1. Introduction

In this final MP, you will use all the components built in previous MPs to build a multi-view surveillance system. You will be able to claim that you have some experience with the “large-scale” system after you finish this MP. Trust me! It will be more complicated than you think.

You are required to work on the Linux platform for this machine problem. You can use the Linux workstations in the EWS lab room SC216 and SC220 or you can use your own laptops. You need all two Logitech Webcams your group has borrowed. You also need your own headphone for the audio playback.

Your final MP will be the part of CS414 class project competition. So, start early to deliver an industry standard workable version to win exciting prizes.

2. Problem Description

2.1 System Overview

For this MP, you have to create at least two servers and a client component. They should run on the separate machines. The functionalities of server and client components are given below.

Server

Each server always listens to a port for any incoming client request. A camera is connected to each server to capture live audio and video streams at a fixed rate. We use 30fps for the video capture and 8000Hz for the audio capture. (Note: you have to capture at these fixed rates even though a client may request for lower rates of audio and video).

Client

A client is a separate machine with Graphical User Interface (GUI) that can send a connection request to the servers for video and audio streams of different rates. The client receives audio/video streams from each server, which can be in two different modes: active mode and passive mode. In active mode, the client requests for both audio and video streams. The video rate may vary between 15fps to 25fps depending on the resource availability and the audio rate is fixed to 8000Hz. The video resolution is high in active mode (640X480). However, in passive mode, the
client requests for only the video stream (no audio) with 10 fps, and the resolution of the video stream is low (320X240). A client can request any combination of active and passive modes from the two servers. For example a client can request active mode communication from server 1 and passive mode from server 2 or vice versa. It may also be possible that the client requests streams in active modes from both servers. In that case, you need to implement the MUTE functionality so that only one audio is played. MUTE does not stop the streaming rather drops the audio frames at the client side. Likewise, a client can also request streams in passive mode from both servers.

![Communication Architecture of a Multi-view Video Surveillance System](image)

Figure 1. Communication Architecture of a Multi-view Video Surveillance System

Figure 1 shows the communication details of this MP. The GUI should display two video windows coming from the two servers. The GUI should also allow the users to change modes for any server at any time (without stopping the current stream).

In active mode, the audio is played on the sound card. Remember that, the audio and video should be properly synchronized in active mode. Use your design imagination to implement the GUI nicely.

### 2.2 Functional Components

As a part of this MP, you will use the following functionalities from MP2 (the descriptions are given in MP2 specification).

- Client Resource Admission
- Resource Negotiation between Server and Client
Machine Problem 3: Multi-View Surveillance System  
CS414 Spring 2012: Multimedia Systems  
Instructor: Klara Nahrstedt  

Posted: April 10, 2012  
Due: 5:00pm April 28, 2012

- **Server Resource Admission**  
- **Server and Client Resource Reservation**  
- **QoS Enforcement (Rate control at the server side)**  
- **Data Plane Communication: video streaming, audio streaming**  
- **Audio Video Synchronization**  
- **Session Control:** START, STOP, PAUSE, RESUME, SWITCH, **MUTE**  
- **Session Adaptation:** Rate should be re-negotiated if bandwidth changes  
- **Session Monitoring:** Synchronization skew, jitter, bandwidth, frame rate

Remember that, any changes in the resource.xml should perform the re-admission control and QoS negotiation. The streaming in active mode should be done with the highest possible rate (between 15 to 25fps) considering the client bandwidth defined in resource.txt.

Client must have two windows (on the same GUI) and each window must have their own session control buttons (STOP, START, PAUSE, RESUME, MUTE, SWITCH).

### 3. Required Features

**Video Streaming (10 points):** The client should be able to display the video from the servers in active and passive modes. Playback in active mode requires video with 640X480 resolution and 15 to 25 fps. Playback in passive mode requires video with 320X240 resolution with 10 fps.  

**Audio Streaming (10 points):** The client should be able to play the audio from the server. Note that, in passive mode, there is no audio. However, if there are two active modes, only one audio should be played and other should be muted.  

**A/V Synchronization (10 points):** In active mode of playback, the audio and video captured from the server should be properly synchronized. You will be judged depending on your synchronization perfection.
Machine Problem 3: Multi-View Surveillance System
CS414 Spring 2012: Multimedia Systems
Instructor: Klara Nahrstedt

Posted: April 10, 2012  Due: 5:00pm April 28, 2012

**Resource Admission and Negotiation (10 points):** You should implement a simple resource admission and negotiation protocol. The server and client bandwidths are given in resource.txt file at each machine.

**QoS Enforcement (Rate control) (10 points):** You should create a proper rate control mechanism so that the network always gets the packet on the negotiated rate. You can use leaky bucket, token bucket or gstreamer-java rate control mechanism. You should not stop any time while switching modes.

