Streaming Video over Wireless Networks for Eye Movement Monitoring

Sumedha Swamy, Kaushal Patel and Keshav Seshadri
Carnegie Mellon University
{sumedhas,kaushalp,kseshadr}@andrew.cmu.edu

Abstract—It has been observed that abnormal eye movement can be an indicator of various medical problems such as balance disorders, Diabetic Retinopathy, Strabismus, Cerebral Palsy, Multiple Sclerosis etc. This paper describes an end to end wireless video streaming system that aims at continuously monitoring eye motion and relaying this information to a doctor for immediate diagnosis of a subject. The system aims at being flexible so that even if the subject is in motion, the data would still be collected and continuously streamed over the network. We propose a way of implementing such a system and analyze its performance and the various issues that are involved in building it.

Index Terms— Real -Time Transport Protocol (RTP), Joint Photographic Experts Group (JPEG), Region Of Interest (ROI), Frames per Second (FPS)

I. INTRODUCTION

TODAY’S research suggests that abnormal eye motion can be used to diagnosis subjects and give an insight into the ailment they may be suffering from. Relaying this information to a doctor in a timely fashion would enable a quick diagnosis and aid people suffering from diseases such as diabetic retinopathy which affects nearly 5.7 people in the US and is leading cause of vision loss. This would involve continuously monitoring of the subject’s eyes and constantly updating information on the horizontal, vertical and torsional motion of the eye. An elegant solution to the problem would be the use of a wireless video streaming to send the data in real time. Such a system would not only enable quick diagnosis but could also be portable and hence be able to monitor a patient at all times. In this paper we develop such a system, which with some improvements could be used for such an application.

Figure 1 is a simple diagram of the system we are aiming to build. A pair of cameras are attached to goggles (high speed monitoring system). Two hot mirrors serve as reflectors to reflect images of the eye to the cameras. Two cables connect the cameras to an IEEE 1394b Firewire card which is plugged into a computer. Ideally, this would computer would be a small portable device which could be worn by the subject so that the entire system could be portable.

The portable computer would serve a number of functions, since the camera is simply an image capturing device with no means to analyze the images and transmit them. The computer would acquire the images from the camera, process them and send them over a wireless network to a receiver which would then display the images to a doctor.

The rest of this paper is organized as follows. Section II describes prior work on this area and associated papers that have dealt with wireless streaming and eye monitoring. Section III gives an in depth description of the issues to be addressed while building such a system and then goes on to describe the system we have built. Section IV analyzes the performance of the system. Section V suggests improvements and additions that could be made to it and finally section VI offers concluding remarks.

II. PRIOR WORK

Several papers have dealt with wireless video streaming and its associated applications. [2] dealt with the areas that need to be addressed whenever streaming over a wireless network is involved. It first dealt with the history and motivation for streaming video before moving on to the technical aspects. These include the concept of Intra, Bi-directional and Predicted frames. Between Intra-Frames a full image is not saved, but rather only the parts of the picture that have changed between it and the frame before. These ‘difference frames’ are known as Predicted Frames (P-Frames). As well as P-Frames, Bi-directional frames (B-Frames) are also kept. B-Frames store in between data about the previous frame, and the next frame after it. [3] addressed the challenges to obtaining low latency video, including highly variable delays, losses and bandwidth of 802.11 b wireless networks. To achieve low
building an end to end streaming video system. First the data
journalism (when the media are unable to reach a scene in
people find their way, tracking of small children, photo
primarily intended for aiding in medical diagnosis, [7]
suggests other potential applications such as helping blind
people find their way, tracking of small children, photo journalism (when the media are unable to reach a scene in
time) etc.

[1] serves as our primary reference paper as it focuses on a
high speed eye tracking system and is a primary part of the
application for which our system is suited. It dealt with a
method to track the pupil and acquire required video frames.
These frames will have to subsequently be sent over the
wireless network to the receiver. According to [1], an image of
size 320x240 (Region of Interest) pixels is good enough to
capture the eye, subsequently the image can be reduced to size
120X120 pixels, once the pupil location is determined. This
data will now have to be sent and received.

