

**TRƯỜNG ĐẠI HỌC PHENIKAA**

**KHOA CÔNG NGHỆ THÔNG TIN**

**------------oOo-----------**

**Báo cáo bài tập lớn Thị Giác Máy Tính**

**ĐỀ TÀI: XÂY DỰNG GAME ỨNG DỤNG MÔ HÌNH NHẬN DIỆN KHUÔN MẶT VÀ LANDMARKS**

**NHÓM THỰC HIỆN : NHÓM 1**

**GIẢNG VIÊN HƯỚNG DẪN : PHẠM TIẾN LÂM**

**KHÓA : K15**

**THÀNH VIÊN NHÓM : VŨ THÀNH ĐẠT- 21010589**

**NGUYỄN MINH ĐỨC - 21010602**

**NGUYỄN HOÀNG DƯƠNG – 21013110**

**HÀ NỘI, 07/2024**

# **Bảng Phân Công Công Việc Theo Tiến Độ**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Vai trò** | | **Tên SV1** | **Tên SV2** | **Tên SV3** |
| Vũ Thành Đạt | Nguyễn Minh Đức | Nguyễn Hoàng Dương |
| **Milestone** | **Nội dung công việc** |  |  |  |
| I | Xây dựng ý tưởng đề tài | 30% | 30% | 40% |
| Xây dựng kế hoạch đề tài | 30% | 30% | 40% |
| II | Nghiên cứu về các thuật toán nhân diện khuôn mặt và sử dụng Yolov8 | 25% | 50% | 25% |
| Thu thập dữ liệu ảnh khuôn mặt của nhưng người được nhận diện | 33.33% | 33.33% | 33.33% |
| Nghiên cứu về thư viện MediaPipe và sử dụng Face Landmarks | 25% | 25% | 50% |
| Code đưa mô hình vào sử dụng | 25% | 25% | 50% |
| III | Tìm hiểu và lập trình game với Godot | **40%** | 30% | 30% |
| Viết báo cáo | 33.33% | 33.33% | 33.33% |
| Toàn bộ dự án |  | 30% | 30% | 40% |

# **MỞ ĐẦU**

Hiện nay, cùng với sự phát triển của xã hội, vấn đề án ninh bảo mật đang được yêu cầu khắt khe ở mọi nơi trên thế giới. Các hệ thống nhận dạng con người được ra đời với độ tin cậy ngày càng cao. Một trong các bài toán nhận dạng con người rất được quan tâm hiện nay là nhận dạng khuôn mặt. Vì nhận dạng khuôn mặt là cách cơ bản và dễ dàng nhất mà con người dùng để phân biệt nhau. Bên cạnh đó, ngày nay việc thu thập, xử lý thông tin qua ảnh để phân biệt đối tượng đang được quan tâm và ứng dụng rộng rãi. Với phương pháp này, chúng ta có thể thu nhận được nhiều thông từ đối tượng mà không cần tác nhiều đến đối tượng nghiên cứu.

Một trong những bài toán được nhiều người quan tâm nhất của lĩnh vực xử lý ảnh hiện nay đó là nhận dạng khuôn mặt (Face Recognition). Như chúng ta đã biết, khuôn mặt đóng vai trò quan trọng trong quá trình giao tiếp giữa người với người, nó mang một lượng thông tin giàu có, chẳng hạn như từ khuôn mặt chúng ta có thể xác định giới tính, tuổi tác, chủng tộc, trạng thái cảm xúc, đặc biệt là xác định mối quan hệ với đối tượng (có quen biết hay không). Do đó, bài toán nhận dạng khuôn mặt đóng vai trò quan trọng trong nhiều lĩnh vực đời sống hằng ngày của con người như các hệ thống giám sát, quản lý vào ra, tìm kiếm thông tin một người nổi tiếng,…đặc biệt là an ninh, bảo mật. Có rất nhiều phương pháp nhận dạng khuôn mặt để nâng cao hiệu suất tuy nhiên dù ít hay nhiều những phương pháp này đang vấp phải những thử thách về độ sáng, hướng nghiêng, kích thước ảnh, hay ảnh hưởng của tham số môi trường.

Trong khuôn khổ đồ án này, chúng em sẽ tiếp tục giải quyết bài toán nhận dạng offline. Trong đó đối tượng được thu thập thành các file dữ liệu và được chuyển về trung tâm. Tại đó, các số liệu sẽ được phân tích xử lý. Chúng em sẽ lấy đó làm nền tảng để có thể xây dựng các ứng dụng lớn hơn sau này.

## **DANH MỤC VIÊT TẮT**

|  |  |  |
| --- | --- | --- |
| **STT** | **Chữ viết tắt** | **Viết đầy đủ** |
| 1 | AI | Artificial Intelligence |
| 2 | CNN | Convolution Neural Network |
| 3 | MTCNN | Multi-task Cascaded Convolutional Networks |
| 4 | DL | Deep Learning |
| 5 | TF | TensorFlow |
| 6 | ML | Machine Learning |

**MỤC LỤC**

[Bảng Phân Công Công Việc Theo Tiến Độ 2](#_Toc171804911)

[MỞ ĐẦU 3](#_Toc171804912)

[DANH MỤC VIÊT TẮT 4](#_Toc171804913)

[CHƯƠNG 1. TỔNG QUAN 7](#_Toc171804914)

[1.1. Lí do chọn đề tài 7](#_Toc171804915)

[1.2. Mục tiêu đề tài 7](#_Toc171804916)

[**1.2.1 Mục tiêu tổng quát** 7](#_Toc171804917)

[**1.2.2 Mục tiêu cụ thể** 7](#_Toc171804918)

[**1.2.3 Phương pháp nghiên cứu** 8](#_Toc171804919)

[**1.2.4 Hướng thực hiện** 8](#_Toc171804920)

[**1.2.5 Ngôn ngữ và các framework sử dụng** 8](#_Toc171804921)

[2.1 Các phương pháp nhận diện khuôn mặt 10](#_Toc171804922)

[**2.1.1 Nhận diện sử dụng thuật toán PCA** 11](#_Toc171804923)

[2.2 Mô hình chính sử dụng Yolov8 Face Detection 13](#_Toc171804924)

[2.3 Mạng nơ-ron tích chập (Convolutional Neural Network) 14](#_Toc171804925)

[**2.3.1 Định nghĩa** 14](#_Toc171804926)

[**2.3.2 Cấu trúc đơn giản của CNN** 15](#_Toc171804927)

[2.8 Ngôn ngữ lập trình C++ 18](#_Toc171804928)

[2.9 Một số thư viện hiện nay 18](#_Toc171804929)

[**2.9.1 TensorFlow** 18](#_Toc171804930)

[**2.9.2 Sklearn** 19](#_Toc171804931)

[**2.9.3 Keras** 19](#_Toc171804932)

[**2.9.4 Dlib** 20](#_Toc171804933)

[**2.9.5 OpenCV** 20](#_Toc171804934)

[CHƯƠNG 2: Mô hình Face Landmarks nhận diện đóng mở miệng 23](#_Toc171804935)

[2.1 Giới thiệu về MediaPipe 23](#_Toc171804936)

[2.2 Face Mesh 24](#_Toc171804937)

[CHƯƠNG 3. XÂY DỰNG GAME BẰNG CÔNG CỤ LÀM GAME GODOT 27](#_Toc171804938)

[3.1 Các giải pháp đã có 27](#_Toc171804939)

[3.2 Giải pháp đề xuất 29](#_Toc171804940)

[3.3 Mô hình hệ thống / Thiết kế giải pháp 30](#_Toc171804941)

[**3.3.2 Quy trình xây dựng game** 31](#_Toc171804942)

[**3.3.2.1 Xây dựng các thành phần của game** 32](#_Toc171804943)

[**3.3.3 Mô hình GameObject và Scene** 33](#_Toc171804944)

[**3.3.3.1 Player** 34](#_Toc171804945)

[**3.3.3.2 GameManager** 35](#_Toc171804946)

[**3.3.4 Các màn hình giao diện người dùng** 36](#_Toc171804947)

[**3.3.5 Thiết kế của đạn** 37](#_Toc171804948)

[1.Kết quả đạt được 38](#_Toc171804949)

[2. Hướng phát triển: 38](#_Toc171804950)

[CHƯƠNG 4: MỘT SỐ THÀNH PHẦN KHÁC CỦA ĐỒ ÁN 39](#_Toc171804951)

[4.1 Kế hoạch dự án 39](#_Toc171804952)

[4.2 Đảm bảo thực hiện đúng làm việc nhóm 39](#_Toc171804953)

[4.3 Các vấn đề về đạo đức và làm việc chuyên nghiệp 40](#_Toc171804954)

[4.4 Kế hoạch cho kiến thức mới và chiến lược học tập 41](#_Toc171804955)

[DANH MỤC TÀI LIỆU THAM KHẢO 42](#_Toc171804956)

# 

# **CHƯƠNG 1. TỔNG QUAN**

## **1.1. Lí do chọn đề tài**

Hiện nay việc nhận diện gương mặt đã trở nên cần thiết đối với nhiều ngành nghề, lĩnh vực khác nhau. Việc xây dựng một hệ thống gương mặt có thể đáp ứng được mọi nhu cầu sử dụng vẫn đang là một vấn đề nan giải mặt dù đã tồn tại rất nhiều mô hình nhận diện gương mặt.

Bên cạnh đó với mong muốn xây dựng một dự án mang lại sự thú vị mà cũng ứng dụng những kiến thức đã học nên nhóm em quyết định chọn xây dựng một game nho nho trên nền tảng godot có sử dụng mô hình nhận diện.

## **1.2. Mục tiêu đề tài**

### **1.2.1 Mục tiêu tổng quát**

Xây dựng một ứng dụng game trên nền tảng godot và tích hợp nhúng mô hình model nhận diện khuôn mặt và nhận diện sự đóng mở miệng bằng landmark để điều khiển nhân vật.

### **1.2.2 Mục tiêu cụ thể**

|  |  |
| --- | --- |
| **Tính cụ thể (Specific)** | Tìm hiểu một số phương pháp Học máy điển hình, áp dụng thuật toán trên tập dữ liệu đã thu thập, đánh giá kết quả của hai thuật toán để từ đó chọn lựa được một mô hình phù hợp với độ chính xác cao. |
| **Tính đo lường**  **(Measurable)** | Thu thập hình ảnh và thông tin của sinh viên, với mỗi sinh viên thì thu thập khoảng từ 20-30 bức ảnh để đưa vào quá trình huấn luyện, sau đó thực hiện tiền xử lý hình ảnh để kết quả dự đoán cao hơn 90%. |
| **Tính khả thi (Achievable)** | Dự án được xây dựng, và có thể chạy thử nghiệm nhận diện khuôn mặt và miệng |
| **Tính thực tế (Realistic)** | Phạm vi nghiên cứu của đề tài này là phù hợp với trình độ của sinh viên thực hiện cũng như là kết quả mà đề tài mang lại là phù hợp với tình hình thực tế hiện nay. |
| **Tính thời hạn**  **(Timely)** | Đề tài sẽ hoàn thành theo đúng tiến độ của thời hạn báo cáo. |

### **1.2.3 Phương pháp nghiên cứu**

Đề tài này sử dụng một vài phương pháp nghiên cứu nhưng chủ yếu là phương pháp phân tích và tổng kết kinh nghiệm. Cụ thể là nghiên cứu từ những paper và document có trên mạng từ đó đưa ra phân tích và hướng làm cụ thể.

