**ĐẠI HỌC ĐÀ NẴNG**

**TRƯỜNG ĐẠI HỌC BÁCH KHOA**

**KHOA CÔNG NGHỆ THÔNG TIN**



**PBL4: DỰ ÁN HỆ ĐIỀU HÀNH & MẠNG MÁY TÍNH**

**Xây dựng hệ thống thu nhận hình ảnh từ webcam qua internet**

**SINH VIÊN THỰC HIỆN:**

**Võ Trần Quốc Khánh LỚP: 22T\_DT1 NHÓM: 22Nh13A**

**Nguyễn Mạnh Dũng LỚP: 22T\_DT1 NHÓM: 22Nh13A**

**GIẢNG VIÊN HƯỚNG DẪN:**

**Th.S Nguyễn Văn Nguyên**

**Đà Nẵng 11/ 2024**

MỤC LỤC

[1](#_Toc181793014)

[MỤC LỤC 2](#_Toc181793015)

[DANH SÁCH HÌNH VẼ 3](#_Toc181793016)

[DANH SÁCH BẢNG BIỂU 4](#_Toc181793017)

[GIỚI THIỆU ĐỀ TÀI 5](#_Toc181793018)

[CHƯƠNG 1. CƠ SỞ LÝ THUYẾT 6](#_Toc181793019)

[1.1. Mô hình Client-Server 6](#_Toc181793020)

[1.2. Lập trình đa luồng ( Multi-Thread ) 6](#_Toc181793021)

[1.3. Lập trình Socket TCP 6](#_Toc181793022)

[1.4. Xử lý đồng bộ 7](#_Toc181793023)

[CHƯƠNG 2 : TRIỂN KHAI 7](#_Toc181793024)

[2.1. Lựa chọn công nghệ sử dụng 7](#_Toc181793025)

[2.2. Triển khai phần giao diện 7](#_Toc181793026)

[2.3. Triển khai mô hình client-server trong truyền dữ liệu – Lập trình socket TCP 7](#_Toc181793027)

[2.4. Triển khai đa luồng 8](#_Toc181793028)

[2.5. Triển khai xử lý đồng bộ 8](#_Toc181793029)

[CHƯƠNG 3. Kết quả chương trình – Đánh giá 9](#_Toc181793030)

[3.1. Chương trình phía Client 9](#_Toc181793032)

[3.2. Chương trình phía Server 9](#_Toc181793033)

[3.3. Đánh giá 10](#_Toc181793034)

[CHƯƠNG 4. KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN 10](#_Toc181793035)

[TÀI LIỆU THAM KHẢO 11](#_Toc181793036)

[PHỤ LỤC 12](#_Toc181793037)

DANH SÁCH HÌNH VẼ

[Hình 1. Mô hình Client-Server 8](#_Toc181741074)

[Hình 2. Chương trình phía Client 11](#_Toc181741075)

[Hình 3. Chương trình phía Server 12](#_Toc181741076)

DANH SÁCH BẢNG BIỂU

[Bảng 1. Mô tả mô hình client-server 9](#_Toc181741847)

GIỚI THIỆU ĐỀ TÀI

Đề tài “Xây dựng hệ thống thu nhận hình ảnh từ webcam qua internet” hướng tới phát triển hệ thống cho phép thu thập và truyền tải hình ảnh được trích xuất từ webcam tại máy khách(client) đến máy chủ(server) thông qua internet từ đó server có thể tiếp nhận và hiển thị các hình ảnh ấy lên màn hình. Bên cạnh đó hệ thống còn hỗ trợ điều chỉnh các thông số (tần suất gửi, chất lượng nén, kích thước ảnh) từ xa ở cả phía client và server. Mục tiêu của đề tài là có thể xây dựng giải pháp gửi nhận ảnh thời gian thực với hiệu suất cao để có thể ứng dụng trong giám sát hoặc truyền tải hình ảnh.

# CHƯƠNG 1. CƠ SỞ LÝ THUYẾT

## Công nghệ sử dụng

Chương trình sử dụng ngôn ngữ lập trình Java với 2 thư viện hỗ trợ JavaFx và OpenCV dùng để xây dựng giao diện cho phía Client và cả Server.

