Real-time Video Streaming from Mobile Underwater Sensors

Seongwon Han, Roy Chen, Youngtae Noh†, Mario Gerla
Dept. of Computer Science, UCLA †Cisco Systems Inc.
{swhan, rschen002, gerla}@cs.ucla.edu, †ynoh@cisco.com

ABSTRACT

Underwater sensor networking is generally regarded as an emerging technology to conduct oceanic exploration and research in an automated and effective manner. As underwater operations become more sophisticated, there is an increasing demand for real-time video streaming. However, real-time video streaming requires high bandwidth as well as low latency. Amongst the resources, bandwidth is the most critical limitation. To help overcome this obstacle, we propose a hybrid solution that combines acoustic and optical communications. Optics provides good quality real time video. Acoustic maintains a "thin" channel for network topology and transmission control, and for still frame video delivery when the optical channel fails. In particular, we enable optical communications by acoustic-assisted alignment and use acoustic communications as a back up when the optical signal is interrupted. The main contribution is to enable reliable, real-time video streaming without underwater optical cables. Another important contribution is the smooth transition between the acoustic and optical video delivery mode, using popular image processing algorithms to compress the video before transmitting it on the acoustic channel.

Categories and Subject Descriptors

C.3 [**Special-purpose and application-based systems**]: Real-time and embedded systems

General Terms

Design, Experimentation

Keywords

underwater wireless networks, acoustic communications, optical communications

1. INTRODUCTION

Although ocean covers more than 70 percent of the Earth surface, a vast majority of it, about 95 percent, remains unexplored according to the National Oceanic and Atmospheric Administration in the United States. Since the ocean can be thousands of feet deep and is difficult for human divers to explore, researchers have turned to underwater sensor networks to gather information in an efficient and automated manner. While traditional sensors provide tabular data (i.e., salinity, temperature, and pressure), recently the need for still images and even real-time video streaming has emerged in the research community. Autonomous Underwater Vehicles (AUVs) have the ability to meet the video demands posed by applications such as ocean bottom monitoring, oil spill detection and search for minerals. If there was a way to provide real-time video streaming from AUVs to support ships, the above mentioned tasks could be carried out much more efficiently in interactive mode.

Traditional underwater modems have used acoustic communications to transmit data through the water using acoustic waves. Acoustic waves travel underwater over distances of several kilometers and do not require direct line-of-sight between the sender and the receiver. However, acoustic communications have disadvantages, namely: the long propagation delay of sound waves compared to electromagnetic or light waves, limited bandwidth and the ease of detection and eavesdropping by the enemy (a critical issue in tactical operations).

Optical communications have received significant attention recently and are the subject of ongoing research [11–14, 21]. Farr *et al.* [14] presented insightful optical communication scenarios and demonstrated video monitoring up to 15 meters. Doniec *et al.* [12] developed AquaOptical II, a bidirectional underwater optical communication system capable of transmitting a few Mbps with the effective range over 15 meters. They further elaborated the previous system mainly focusing on video streaming and successfully transmitted over a range of 25 meters [13]. Generally, optical communications have a higher bandwidth capacity and consume much less energy than acoustics. The speed of light through water is faster than the speed of sound, which results in a lower latency in optical communications. However, light waves have a larger attenuation. At most, optical modems can transmit at distances of about 100 meters [14], in excellent water conditions. Due to the nature of optical communications, a clear line-of-sight is required between the sender and receiver, although there are efforts to overcome this obstacle [6, 7]. This requires for the optical modems to be aligned in order to provide reliable data transmission. It has been reported that optical modems can have a field of view of up to 120 degrees. However, the transmission distance decreases as the transmission angle increases [16].

In this paper, we propose a hybrid solution for real-time video streaming from sensors to a monitoring center (e.g., surface ship). The primary objective is to realize real-time

WUWNET '14 November 12 - 14, 2014, Rome, Italy

http://dx.doi.org/10.1145/2671490.2674582

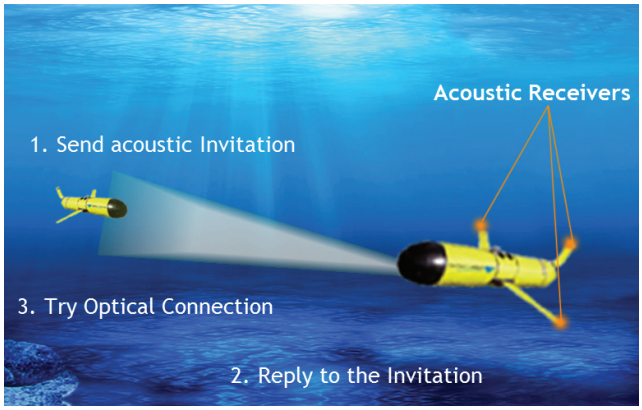

Figure 1: Optical/Acoustic alignment between AUVs

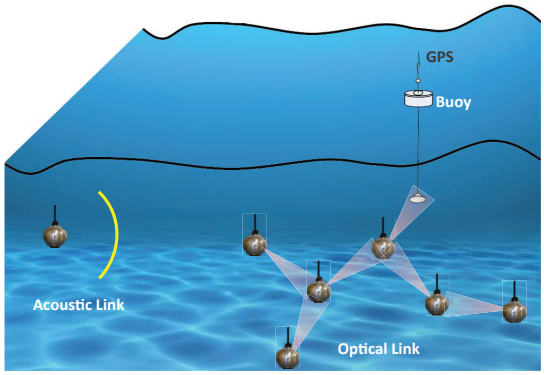

Figure 2: Bottom video exploration

underwater video streaming over both the acoustic and optical modes (where the acoustic mode is the backup mode). However, in case of optical channel failure, the acoustic channel has low bandwidth and long propagation delay, and is not quite adequate to take over the support of the real time application (e.g. remote search of the ocean floor). A possible solution is to dramatically reduce the size and rate of video frames, so that they can fit the acoustic channel and yet allow application interactivity. To this end, we propose to use image processing techniques to compress the video frames and reduce the size to fit the acoustic channel. We use the acoustic modems to transmit ACKs and other control messages to prevent collisions between AUVs.