Based on the information contained in the above papers, we
were able to design our system to suit the application better
and also were able to settle on the protocols required for
transmission of data over a wireless network.

III. SYSTEM IMPLEMENTATION

There are still a lot of areas that need to be addressed while
building an end to end streaming video system. First the data
has to be collected. Camera and drivers need to be configured
to collect images of the eye. For the purpose of implementation we are using the firefly camera and its
associated drivers developed by Point Grey Research (PGR)
[8]. PGR’s firefly camera has a GUI that has been
implemented in Microsoft’s Visual studio 2008. For this
reason we constructed our system to be compatible with it and
used Visual C++ as the software tool for the purpose. Our
system has been designed using one camera but could easily be
extended to work for two. The images collected are of size
640X480 while the camera could operate at speeds of 3.75,
7.5, 15 and 30 FPS which are suitable for the testing of our
system.

This brings us to the second issue, given the bandwidth
constraints the 802.11X wireless technology is the best bet.
802.11a and b have throughputs that could meet the project
requirements provided ROI is implemented carefully. The
video has to be compressed, and it has to be a lossless coding
in the critical region, i.e., the pupil and iris area, because that
is the part of the image that the clinician based their diagnosis
on. We estimate the typical size of the critical region to be
about 200*200 pixels, and the bandwidth requirement to
transmit the area at 30Hz for a monocular system is 9 Mbps.
This makes the bandwidth requirement more manageable. For
the purposes of implementation we have experimented with a
monocular system but the same concepts could easily be
extended to suit the needs of a binocular system.

Packet sequence, arrival of packets in a timely fashion and
compression are further issues that need to be addressed.
Packets that do not arrive in a timely fashion are as good as
lost packets and will have to be discarded as the margin for
error with video/audio streaming is very small. Hence only
small delays of the order of a few seconds will be acceptable.
Going by prior work in this field, RTP (Real Time Transport
Protocol) [9] would be best suited for this application.

Finally the data would be received and decoded to obtain the
images. Real time streaming may not be perfectly implemented
and delays of a few seconds would be acceptable. Visual C++
would be used to extract the data and then display it.

Packet sequence, arrival of packets in a timely fashion and
compression are further issues that need to be addressed.
Packets that do not arrive in a timely fashion are as good as
lost packets and will have to be discarded as the margin for
error with video/audio streaming is very small. Hence only
small delays of the order of a few seconds will be acceptable. Visual C++
would be used to extract the data and then display it.

Now that the fundamental principles outlining the design of a
wireless video streaming system have been outlined, we
present our design of such a system. Our primary aim was to
first build a system that simply captures images and streamed
them over the network at low resolution, low speed and maybe
even low image quality using RTP as the protocol to send
packets. Once this was done, the focus could shift to
improving performance.

Our use of RTP is through the library JRTPLIB [10], which is
an object- oriented RTP library written in C++. The library
offers support for the Real-time Transport Protocol (RTP). It
makes it very easy to send and receive RTP packets as the
RTCP (RTP Control Protocol) functions are handled entirely
internally.

First we set about designing a simple system to just break
down 1 image and send it. The same code was then extended
to handle multiple images and also ensure low lag between
them. A simplified block diagram of the system we have in
place is shown in figure 2. An explanation of each component follows.

Figure 2 – System Block Diagram

1) Point Grey Camera - For the time being we are working
with a single camera that can acquire either black and white or
color images at an adjustable frame rate (maximum of 30 fps
and minimum of 7.5 fps). The camera was set to acquire
640X480 pixel images. Visual Studio codes are in place to
control every operation of the camera and hence present an
easy to use GUI.