## Giao diện hệ thống

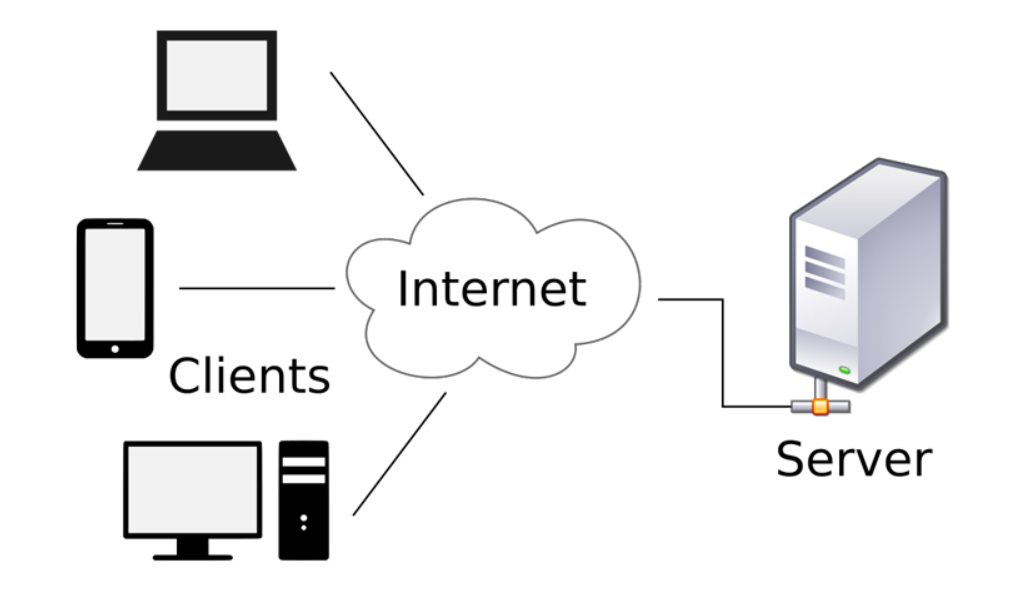
Bên phía Client, giao diện được xây dựng bao gồm một Sence chính chứa tất cả các phần tử khác, một ImageView được gán VideoCapture của thư viện OpenCV để hiển thị hình ảnh chiếu từ camera thiết bị , một propertiesBox là các thuộc tính của hình ảnh chiếu bao gồm tần suất (frequency), kích cỡ (size : width x height ) và chất lượng nén khi gửi (compression ). Ngoài ra các button cần thiết để gửi thông tin và đóng - mở camera cũng được thêm vào để thực hiện các chức năng trên.

Thư viện OpenCV được sử dụng trong việc đóng - mở camera, chuyển đổi định dạng ảnh sang định dạng số MatofBytes để gửi qua Server, lật khung nhìn camera để cho ảnh hiển thị tự nhiên nhất.

Bên phía Server, giao diện được xây dựng gồm một ScrollPane chứa một mảng VBox là thành phần chứa giao diện của từng Client.

## Mô hình Client-Server

Client Server là mô hình mạng máy tính mà trong đó, các máy tính con sẽ đóng vai trò như máy khách, chúng làm nhiệm vụ gửi yêu cầu đến các máy chủ. Sau đó, máy chủ sẽ xử lý yêu cầu và trả kết quả về cho máy khách.

Trong chương trình, Client sẽ có vai trò là bên gửi thông tin bao gồm ảnh và các thuộc tính của ảnh, Server sẽ là bên nhận các thông tin gửi từ Client và hiển thị trên màn hình.

Hình . Mô hình Client-Server

## Lập trình đa luồng

Lập trình đa luồng là kỹ thuật một ứng dụng có thể thực thi nhiều phần công việc một cách song song, đồng thời.

Đối với mô hình client-server được ứng dụng trong hệ thống thì việc nhiều client cùng lúc gửi ảnh đến cho server là điều đương nhiên do đó hệ thống có sử dụng lập trình đa luồng.