The rest of the paper is organized as follows. Section 2 describes the hybrid solution proposed in this paper and presents representative scenarios. Section 3 examines image processing algorithms that can reduce the bandwidth required for real-time video streaming. In Section 4, we briefly discuss the implementation of the acoustic system and evaluate the overall performance on tested images. We conclude this paper in Section 5.

2. THE HYBRID SOLUTION

We propose the hybrid solution illustrated in Fig. 1 in order to take advantage of the benefits of both the acoustic and optical communications media. Our primary objective of acoustics is to provide real-time video monitoring between AUVs and Base Station at all times even if no optical link is available. Moreover, the acoustic channel carries back up video frames when the optical channel fails. We propose image processing techniques such as Canny edge detection or Sobel to reduce image size (like vector graphics) to 90 percent smaller than a grayscale original image. This way, we can expect at most 3-5 frames per second (good enough for real-time video monitoring). A low resolution greenscale image is also transferred via Acoustic channel (either predefined interval or specific image which is requested by user). On the other hand, high definition video is always transmitted via optical communications. Naturally, the system should withstand mobility (submarines, drifters and base all drift with the currents). As in Fig. 2, a hybrid acousto-optic mesh can be dynamically configured and can maintain the equivalent of high speed optical links between its nodes. The control is completely supported by acoustic and the high speed data travels on optical links.

2.1 Design

Fig. 1 shows a mobile AUV unit composed of both an acoustic and an optical modem. Also three additional acoustic receivers are included to improve transmission redundancy as well as to provide additional data for localizing another transmitter and aligning the optical modems. In order to carry this out, we use time-difference-of-arrival to determine the relative position of the sender in three dimensions, similar to the technique described by Liu *et al.* [20]. We also use the information from the depth sensor to determine the current node's position on the z-axis, and transmit it to other nodes as necessary. During the alignment process, we continue sending data through acoustic communications, until the optical modems are aligned properly. Once the optical modems are aligned, we send data using optical communication, to take advantage of the increased bandwidth, but continue to use acoustic communication for sending control signals such as acknowledgements (ACKs).

Alignment takes place as follows: the sending node sends a request to the receiving node over the acoustic medium. The receiving node responds to the request with the acoustic transmitter as well, and both sender and receiver use the trilateration information to move towards each other and to get into optical alignment. Once the nodes can communicate over the optical modem and are within range of each other, they can begin to use the optical medium for data transmissions. In the event that there are multiple source nodes present that are transmitting, it is more difficult to determine where the node that is sending the data is located, and to align the potential receiving optical sensor properly. Therefore, previous work [18] includes a preliminary solution to address this issue, which involves the proper node responding to the communication request packet that the sending node transmits to nearby receivers.

2.2 Scenarios for The Hybrid Solution

This section shows a number of possible scenarios which can take advantage of the proposed hybrid solution.

2.2.1 Bottom video exploration

One important reason for using the optical channel underwater is to exploit its high bandwidth (up to several Mbps) for interactive video. The classic application is the deployment of an U/W robot equipped with video camera at great depths from a surface vessel for exploratory or recovery operations. While at low depths the robot can be guided via a cable that carries data and possibly also power; at great depths the guiding cable is not practical, it may get entangled. A proposed solution is to drop a "base station" to the bottom from the surface ship. The base station is stationary and is connected by cable to the surface ship. Multiple robots can roam from the base station in several directions,