### **1.2.4 Hướng thực hiện**

Để hoàn thành dự án một cách khoa học và đúng tiến độ, nhóm em quyết định chia ra thành từng giai đoạn như sau.

- Giai đoạn 1: Nghiên cứu và tìm hiểu mô hình Yolo v8, đưa mô hình này hoạt động và nhận diện được khuôn mặt.

- Giai đoạn 2: Nghiên cứu và tìm hiểu mô hình MediaPipe của Google, sử dụng Face Landmarks, tìm ra các điểm landmarks của khuôn miệng rồi đưa mô hình vào hoạt động nhận diện được khuôn miệng mở hay đóng.

- Giai đoạn 3: Tìm hiểu về Godot, một nền tảng lập trình game để có thể xây dựng UI đơn giản ứng dụng 2 mô hình của giai đoạn trên.

- Giai đoạn 4: Nhúng mô hình vào godot ( ngôn ngữ nhóm sử dụng là C++ với framework QT)

### **1.2.5 Ngôn ngữ và các framework sử dụng**

Đối với việc sử dụng mô hình, chúng em sử dụng ngôn ngữ lập trình là C++ cùng với sự hỗ trợ của framework là QT. Lí do sử dụng là do C++ có thể hỗ trợ hiệu năng thực tế cao hơn với việc xử lí trên GPU.

Về phần lập trình game thì nhóm sử dụng Godot với ngôn ngữ lập trình GDScript, với giao diện thân thiện, sử dụng dễ dàng và tiện lợi.

Đối với phần khó nhất là nhúng code C++ vào Godot thì sử dụng GDExtensio

**CHƯƠNG 2. MÔ HÌNH NHẬN DIỆN KHUÔN MẶT VỚI YOLO V8**

* 1. **Các phương pháp nhận diện khuôn mặt**

Hiện nay các phương pháp nhận dạng mặt được chia thành nhiều hướng theo các tiêu chí khác nhau: nhận dạng với dữ liệu đầu vào là ảnh tĩnh 2D là phổ biến nhất, tuy nhiên tương lai có lẽ sẽ là 3D (vì việc bố trí nhiều camera 2D sẽ cho dữ liệu 3D và đem lại kết quả tốt hơn, đáng tin cậy hơn), cũng có thể chia thành 2 hướng là: làm với dữ liệu ảnh và làm với dữ liệu video. Trên thực tế người ta hay chia các phương pháp nhận dạng mặt ra làm 3 loại: phương pháp tiếp cận toàn cục (Global) như Eigenfaces - PCA, Fisherfaces - LDA; phương pháp tiếp cận dựa trên các đặc điểm cục bộ (Local Feature Based) như LBP, Gabor Wavelets và phương pháp lai (là sự kết hợp của phương pháp toàn cục và phương pháp cục bộ). *Phương pháp nhận dạng khuôn mặt dựa trên các đặc điểm cục bộ:* Đây là phương pháp nhận dạng khuôn mặt dựa trên việc xác định các đặc trưng hình học của các chi tiết trên một khuôn mặt (như vị trí, diện tích, hình dạng của mắt, mũi, miệng,…) và mối quan hệ giữa chúng (như khoảng cách của hai mắt, khoảng cách của hai lông mày,…).

Phương pháp nhận dạng khuôn mặt dựa trên xét toàn diện khuôn mặt:Nhận dạng dựa trên toàn diện khuôn mặt, có nghĩa là sẽ không đi xét đến từng thành phần đặc trưng trên khuôn mặt nữa mà sẽ xem khuôn mặt là một không gian cụ thể và sẽ tìm những đặc trưng, những đặc điểm chính trên không gian đó.

Phương pháp dựa trên các đặc điểm cục bộ đã được chứng minh là ưu việt hơn khi làm việc trong các điều kiện không có kiểm soát và có thể nói rằng lịch sử phát triển của nhận dạng mặt là sự phát triển của các phương pháp trích chọn đặc trưng (Feature Extractrion Methods) được sử dụng trong các hệ thống dựa trên feature based. Các ứng dụng cụ thể của nhận dạng mặt dựa trên 2 mô hình nhận dạng: identification (xác định danh tính, bài toán 1-N), và verification (xác thực danh tính, bài toán 1-1). Trong bài toán identification, ta cần xác định danh tính của ảnh kiểm tra, còn trong bài toán verification ta cần xác định 2 ảnh có cùng thuộc về một người hay không.

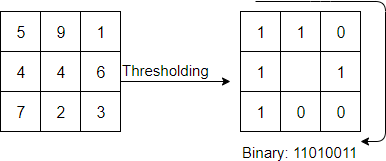
**2.1.1 Nhận diện sử dụng thuật toán PCA**

Kohonen đã đưa ra phương pháp dùng vector riêng để nhận dạng khuôn mặt, ông dùng một mạng neural đơn giản để chứng tỏ khả năng của phương pháp này trên các ảnh đã được chuẩn hóa. Mạng neural tính một mô tả của khuôn mặt bằng cách xấp xỉ các vector riêng của ma trận tương quan của ảnh. Các vector riêng sau này được biết đến với cái tên Eigenface . Kirby và Sirovich chứng tỏ các ảnh có khuôn mặt có thể được mã hóa tuyến tính bằng một số lượng vừa phải các ảnh cơ sở. Tính chất này dựa trên biến đổi Karhunen-Lòeve, mà còn được gọi dưới một cái tên khác là PCA và biến đổi Hotelling. Cho một tập các ảnh huấn luyện có kích thước n x m được mô tả bởi các vector có kích thước m x m, các vector cở sở cho một không gian con tối ưu được xác định thông qua lỗi bình phương trung bình khi chiếu các ảnh huấn luyện vào không gian con này. Các tác giả gọi tập các vector cơ sở tối ưu này là ảnh riêng sau đó gọi cho đơn giản là vector riêng của ma trận hiệp phương sai được tính từ các ảnh khuôn mặt đã vector hóa trong tập huấn luyện. Nếu cho 100 ảnh, mà mỗi khuôn mặt có kích thước 91x50 thì có thể chỉ dùng 50 ảnh riêng, trong khi vẫn duy trì được một khả năng giống nhau hợp lý (giữ được 95% tính chất).

Tương tự, dùng PCA trên tập huấn luyện ảnh các khuôn mặt để sinh các ảnh riêng (còn gọi là Eigenface) để tìm một không gian con (không gian khuôn mặt) trong không gian ảnh. Các ảnh khuôn mặt được chiếu vào không gian con này và được gom nhóm lại. Tương tự các ảnh không có khuôn mặt dùng để huấn luyện cũng được chiếu vào cùng không gian con và gom nhóm lại. Các ảnh khi chiếu vào không gian khuôn mặt thì không bị thay đổi tính chất cơ bản, trong khi chiếu các ảnh không có khuôn mặt thì xuất hiện sự khác nhau cũng không ít. Xác định sự có mặt của một khuôn mặt trong ảnh thông qua tất cả khoảng cách giữa các vị trí trong ảnh và không gian ảnh. Khoảng cách này dùng để xem xét có hay không có khuôn mặt người, kết quả khi tính toán các khoảng cách sẽ cho ta một bản đồ về khuôn mặt. Có nhiều nghiên cứu về xác định khuôn mặt, nhận dạng, và trích đặc trưng từ ý tưởng vector riêng, phân rã, và gom nhóm. Sau đó Kim phát triển cho ảnh màu, bằng cách phân đoạn ảnh để không gian tìm kiếm giảm xuống.

**2.1.2 Nhận diện sử dụng phương pháp LBP**

Mẫu nhị phân cục bộ (Local Binary Pattern - LBP) là phương pháp trích chọn kết cấu của ảnh thành vector đặc trưng gọi là đặc trưng LBP. Ý tưởng ban đầu của phương pháp này được tác giả Ojala và các đồng nghiệp giới thiệu, ban đầu xử lý trên ảnh xám. Ví dụ với một điểm ảnh xé t 8 điểm lân cận và sử dụng chính giá trị của điểm ảnh tại trung tâm để phân ngưỡng 8 giá trị lân cận. Kết quả tìm được một chuỗi bit nhị phân có chiều dài bằng 8 tương ứng với 8 điểm lân cận được định nghĩa theo thứ tự nhất định. Chuỗi bit sẽ được chuyển sang hệ thập phân và giá trị thập phân này thay thế giá trị ban đầu của điểm ảnh đang xét. Số điểm lân cận có thể được thay đổi bằng đại lượng bán kính. *Hình 2.1* mô tả quá trình tính toán LBP cho bài toán nhận dạng mặt người.



Hình 1: Mô tả tính toán với bán kính 1 điểm ảnh và lấy mẫu 8 điểm lân cận

Ưu điểm của phương pháp LBP là có chi phí tính toán thấp, ổn định khi cường độ thay đổi đơn điệu và dễ mở rộng lên không gian nhiều chiều như ảnh màu trong hệ RGB. Tuy nhiên, nhược điểm của đặc trưng dựa trên gradient là:

Đầu tiên, gradient phác họa cường độ phân phối xung quanh điểm ảnh một cách khá thô do chỉ biểu thị bởi hai giá trị 0 và 1. Hai mức gradient giống nhau có thể thuộc hai đối tượng cục bộ khác nhau, do đó không rõ ràng.

Thứ hai, thông tin về gradient không đủ biểu diễn cho các đối tượng trong trường hợp gặp môi trường phức tạp gây ra nhiễu.

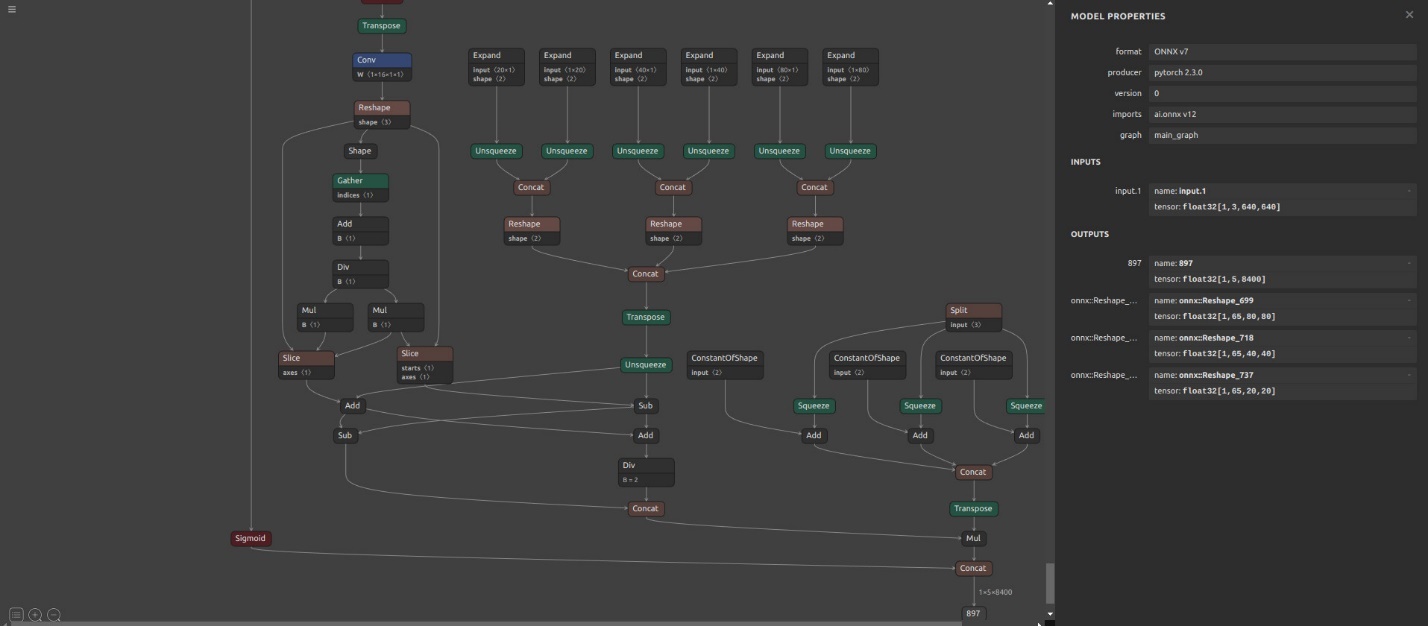
* 1. **Mô hình chính sử dụng Yolov8 Face Detection**



**Hình 2: Lịch sử phát triển của Yolo**

Yolo là một mô hình mạng CNN cho việc phát hiện, nhận dạng, phân loại đối tượng. Yolo được tạo ra từ việc kết hợp giữa các convolutional layers và connected layers. Trong đó các convolutional layers sẽ trích xuất ra các feature của ảnh, còn full-connected layers sẽ dự đoán ra xác suất đó và tọa độ của đối tượng.