**Session Control (20 points):** The client GUI should implement the STOP, START, PAUSE and RESUME functionalities. You should also be able to SWITCH modes of the viewing across the servers at any time. In addition, MUTE button functionality should be implement that can be used to mute the audio in active mode communication. You should not stop any time while switching modes.

**Session Monitoring (20 points):** Display the monitoring output on the GUI in the text form. You must monitor failure rate, jitter, frame rate, synchronization skew and bandwidth at the client for individual streams.

**Report Writing (10 points):** Write a clean report describing your approach, algorithm and assumptions. You should submit two reports: development manual and user manual. The development manual should describe your design approach, assumptions and architecture. On the other hand, the user manual should contain instructions on how to compile and run your code.

### 4. Bonus Features

**Graphical Plot of Monitoring (10 Points):** Similar to MP1, showing the GUI for the monitored output (graphs, plots) will be awarded 10 points as bonus points. Remember that to get the bonus point full, you have to implement all monitoring graphs and append them nicely in your video display GUI.

### 5. System Functionality

The design architecture of server side distribution is same as MP2. Therefore, no modification is required at the server side if you have already implemented all the functionalities of MP2. However, for getting streams from two servers, you have to maintain two threads at the client side.

Here I give an example scenario. Suppose, average video frame size for active mode is 15000bits and audio bandwidth is 8000*16 = 128000bps. The available bandwidth at the client is 931000bps. When the client request for active mode to
server 1, the admission control at the client uses the formula: $931000 = (15000*fr) + 128000$ (the general formula is given in MP2 lecture slide) to compute the possible video frame rate $fr$. Since, here the value $fr$ is higher than 25, the server 1 is negotiated to deliver 25fps of video with 8000Hz audio (provided that server can support it). The bandwidth used in this communication is 503000bps ($=15000*25 + 128000$). The remaining bandwidth at the client is 428000bps. Now, suppose, client also requests for an active mode from server 2. However, the required bandwidth (503000 bps) to support in 25fps video and required audio is higher than the current available bandwidth (428000bps) at the client. So, client will try to lower the video frame rate. To find the appropriate video frame rate ($fr$), the client would use formula: $428000 = (15000 * fr) + 128000$, which gives $fr = 20$fps. Therefore the active mode with server 2 would be negotiated with 20 fps of video and 8000Hz audio. Your GUI should display all these computations. Remember, this is a sample computation. Your average frame size might be different depending on your compression format. The available bandwidth is given at resource.txt.

Every time a user changes the mode in the client the above computation should be done. If a client requests for active mode audio and video from both servers, one of them should be muted. However, the client can switch the MUTE functionality across the videos.

6. Submission

Pack all source codes and documents into a zip file or tar ball and submit them through Compass. Do not submit your solutions through email unless there are technical problems with the Compass system. The submission deadline is April 28th at 5:00 pm. You can get up to two days bonus for each MP, but please remember you can use only 3 bonus days throughout the semester. Further late submissions are not accepted and will get 0 point.

6.1 Source Code

You should only submit your source codes (.java, .c or .cpp files), Makefile, and any open source libraries (or jar files) you use in your solution into compass. Do not include any pre-compiled obj files (.o), binary execution, or pre-recorded media files in the directory.

6.2 Documentation

Your documentation should include two parts: user manual and development manual. The user manual should include all instructions on how to compile and
Machine Problem 3: Multi-View Surveillance System
CS414 Spring 2012: Multimedia Systems
Instructor: Klara Nahrstedt

Posted: April 10, 2012  Due: 5:00pm April 28, 2012

install your source code, and how to run your program and test all features. If you have designed any GUI, it is better to attach some screen shots to explain. The development manual should have the implementation details of all features. The implementation details include program flow, key data structures; media file formats, important algorithms, and so on.

7. Evaluation

The evaluation will be done by face-to-face interview with each group. You will run your program. We may ask you to show the source code implementing different functionalities. Your solution is evaluated based on how many features you have implemented and demonstrated. In order to get full score (100 points) of this MP, you need to implement all 100-point required features. 10-point optional features will be used to offset your lower marks at MPs, homework’s or exams. Evaluation will happen on Monday April 30th. We will divide evaluation of those who are using laptop to minimize the wireless inference. Groups demonstrating with laptops will be interviewed at room SC 3401 at 5:00 pm. However, groups who are planning to use wired network with lab machines will be interviewed at lab SC216 at 6:00 pm. A sign-up sheet will be provided (by email or using compass) to schedule your demonstration slot. You MUST use at least THREE pcs for the demonstration.

[Please Attend MP3 Lecture on Friday April 13th, 2012. TA will discuss various design alternatives and gstreamer functionalities that you can use for implementing this MP.]

[A list of evaluation case studies will be posted later and will also be discussed in the class. Your code must be able to to meet those case studies.]