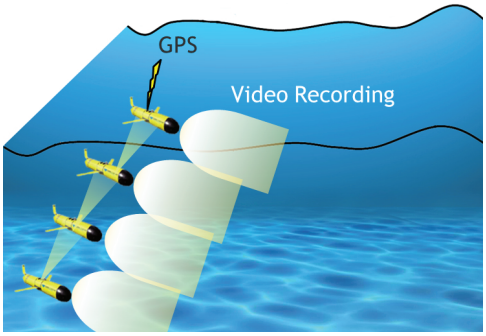

Figure 3: Shallow water inter-submarine video communications

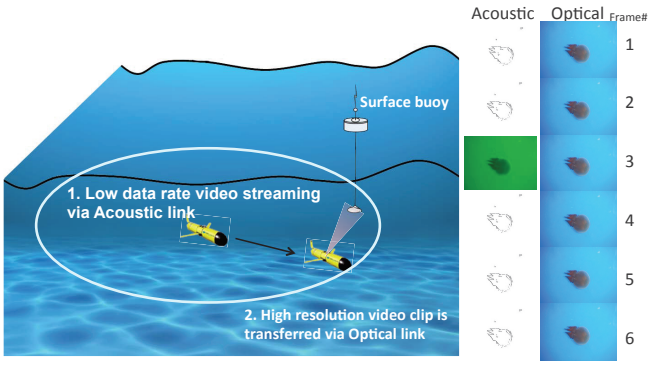

Figure 4: Real-time video streaming via Acoustic/Optical

they carry video cameras that are monitored by operators on the ship and are used to remotely guide the robots. If the robots are at more than 50 m distance from the base station, they will not be able to communicate directly. One interesting scenario is the autonomous deployment of a mesh network that supports one or more video streams from robots to base station and ship. The key idea is to create an optical tree from robots to base station. The data and commands in the reverse direction (base station to robot) are carried via acoustic channels. The short distance, say < 200m, guarantees low delays (< 200ms) even for acoustic propagation and does not compromise real time interactivity. The challenge is to develop a totally distributed algorithm that takes the relay nodes from base station to the field and positions/orients them in the proper way (note, the nodes can localize themselves relative to the base station).

2.2.2 *Shallow water inter-submarine video communications*

Another application of the hybrid concept is the establishment of high speed video connections among a team of mini-submarines participating in a scouting expedition. As depicted in Fig. 3, each submarine sends its video to all others. In this case, the acoustic modems are used to position the submarines and to align their lasers. The submarines provide the optical multihop mesh. In principle, each submarine carries two or more optical transmitters and several optical receivers, so that a mesh can be maintained at all times. As a difference from the sea bottom robot operation, the submarines are equipped with independent, separately oriented lasers, so that they can beam two separate neighbors thus creating a mesh. Moreover, the links are full duplex. There are several challenges: first, the acoustic positioning; second the maintenance of a fully connected acoustic mesh among submarines; third, the alignment of the optical transmitters and receivers and the establishment of a connected, multihop optical mesh; finally, the support of video many to manycast.

2.2.3 *Mixed pure and murky environments*

We can use optical transmissions only in clear waters. In murky water we must use acoustic instead. Data rate must be dramatically scaled down from Mbps to a few hundreds bps (from motion video to still frames). In U/W operations near the surface or in shallow waters, paths conditions may change continuously and intermittently (from clear path to murky path). This is where the hybrid solution shines. We switch back and forth from optic to acoustic channel. If the switchover frequency is very high, it may be more convenient to use the two paths (optical and acoustic) in parallel with combined still frame and video traffic. Network Coding may be considered in this case, to allow the exploitation of both streams in parallel, without having to switch path/technology on a continuous basis. Naturally, the challenges of merging two *Network Coded* streams from channels of such disparate speed and propagation delay are not indifferent.

2.2.4 *Covert AUV to Ship communications*

Suppose a ship deploys several AUVs in a 1000m radius to detect possible attackers or to scout the terrain with cameras. Enemy submarines may be listening and the operation must be covert in the sense that the AUVs should not be detected by submarines when they transmit. A possible strategy works as follows. The ship trails an underwater acoustic/optical mini base station. The base station periodically sends a ping via acoustic communications. When a friendly AUV in the area recognizes the ping, it uses it to align its transmitter to the base station. This enables the establishment of an optical link that cannot be detected by the enemy. In fact, since the ping is repeated periodically, the enemy does not know if there is indeed an AUV in the area.

3. IMAGE PROCESSING AND VIDEO COMPRESSION

Fig. 4 shows an example of a device and how the video would stream. The concept is to compress the video file using image processing techniques. The video quality may be coarse but with every key frames having a greenscale that makes it acceptable until the optical link becomes available again. To test the feasibility of this concept, we implemented and evaluated a few image processing algorithms and examined the file size of the resulting images.

We used a series JPEG images of underwater scenes to conduct our tests. Some level of compression, such as the compression that the JPEG file format uses, is necessary in order to achieve the bandwidth savings. This compression is considered “lossy”, however, in that each time the file is saved and compressed, some data is lost, even if there were no other changes performed to the image. These JPEG images are uncompressed by our programs before they are used; the uncompressed versions are composed of units known as pixels, arranged in a multidimensional array. Each pixel has a set of three RGB values from 0 to 255; these represent the red, green, and blue values for the color that the pixel should display. There is no differentiation between a grayscale image and a full-color image in the JPEG file format [3]. Our results comparing the average file size reductions from each technique are located in Fig. 11.

3.1 Simple Image Processing Techniques

Averaging the RGB values for each pixel, and setting the new RGB values for each pixel to this average converts a full-color JPEG image to grayscale, with the appropriate shade of gray that corresponds to the color. This simple processing algorithm results in a file size reduction to 33 percent of the original size on average. By replicating the same value for all three RGB components of a pixel, we take advantage of the compression scheme and thus are able to reduce the file size greatly.

The blue and green colors travel the longest in the water because of their shortest wavelengths. Therefore, the images taken in the water are always dominated by blue-green color. In fact, red, orange and yellow colors disappear in relatively shallow depth in the water due to their wavelength [22]. Therefore, underwater image enhancing techniques mainly focus on skewed color compensation caused by light scattering and color change [10]. Accordingly, we can use a variant of this technique that we call "greenscaling" to only set the green values of each pixel to this new value, and setting the other red and blue values to zero, resulting in an image with a green hue, but where the prominent features of the image can still be distinguished. This results in a slightly more optimal solution with an average resulting file size of 24 percent of the original, as the value for all the red and blue pixels is the same (zero), thus making the compression even more effective. However, these images still retain the same quality as the grayscale images, and only have an additional green hue. Both the grayscale and greenscale algorithms run in linear time, and are computationally efficient.