YOLOv8 là một mô hình nhận dạng đối tượng dựa trên mạng convolutional neural network (CNN) được phát triển bởi Joseph Redmon và nhóm nghiên cứu của ông tại Đại học Washington.

YOLOv8 là phiên bản nâng cấp của YOLOv7, với khả năng nhận diện đối tượng nhanh hơn và chính xác hơn. Điều này được đạt được thông qua một số cải tiến, bao gồm mạng kim tự tháp đặc trưng, các mô-đun chú ý không gian và các kỹ thuật tăng cường dữ liệu tiên tiến.

**Hình 3: Cấu trúc Yolov8 Face Detection**

Mô hình YOLOv8 sử dụng một mạng neural kiến trúc darknet-53 để trích xuất đặc trưng của hình ảnh và áp dụng thuật toán nhận dạng đối tượng YOLOv8 trên các đặc trưng đó.

Mặc dù YOLO ban đầu được thiết kế để phát hiện vật thể nói chung, bao gồm con người, động vật và phương tiện, nhưng nó có thể được điều chỉnh để phát hiện khuôn mặt. Phát hiện khuôn mặt bằng YOLO liên quan đến việc đào tạo mô hình trên một tập dữ liệu bao gồm khuôn mặt và sau đó sử dụng mô hình đã đào tạo để phát hiện khuôn mặt trong hình ảnh hoặc video mới.

**2.3 Mạng nơ-ron tích chập (Convolutional Neural Network)**

**2.3.1 Định nghĩa**

Mạng nơ-ron tích chập là một trong những mô hình mạng học sâu phổ biến nhất hiện nay, có khả năng nhận dạng và phân loại hình ảnh với độ chính xác rất cao (khoảng 98%). Mô hình này đã và đang được phát triển, ứng dụng vào các hệ thống xử lý ảnh lớn của Facebook, Google hay Amazon… cho các mục đích khác nhau như các thuật toán gắn thẻ tự động, tìm kiếm ảnh hoặc drone giao hàng tự động.

Sự ra đời của mạng nơ-ron tích chập là dựa trên ý tưởng cải tiến cách thức các mạng nơ-ron nhân tạo truyền thống học thông tin trong ảnh. Do sử dụng các liên kết đầy đủ giữa các điểm ảnh vào nút, các mạng nơ-ron nhân tạo truyền thẳng bị hạn chế rất nhiều bởi kích thước của ảnh, ảnh càng lớn thì số lượng liên kết càng tăng nhanh và kéo theo sự bùng nổ khối lượng tính toán. Ngoài ra sự liên kết đầy đủ này cũng là sự dư thừa khi với mỗi bức ảnh, các thông tin chủ yếu thể hiện qua sự phụ thuộc giữa các điểm ảnh với những điểm xung quanh nó mà không quan tâm nhiều đến các điểm ảnh ở cách xa nhau. Mạng nơ-ron tích chập ra đời với kiến trúc thay đổi, có khả năng xây dựng liên kết chỉ sử dụng một phần cục bộ trong ảnh kết nối đến nút trong lớp tiếp theo thay vì toàn bộ ảnh như trong mạng nơ-ron truyền thẳng.

Trong luận văn này, nhóm thực hiện sẽ trình bày về tích chập cũng như ý tưởng của mô hình CNN áp dụng trong bài toán nhận diện cảm xúc trên khuôn mặt.

**2.3.2 Cấu trúc đơn giản của CNN**

Các lớp cơ bản trong một mạng nơ-ron tích chập bao gồm: lớp chập, lớp gộp và lớp kết nối đầy đủ, được thay đổi về số lượng và cách sắp xếp để tạo ra các mô hình huấn luyện phù hợp cho từng bài toán khác nhau.

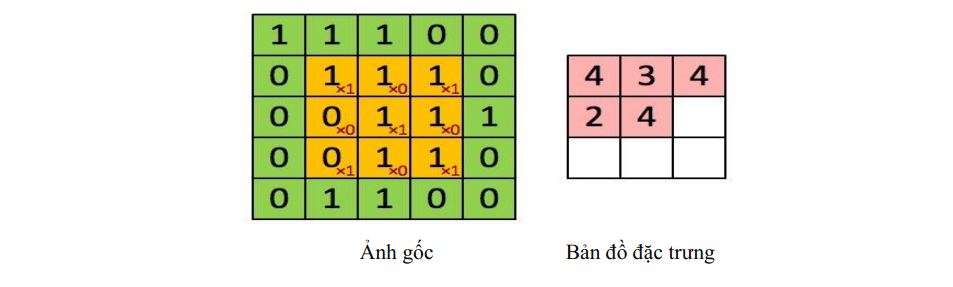
A diagram of a chip and lop dip

Description automatically generated

Hình 4: Cấu trúc của mạng nơ-ron tích chập

* **Lớp chập**

Là lớp quan trọng nhất trong cấu trúc của mạng nơ-ron tích chập. Chập dựa trên lý thuyết xử lý tín hiệu số, việc chập sẽ giúp trích xuất được những thông tin quan trọng từ dữ liệu. Để dễ hình dung, có thể xem tích chập như một cửa sổ trượt áp đặt lên một ma trận. Cơ chế của tích chập được minh họa qua *Hình 6*  bên dưới.



Hình 5: Minh họa chập với kernel 3x3

Từ *Hình 6,* ma trận bên trái là một bức ảnh đen trắng. Mỗi giá trị của ma trận tương đương với một điểm ảnh, 0 là màu đen, 1 là màu trắng (nếu là ảnh xám thì giá trị biến thiên từ 0 đến 255).

Cửa sổ trượt còn có tên gọi là kernel hay filter. Ở đây, ta dùng một ma trận

kernel 3×3 nhân từng thành phần tương ứng với ma trận ảnh bên trái. Giá trị đầu ra do tích của các thành phần này cộng lại. Kết quả của tích chập là một ma trận sinh ra từ việc trượt ma trận kernel và thực hiện tích chập cùng lúc lên toàn bộ ma trận ảnh bên trái.

* **Lớp gộp**

Bản đồ đặc trưng được tạo ra thông qua việc tính toán của lớp chập, tuy nhiên kích thước của nó so với kích thước đầu vào giảm không đáng kể. Nếu kích thước không thay đổi, sẽ tạo nên một số lượng tính toán lớn và việc học tập cũng trở nên rất khó khăn. Lớp gộp này sẽ giải quyết vấn đề về làm giảm kích thước và số lượng tính toán của bản đồ đặc trưng.

Có nhiều toán tử gộp như Sum-Pooling, Max-Pooling, L\_2-Pooling nhưng Max-Pooling thường được sử dụng. Về mặt ý nghĩa thì Max-Pooling xác định vị trí cho tín hiệu mạnh nhất khi áp dụng một loại kernel. Điều này cũng tương tự như là một bộ lọc phát hiện vị trí đối tượng bằng kernel trong bài toán phát hiện đối tượng trong ảnh.

Về mặt lý thuyết với ma trận đầu vào có kích thước a\*b\*c và thực hiện toán tử gộp trên ma trận con của ma trận đầu vào có kích thước f\*f với bước nhảy pixel s thì ta được ma trận đầu ra a’\*b’\*c’ trong đó:

A diagram of a single depth slice

Description automatically generated

Hình 6: Phép toán sử dụng phép gộp

*Hình 7*là ví dụ về sử dụng toán tử gộp. Trong đó miêu tả cách thức lớp gộp xử lý đối với một đầu vào, kích thước của đầu vào là [224\*224\*64] được thực hiệnvới các thông số f = 2 và s = 2 thì đầu ra có kích thước [112\*112\*64] được thể hiện ở phía trái của *Hình 7*. Phía phải mô tả chi tiết cách thức hoạt động của max-pooling trong đó f = 2 và s = 2 kết quả đầu ra ma trận tương ứng.

* **Lớp ReLU**

Chập là một phép biển đổi tuyến tính. Nếu tất cả các nơ-ron được tổng hợp bởi các phép biến đổi tuyến tính thì một mạng nơ-ron đều có thể đưa về dưới dạng một hàm tuyến tính. Do đó tại mỗi nơ-ron cần có một hàm truyền dưới dạng phi tuyến.

Có nhiều dạng hàm phi tuyến được sử dụng trong quá trình này như sau:

**A group of math equations

Description automatically generated**

***Hình 7: Một số hàm phi tuyến.***

Các nghiên cứu gần đây chứng minh được việc sử dụng hàm ReLU cho kết quả tốt hơn ở các khía cạnh: tính toán đơn giản, tạo ra tính thưa ở các nơ-ron ẩn (ví dụ như sau bước khởi tạo ngẫu nhiên các trọng số, khoảng 50% các nơ-ron ẩn được kích hoạt) và quá trình huấn luyện nhanh hơn ngay cả khi không phải trải qua bước tiền huấn luyện.

* **Lớp kết nối đủ**

Cách kết nối các nơ-ron ở hai tầng với nhau trong đó tầng sau kết nối đẩy đủ với các nơ-ron ở tầng trước nó. Đây cũng là dạng kết nối thường thấy ở mạng nơ-ron nhân tạo, trong mạng nơ-ron tích chập tầng này thường được sử dụng ở phía cuối của kiến trúc mạng.

Hiện nay có một số cấu trúc CNN đạt được kết quả khả quan như:

-LeNet: Đây là mô hình CNN thành công đầu tiên, đặc biệt là trong việc nhận dạng chữ số, kí tự trong văn bản. Được phát triển bởi Yann Lecun vào cuối những năm 90.

-AlexNet: Được phát triển bởi Alex Krizhevsky, Ilya Sutskever và Geoff Hinton. Làn đầu được giới thiệu vào năm 2012 với cấu trúc khá tương tự như LeNet nhưng với số lượng neuron, filter và layer lớn hơn. Được coi là mạng neural đầu tiên phổ biến rộng rãi khả năng của CNNs.