2) Isolation of Images - Having setup the camera, our next step
was to understand its control codes to see how it acquired
images and to modify these codes to suit our needs. We were
able to isolate one Visual Studio project named
FlyCaptureTest that performed this function. By modifying this code we were able to isolate images captured by the camera on a frame by frame basis and even save these images in JPEG format (or for that matter any suitable format specified by us). The number of images to be captured could also be set by us.

3) TX/RX of images using JRTPLIB 3.7.1 and Display of Received Images - The next stage in the system is the transmission of the images using JRTPLIB.

First we will go into the details of the sender. The sender must break down the image into packets of equal size and then send them. It first creates an RTP session and sets up the sender and receiver ports as well as the IP address of the receiving machine. It then creates a buffer to store the packets. It then cuts up into packets of size 1400 bytes (RTP specifies this) with 1388 bytes of data and 12 bytes for header information. The main case that has to be taken care of is when the end of an image is reached. This packet is specially marked in order to signify end of transmission. It also has less user data than the other packets. The sender was placed in an infinite loop to continuously send images until terminated by the user. PGR’s GUI display unit was used by us to display all images being sent. This ended the sender code.

The receiver code worked in a similar fashion to receive packets on port 9000. The receiver keeps accessing payload data and displaying it, until the last frame is reached. In addition to this it could display the sequence number of the packet (in order to show that packets are being received in order) and also the length of each packet, which is the same except for the last packet. In order to display images at the receiver we wrote our own code to use forms in visual C++ to acquire each image and display it using something known as a picture box. The form has to keep pace with the images being acquired and hence refresh itself. This required interfacing the receiver code and the display code (i.e. linking of two different visual studio projects and their subsequent simultaneous running).

Thus, we now had an end to end system that captured, displayed, sent and received multiple images. The one channel scheme we implemented sent images using JPEG compression, i.e. lossless compression. The number of images acquired could be as many as required as the images were continuously captured in a while loop which could be exited whenever required by the user. The same principles used to design an end to end image streaming system would have to be extended in order to capture the ROI region of the eye. Thus, the system we have developed would serve as the foundation to implement a 2 channel transmission scheme with only the ROI being sent in the second channel. Also, the ROI channel would have to be sent using reliable lossless coding and without compression, in order to convey full information to the doctor. This is explained in section V.

IV. PERFORMANCE ANALYSIS OF THE SYSTEM

Now that we were able to construct an end to end system, the next step was to analyze it and determine what parameters could be adjusted in order to improve performance. The following tables show the simple tests we conducted in order to gauge performance. In Table 1 the sender and the receiver were in the same room (separated by a few meters), while in Table 2, the sender and receiver were placed in different rooms separated by 50 meters.

<table>
<thead>
<tr>
<th>Camera FPS</th>
<th>Throughput (kbps)</th>
<th>Packets/sec</th>
<th>Frames/sec</th>
</tr>
</thead>
<tbody>
<tr>
<td>TX RX</td>
<td>TX RX</td>
<td>TX RX</td>
<td>TX RX</td>
</tr>
<tr>
<td>3.75</td>
<td>450 440</td>
<td>40.1 40.0</td>
<td>3.74 3.72</td>
</tr>
<tr>
<td>15</td>
<td>680 660</td>
<td>61.9 61.2</td>
<td>6.21 6.14</td>
</tr>
<tr>
<td>30</td>
<td>670 650</td>
<td>60.3 59.6</td>
<td>6.15 6.01</td>
</tr>
</tbody>
</table>

TABLE I: PERFORMANCE WITH SENDER AND RECEIVER IN SAME ROOM

<table>
<thead>
<tr>
<th>Camera FPS</th>
<th>Throughput (kbps)</th>
<th>Packets/sec</th>
<th>Frames/sec</th>
</tr>
</thead>
<tbody>
<tr>
<td>TX RX</td>
<td>TX RX</td>
<td>TX RX</td>
<td>TX RX</td>
</tr>
<tr>
<td>3.75</td>
<td>510 490</td>
<td>46.1 43.6</td>
<td>3.72 3.56</td>
</tr>
<tr>
<td>15</td>
<td>690 630</td>
<td>62.8 59.7</td>
<td>6.46 6.20</td>
</tr>
<tr>
<td>30</td>
<td>690 640</td>
<td>61.8 57.1</td>
<td>5.74 5.41</td>
</tr>
</tbody>
</table>