Following this, we run a simplistic edge detection algorithm that emphasizes sharp differences in RGB values, traversing the image horizontally. Using the grayscale image as a starting point, we used the mathematical differences in the average of the RGB values to determine where the edges are in the image. We found that this simplistic algorithm reduced the average size to 17 percent of the original. However, this was not a significant improvement over the greenscale algorithm because of the complexity of the resulting images since our algorithm found too many edges in the images. Attempts to remove the edges representing smaller numerical differences did not significantly improve the results. In addition to this, three out of the five resulting images produced pixels that were too lightly colored to detect, making the entire image appear to be nearly white.

Therefore, we decided to investigate more complex edge detection algorithms in order to detect a smaller number of edges with a higher resulting contrast, and to produce a less complex picture (and a smaller image file) as a result.

3.2 Sobel Image Processing

We first examine the Sobel edge detection algorithm for use in our proposed solution [15, 24]. There is ongoing research that would allow for Sobel edge detection to be performed in real time by a FPGA, which could easily be incorporated into an AUV, and would reduce the computational cost of performing this algorithm by converting the software computation into a hardware set of operations [19].

Edge detection algorithms are generally classified as based on either gradient or Laplacian methods. The Sobel algorithm [15] is a gradient algorithm, which means that it looks for the maximum and minimum of the first derivative of the values in the image. The primary component of the Sobel algorithm is the Sobel operator, which we use on a 5x5 block of pixels. While using the default 3x3 operator does provide true edge detection, we found that in pictures with noise the file size was not decreased enough for the Sobel algorithm to be effective. Therefore, we use a larger operator to find the larger edges in the image rather than noise.

We calculate the gradient for a particular pixel twice, in order to determine the derivative by both row and column. Following this, we calculate the magnitude of the gradient vector to get one gradient to perform our Sobel calculations. Instead of handling the different RGB components, we start with the grayscale image and apply the result to the red, green, and blue components of the resulting pixels. Pixels on the same vertical or horizontal axis are factored into the calculations, but are not given as much weight as the pixels directly next to the current one. Pixels to the diagonal of the pixel in question are also factored into the calculations, but are given an even lesser weight compared to the ones that are both directly next to the current pixel and also those on the same direction as the one being analyzed. We illustrate this calculation below, displaying the coefficients that are multiplied by the values of the neighboring pixels, and representing the current pixel as the center:

-5	-4	0	4	5
-8	-10	0	10	8
-10	-20	0	20	10
-8	-10	0	10	8
-5	-4	0	4	5

Table 1: Horizontal Sobel 5x5 mask [15]

5	8	10	8	5
4	10	20	10	4
0	0	0	0	0
-4	-10	-20	-10	-4
-5	-8	-10	-8	-5

Table 2: Vertical Sobel 5x5 mask [15]

Following this, we divide by the sum of the coefficients, in order to obtain a weighted average.

For our specific implementation, the pixels on the border of the image are set to white, as they cannot have a proper gradient calculated; this also decreases the final size of the file, while losing little data. We invert the result of the gradient and set a pixel with a small gradient to white, and one with a large gradient to black. In order to determine the combined magnitude of the horizontal and vertical gradients, we use the following formula:

$$|G| = \sqrt{Gx^2 + Gy^2} \quad (1)$$

where Gx and Gy are the horizontal and vertical components.

The result of this algorithm generates an image that reflects all the gradients of the original image, regardless of how large or small the gradients are. We find that there is a significant file size reduction in most cases, at roughly 27 percent of the original size of the image. In most cases, this new technique outperforms grayscale and greenscaling, and provides usable images where the primary features can be distinguished in all of the five cases, as opposed to the primitive edge detection algorithm. However, this is not the most efficient solution, and thus we make an adjustment since we assume that there are many small gradients that are contributing to the file size.

We discard the gradients that are small, and set those pixels to white to decrease the file size and to emphasize the major edges. Our determination is made based on the RGB value of a pixel: lighter values are numerically greater than

Figure 5: Image in grayscale

Figure 6: Sobel w/o the adaptive algorithm

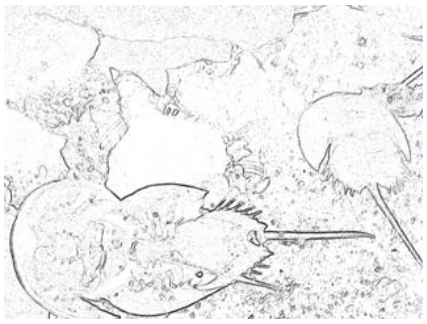

Figure 7: Sobel with the adaptive algorithm

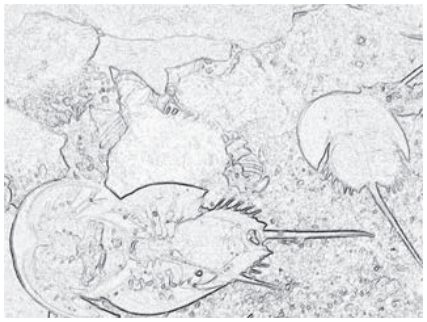

Figure 8: Gaussian and Sobel w/o the adaptive algorithm

Figure 9: Gaussian and Sobel with the adaptive algorithm

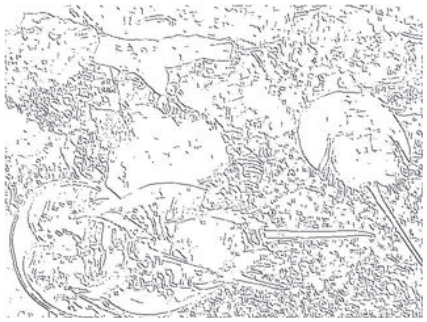

Figure 10: Canny edge detection

the threshold, and darker values are lesser than the threshold. Past attempts to do this with a universal threshold were unsuccessful, as if the threshold was too low, then some of the images would be white or almost entirely white and would be completely useless. But if the threshold was too high, then other images would have very few pixels removed, and have a much larger file size than necessary. Thus, it is necessary to have an adaptive threshold for multiple types of images, including those images with lots of edges and those that do not have them.