-ZF Net: Là mạng CNNs tốt nhất năm 2013 được phát triển bởi Matthew Zeiler và Rob Fergus (ZF là viết tắt của Zeiler và Fergus). Mạng neural này được phát triển từ AlexNet với việc tinh chỉnh các hyperparameter (filter size, stride, ...). Với những layer gần input layer, filter size và stride nhỏ hơn.

-GoogLeNet: Là mạng CNNs tốt nhất năm 2014 được phát triển bởi Szegedy từ Google. Với một số thay đổi như giảm thiểu số lượng tham số trong AlexNet từ 60 triệu xuống 4 triệu, sử dụng Average Pooling thay cho FC-Layer.

-VGGNet: Là mạng CNNs tốt nhất năm 2015 được phát triển bởi Karen Simonyan và Andrew Ziserman. Sử dụng filter 3x3 và pooling 2x2 từ đầu tới cuối mạng. Và vẫn còn được tiếp tục phát triển cho đến nay.

## **2.8 Ngôn ngữ lập trình C++**

Lí do chọn C++ với sự hỗ trợ của framework QT thay cho Python

1. Là ngôn ngữ cơ bản

2. Framework tiện lợi dễ sử dụng

3. Độ hiệu quả cao khi hỗ trợ trên GPU

4. Nền tảng độc lập

5. Hỗ trợ cộng đồng

## **2.9 Một số thư viện hiện nay**

### **2.9.1 TensorFlow**

Thư viện TensorFlow là thư viện mã nguồn mở dùng cho tính toán số học sử dụng đồ thị luồng dữ liệu.

TensorFlow ban đầu được phát triển bởi các nhà nghiên cứu và kỹ sư làm việc trong nhóm Google Brain trong tổ chức Nghiên cứu trí thông minh máy của Google với mục đích tiến hành học máy và nghiên cứu mạng lưới thần kinh sâu. Hệ thống có thể áp dụng trong nhiều lĩnh vực khác nhau.

Ưu điểm của TensorFlow:

* Tích hợp sẵn rất nhiều các thư viện machine learning.
* Có khả năng tương thích và mở rộng tốt. Được Google phát triển cho machine learning phục vụ nghiên cứu lẫn xây dựng các ứng dụng thực tế.
* Phổ biến

TF có tài liệu đầy đủ và bao gồm nhiều thư viện Học máy nên rất phổ biến hiện nay. Vì là một sản phẩm của Google, nên hiện tại, TF đang được ứng dụng rất nhiều trong các sản phẩm của Google như phân loại chữ viết tay, nhận diện hình ảnh, xử lý ngôn ngữ tự nhiên,...

**2.9.2 Sklearn**

Sklearn là một thư viện Học máy mã nguồn mở hữu ích và mạnh mẽ trong Python.Dự án của David Cournapeau bắt đầu vào năm 2007 với tư cách là một dự án của Google Summer of Code. Hiện tại, Sklearn đang được duy trì bởi một đội ngũ các tình nguyện viên. Sklearn cung cấp một sự lựa chọn các công cụ hiệu quả cho Học máy và mô hình thống kê, bao gồm phân loại, hồi quy, phân cụm và giảm chiều dữ liệu với giao diện nhất quán trong Python. Thư viện này phần lớn được viết bằng Python, được xây dựng dựa trên NumPy, SciPy và Matplotlib.

Một số sản phẩm thương mại sử dụng Sklearn như Spotify, Evernote, Booking.com, J.P.Morgan, Hugging Face, Télécom ParisTech, Aweber,...

### **2.9.3 Keras**

Keras là một thư viện được phát triển vào năm 2015 bởi François Chollet, là một kỹ sư nghiên cứu deep learning tại Google. Nó là một mã nguồn mở cho mạng nơ-ron được viết bởi ngôn ngữ Python. Keras là một API bậc cao có thể sử dụng chung với các thư viện deep learning nổi tiếng như Tensorflow (được phát triển bởi Google), CNTK (được phát triển bởi microsoft), Theano (người phát triển chính Yoshua Bengio). Keras có một số ưu điểm như:

* Dễ sử dụng, xây dựng model nhanh.
* Có thể chạy trên cả CPU và GPU
* Hỗ trợ xây dựng CNN, RNN và có thể kết hợp cả hai

### **2.9.4 Dlib**

Dlib được tạo ra từ 2002 bởi tác giả Davis King, được viết trên ngôn ngữ lập trình C++. Khác với mục đích của OpenCV là cung cấp hạ tầng thuật toán cho các ứng dụng xử lý ảnh và thị giác máy tính, dlib được thiết kế cho các ứng dụng máy học (machine learning) và trí tuệ nhân tạo với các thư viện con chính như sau:

* Classification: các kỹ thuật phân lớp chủ yếu dựa trên hai phương pháp cơ sở là kNN và SVM.
* Data transformation: các thuật toán biến đổi dữ liệu nhằm giảm số chiều, loại bỏ các dữ liệu dư thừa và tăng cường tính khác biệt (discriminant) của các đặc điểm được giữ lại.
* Clustering: các kỹ thuật phân cụm.
* Regression: các kỹ thuật hồi qui.
* Structure prediction: các thuật toán dự đoán có cấu trúc.
* Markov Random Fields: các thuật toán dựa trên các trường Markov ngẫu nhiên

### **2.9.5 OpenCV**

OpenCV (OpenSource Computer Vision) là một thư viện mã nguồn mở. OpenCV được phát hành theo giấy phép BSD, do đó nó hoàn toàn miễn phí cho cả học thuật và thương mại. Nó có các interface C++, C, Python, Java và hỗ trợ Windows, Linux, Mac OS, iOS và Android. OpenCV được thiết kế để tính toán hiệu quả và với sự tập trung nhiều vào các ứng dụng thời gian thực. Được viết bằng tối ưu hóa C/C++, thư viện có thể tận dụng lợi thế của xử lý đa lõi. Được sử dụng trên khắp thế giới, OpenCV có cộng đồng hơn 47 nghìn người dùng và số lượng download vượt quá 6 triệu lần. Phạm vi sử dụng từ nghệ thuật tương tác, cho đến lĩnh vực khai thác mỏ, bản đồ trên web hoặc công nghệ robot.

Một số module mà chúng ta có thể làm việc thường xuyên nếu chúng ta sử dụng thư viện này:

* *Core*: Đây là module chứa các cấu trúc, class cơ bản mà OpenCV sẽ sử dụng trong việc lưu trữ và xử lý hình ảnh như Mat, Scale, Point, Vec… và các phương thức cơ bản sử dụng cho các module khác.
* *Imgproc*: Đây là một module xử lý hình ảnh của OpenCV bao gồm các bộ lọc (filter) linear và non-linear và các phép biến đổi hình học (tranformation) như resize hoặc các phép biến đổi affine, chuyển đổi hệ máy và các thuật toán liên quan đến histogram (biểu đồ) của hình ảnh.
* *Highgui*: Đây là một module cho phép tương tác với người dùng trên UI (User Interface) như hiển thị hình ảnh, video capturing.
* *Features2d*: Module tìm các đặc trưng (feature) của hình ảnh. Trong module có implement các thuật toán rút trích đặc trưng như PCA…
* *Calib3d*: Hiệu chuẩn máy ảnh và xây dựng lại 3D.
* *Objdetect*: Module cho việc phát hiện các đối tượng như khuôn mặt, đôi mắt, cốc, người, xe hơi, … trong hình ảnh. Các thuật toán được sử dụng trong module này là Haar‐like Features.
* *ml*: Module này chứa các thuật toán về Machine Learning phục vụ cho các bài toàn phân lớp (Classfitication) và bài toán gom cụm (Clustering). Ví dụ như thuật toán SVM (Support Vector Machine), ANN…
* *Video*: Module phân tích video gồm ước lượng chuyển động, trừ nền, và các thuật toán theo dõi đối tượng (object tracking).

OpenCV đang được sử dụng rộng rãi trong các ứng dụng bao gồm:

* Nhận diện hình ảnh
* Kiểm tra và giám sát tự động
* Robot và xe hơi tự lái
* Phân tích hình ảnh y tế
* Tìm kiếm và phục hồi hình ảnh/video
* Phim - cấu trúc 3D từ chuyển động
* Nghệ thuật sắp đặt tương tác

Đối với sinh viên ngành điện tử viễn thông việc ứng dụng thư viện mã nguồn mở OpenCV có thể thực hiện được rất nhiều các bài toán lý thú trên các bo mạch phát triển sẵn như Raspberry pi hay Adruino.

# **CHƯƠNG 2: Mô hình Face Landmarks nhận diện đóng mở miệng**

## **2.1 Giới thiệu về MediaPipe**

AI (Trí tuệ nhân tạo) đang dần "xâm chiếm" hầu như tất cả các lĩnh vực trong cuộc sống. Với tính chất tự động hóa cao, có thể thực hiện các công việc khó và có độ chính xác ngày càng được cải thiện, AI đang dần trở thành một công cụ khó có thể thay thế được. Tuy nhiên, các mô hình AI hiện tại được đánh giá là "nặng, yêu cầu phần cứng cao" khiến cho việc áp dụng vào trong các dự án thực tế trở nên khá khó khăn, nhất là với các thiết bị mobile hoặc edge devices - xu thế hiện tại. Vì vậy, Google đã đưa ra một giải pháp, chính xác hơn là một bộ công cụ, cung cấp các công cụ cho các bài toán AI/ML đã được tối ưu để chạy trên nhiều nền tảng khác nhau, với tên gọi là [MediaPipe](https://github.com/google/mediapipe).

