizing the packet delivery on the primary channel in relation to higher channels, the main image will take precedence over the additional information and will arrive at the destination with minimal delay. The same can be done regarding a second channel in relation to a third one and so on.

Frame rate could also be controlled by using multiple channels or by limiting it so that the primary channel will try to prioritize image quality and resolution once the desired frame rate is achieved.

This way, we can arrange for dynamically adjustable channel usage and prioritize new information and frame rate in relation to resolution or quality information.

3 Conclusions

In this paper, the problem of controlling AUVs-I in real-time and the need to send information to allow a supervised control has been introduced. Communication is a crucial subsystem in any robotic application, specially the ones that permit the user to interact remotely with the system. For that, image compression and transmission is necessary in order to send the required information at the lowest time-delay and without compromising the network and the whole system. To tackle this issue, several progressive image compression and transmission techniques have been presented, considering different ways to connect the robot to the user, such as RF, Sonar and Ethernet connection through the umbilical. Further work will focus on the implementation of the selected algorithm on the AUV-I platform at IRSLab.

Acknowledgements

This research was partly supported by Spanish Ministry of Research and Innovation DPI2011-27977-C03 (TRITON Project), by Universitat
Jaume I and Foundation Caixa Castelló-Bancaixa under grant PI-1B2011-17 and by the programme Ciencia Sem Fronteiras del Brasil.

References