To incorporate an adaptive threshold, we begin with a numerical threshold of RGB value = 250, which already eliminates a small portion of the grayscale spectrum. Then, we evaluate the percentage of pixels that represent edges, compared to all of the pixels of the image. If the percentage of pixels representing edges is greater than 25 percent, then we decrease the threshold by 10 and reevaluate the percentage until less than 25 percent of the pixels represent the edges. In this way, lighter edges that are not essential to the understanding of what the image represents are removed. The results can be seen in Fig. 6 and 7.

However, even the 5x5 Sobel operator struggles to handle JPEG images that have a lot of noise, since these edges are detected in the image, resulting in a larger file size due to the unnecessary edges. The adaptive method described above fails to distinguish between darker edges that are essential to the image and those that are a result of noise. Therefore, there is room for improvement in this edge detection method.

3.3 Gaussian Blurring

To counteract the effects of noise in the image on edge detection, we implement the technique of Gaussian blurring [8]. The objective of the blurring is to remove noise from the image by calculating a weighted average of the current pixel as well as the neighboring pixels, and replacing the current pixel with the result. This will proceed to remove sudden color changes that are generally isolated and that are not essential to the understanding of the image by using the average to lessen the effects of an outlier. We use a 5x5 Gaussian matrix, as calculated using the method described at [4] that uses integration over the following Gaussian equation rather than approximation:

$$G(x,y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}} \tag{2}$$

The coefficients are rounded to be integers for ease of calculation, and the standard deviation is set at the value of 1. Following this, we divide by the sum of the coefficients in order to obtain a weighted average. Below is the coefficient matrix, with the weights for the neighboring pixels listed below, and with the center square representing the current pixel:

1	4	7	4	1
4	16	26	16	4
7	26	41	26	7
4	16	26	16	4
1	4	7	4	1

Table 3: 5 x 5 Gaussian matrix [4]

Preliminary tests suggested that a 3x3 operator would not be sufficient, so we chose a 5x5 matrix as a compromise between functionality, risk of making the edges too wide, and concerns about the additional computational time required. We do not calculate the Gaussian blur for the outermost two rows and columns on each edge of the image, since the matrix would extend outside the image, but we do use those pixels in our other calculations as necessary.

Following Gaussian blurring, we perform the Sobel algorithm as described above. This results in a file size reduction

to 19 percent of the original size, which is slightly more efficient than the Sobel-only solutions, and is arguably better than the primitive edge detection in that the quality of the images is much better, considering the difference of only 2 percent in the file sizes.

We also implement the adaptive cutoff method following the Gaussian blurring and Sobel algorithm to further reduce the file size. With this, we are able to reduce the file size to an average of 15 percent of the original size. In addition, this method produced images that contained the essential edges for all five of our test cases. We believe that the adaptive method works better in this case than in the Sobel-only case because the grayscale values have a higher probability of being edges rather than noise, since the pixels representing noise have been reduced. We show the differences in Fig. 8 and 9.

Therefore, we believe this method is the best among the ones studied so far because of the smallest average resulting file size, combined with the usability of all of the output images.

3.4 Canny Edge Detection

We examined a few other edge detection methods for comparison. The first method we evaluated was Canny edge detection [9, 11]. This method builds on Gaussian blurring as well as Sobel edge detection, and adds the additional processing step of finding the direction of the gradient, using the formula and the Sobel gradients calculated in both the horizontal and vertical directions:

$$\theta = \tan^{-1}\left(\frac{\Delta y}{\Delta x}\right) \quad (3)$$

Following Gaussian, we use the technique described in [2] to determine the numerical difference in the direction of the gradient. We use the degree measure of the direction angle found and round to the nearest 45-degree increment. The next step is to determine if a pixel is part of an image. We examine the two neighboring pixels on either side of the current pixel in the direction that we found in the previous step. If the current pixel's magnitude of the gradient is larger than that of both of these neighboring pixels, then we consider it an edge and mark the pixel as black, or (0,0,0) in RGB space. Otherwise, we decide that this is not an edge, and we mark it as white, or (255,255,255) in RGB space.

Then, we use a simplified form of a hysteresis threshold to remove the minor edges. A hysteresis threshold has two numerical values, a higher value and a lower value. Any pixel with a gradient magnitude below the lower threshold is considered to be a minor edge and is removed from the output image, and a magnitude above the higher threshold is considered to be a major edge. A pixel with a gradient magnitude in between the thresholds could be considered part of a minor or major edge, depending on whether this pixel is an essential part of another edge that is considered major. For our purposes, we look at the neighboring pixels and ensure that one of the neighboring pixels is also above the lower threshold to determine if it is a major edge. This choice preserves quality over file size, though it is possible to choose other methods of handling values in between the thresholds. More advanced methods recursively examine the neighbors of the neighboring pixels of the current pixel if the magnitudes of all the neighboring pixels fall in between the higher and lower thresholds.

