Today’s Lecture

- Coding
  - Pthreads / etc.
- End-to-End Application Issues
  - Case Study - Skype

Discussion - Tracker

- Tracker purpose
  - Record nodes that are participating
  - “Bootstrap” phase
- How might we design it?
  - TCP, UDP?
  - Data structures
  - C / C++
Discussion - Client

- Client serves two roles
  - Actual client (main thread)
    - Respond to console input
    - Search / get files from other clients
  - Server (pthreads)
    - One pthread - Accept new connections
    - ? pthreads – Handle those connections

Code Sketch – Accept in a pthread

- See wiki / notes for code

```c
void *listenNewRequests(void *pData)
{
  // Convert the pointer / etc.
  while (1)
  {
    clientSocket = accept(serverSocket...);
    // Spawn a new pthread
    pthread_create(handleClient,...);
  }
}
```

This is your loop to listen / accept new incoming client connections

```c
void *handleClient(void *pData)
{
  // Convert the pointer / etc.
  // Read some from the socket
  // Parse the command
  // Write some to the socket
  // Close the socket and done
}
```

This handles each client as they arrive to the server

Discussion - Structure

- Two+ threads running
  - Console input – cin, connect / query, etc.
    - Work based on the list from the tracker
  - Listen thread – listening on the data port
    - Dynamically process new requests
End-to-End Data

Outline
  Case Study – Video Chat

Example Application – Video Chat

• Skype
  – Logon to a central server

• Think about it
  – Well-known address
    • Leverage DNS
    • Resolve for IP address

Chat - Video

• How might a chat work?
  – Connect directly?
  – Connect through third party?

We’ll come back to this in a later lecture, for now just assume we can make it work.
Chat - Text

- Text chat
  - Just send the data
  - Two threads
    - Main -> Reading in your console input
    - Reader -> Read / display data from the other side

Add in Video / Audio

- Digitization of audio / video
  - A/D and D/A
  - Sample rate
  - Resolution
  - What does it mean?

Latency

- We need to stay synchronized
  - Audio / video
- Uncanny
  - Something just seems weird
  - Need to keep within bounds of normal human capabilities
Processing

- Cycle time for processors
  - TP ~ 100 ms [50-200 ms]
  - TC ~ 70 ms [30-100 ms]
  - TM ~ 70 ms [25-175 ms]
- Concepts – MHP
  - Slowman, Middleman, Fastman
  - Fastman may be 10x faster than Slowman

What does that mean for us?

- Where do we get latency?
  - Sampling delay
  - Processing
    - Read from microphone / camera
    - Write packet to OS
  - Packetization delay
  - Make a packet on the network
  - Physical delay
    - Raw transmission time
    - Propagation delay
    - Congestion delay
    - Other packets / flows
  - Device driver
    - Receive packet / deliver to OS
  - App delay
    - Note that packet has come in
    - Queue packet for playback
Thought Exploration - Skype

• How does it work?
  – How is Skye different from listening to a radio stream?
  – Is it easier or harder?

Project 4 – Building a server to do immersive applications with interactions