|  |
| --- |
| **ĐẠI HỌC QUỐC GIA HÀ NỘI** |
| **TRƯỜNG ĐẠI HỌC CÔNG NGHỆ** |
|  |
| **BÁO CÁO BÀI TẬP LỚN**  **MÔN NHẬP MÔN LẬP TRÌNH** |
| **ĐỀ TÀI: MÔ PHỎNG ROBOT 3D** |
| **Nguyễn Ngọc Sơn – 24021197**  **Trương Hữu Nghĩa – 24021169** |
| **Giảng viên : Trần Thanh Hải**  **Lớp: K69M-MT3**  **Mã học phần: INT1008-2** |
| **HÀ NỘI - 2025** |

# LỜI CẢM ƠN

Chúng em xin phép được bày tỏ lòng biết ơn sâu sắc đến thầy Trần Thanh Hải vì đã dành thời gian và tâm huyết để hướng dẫn chúng em thực hiện dự án. Sự hỗ trợ và chỉ dẫn tận tình của thầy Trần Thanh Hải đóng vai trò then chốt trong thành công của dự án này.

Ban đầu, chúng em cảm thấy khá hoang mang và gặp nhiều khó khăn trong việc định hướng nghiên cứu cũng như giải quyết các vấn đề kỹ thuật. Tuy nhiên, nhờ sự định hướng rõ ràng, những lời khuyên xác đáng và những buổi trao đổi chuyên môn sâu sắc của thầy, chúng em đã dần nắm bắt được vấn đề, tìm ra hướng giải quyết tối ưu và hoàn thiện dự án theo đúng tiến độ.

Chúng em đặc biệt trân trọng sự kiên nhẫn và nhiệt tình của thầy khi luôn sẵn lòng giải đáp mọi thắc mắc của chúng em, dù là những câu hỏi nhỏ nhất. Thầy không chỉ truyền đạt kiến thức chuyên môn mà còn chia sẻ những kinh nghiệm thực tế, giúp chúng em hiểu rõ hơn về lĩnh vực Công nghệ thông tin, lập trình và định hướng sự nghiệp của mình.

Hơn thế nữa, sự khích lệ và động viên từ thầy đã giúp chúng em có thêm động lực để vượt qua những trở ngại và không ngừng nỗ lực để đạt được kết quả tốt nhất. Chúng em xin hứa sẽ tiếp tục học hỏi và trau dồi kiến thức để xứng đáng với sự kỳ vọng của thầy.

Một lần nữa, chúng em xin chân thành cảm ơn thầy đã đồng hành cùng chúng em trong suốt quá trình thực hiện dự án. Chúng em xin chúc thầy luôn mạnh khỏe, hạnh phúc và thành công trong sự nghiệp trồng người

# MỤC LỤC

[1. GIỚI THIỆU 4](#_Toc198324900)

[1.1 Giới thiệu về Codeblock 4](#_Toc198324901)

[1.2 Giới thiệu về thư viện OpenGL 4](#_Toc198324902)

[1.3 Lí do chọn đề tài 5](#_Toc198324903)

[2. CHI TIẾT CHƯƠNG TRÌNH 7](#_Toc198324904)

[2.1 Thư viện khai báo 7](#_Toc198324905)

[2.2 Sơ đồ khối 8](#_Toc198324906)

[2.3 Các module được xây dựng 10](#_Toc198324907)

[2.4 Lớp, Enum 10](#_Toc198324908)

[2.5 Hàm 10](#_Toc198324909)

[3. CÀI ĐẶT VÀ SỬ DỤNG 11](#_Toc198324910)

[3.1 Hướng dẫn cài đặt CodeBlock và cài thư viện OpenGL trên CodeBlock 11](#_Toc198324911)

[3.2 Hướng dẫn các thao tác với chương trình. 14](#_Toc198324912)

[4. ĐÁNH GIÁ VÀ PHƯƠNG HƯỚNG PHÁT TRIỂN 17](#_Toc198324913)

[4.1 Đánh giá 17](#_Toc198324914)

[4.2 Phương hướng phát triển 17](#_Toc198324915)

# GIỚI THIỆU

## Giới thiệu về Codeblock

Để thực hiện dự án này, em đã lựa chọn Code::Blocks làm môi trường phát triển chính. Em xin phép được trình bày lý do lựa chọn này:

* Code::Blocks là một IDE mã nguồn mở, miễn phí và đa nền tảng. Điều này cho phép em dễ dàng tiếp cận và sử dụng mà không lo lắng về vấn đề bản quyền.
* Code::Blocks hỗ trợ tốt cho việc lập trình C++, ngôn ngữ chính mà em sử dụng trong dự án.
* Code::Blocks cung cấp khả năng tích hợp và quản lý thư viện OpenGL một cách dễ dàng. Việc thiết lập và sử dụng các hàm của OpenGL trở nên đơn giản hơn nhiều.
* Code::Blocks có giao diện thân thiện và dễ sử dụng, giúp em tập trung vào việc viết code và debug chương trình một cách hiệu quả.