Using this method with a lower threshold of 15 and a higher threshold of 30, we attain the average of 24 percent of the original file size. However, the average is not a fair representation of this method, because the median is 12 percent,

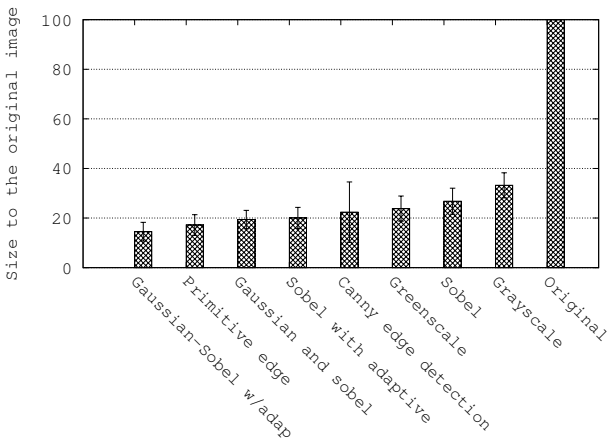

Figure 11: Results of image processing on full-size images(2592x1944)

and there is an outlier of 75 percent that skews the average. With the exception of the outlier, the file sizes are approximately the same, or in some cases slightly smaller, than those generated with the Gaussian and Sobel algorithms with the adaptive method as described in the previous section, as can be seen in Fig. 11. It is possible that with further threshold and/or algorithm adjustments the issue of the outlier would be resolved, as there are many extra edges in the image that can be seen in Fig. 10.

For simplicity, in our evaluations we will use the Sobel and Gaussian algorithms with the adaptive method, since this method consistently provides nearly optimal results, and because the Canny method typically generates files of approximately the same size. However, the Canny edge detection method clearly shows promise for future exploration.

3.5 Other Methods

Using median filtering, where the median of the adjacent pixels and the current pixel is calculated and then used to replace the value of the current pixel, is another option to correct image noise. However, in order to calculate the median of the matrix of pixels, a sorting algorithm must be used [23]. This would not be ideal for the application that we propose, where there are constraints on both computational time (since this is real-time video) as well as the power available to the underwater networking devices.

The other main category of edge detection methods is Laplacian methods, which uses the Gaussian blurring algorithm as the first step to remove noise from the image. Following this, the Laplacian operator uses the second derivative of the image to determine where the edges are. However, this makes the algorithm more vulnerable to the effects of noise, a problem that is prevalent in many remote sensing applications [5]. Therefore, we decided to focus on the other two methods in our design evaluation.

4. EVALUATION

To evaluate our solution, we perform an experiment with a system that takes pictures, applies image processing to them, and then transmits them between computers using acoustic communications.

4.1 Implementation Details

To measure sustainability of acoustic communication, we perform data communication between two AquaSeNT acoustic modems [1] as depicted in Fig. 12. The application allows for a set number of JPEG images to be transmitted

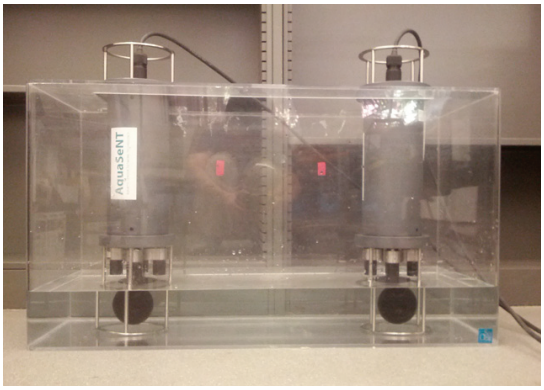

Figure 12: AquaSeNT OFDM modems

over acoustic communication in a small fish tank with a base roughly 2 feet by 1 foot in size. The acoustic modem has a bandwidth of 14-20 kHz, as well as a data rate of 3200 *bps*. This provides an upper limit on the transmission rate and makes it difficult to transfer larger images with the bounded latency of real-time video streaming.

The framework does not use any familiar network protocols since the modem does not include such support. The only features supported are file size verification as well as partial acknowledgment transmissions. In order to perform the transmissions, the image files were resized to be small enough to transmit over the acoustic medium. Before each transmission, both the sending and receiving buffers are flushed before 608 *bytes* are transmitted at a time. The program inserts delays into the transmission time in order to ensure that the receiver has enough time to receive and process the signals.

The images are captured through a webcam using the libav *avconv* package. The pictures are displayed upon successful transmission of the file to the receiving program using the GNU *display* utility.

4.2 Data Rate and Latency

The above system is estimated to have a transmission rate of 83.29 *bytes* per second, or 12.294 seconds per kilobyte (sec/kB), when a baud rate of 9600 *bps* is used. Although the transmission rate can go up to 400 *bytes* per second according to the specifications [1], we are conducting the test in a small water tank so the modems are inevitably affected by severe multipath fading due to tank wall reflections. Using this information combined with the size of the typical image file, we can determine what effects image processing will have on transmission time and consequently, the ability to transmit real-time video.