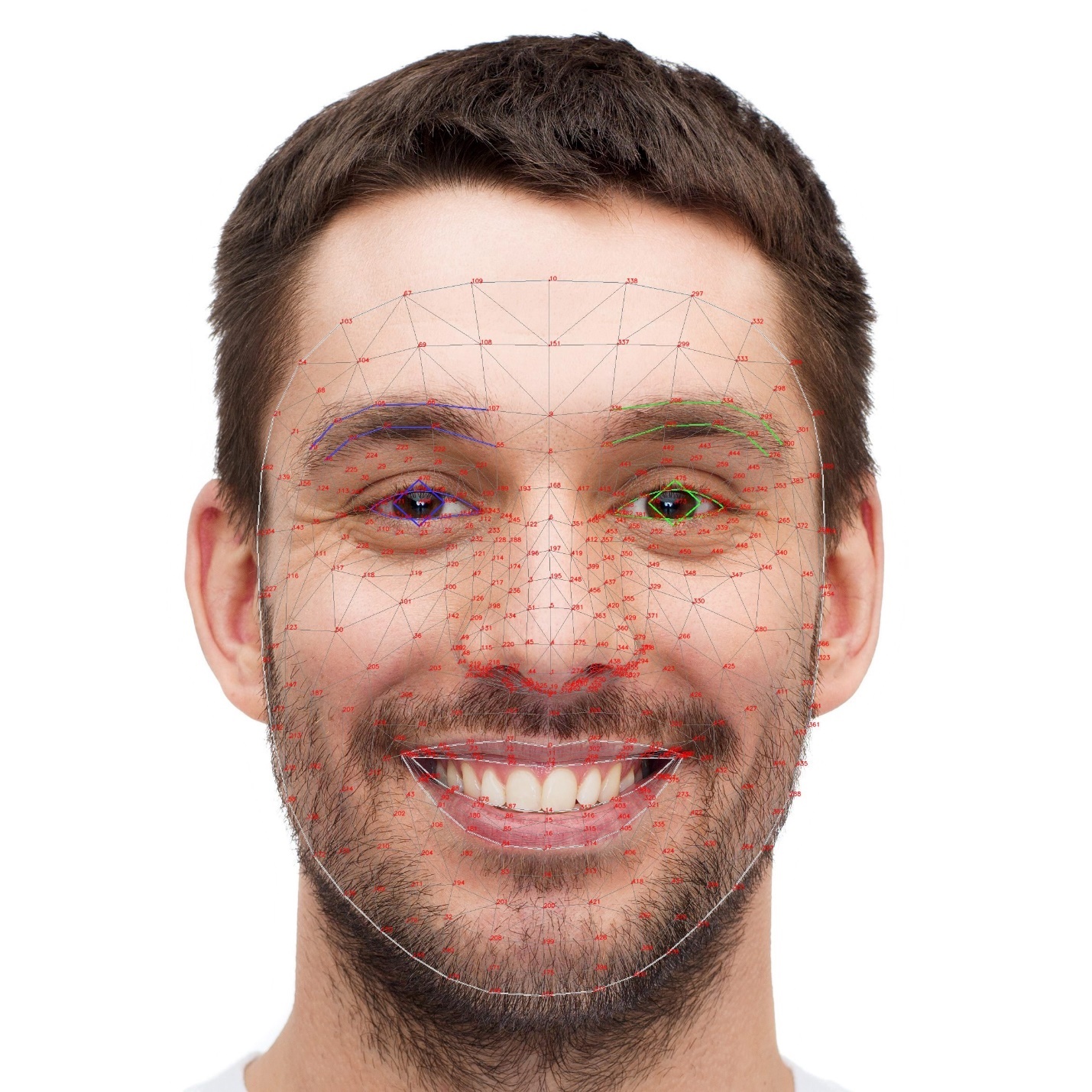
Về tổng quan, [MediaPipe](https://github.com/google/mediapipe) là tập hợp của một loạt các giải pháp Machine Learning đa nền tảng, có thể can thiệp được và cực kỳ lightweight. Một số ưu điểm có thể kể tới của giải pháp này bao gồm:

* Cung cấp một giải pháp inference nhanh chóng: Google khẳng định rằng bộ công cụ này có thể chạy ổn định trên hầu hết các cấu hình phần cứng thông dụng.
* Dễ dàng cài đặt và triển khai: Việc cài đặt cực kỳ dễ dàng và tiện lợi, có thể triển khai trên nhiều nền tảng khác nhau như Mobile (Android/iOS), Desktop/Cloud, Web và IoT devices.
* Mã nguồn mở và miễn phí: Toàn bộ source code được công khai trên [MediaPipe](https://github.com/google/mediapipe), người dùng hoàn toàn có thể sử dụng và tùy chỉnh trực tiếp để phù hợp với bài toán của mình.

## **2.2** **Face Mesh**

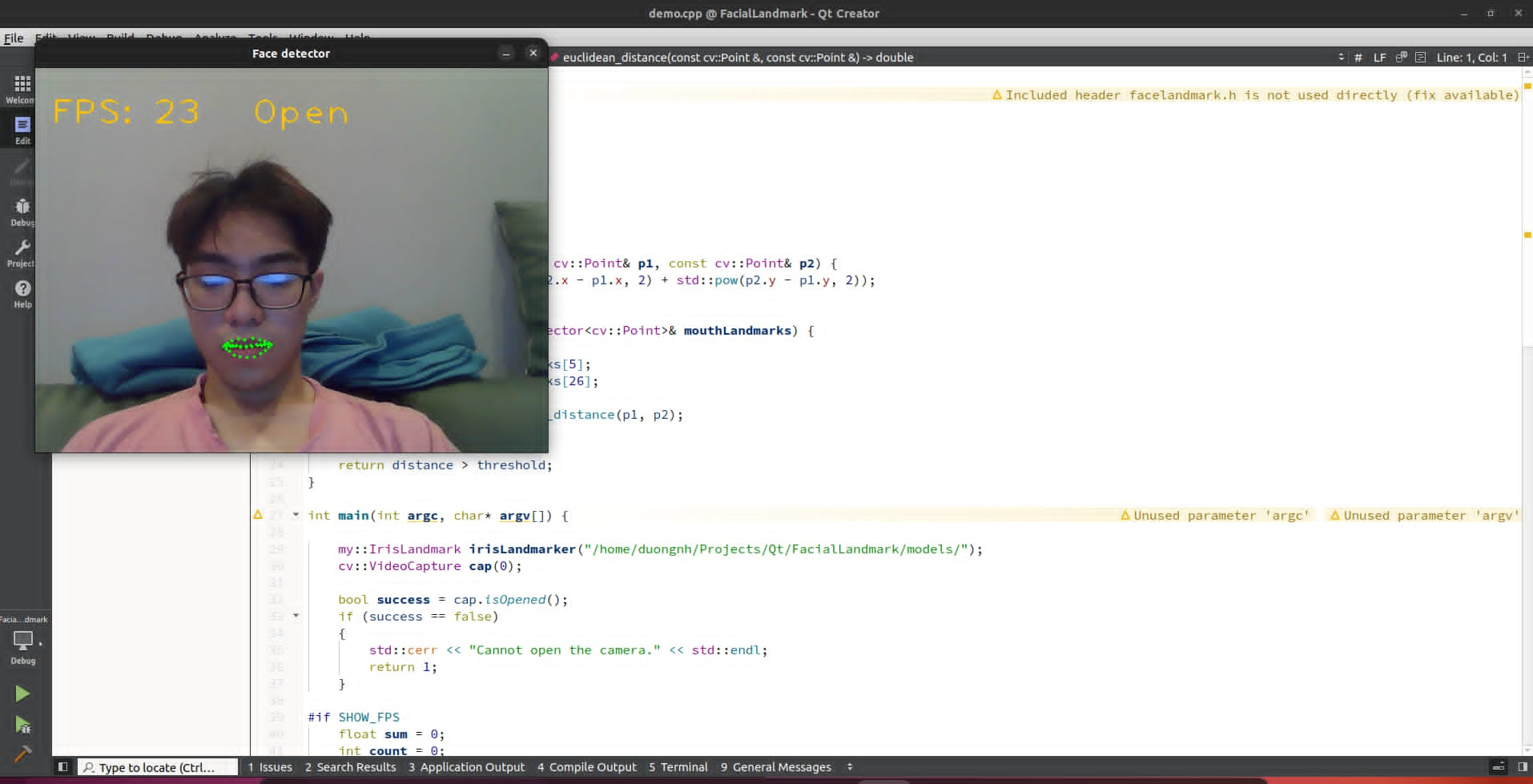
**Hình 8: Các giải pháp được cung cấp và tính khả dụng trên các nền tảng**

Thay vì tìm bounding box bao quanh mặt, Face Mesh là một bài toán nhận diện một loạt các điểm trên khuôn mặt, từ đó tạo thành 1 lưới (mesh) của mặt. Chiếc lưới này sẽ được áp dụng vào các bài toán chỉnh sửa ảnh mặt 3D hay các tác vụ liên quan tới 3D Alignment và Anti-spoofing. Công cụ trong MediaPipe sẽ sinh ra tổng cộng 468 điểm trên mặt và tạo thành lưới mà không đòi hỏi quá nhiều năng lực tính toán cũng như số camera (chỉ cần 1 camera chính diện).



**Hình 9: Hình ảnh 468 điểm landmarks trên khuôn mặt theo MediaPipe**

Để nhận diện khuôn miệng, nhóm đã lấy ra tập hợp các landmarks của vùng miệng, cụ thể là môi trên và môi dưới dựa theo ảnh trên rồi khoanh vùng và detect



**Hình 10: Ảnh thực nghiệm Mouth Landmarks**

# 

# **CHƯƠNG 3. XÂY DỰNG GAME BẰNG CÔNG CỤ LÀM GAME GODOT**

## **3.1 Các giải pháp đã có**

Ngoài Unity, còn có hai công cụ làm game phổ biến khác là Godot và Unreal Engine. Cả ba công cụ này đều có những đặc điểm riêng, hỗ trợ người phát triển tạo ra những trò chơi đáng kinh ngạc và đẹp mắt.

**Godot:**  Godot là một công cụ mã nguồn mở và miễn phí, được phát triển bởi cộng đồng và cung cấp một môi trường phát triển trực quan.

* **Ưu điểm**: - Một điểm mạnh của Godot là giao diện đơn giản và dễ sử dụng. Nó cung cấp một môi trường phát triển trực quan và trực quan giúp người dùng dễ dàng tạo và quản lý các thành phần trong trò chơi. Bên cạnh đó, Godot hỗ trợ nhiều ngôn ngữ lập trình như C++, C#, và GDScript - ngôn ngữ lập trình riêng của Godot. Điều này cho phép người phát triển lựa chọn ngôn ngữ phù hợp với kỹ năng và sở thích của mình.
* Godot cũng nổi tiếng với tính đa nền tảng. Người phát triển có thể tạo ra trò chơi cho nhiều nền tảng như PC, điện thoại di động và máy tính bảng. Godot hỗ trợ việc xuất bản game dễ dàng trên các nền tảng này, giúp mở rộng đối tượng người chơi và tiếp cận thị trường đa dạng.
* Với Godot, người phát triển cũng có lợi thế của cộng đồng đông đảo và hỗ trợ tài liệu phong phú. Godot có một cộng đồng tích cực và sẵn lòng chia sẻ kiến thức, tài liệu và tài nguyên giúp người dùng giải quyết các vấn đề và tìm kiếm giải pháp. Hơn nữa, do Godot là công cụ mã nguồn mở, người dùng có thể truy cập và tùy chỉnh mã nguồn theo ý muốn.
* **Hạn chế**: Tuy nhiên, một điểm hạn chế của Godot là hiệu suất đồ họa không mạnh như Unreal Engine. Mặc dù Godot cung cấp các công cụ và tính năng đồ họa tốt, nhưng nó không thể đạt được mức độ chất lượng cao như Unreal Engine. Điều này có thể ảnh hưởng đến khả năng tạo ra các trò chơi với đồ họa phức tạp và hiệu ứng đặc sắc.

**Unreal Engine**: Unreal Engine là một công cụ phát triển game mạnh mẽ và phổ biến, được phát triển bởi Epic Games.

* **Ưu điểm**: Nó cung cấp một hệ thống đồ họa cao cấp, cho phép người phát triển tạo ra các trò chơi với đồ họa chất lượng cao và hiệu ứng đặc sắc. Unreal Engine hỗ trợ ngôn ngữ lập trình C++ và Blueprint, một công cụ trực quan giúp người dùng kết hợp các hành động và chức năng một cách trực quan. Ngoài ra, Unreal Engine cũng có cộng đồng đông đảo và nhiều tài liệu hỗ trợ, giúp người phát triển tìm kiếm giải pháp và chia sẻ kiến thức.
* **Hạn chế:** Tuy nhiên, một điểm hạn chế của Unreal Engine là khả năng học và sử dụng. Đối với những người mới bắt đầu hoặc không có nền tảng vững chắc về lập trình, việc làm quen với công cụ và học cú pháp Blueprint (ngôn ngữ trực quan của Unreal Engine) có thể đòi hỏi thời gian và nỗ lực. Hơn nữa, vì Unreal Engine tập trung vào đồ họa và hiệu suất cao, việc làm việc với nó có thể yêu cầu cấu hình máy tính mạnh và tài nguyên phần cứng cao hơn so với các công cụ khác.

