LIVE VIDEO STREAMING APPLICATION

CS536 COURSE PROJECT

(I agree to release my video presentation on Youtube)

Sahithi Kodali - 0034789866

PURDUE UNIVERSITY | Department of Computer Science

Agenda

1. Objectives
2. Design and Implementation
   a. Server End
   b. Web GUI Client Implementation
   c. Android Application Client Implementation
   d. New Design Implementation
3. Deploying Android Application
4. Demo
5. Performance Evaluation
6. Challenges and Future Work

Objectives

- Create an user-friendly platform that transmits live video from mobile devices to remote servers in real-time.
- Aims to facilitate seamless communication through live video-streaming.
- Involves developing a reliable server side infrastructure, a cutting-edge mobile application that leverages the power of live video-streaming, enabling users to connect, share, and interact in real-time.

Design and Implementation

Server end

- Create a **UDP socket** and bind it to a **specific IP address and port.** This socket is used to receive data (i.e., video frames) from a client.
- The video streamed by the client is sent as image frames in the form of bytes. The data received as bytes is unpickled and decoded as image frames and displayed on the server end.
- If the client closes the connection, the server ends and the performance metrics are computed. Finally the socket is closed.

Design and Implementation

Web GUI Client Implementation

- A library called “**tkinter**” is used to create a basic GUI (Graphical User Interface) with 2 buttons “**start stream**” and “**stop stream**”.
- The video stream is captured using OpenCV’s “**cv2.VideoCapture()**”. OpenCV is a library that provides functions for image/video processing.
- When the “start stream” button is pressed, a start_stream method is called to initialize a UDP socket and set socket’s send buffer size.
- The update_gui method is called to start updating GUI and stream video as image frames, thereby display the captured frame.
- Each frame is encoded as a **JPEG image using “cv2.imencode”** and the encoded frame is **pickled and sent as bytes** to server via socket.
- When the “stop stream” button is pressed the stop_stream method is called and the server receives stop signal and closes the stream. The performance metrics are computed.

Design and Implementation

Android Application Client Implementation

- Utilizes **Kivy framework** (a python library) to develop android based application GUI for video streaming.
- GUI is defined using a vertical "**BoxLayout**" that contains several widgets. The widgets used are "**TextInput**" to enter server's IP address, "**Image**" for displaying streamed video, "**Camera**" for capturing video frames from device's camera.
- "**Start Stream**" and "**Stop Stream**" buttons are used for initiating and stopping the video streaming, which works similar to the web GUI framework.
- The update method is called periodically (every 1/30 of a second as scheduled by clock.schedule_interval).
- Converts the new frame pixels to a numpy array and processes the frame to be displayed in the Kivy "Image" widget. The frames are **encoded as JPEG** and sent to server as bytes along with the timestamp via UDP socket.
- When the "stop stream" button is pressed the stop_stream method is called and the server receives stop signal and closes the stream. The performance metrics are computed.

Design and Implementation

New Design Implementation

- Implemented **compression technique** to reduce network usage by reducing the amount of data that needs to be transmitted over network.
- Utilizes "**zlib**" library to enable data compression.
- "**zlib.compress**" is used to compress the pickled frame data before sending it to server.
- At the server side the received data is decompressed using "**zlib.decompress**" and the bytes are decoded to display the video frames.

Deploying Android Application

- To deploy the client code in an android application an “APK (Android Application Package) file” needs to be installed on the mobile device.
- **“Builddozer”** is a tool that works with Kivy to develop standalone packages that can be deployed on android device.
- Contains a “**builddozer.spec**” file which is a configuration file used to specify various settings of application such as permissions, requirements, and dependencies.
- To build the android package the command “**builddozer -v android debug**” is executed from the folder where client is located.
- This APK file can now be installed on the phone to run the application.
- **ADB (Android Debug Bridge)** is used to debug the application from the connected computer by enabling USB debugging (Settings -> Enable Developer Mode -> Enable USB debugging) on the mobile device.

Demo

Web GUI & Android Application demo

Performance Evaluation

Without Compression

Server side

```
Connection closed from client
Average Latency at server: 0.0005859429495675224 seconds
Average Received Data at server: 8223.57142857143 bytes
Average Network Usage at server: 8223.57142857143 bytes
Video streaming ended
```

Client side

```
Streaming stopped
Average Latency at client: 0.000395550046648298 seconds
Average Sent Data at client: 8223.57142857143 bytes
[INFO   ] [Base        ] Leaving application in progress...
```

With Compression

Server side

```
Connection closed from client
Average Latency at server: 0.0005190936663678584 seconds
Average Received Data at server: 5135.486005089058 bytes
Video streaming ended
```

Client side

```
Streaming stopped
Average Latency at client: 0.0002875880127341389 seconds
Average Sent Data at client: 7493.900763358779 bytes
Average Compressed Sent Data at client: 5135.486005089058 bytes
[INFO   ] [Base        ] Leaving application in progress...
```

Challenges and Future Work

- The main challenge was to find the issues of deploying android application and debug to resolve the issues.
- Implement more algorithms like adaptive bitrate streaming, dynamic frame rate adjustment etc., along with data compression methods to reduce video streaming latency and network usage.
- Develop more GUI features to enhance live video streaming experience.

THANK YOU