For our purposes, we run our primary algorithms on 64 pixels by 48 pixels, 128x96, 320x240, and 640x480 size versions of our test images; the results are shown in Fig. 13. While the file size varies depending on the contents of the original image, there are clear trends that reflect the choice of image processing method. As the bandwidth of an acoustic modem is very constrained, it is not feasible to send full-size images (taken by ordinary digital cameras); sending an unprocessed image would take a few hours in the worst case. Using our proposed Gaussian and Sobel method will reduce the transmission time up to twenty minutes. It is a large improvement but it is still unfeasible for real-time streaming.

We note that as the dimensions of the image decrease, the effectiveness of the grayscale, greenscale, and primitive edge detection algorithms increase, while the performance of the Sobel and Gaussian algorithms suffers. In 640x480

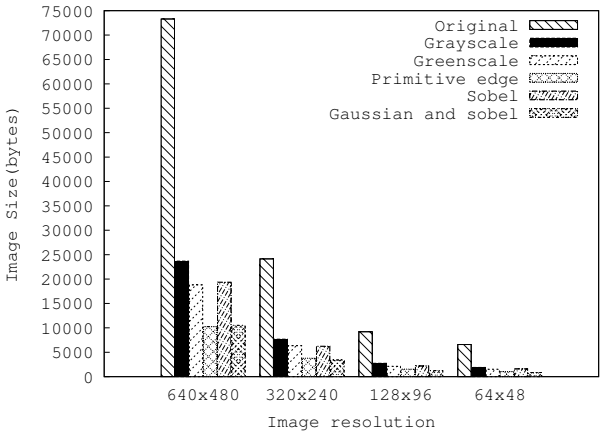

Figure 13: Average file sizes for different image resolutions

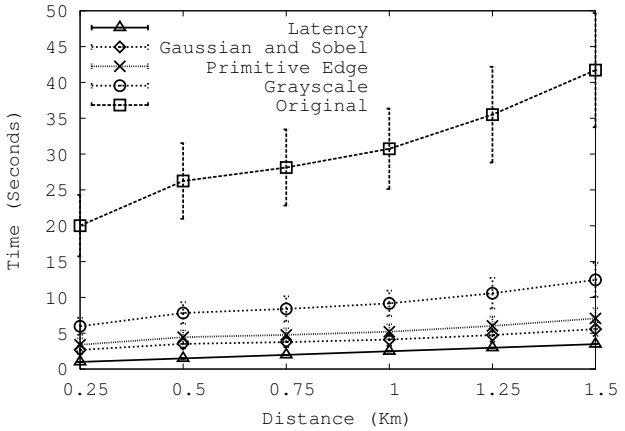

Figure 14: Average time interval for video frame delivery (128x96)

case, the primitive edge detection outperforms all the other algorithms by a small margin but the image quality is often not sufficient. However, the Sobel and Gaussian algorithm combination consistently outperforms the remaining methods both in the image size reduction and the image quality, and outperforms the primitive edge detection method in the 64x48, 128x96 and 320x240 cases.

We also analyze the estimated time needed to transmit the images over the acoustic medium. In the 640x480 case, it takes about 15 minutes to transmit the unprocessed JPEG file from the sender to the receiver. Converting the image to grayscale reduces the latency to about 5 minutes, while Sobel reduces the latency to 3.5 minutes and adding Gaussian blurring results in 2 minutes. While this is a large improvement, this still does not support real-time video transmission. However, we note that with image sizes of 128x96 and 64x48, we are able to transmit the unprocessed images in 113 seconds and 81 seconds, respectively. With the Sobel and Gaussian combination, an image can be transmitted every 16 seconds at a resolution of 128x96, and every 10 seconds at a resolution of 64x48. While not real-time video grade, this rate still allows the user to interact in real time with the environment and, for instance to determine what the sensors are detecting. It is possible that differences in the baud rate, the distance between the sender and receiver, or in the specific model of acoustic modem may improve the latency of the data transmissions. For example, one possible way to get more frames per second is to use multiple

bands for MIMO (Multi-Input Multi-Output) communication for faster overall transmission rate. Another way is to use the interframe prediction-based video compression techniques such as $H.264$, $MPEG-2$, Ogg, and $Theora$. We do not consider this in this paper but assume that if we only transmit the differences of continuous images, we can still send more images. Further improvements in our image processing techniques will continue to reduce latency by decreasing the file size that needs to be transmitted, while retaining the important details of the images.

To evaluate our proposed solution with existing solutions, we also perform simulation and measure elapsed time interval to deliver an image frame with different distances via QualNet. We set data rate to 9600 *bps*, fix packet size to 512 *bytes* and select image resolution = 128x96 pixels. We use recently proposed M-FAMA MAC protocol [17] and measure the latency. As shown in Fig. 14, we note the general behavior, namely the delivery time of all solutions is proportional to distance. Required latency means the time required to deliver one packet from the AUV to destination to support real-time streaming. Gaussian and Sobel outperforms other existing solutions according to average time interval for image frame delivery. It is noteworthy that original image for 250m distance requires 20 seconds but our proposed Gaussian and Sobel only require 2.7 seconds.