Được ứng dụng mỗi khi có client kết nối đến server thì chương trình phía server sẽ tạo ra “Thread” tương ứng với client ấy và lưu vào danh sách các thread, các thread này sẽ được chương trình phía server kiểm soát và xử lý song song.

* Định nghĩa các Thread:

## Lập trình Socket TCP

Lập trình socket TCP là kỹ thuật giao tiếp giữa client và server bằng giao thức TCP(Transmission Control Prototol). Giao thức TCP là giao thức có liên kết đảm bảo việc truyền dữ liệu một cách tin cậy, gửi các gói tin theo thứ tự và không bị mất mát. Khi thực hiện truyền dữ liệu thì phía server sẽ mở một socket để lắng nghe các kết nối từ client trên một cổng nào đó, phía client cũng sẽ mở socket kết nối đến IP và cổng kết nối của server. Sau khi kết nối được thiết lập, việc truyền tin được diễn ra thông qua socket cho đến khi ngắt kết nối.

## Xử lý đồng bộ

Xử lí đồng bộ liên quan đến việc điều phối truy cập tài nguyên sử dụng chung nhằm tránh gây ra lỗi khi nhiều luồng cùng muốn truy cập vào tài nguyên này, đảm bảo rằng tại một thời điểm chỉ có một luồng chiếm đượt miền găng và thực thi khối mã được đồng bộ hóa.

Đối với hệ thống thì việc các client kết nối và ngắt kết nối với server diễn ra liên tục nên giao diện sẽ có nhiều thay đổi mỗi khi như vậy, do đó việc nhiều thread cùng thực thi chỉnh sửa giao diện cùng lúc nếu không xử lý sẽ gây xung đột do đó cần áp dụng xử lí đồng bộ hóa ở đây để đảm bảo giao diện được cập nhật mà không gây ra lỗi.

# CHƯƠNG 2 : XÂY DỰNG HỆ THỐNG CLIENT-SERVER

## 2.1. Lựa chọn công nghệ sử dụng

Chương trình sử dụng ngôn ngữ lập trình Java với 2 thư viện hỗ trợ JavaFx và OpenCV dùng để xây dựng giao diện cho phía Client và cả Server

## 2.2. Triển khai phần giao diện

## 2.3. Triển khai mô hình client-server trong truyền dữ liệu – Lập trình socket TCP

Trước tiên, server sẽ tạo ServerSocket để lắng nghe các kết nối từ client trên cổng “5000”.

* ServerSocket serverSocket = *new* ServerSocket(5000)

Tiếp đến, client để kết nối đến server cũng sẽ tạo một socket kết nối đến IP của server thông qua cổng “5000”.

* Socket socket = *new* Socket("10.10.56.171", 5000)

Ở phía client và server khi bắt đầu truyền thông đều sử dụng vòng lặp “while true” để liên tục gửi và nhận dữ liệu (các tham số ảnh, mảng byte chứa thông tin ảnh).

Mô tả quá trình:

|  |  |  |
| --- | --- | --- |
| Bước | Client | Server |
| 1 | Kết nối đến Server | Chấp nhận kết nối từ Client |
| 2 | Chụp ảnh từ webcam |  |
| 3 | Điều chỉnh các thông số(tần suất, kích thước, chất lượng nén ảnh) nếu cần |  |
| 4 | Gửi ảnh đến Server | Nhận ảnh từ Client |
| 5 | Lặp lại bước 2-4 | Nhận và hiện thị ảnh lên màn hình |
| 6 | Nhận và điều chỉnh các thông số gửi từ Server(nếu có) | Gửi các điều chỉnh đến Client(nếu cần) |
| 7 | Lặp lại bước 2-6 | Lặp lại bước 4-6 cho mỗi Client |

Bảng . Mô tả mô hình client-server

## 2.4. Triển khai đa luồng

Hệ thống bao gồm 1 server- nhiều client , các client cùng lúc lấy ảnh từ webcam gửi đến server cho nên ở phía server mỗi khi có một client gửi yêu cầu đến thì sẽ tạo một luồng(thread) để gửi nhận thông tin với client này, server sẽ quản lý danh sách các thread này để vận hành hệ thống.