## Giới thiệu về thư viện OpenGL

OpenGL (Open Graphics Library) là một API (Application Programming Interface) đồ họa 2D và 3D đa nền tảng. Nó không phải là một ngôn ngữ lập trình, mà là một tập hợp các hàm cho phép các ứng dụng phần mềm giao tiếp với phần cứng đồ họa (GPU) để tạo ra các hình ảnh, video và các hiệu ứng đồ họa phức tạp.

### **1.2.1 Các tính năng nổi bật của OpenGL:**

* Kết xuất 3D (3D Rendering): OpenGL cung cấp các hàm để định nghĩa và kết xuất các đối tượng 3D, bao gồm việc xác định hình dạng, màu sắc, vật liệu, và ánh sáng. Dự án của em sử dụng OpenGL để tạo ra mô hình 3D của robot và môi trường xung quanh.
* Biến đổi hình học (Geometric Transformations): OpenGL cho phép thực hiện các phép biến đổi như xoay, tịnh tiến, co giãn các đối tượng 3D. Điều này rất quan trọng trong dự án của em để di chuyển và điều khiển các khớp của robot.
* Ánh sáng và Vật liệu (Lighting and Materials): OpenGL hỗ trợ các mô hình ánh sáng khác nhau (ambient, diffuse, specular) và cho phép định nghĩa các thuộc tính vật liệu (màu sắc, độ bóng, độ phản xạ) để tạo ra hình ảnh chân thực hơn. Dự án của em sử dụng các kỹ thuật này để làm cho robot trông sống động hơn.
* Texture Mapping: OpenGL cho phép áp dụng các hình ảnh (textures) lên bề mặt các đối tượng 3D, tạo ra các chi tiết phức tạp và tăng tính chân thực.
* Shader Programming: OpenGL hỗ trợ ngôn ngữ shader (GLSL), cho phép các nhà phát triển viết các chương trình nhỏ chạy trực tiếp trên GPU để tùy chỉnh các hiệu ứng đồ họa. Dự án của em có thể sử dụng shaders để tạo ra các hiệu ứng đặc biệt.

### **1.2.2 Ưu điểm của OpenGL:**

* Đa nền tảng: OpenGL hoạt động trên nhiều hệ điều hành (Windows, Linux, macOS) và phần cứng khác nhau. Điều này giúp các ứng dụng OpenGL có thể được triển khai trên nhiều nền tảng khác nhau mà không cần thay đổi mã nguồn.
* Hiệu suất cao: OpenGL được thiết kế để tận dụng tối đa sức mạnh của GPU, cho phép tạo ra các ứng dụng đồ họa phức tạp với hiệu suất cao.
* Tính linh hoạt: OpenGL cung cấp một loạt các hàm và tính năng cho phép các nhà phát triển tạo ra các ứng dụng đồ họa đa dạng.
* Cộng đồng lớn: OpenGL có một cộng đồng người dùng và nhà phát triển lớn và tích cực, cung cấp nhiều tài liệu, hướng dẫn và ví dụ mã nguồn.

## Lí do chọn đề tài

Em xin trình bày lý do em lựa chọn đề tài "Lập trình mô phỏng Robot 3D bằng OpenGL" cho dự án của mình. Quyết định này được dựa trên nhiều lí do cụ thể:

### **1.3.1 Tính cấp thiết và ứng dụng thực tiễn:**

* Mô phỏng robot là một lĩnh vực quan trọng: Trong bối cảnh tự động hóa ngày càng phát triển, mô phỏng robot đóng vai trò quan trọng trong việc thiết kế, phát triển, kiểm thử và huấn luyện robot. Mô phỏng giúp tiết kiệm chi phí, thời gian và giảm thiểu rủi ro trong quá trình phát triển robot thực tế.
* Ứng dụng rộng rãi trong nhiều ngành: Mô phỏng robot được ứng dụng rộng rãi trong các lĩnh vực như công nghiệp sản xuất, y tế, logistics, nông nghiệp và nghiên cứu khoa học. Việc xây dựng một môi trường mô phỏng robot cho phép người dùng kiểm tra các thuật toán điều khiển, tối ưu hóa quá trình làm việc và đào tạo nhân viên vận hành một cách an toàn và hiệu quả.