5. CONCLUSIONS

In this paper, we investigate a hybrid solution involving both acoustic and optical communications as a step towards real-time video transmission for the underwater mobile sensor networks. The optical channel is "fragile" and intermittent (it can easily break up because of modems misalignment, mobility, water impurity etc). In the proposed hybrid system, when the optical channel quits, the acoustic channel takes over. To this end, the video signal must be compressed to such a level that the acoustic channel can deliver the video (or images) at a rate that allows real time interaction. We have examined the use of image processing as a compression technique in order to reduce the size of the data files. We have evaluated the feasibility of several image processing methods, and have shown that this is a viable technique that makes a significant impact on latency and bandwidth. We have proposed two possible solutions that reduce the images between 10 and 20 percent of the original file size and estimated the latency of the data transfer.

Future work includes analyzing these methods to conform to the computational time and power constraints found in typical underwater networking devices. It is possible that further modifications to the files can be made in order to minimize both file size and computational time, by analyzing the critical points of the algorithm to eliminate bottlenecks and to even parallelize portions of the algorithm. In addition, work needs to be done regarding the extension of these techniques to video transmission such as $H.264$ or $Theora$ (as opposed to JPEG files), as well as the evaluation of additional methods of image compression and processing.

Acknowledgment

This work was supported in part by the US National Science Foundation under Grant No. 1205757 and Northrop Grumman Corporation.

6. REFERENCES

[1] Aquasent ofdm user manual, http://www.aquasent.com/.

[2] Canny edge detection tutorial, http://dasl.mem.drexel.edu/alumni/bgreen/www.pages.drexel.edu/_weg22/can_tut.html.

[3] Jpeg file interchange format, version 1.02, http://www.w3.org/graphics/jpeg/jfif3.pdf.

[4] Spatial filters - gaussian smoothing, http://homepages.inf.ed.ac.uk/rbf/hipr2/gsmooth.htm.

[5] Ali, M., and Clausi, D. Using the canny edge detector for feature extraction and enhancement of remote sensing images. In *Geoscience and Remote Sensing Symposium, 2001. IGARSS '01. IEEE 2001 International* (2001), vol. 5, pp. 2298–2300 vol.5.

[6] Anguita, D., Brizzolara, D., and Parodi, G. Vhdl modules and circuits for underwater optical wireless communication systems. *WTOC 9*, 9 (Sept. 2010), 525–552.

[7] Arnon, S., and Kedar, D. Non-line-of-sight underwater optical wireless communication network. *J. of the Optical Soc. of America A 26*, 3 (Mar. 2009), 530–539.

[8] Basu, M. Gaussian-based edge-detection methods-a survey. *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on 32*, 3 (Aug 2002), 252–260.

[9] Canny, J. A computational approach to edge detection. *Pattern Analysis and Machine Intelligence, IEEE Transactions on PAMI-8*, 6 (Nov 1986), 679–698.

[10] Chiang, J., and Chen, Y.-C. Underwater image enhancement by wavelength compensation and dehazing. *Image Processing, IEEE Transactions on 21*, 4 (April 2012), 1756–1769.

[11] Ding, L., and Goshtasby, A. On the canny edge detector. *Pattern Recognition 34* (2001), 721–725.

[12] Doniec, M., and Rus, D. Bidirectional optical communication with aquaoptical ii. In *Communication Systems (ICCS), 2010 IEEE International Conference on* (Nov 2010), pp. 390–394.

[13] Doniec, M., Xu, A., and Rus, D. Robust real-time underwater digital video streaming using optical communication. In *Robotics and Automation (ICRA), 2013 IEEE International Conference on* (May 2013), pp. 5117–5124.

[14] Farr, N., Bowen, A., Ware, J., Pontbriand, C., and Tivey, M. An integrated, underwater optical /acoustic communications system. In *OCEANS 2010 IEEE - Sydney* (May 2010), pp. 1–6.

[15] Gupta, S., and Mazumdar, S. Sobel edge detection algorithm. *International Journal of Computer Science and Management Research 2* (2013), 1578–1583.

[16] Han, J., Asada, A., and Yagita, Y. Short range high speed ppp-based underwater acoustic network system. In *OCEANS 2008 - MTS/IEEE Kobe Techno-Ocean* (April 2008), pp. 1–4.

[17] Han, S., Noh, Y., Lee, U., and Gerla, M. M-fama: A multi-session mac protocol for reliable underwater acoustic streams. In *INFOCOM, 2013 Proceedings IEEE* (April 2013), pp. 665–673.

[18] Han, S., Noh, Y., Liang, R., Chen, R., Cheng, Y., and Gerla, M. Evaluation of underwater optical-acoustic hybrid network. *Communications, China* (2014).

[19] Kong, W., Cheng, P., and Bi, Z. Real-time sobel edge detector. *Proceedings of the 6th PSU-UNS International Conference on Engineering and Technology* (May 2013).

[20] Liu, S., Zhang, C., and Huang, Y. Research on acoustic source localization using time difference of arrival measurements. In *Measurement, Information and Control (MIC), 2012 International Conference on* (May 2012), vol. 1, pp. 220–224.

[21] Ochi, H., Watanabe, Y., Shimura, T., and Hattori, T. The acoustic communication experiment at 1,600 m depth using qpsk and 8psk. In *OCEANS 2010* (Sept 2010), pp. 1–5.

[22] Schettini, R., and Corchs, S. Underwater image processing: State of the art of restoration and image enhancement methods. *EURASIP J. Adv. Signal Process 2010* (Jan. 2010), 14:1–14:7.

[23] Shapiro, L., and Stockman, G. *Computer Vision*. Prentice Hall, 2001.

[24] Sobel, I. An isotropic 3x3 gradient operator. *Machine Vision for Three-Dimensional Scenes* (1990), 376–379.