Định nghĩa thread:

* *class* ServerThread *implements* Runnable

Tạo thread và lưu vào danh sách các thread được server quản lý

* ServerThread serverThread = *new* ServerThread(clientSocket,timeConnect);  
  clientThreads.put(clientSocket.getInetAddress().getHostAddress(), serverThread);  
  Thread thread = *new* Thread(serverThread);  
  thread.start();

## 2.5. Triển khai xử lý đồng bộ

Với việc nhiều client cùng gửi dữ liệu đến server, thì việc cập nhật giao diện để hiển thị ảnh từ client gửi đến cần được xử lý đồng bộ để tránh lỗi khì các thread cùng lúc cập nhật giao diện.

Có 2 hàm cần xử lý đồng bộ, đó là hàm xử lý khi client kết nối đến(displayImage) và hàm xử lý khi client ngắt kết nối(removeClient)

Trước tiên tạo một đối tượng final(lock) – sử dụng đối tượng này như là “tài nguyên chung” :

* *private static final* Object lock = *new* Object();

Tiếp đến thực thi đồng bộ cho 2 hàm xử lý phía trên:

Hàm xử lý khi một client kết nối đến server

* *public static void* displayImage(parameter … )

*{*

*synchronized* (lock){ //triển khai hàm}

*}*

Hàm xử lý khi một client ngắt kết nối

* *public static void* removeClient(parameter … )

{

*synchronized* (lock) { // triển khai hàm }

}

## 3.1. Chương trình phía Client

Chương trình hiển thị được khung camera của Client và phần nhập thông số cho hình ảnh được gửi đi về phía Server.