TABLE II: PERFORMANCE WITH SENDER AND RECEIVER IN DIFFERENT ROOMS

After obtaining these measurements, we concluded that the maximum frame rate we could obtain at the receiver was around 6.5 fps. This meant that the system we built would have to be improved in order to get as close to the actual camera speed. However, we did note that our system was close to the camera speed when it was set at 3.75 fps. In order to capture rapid eye motion, much higher frame rates (of at least 30 fps) would be required. So clearly though 7 fps conveys the impression of continuity of images to the naked eye, to be of more value to a doctor seeing the streaming video, a considerable boost in receiver fps would be required. This would require better synchronization between sender and receiver, a speeding up of the receiver and also a faster camera.

It was also clear that distance between the sender and receiver was not hampering performance as good frame rates at the receiver could even be obtained when it was far away. This is because the frame rate, packet rate and throughput are only dependent on the rate of data being captured by the camera (i.e. its frame rate).

V. POSSIBLE IMPROVEMENTS

There are several additions that could be made to our system in order to boost performance and make it more easy to use.
1) 2 Channel System and ROI – A second channel would have to be created to send the ROI information (which would have to be extracted from each image before it is sent). This channel would have to carry lossless (uncompressed) and reliable data. Since it would be carrying uncompressed data, its bandwidth consumption would be large, so we could save on bandwidth by down sampling the images outside the ROI (360X250 region out of which the pupil actually occupies only a 150X150 region) to obtain low resolution images and then sending these on the first (JPEG) channel. Also, inter-frame video compression methods such as MPEG could be used instead of JPEG. Next, the two channels would have to be combined at the receiver and a method to relate the ROI and the specific bigger image it belongs to would need to be in place. Figures 3 illustrate ROI extraction.

2) Horizontal, Vertical and Torsional Eye Motion - The main issue at hand is whether the horizontal, vertical and torsional eye motion could be conveyed to a doctor to diagnose a patient based on these movements. To do so, a much higher frame rate of the camera as well as the receiver would be required. Also, in order to save bandwidth, the portable computer attached to the subject could do some form of processing and only send relevant information i.e. motion of the eye since last image was sent. This would considerably reduce bandwidth required as only the delta (difference) motion would be sent. Though our system currently works at low frame rates, we are confident that given a faster camera capturing images and a few modifications to the code we have implemented, the system we have built would perform well enough to aid doctors in their diagnosis of a patient.

3) Buffered Displays – In order to obtain smooth playing of received video at a good frame rate, buffering would be needed at the receiver end for both channels. This would ensure that frames would be saved in a buffer and played back once enough of them were saved. The process would continue until the end of transmission.

4) Better GUI and Display Unit – Lastly, we could improve the GUI of our system and also have a better/easier to handle display unit.

VI. CONCLUSIONS

We were successful in designing and building and end to end video streaming system which could capture, transmit and receive images using RTP as the transmission protocol. Packet loss was minimum and good synchronization between the sender and receiver was achieved with a perfect match of images between the two. The system was capable of receiving images with speeds of around 6 FPS and was also able to operate even when the sender and receiver were separated by large distances. Thus, our primary goal of an end to end system was achieved.

Though a lot of work could be done to extend this project for use as an eye tracking system (rather than a general video streaming system), it is our opinion that the basic setup required to do so is in place. Based on the performance of our system, we are confident that given a faster camera capturing images and a few modifications to the code we have implemented, the system we have built would perform well enough to aid doctors in their diagnosis of a patient.

REFERENCES