**Unity:** Unity là một công cụ phát triển game mạnh mẽ và phổ biến, với nhiều điểm mạnh và điểm hạn chế đáng chú ý.

* **Ưu điểm:** Một điểm mạnh của Unity là tính đa nền tảng. Người phát triển có thể tạo ra trò chơi cho nhiều nền tảng như PC, điện thoại di động, máy tính bảng và cả hệ máy chơi game. Điều này giúp mở rộng đối tượng người chơi và tăng khả năng tiếp cận đến thị trường đa dạng. Unity cung cấp một hệ thống xuất bản linh hoạt giúp người dùng dễ dàng triển khai trò chơi trên các nền tảng khác nhau. Unity cũng có một cộng đồng lớn và phong phú. Người dùng Unity có thể tìm thấy nhiều tài liệu, hướng dẫn và tài nguyên trên mạng. Cộng đồng Unity rất tích cực và sẵn lòng chia sẻ kiến thức và kinh nghiệm, giúp người dùng giải quyết các vấn đề và tìm kiếm giải pháp. Điều này tạo điều kiện thuận lợi cho việc học và phát triển trong quá trình sử dụng Unity. Unity cung cấp một công cụ phát triển trực quan và dễ sử dụng. Giao diện Unity cho phép người dùng tạo và quản lý các thành phần trong trò chơi một cách trực quan và thuận tiện. Nó cũng hỗ trợ nhiều ngôn ngữ lập trình như C#, JavaScript và Boo, cho phép người dùng lựa chọn ngôn ngữ phù hợp với kỹ năng và sở thích của mình.
* **Nhược điểm:** Tuy nhiên, một điểm hạn chế của Unity là hiệu suất và tối ưu hóa. Trong những trò chơi có đồ họa phức tạp và yêu cầu xử lý nặng, Unity có thể gặp khó khăn trong việc đạt được hiệu suất cao và tối ưu hóa tốt. Điều này đặc biệt đúng khi phát triển trò chơi cho các thiết bị di động có tài nguyên phần cứng hạn chế. Người dùng Unity cần có kiến thức về tối ưu hóa và tìm hiểu kỹ về các phương pháp để tối ưu hóa trò chơi của mình.

Unity, Godot và Unreal Engine đều có ưu điểm riêng và được sử dụng rộng rãi trong ngành công nghiệp game. Unity nổi tiếng với tính linh hoạt và đa nền tảng, hỗ trợ cho cả người mới bắt đầu và những nhà phát triển kỳ cựu. Godot thu hút người dùng với tính đơn giản, mã nguồn mở và miễn phí. Unreal Engine nổi bật với tính năng đồ họa mạnh mẽ và hiệu suất cao. Tùy thuộc vào yêu cầu và ưu tiên của dự án, người phát triển có thể chọn công cụ phù hợp nhất để tạo ra những trò chơi ấn tượng và đáng chú ý trên thị trường hiện nay.

## **3.2 Giải pháp đề xuất**

Chúng em lựa chọn sử dụng Godot để xây dựng game vì những lợi ích sau:

* Đa nền tảng: Godot hỗ trợ phát triển trên nhiều nền tảng, bao gồm Windows, macOS, Linux, Android, iOS và nhiều hơn nữa. Điều này cho phép bạn phát triển game một lần và triển khai trên nhiều nền tảng một cách dễ dàng.
* Miễn phí và mã nguồn mở: Godot là một công cụ mã nguồn mở và hoàn toàn miễn phí để sử dụng. Bạn có thể truy cập và tùy chỉnh mã nguồn của Godot theo nhu cầu của mình mà không phải trả bất kỳ phí nào.
* Đội ngũ phát triển chủ động: Godot có một cộng đồng đông đảo và nhiệt tình, với nhiều người dùng và nhà phát triển đóng góp vào sự phát triển của nó. Điều này đảm bảo rằng Godot được cập nhật thường xuyên và hỗ trợ tốt cho các vấn đề và tính năng mới.
* Giao diện đồ họa dễ sử dụng: Godot cung cấp một giao diện đồ họa trực quan và dễ sử dụng cho việc phát triển game. Nó có một bộ công cụ mạnh mẽ cho việc tạo và quản lý các đối tượng, kịch bản, hiệu ứng và hình ảnh, giúp bạn tập trung vào việc thiết kế và phát triển game một cách hiệu quả.

- Hỗ trợ nhiều ngôn ngữ lập trình: Godot hỗ trợ nhiều ngôn ngữ lập trình như GDScript (ngôn ngữ kịch bản riêng của Godot), C#, và C++. Bạn có thể lựa chọn ngôn ngữ phù hợp với kỹ năng và sở thích của mình để phát triển game

## **3.3 Mô hình hệ thống / Thiết kế giải pháp**

**3.3.1 Các kịch bản của hệ thống ( Use-Case )**

Use-case 1:

* Name: Start ( Bắt đầu )
* Description: Người chơi muốn bắt đầu game
* Actor: Người chơi
* Priority: Cao

Course:

+ Step 1: Người chơi chọn nút Start

+ Step 2: Game chuyển sang panel GamePlay ( giao diện chơi game )

Use-case 2:

* Name: Điều khiển nhân vật
* Description: Người chơi muốn di chuyển nhân vật
* Actor: Người chơi
* Priority: Cao
* Course:

+ Step 1: Người chơi di chuyển khuôn mặt của mình lên xuống theo phương thẳng đứng

+ Step 2: Game sẽ tiếp nhận và xử lý vị trí của nhân vật, nhân vật sẽ di chuyển lên xuống theo khuôn mặt của người chơi được nhận diện.

Use-case 3:

* Name: Tấn công ( Shoot )
* Description: Người chơi muốn bắn đối phương ( enemy )
* Actor: Người chơi
* Priority: Cao
* Course:

+ Step 1: Người chơi há miệng để hệ thống nhận diện

+ Step 2: Game tiếp nhận trạng thái há miệng của người chơi và thực hiện bắn ra đạn.

+ Step 3: Khi người chơi ngậm miệng lại thì nhân vật sẽ không bắn đạn ra nữa.

Use-case 4:

* Name: Quit ( Thoát khỏi trò chơi )
* Description: Người chơi muốn thoát khỏi trò chơi
* Actor: Người chơi
* Priority: Cao
* Course:

+ Step 1: Người chơi chọn nút Quit Game.

+ Step 2: Game tiếp nhận và thực hiện thoát game.

### **3.3.2 Quy trình xây dựng game**

- Thiết kế và lên ý tưởng: Xác định thể loại và cốt truyện của trò chơi. Tạo một bản thiết kế cơ bản về giao diện, màn chơi, nhân vật, và hệ thống gameplay.

- Xác định tài nguyên: Xác định và thu thập tài nguyên cần thiết như hình ảnh, âm thanh, và hiệu ứng đồ họa. Nếu không có khả năng tự tạo tài nguyên, bạn có thể tìm kiếm các nguồn tài nguyên miễn phí hoặc mua từ các trang web.

- Thiết lập môi trường phát triển: Tải và cài đặt Godot để bắt đầu phát triển trò chơi. Tạo một dự án mới và cấu hình cài đặt cơ bản như kích thước màn hình, hệ thống vật lý, và thời gian cập nhật.

- Xây dựng cơ bản: Tạo các đối tượng, nhân vật, và môi trường cần thiết trong trò chơi bằng cách sử dụng các công cụ và tính năng của Unity. Điều chỉnh kích thước, màu sắc, và vị trí của các đối tượng để tạo cảm giác thị giác và gameplay mong muốn.

- Lập trình logic: Sử dụng ngôn ngữ lập trình GDScript để viết mã để điều khiển hành vi của nhân vật, xử lý va chạm, quản lý điểm số, và các tính năng khác của trò chơi. Xử lý các sự kiện như nút bấm, di chuyển, và tương tác với đối tượng.

- Kiểm tra và sửa lỗi: Kiểm tra trò chơi để phát hiện và sửa lỗi, đảm bảo tính ổn định và khả năng chơi. Thử nghiệm từng khía cạnh của trò chơi và điều chỉnh cần thiết.

- Hoàn thiện: Hoàn thiện các yếu tố như âm thanh, hiệu ứng, chuyển cảnh, và giao diện người dùng.

### **3.3.2.1 Xây dựng các thành phần của game**

Sau khi hoàn tất việc chuẩn bị nguyên liệu ( hình ảnh, nhạc nền, nhạc hiệu ứng, hiệu ứng hình ảnh ) thì chúng ta sẽ bắt đầu bắt tay vào làm game. Bước đầu tiên trong quy trình xây dựng Game là tạo ra các GameObject và Scene. Bao gồm:

* Player: Đây là một trong những thành phần quan trọng nhất của game được liên kết với script Player.
  + Lập trình chức năng di chuyển cho nhân vật khi người chơi điều khiển bằng khuôn mặt
  + Lập trình chức năng Shoot giúp người chơi có thể bắn đạn bằng cách nhận diện há miệng trên khuôn mặt.
  + Xây dựng game logic khi nhân vật bị trúng đạn sẽ bị trừ máu. Lượng máu được hiển thị trên thanh HP phía trên.
* AudioController: Đây là đối tượng được đính kèm script AudioController để quản lý các loại âm thanh có trong game như nhạc nền, nhạc hiệu ứng, độ to nhỏ,...
* SceneController: Đối tượng này được đính kèm script SceneController để giúp việc chuyển scene dễ dàng.
* BG ( backgrounnd ): Đây là đối tượng để quản lý phần background của game, giúp game tăng thêm tính chân thật và đẹp mắt.
* FileScript: Đây là đối tượng quản lý logic chính của game, kết nối các thành phần đơn lẻ của game giúp hình thành một game logic.

#### **3.3.2.2 Tạo hiệu ứng âm thanh**

Các hiệu ứng âm thanh giúp người chơi dễ dàng hơn trong việc xác định mình đang làm gì và tạo thêm tính sinh động, hấp dẫn cho game.

* BG music: đây là nhạc nền chính của game
* Sniper : hiệu ứng âm thanh này sẽ được tạo ra khi người chơi thực hiện Shoot
* Win: hiệu ứng âm thanh này sẽ được tạo ra mỗi khi người chơi vượt qua một level nào đó.
* Lose: khi bạn thua cuộc ( gameover ) thì hiệu ứng âm thanh này sẽ được phát ra để thông báo.

