Abstract—This project is a low-level implementation of a real-
time streaming video server. The idea is to create a server
program on a portable computer with a webcam that will
correctly transmit the video feed over a wireless network to a
client computer located somewhere else on the network. This
paper also investigates other streaming video options that are
already available.

Index Terms—real-time video streaming video, WLAN,
wireless local area network

I. INTRODUCTION

This project consists of two main parts. The first part will be
the implementation of a real-time streaming video server that
can transmit video over a wireless network. The second part
of the project will be investigating other tools for streaming
video that are currently available.

The real-time streaming video server will run on a portable
computer with an attached USB webcam that will relay the
video data back to a client process in real time over a wireless
IEEE 802.11g network. This project can be easily extended to
transmit video data over any Internet Protocol (IP) based
network that supports the User Datagram Protocol (UDP). If
connected to the Internet, the server will even be able to
transmit the video feed across the world in real time. This
could be very useful for a variety of applications, including:
security, robotics, reporting, or even simple personal
communication.

The real-time streaming video server will be implemented
using the Windows Sockets Application Programming
Interface. This interface will allow UDP data transfer between
two computers that are connected on the same network via a
wireless router, or two computers connected through the
Internet. The server will then broadcast the video feed to the
client using the connectionless UDP protocol. This protocol
will allow for faster data transmission, but will not guarantee
the reliable transfer of information. For general video
transmission the loss of a few packets is usually tolerable and
the speed improvement should offset any potential packet
losses.

The low level video data processing will be done with Open
Source Computer Vision, an open source software library
containing several functions that are useful for capturing and
processing digital video.

II. WINSOCK, TCP/IP & UDP

A. WinSock Introduction

The software for this project was written in C++ and will
take advantage of two powerful software application
programming interfaces (APIs), Windows Sockets and
OpenCV (open source computer vision). In this section we
will discuss the Windows Sockets Networking API, more
commonly known as WinSock.

WinSock API is basically a library of functions that can be
used to write network applications. It was created by a large
conglomeration of network programmers and TCP/IP protocol
stack vendors and was built based on the solid precedent of the
Berkeley Sockets API, and includes many of the same
functions as Berkeley Sockets [3]. WinSock Version 1.0 was
first released in 1992, and WinSock 2 will be used in this
project.

B. Network Architecture

Modern network development is based on a layered
approach which allows for the different layers to be designed
and implemented independent from one another assuming each
layer knows how to communicate with the layers immediately
above and below it. Initial network designs were often not
compatible with each other, so the International Standards
Organization (ISO) created a reference model for open
systems interconnection (OSI), which is very commonly
referred to in network architecture discussions. This model is
known as the OSI Reference Model and describes seven layers
of network architecture from the low-level physical layer
which specifies the actual modulation schemes and electrical
properties of the network to the application layer.

In practice different layers of the OSI model are often
combined, and some layers are sub-divided into smaller layers.
Fig. 1, shows how WinSock applications are modeled on a
typical PC, and how that compares with the OSI Reference
Model.

In the case of this real-time video project, the application
will be written in C++, taking advantage of the WinSock API
and the OpenCV visual library. The program will transmit
data using the UDP protocol which runs over IP. Finally,
Medium Access Control (MAC) drivers and Network Interface
Cards (NIC) will be used to send and receive the actual data.

WinSock applications run over the TCP/IP protocol. The
general network model for TCP/IP can be seen in Fig. 2.
C. Transmission Control Protocol

Transmission Control Protocol (TCP) one of the core protocols used for the Internet. TCP typically runs over IP and the entire suite of Internet protocols is often simply called “TCP/IP.” In contrast to IP, which handles low level transmission of data between computers and across the Internet, TCP is a slightly higher level protocol through which data is sent as a continuous byte stream between two computers. TCP is called a connection-oriented protocol because in order to transmit and receive using TCP a connection must first be established between a client and server before the bytes streams can be sent. TCP guarantees that all packets sent will arrive error free by having the sender retransmit any packets that are lost or received with errors. TCP also uses a numbering scheme to ensure that all packets arrive at the destination in the same order that they are sent.

D. User Datagram Protocol

The handshaking that is required for TCP to establish connections and guarantee data leads to high overhead which is unacceptable is some cases. There are also many situations that do not require guaranteed transmission of data and do not need (or even want) a specific connection between computers established beforehand. For that reason the User Datagram Protocol (UDP) is often used.

UDP is known as a connectionless protocol. It is simple and fast, but does not guarantee data with respect to delivery or order of arrival [1]. For these reason, this real-time streaming video project will be using the UDP protocol.

III. WIRELESS COMM. & IEEE 802.11

A. Wireless Communication

Wireless communication using electromagnetic waves has been around for over a century and radio transceivers existed well before the creation of the first transistor in 1947. With the advent of modern computing technologies wireless communication capabilities have skyrocketed.

B. IEEE 802.11 Project

In 1997 the Institute of Electrical and Electronics Engineers (IEEE) released its first standard for wireless local area networks (LAN) called IEEE 802.11-1997. Several other standards eventually came out of this project with the most popular being 802.11b (1999), 802.11g (2003).

More to come on this in the Final Draft…

IV. VIDEO PROCESSING & OPENCV

Open Source Computer Vision (OpenCV) is a library of over 300 C functions that are used for various image processing and computer vision applications. OpenCV contains many complicated algorithms used for face recognition, object detection, image filtering and much more. This project could be extended to utilize these functions to not only capture and transmit video data, but to perform pattern recognition and image enhancement that could be useful for many applications.

Currently OpenCV is used to capture image data from the webcam and store the data in a manageable data structure. At the receiver end the client process will collect the video data and restore the image data structure allowing for playback of the video.

Digital video images are captured one frame at a time. To play back the video, frames are displayed one at a time at a specific frame rate so that the images appear as a motion picture.

V. FINAL RESULTS

As of the time of this writing the results are not final. I have been able to successfully transmit and receive data between a server and a client with WinSock using the UDP protocol. This is a big step toward completion of the streaming video project. I have also successfully been able to capture video
from a webcam and replayed the video back on the same computer. The next step is to actually transmit the video data over the wireless network using UDP. I have done the bulk of the work for this task and all that remains is deciding exactly how to send the actual video data, which should be a fairly simple task.

The flowcharts in Fig. 3, show the design goals of the source code.

VI. ALTERNATE STREAMING VIDEO OPTIONS
As suggested by Dr. Latchman, I will be investigating and comparing the various existing software for streaming video transmission over the Internet. The programs to be evaluated are:

- Real Video
- Microsoft Media
- SHOUTcast Video
- Adobe Flash
- Quicktime

VII. CONCLUSION
In conclusion, I still have plenty of work to do to complete this project. I feel like I am very close to getting the actual video transmission part completed. Once that is done satisfactorily, I can focus more on the research aspect of this project.

APPENDIX
The appendix is currently empty, in the future I will put important pieces of my actual code in this section.

ACKNOWLEDGMENT
There are currently no acknowledgments.

REFERENCES

Christopher S. Hett is the author. He graduated from the University of Florida in August 2008 with a dual Bachelors of Science in Computer and Electrical Engineering. He is currently attending the University of Florida and working on his Masters Degree in the same with an expected graduation date of May 2010.