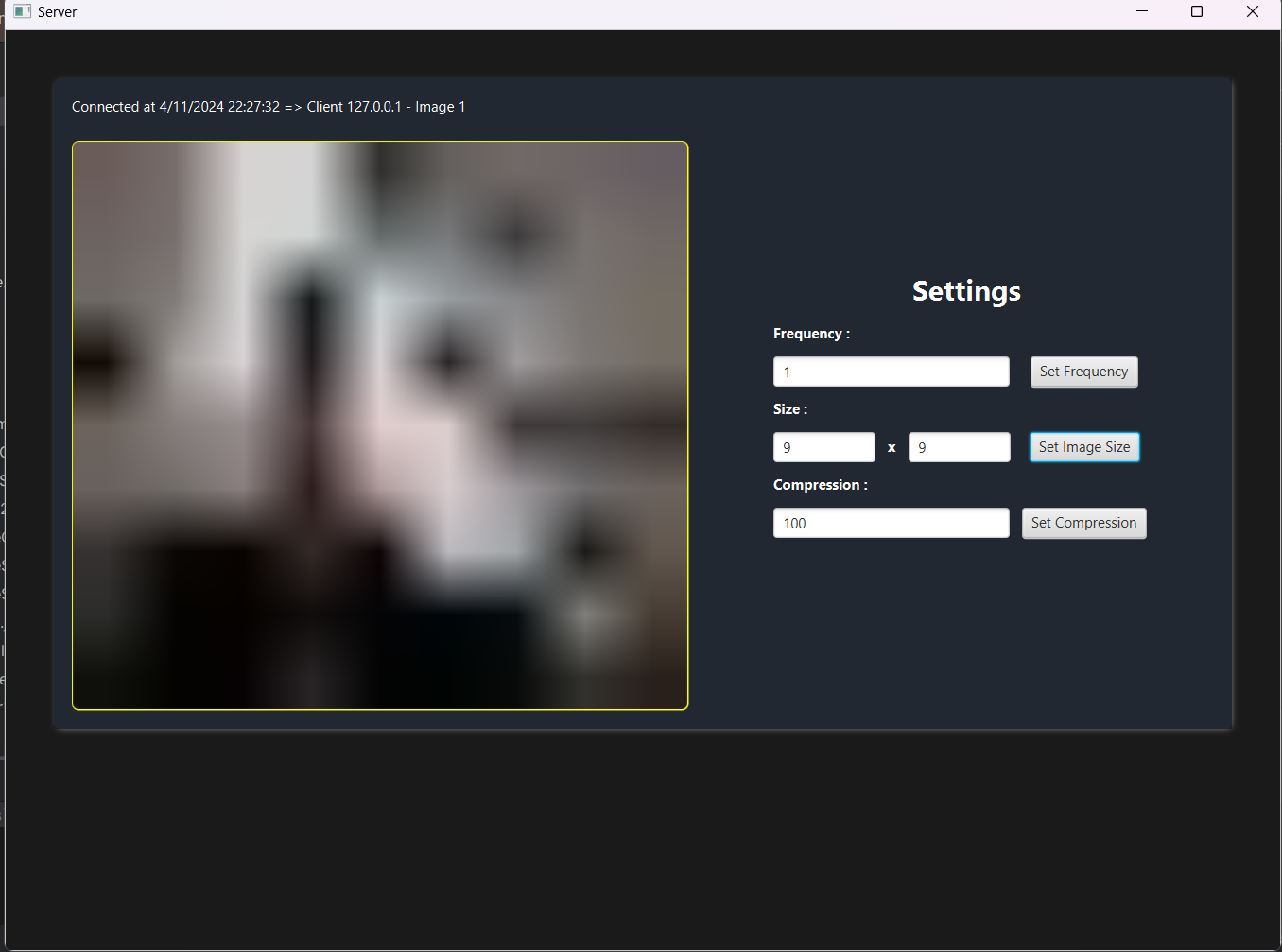
A screenshot of a computer

Description automatically generated

Hình . Chương trình phía Client

## 3.2. Chương trình phía Server

Chương trình hiển thị được giao diện nhận được từ Client và thông số liên quan, thêm vào đó là phần nhập thông số cho hình ảnh tương tự như Client để gửi lại về Client.



Hình . Chương trình phía Server

## 3.3. Đánh giá

Với việc sử dụng OpenCV để lấy thông tin từ các frame với tốc độ nhanh nên việc gửi ảnh có độ trễ rất thấp, nếu giảm tần suất gửi xuống gần bằng không thì có thể thấy được ảnh được gửi đi ở client, nhận và hiển thị ở phía server với sự chêch lệch rất thấp. Đây cũng là lý do chúng tôi sử dụng giao thức TCP để gửi ảnh thay vì UDP.

So sánh việc dùng giao thức UDP và TCP để gửi ảnh nói riêng và các thông tin truyền thông(media) nói chung thì dùng UDP sẽ được ưu tiên hơn, cho ra tốc độ gửi nhanh hơn do các tính chất của UDP như là: không tại liên kết trong quá trình truyền, không duy trì thứ tự các gói tin, không đảm bảo việc mất gói tin,…

Việc dùng giao thức TCP đối với hệ thống này vừa có thể gửi nhận ảnh với tốc độ nhanh vừa có thể đảm bảo độ tin cậy trong lúc truyền ảnh cùng với các tham số kèm theo.

# KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

Hệ thống gần như đã hoàn thành được các yêu cầu đề ra( client chụp ảnh từ webcam và gửi đến server , server nhận ảnh và hiển thị lên màn hình, server có thể nhận ảnh từ nhiều client, ở cả client và server đều có thể chỉnh sửa các thông số, … )

Hệ thống có thể tiếp tục phát triển chức năng gửi nhận âm thanh, hình ảnh màn hình, … để ứng dụng vào việc giám sát, kiểm tra,…

TÀI LIỆU THAM KHẢO

[1] <https://viblo.asia/p/tim-hieu-giao-thuc-tcp-va-udp-jvEla11xlkw>, 28/08/2024

[2] <https://codelearn.io/sharing/lap-trinh-socket-co-voi-tcpip-ava>, 28/08/2024

[3]<https://codelearn.io/sharing/tim-hieu-ve-mo-hinh-client-server>,21/08/2024

PHỤ LỤC