### **3.3.3 Mô hình GameObject và Scene**

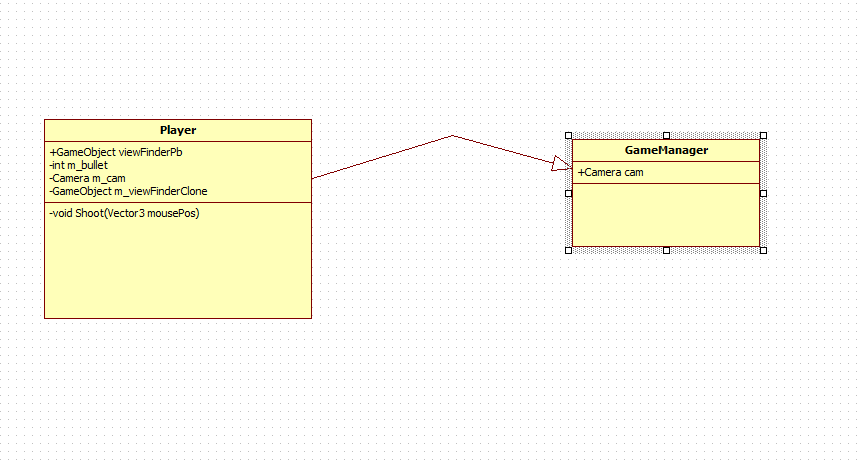
GameObject: GameObject (Đối tượng trò chơi) là thành phần cơ bản nhất trong Godot, đại diện cho các đối tượng trong trò chơi như nhân vật, vật phẩm, môi trường, ánh sáng và nhiều thứ khác. Một GameObject có thể chứa các thành phần khác, như là Mesh Renderer (để hiển thị hình dạng), Collider (để xử lý va chạm), Script (để xử lý logic), và nhiều thành phần khác. GameObject cũng có thể có các GameObject con, tạo thành cây phân cấp.

Scene: Scene (Cảnh) là một màn chơi hoặc khung cảnh trong Unity, đại diện cho một phần của thế giới trò chơi hoặc một trạng thái cụ thể trong ứng dụng. Mỗi Scene chứa các GameObject và các thành phần của chúng, xác định nội dung và logic của màn chơi đó. Ví dụ, có thể có các cảnh như "Màn chơi 1-1", "Màn chơi 1-2" và "Màn chơi cuối cùng". Các cảnh cho phép xây dựng và quản lý các đối tượng và tương tác giữa chúng trong quá trình phát triển trò chơi.

### **3.3.3.1 Player**

Lớp Player thực hiện việc khả năng di chuyển cho người chơi để né đạn.

Tạo ra phương thức Shoot để tạo khả năng bắn kẻ địch cho người chơi.

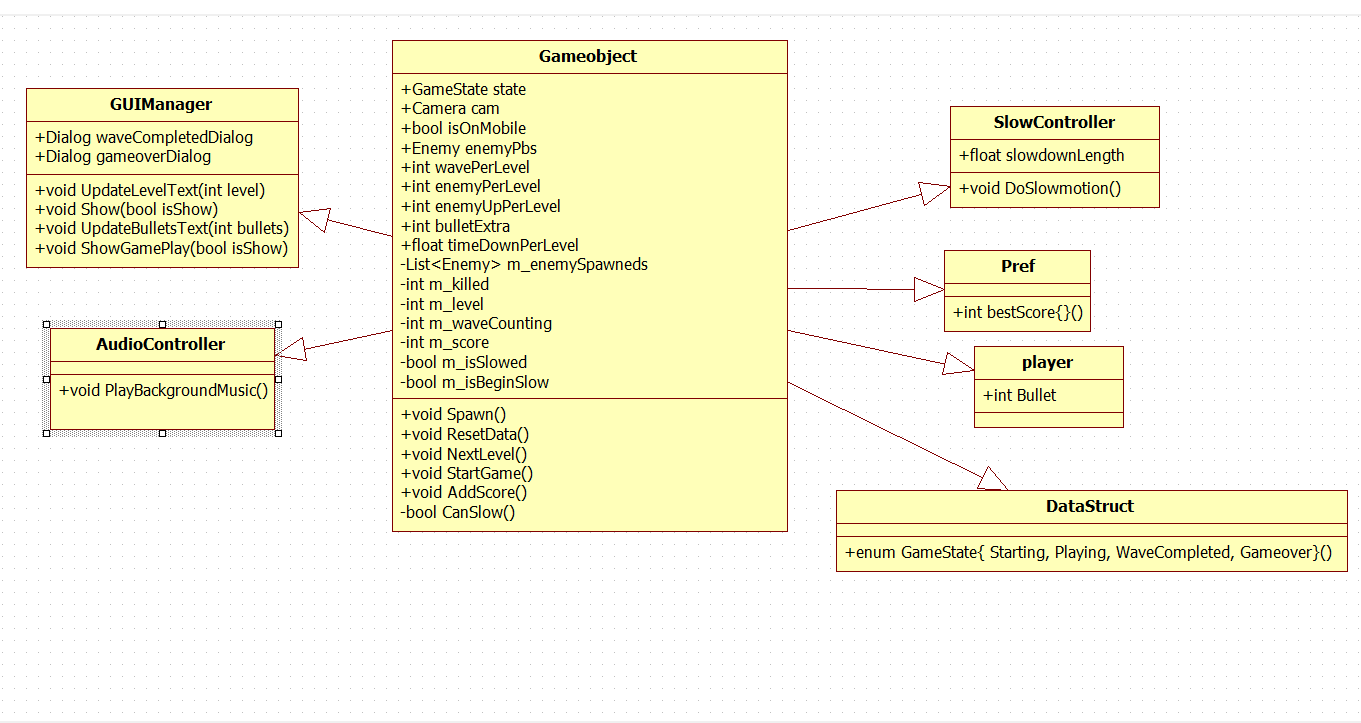


**Hình 11: Biểu đồ Scene của Player**

### **3.3.3.2 GameManager**

Lớp GameManager liên kết nhiều gameobject tạo thành game logic:

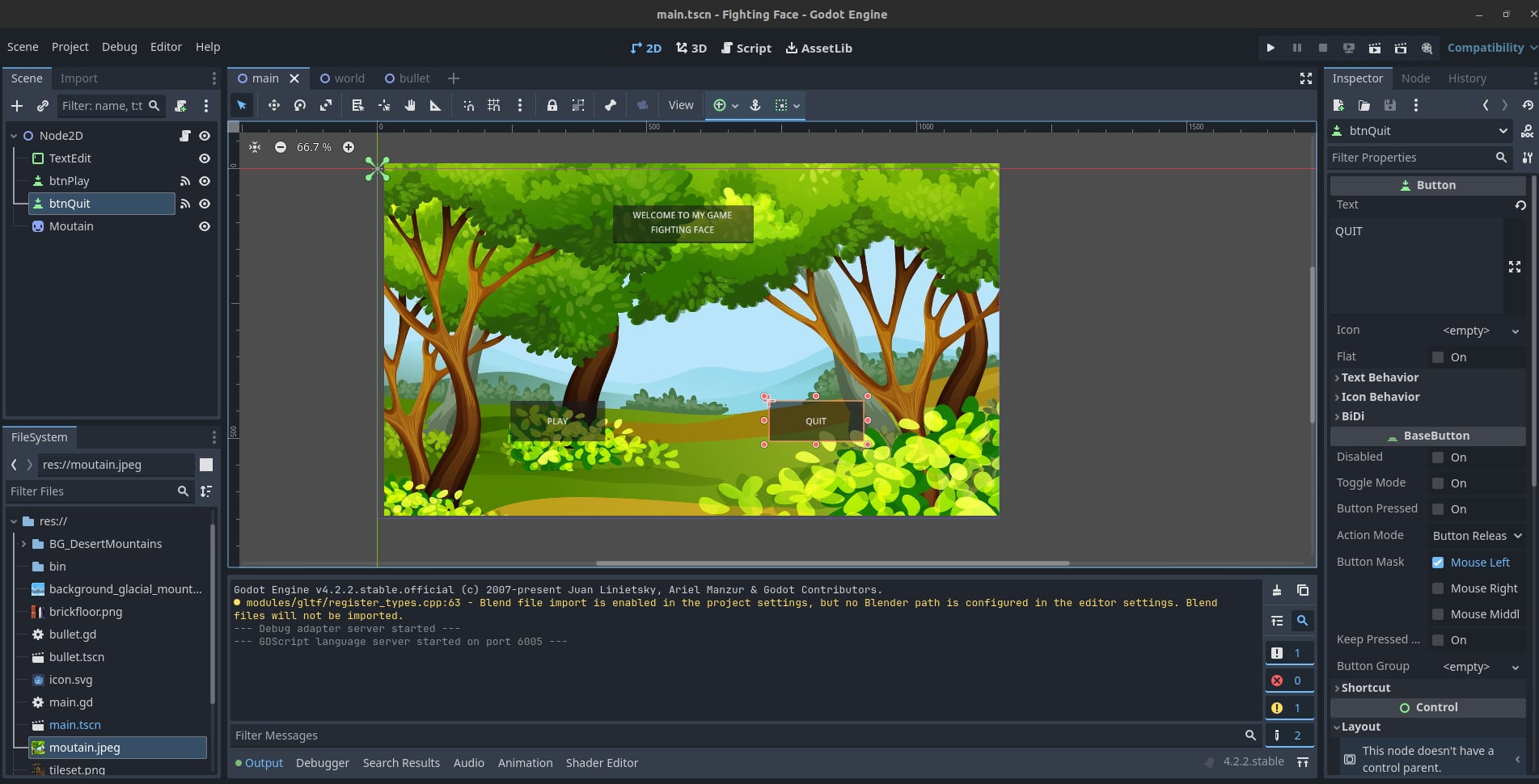
* Phương thức ResetData(): để reset các dữ liệu cho game
* Phương thức NextLevel(): được dùng để tăng level cho game
* Phương thức StartGame(): đươc dùng để start game



**Hình 12: Biểu đồ của GameObject**

### **3.3.4 Các màn hình giao diện người dùng**

3.3.4.1 Giao diện Home



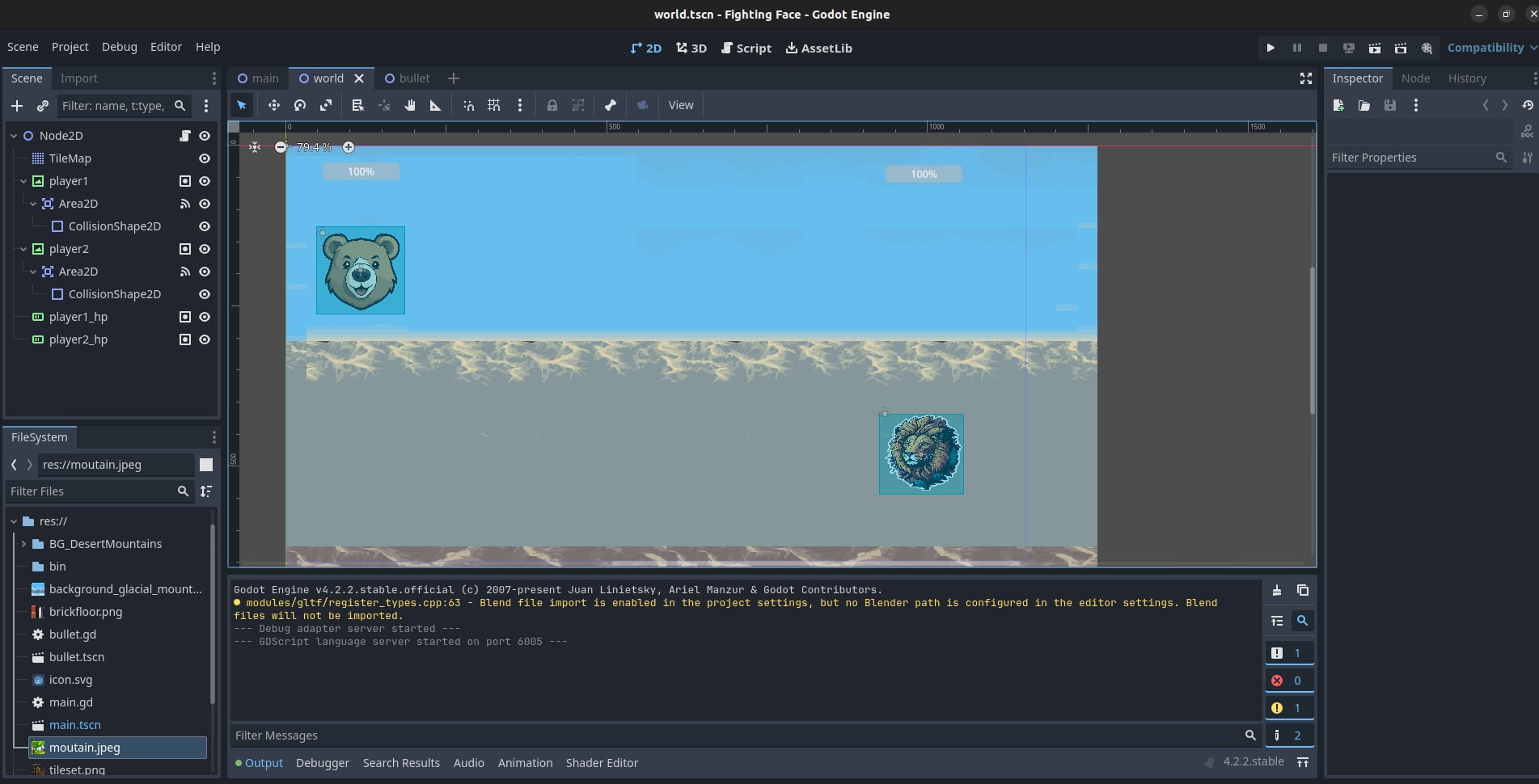
Hình 13: Home

Màn hình chính của Game là nơi bắt đầu của tất cả người chơi. Màn hình này bao gồm tiêu đề trên cùng của Game “Fighting Face”.

Nếu người chơi đã từng chơi trước đó hay là muốn tự tìm hiểu cách chơi thì có thể bấm nút “ Start ” ở để bắt đầu chơi.

Ngoài ra, đối với những người đã chơi xong và muốn thoát game thì cũng có thể bấm nút “ Quit ” để thoát khỏi Game.

3.3.4.2 Giao diện game play



**Hình 14: Game Play**

Đây là giao diện sau khi người chơi nhấn nút Start.

2 người chơi sẽ hiển thị mặt được nhận diện lên giao diện của game.

Có thể di chuyển nhân vật lên xuống và há miệng để bắn đạn về phía người còn lại. Nếu bắn trúng đối phương, đối phương sẽ bị trừ máu.

Bên nào hết máu trước sẽ thua cuộc.

### **3.3.5 Thiết kế của đạn**

**Hình 15: Bullet**

**KẾT LUẬN**

## **1.Kết quả đạt được**

Tạo ra được mô hình game tích hợp nhận diện khuôn mặt và miệng

## **2. Hướng phát triển:**

Trong quá trình làm việc và tìm hiểu, nhóm gặp nhiều khó khan vì tài liệu còn ít, đặc biệt là về MediaPipe. Bên cạnh đó về phần nhúng code C++ vào trong Godot cũng là một cản trở mà nhóm chưa hoàn thành để dự án có thể đồng bộ. Vì vậy các phần như mô hình nhận diện với phần game còn đang bị tách rời. Thời gian tới nhóm sẽ tìm hiểu them và oàn thành sản phẩm thú vị này và update sớm**.**

# **CHƯƠNG 4: MỘT SỐ THÀNH PHẦN KHÁC CỦA ĐỒ ÁN**

## **4.1 Kế hoạch dự án**

Kế hoạch dự án là một tài liệu quan trọng, giúp nhóm nghiên cứu định hướng và theo dõi tiến độ thực hiện đồ án.

Mục tiêu chung của đồ án là xây dựng hệ thống điểm danh bằng nhận diện khuôn mặt bằng MTCNN và FaceNet. Mục tiêu cụ thể của đồ án bao gồm:

## **4.2 Đảm bảo thực hiện đúng làm việc nhóm**

Làm việc nhóm là một kỹ năng quan trọng cần được rèn luyện trong quá trình học tập và làm việc. Để đảm bảo thực hiện đúng làm việc nhóm, cần chú ý đến các vấn đề sau:

Lập kế hoạch làm việc nhóm:

Kế hoạch làm việc nhóm cần được lập ra ngay từ đầu dự án, và được cập nhật thường xuyên trong quá trình thực hiện. Kế hoạch làm việc nhóm giúp nhóm nghiên cứu thống nhất mục tiêu, nhiệm vụ, phân công công việc, và theo dõi tiến độ thực hiện dự án.

Kế hoạch làm việc nhóm của nhóm nghiên cứu đồ án điểm danh bằng nhận diện khuôn mặt bằng MTCNN và FaceNet có thể bao gồm các nội dung sau:

\* Mục tiêu: Xây dựng game ứng dụng mô hình nhận diện khuôn mặt và landmarks.

\* Nhiệm vụ:

\* Nghiên cứu lý thuyết và thu thập dữ liệu.

\* Phát triển mô hình Yolov8 Face Detection.

\* Phát triển mô hình Face Landmarks.

## **4.3 Các vấn đề về đạo đức và làm việc chuyên nghiệp**

Trong quá trình thực hiện đồ án, cần chú ý đến các vấn đề về đạo đức và làm việc chuyên nghiệp, bao gồm:

* Tôn trọng quyền sở hữu trí tuệ: Không sao chép, sử dụng trái phép các tài liệu, ý tưởng của người khác.
* Tuân thủ các quy định của pháp luật: Không thực hiện các hoạt động vi phạm pháp luật.
* Tác phong chuyên nghiệp: Cẩn thận, tỉ mỉ trong công việc, đúng giờ, đúng hẹn,...

Các vấn đề về đạo đức

Trong lĩnh vực công nghệ thông tin, các vấn đề về đạo đức cần được đặc biệt quan tâm. Một số vấn đề về đạo đức cần lưu ý trong quá trình thực hiện đồ án bao gồm:

* Bảo mật thông tin: Khi thu thập dữ liệu, cần đảm bảo bảo mật thông tin của người tham gia.
* Chống phân biệt đối xử: Không sử dụng hệ thống nhận diện khuôn mặt để phân biệt đối xử với bất kỳ nhóm người nào.
* Quyền riêng tư: Không sử dụng hệ thống nhận diện khuôn mặt để xâm phạm quyền riêng tư của người khác.

Các vấn đề về làm việc chuyên nghiệp

Trong quá trình thực hiện đồ án, cần có tác phong chuyên nghiệp, thể hiện qua các khía cạnh sau:

* Trách nhiệm: Mỗi thành viên trong nhóm cần có trách nhiệm với công việc của mình. Các thành viên cần hoàn thành đúng thời hạn và chất lượng công việc được giao.
* Kỷ luật: Các thành viên trong nhóm cần tuân thủ các quy định của nhóm, đảm bảo tiến độ thực hiện đồ án.
* Tôn trọng lẫn nhau: Các thành viên trong nhóm cần tôn trọng lẫn nhau, không phân biệt đối xử với nhau.

## **4.4 Kế hoạch cho kiến thức mới và chiến lược học tập**

Trong quá trình thực hiện đồ án, nhóm nghiên cứu cần tìm hiểu và nắm vững một số kiến thức mới, bao gồm:

* Kiến thức về nhận diện khuôn mặt: Nhóm nghiên cứu cần tìm hiểu về các thuật toán nhận diện khuôn mặt phổ biến, chẳng hạn như MTCNN và FaceNet.
* Kiến thức về phát triển mô hình học máy: Nhóm nghiên cứu cần tìm hiểu về các phương pháp phát triển mô hình học máy, chẳng hạn như học máy có giám sát và học máy không giám sát.
* Kiến thức về xây dựng hệ thống: Nhóm nghiên cứu cần tìm hiểu về các bước xây dựng hệ thống, chẳng hạn như thu thập dữ liệu, phát triển mô hình, triển khai hệ thống chiến lược học tập

Để nắm vững các kiến thức mới, nhóm nghiên cứu cần có chiến lược học tập hiệu quả, bao gồm:

* Lập kế hoạch học tập: Nhóm nghiên cứu cần lập kế hoạch học tập cụ thể, bao gồm các mục tiêu học tập, thời gian học tập, phương pháp học tập,...
* Tìm kiếm tài liệu học tập: Nhóm nghiên cứu cần tìm kiếm các tài liệu học tập chất lượng, chẳng hạn như sách, bài báo, tài liệu trực tuyến,...
* Tham gia các khóa học trực tuyến: Nhóm nghiên cứu có thể tham gia các khóa học trực tuyến để nâng cao kiến thức và kỹ năng.
* Tham gia các cộng đồng học tập: Nhóm nghiên cứu có thể tham gia các cộng đồng học tập để trao đổi, học hỏi kinh nghiệm từ các thành viên khác.

# **DANH MỤC TÀI LIỆU THAM KHẢO**

[1] G. Yang and T. S. Huang, “Human Face Detection in Complex Background”, Pattern Recognition, vol. 27, no. 1, pp. 53-63, 1994.

[2] A. Lanitis, C.J. Taylor, and T.F. Cootes, “An Automatic Face Identification System Using lexible Appearance Models”, Image and Vision Computing, vol. 13, no. 5, pp. 393-401, 1995.

[3] Andrew King, “A Survey of Methods for Face Detection”, 2003.

[4] A. Yuille, P. Hallinan, and D. Cohen, “Feature Extraction from Faces Using eformable Templates”, Int’l J. Computer Vision, vol. 8, no. 2, pp. 99-111, 1992.

[5] D.G. Kendall, “Shape Manifolds, Procrustean Metrics, and Complex Projective Shapes”, Bull. London Math. Soc., vol. 16, pp. 81-121, 1984.

[6] E. Osuna, R. Freund, and F. Girosi, “Training Support Vector Machines: An Application to Face Detection” Proc. IEEE Conf. Computer Vision and Pattern Recognition, pp. 130-136, 1997.

[7] Fuzhen Huang and Jianbo Su, “Multiple Face Contour Detection Using adaptive Flows”, Sinobiometrics 2004, LNCS 3338, pp. 137-143, Springer-Verlag Berlin eidelberg, 2004.

[8] H.P. Graf, T. Chen, E. Petajan, and E. Cosatto, “Locating Faces and Facial Parts”, Proc. First Int’l Workshop Automatic Face and Gesture Recognition, pp. 41-46, 1995.

[9] J. Canny, “A Computational Approach to Edge Detection”, IEEE Trans. Pattern Analysis and Machine Intelligence, vol. 8, no. 6, pp. 679-698, June 1986. [10] J. Sobottka and I. Pitas, “Segmentation and Tracking of Faces in Color Images”, Proc. Second Int’l Conf. Automatic Face and Gesture Recognition, pp. 236-241, 1996