### **1.3.2 Tính khả thi và phù hợp với kiến thức:**

* Kiến thức về lập trình C++: Em đã có kiến thức nền tảng vững chắc về lập trình C++, đây là ngôn ngữ chính được sử dụng để phát triển ứng dụng đồ họa với OpenGL.
* Kiến thức về đồ họa máy tính: Trong quá trình học tập, em đã được trang bị kiến thức về các khái niệm cơ bản của đồ họa máy tính như không gian 3D, biến đổi hình học, ánh sáng, vật liệu, texture mapping... Đây là những kiến thức cần thiết để xây dựng môi trường mô phỏng 3D.
* Khả năng sử dụng thư viện OpenGL: Em đã có kinh nghiệm làm việc với thư viện OpenGL thông qua các bài tập và dự án nhỏ. Việc sử dụng OpenGL để xây dựng môi trường mô phỏng 3D là một thử thách phù hợp để em nâng cao kỹ năng và hiểu sâu hơn về thư viện này.

### **1.3.3 Tính sáng tạo và khả năng phát triển:**

* Khả năng tùy biến và mở rộng: OpenGL là một thư viện đồ họa rất linh hoạt và cho phép tùy biến cao. Em có thể tùy chỉnh các tham số của ánh sáng, vật liệu, texture và shader để tạo ra các hiệu ứng đồ họa độc đáo và phù hợp với mục tiêu của dự án.
* Khả năng mở rộng dự án: Dự án có thể được mở rộng để hỗ trợ nhiều loại robot khác nhau, tích hợp các cảm biến ảo, mô phỏng môi trường làm việc phức tạp và kết nối với các hệ thống điều khiển robot thực tế.

Tóm lại, việc lựa chọn đề tài "Lập trình mô phỏng Robot 3D bằng OpenGL" là một quyết định hợp lý dựa trên tính cấp thiết, tính khả thi và tính sáng tạo của đề tài. Em tin rằng dự án này không chỉ giúp em củng cố kiến thức và kỹ năng mà còn mang lại những giá trị thiết thực cho cộng đồng.

# CHI TIẾT CHƯƠNG TRÌNH

## Thư viện khai báo

Các thư viện được sử dụng trong Project bao gồm:

### <GL/glut.h> // Thư viện OpenGL dùng để vẽ hình 2D và 3D

### <windows.h> //thư viện để xử lí hiển thị trên màn hình Console

### <memory> //Tổ hợp nhiều thư viện để sử dụng unique pointer

### <unordered\_map> // sử dụng CTDL unordered\_map

### <string> // sử dụng lớp string xử lí chuỗi

### <cmath> // Thư viện để xử lí các hàm toán học

### <iostream> // Thư viện để sử dụng luồng vào, ra chuẩn

### <vector> //Thư viện để sử dụng lớp vector

### <map> //Thư viện để sử dụng CTDL map

### <set> //Thư viện để sử dụng CTDL set

### <multimap> //Thư viện để sử dụng CTDL multimap

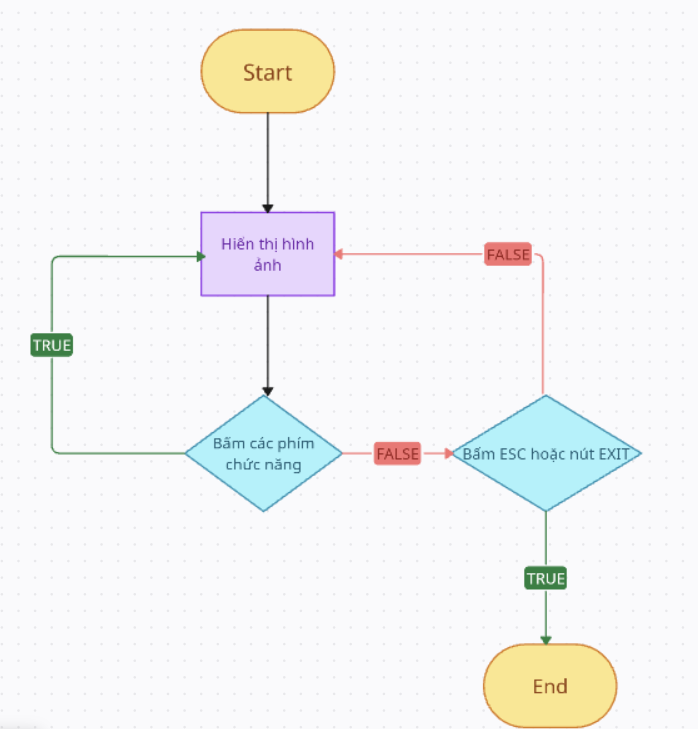
### <multiset> //Thư viện để sử dụng CTDL multiset

### <Algorithm> //Thư viện để sử dụng các thuật toán built-in

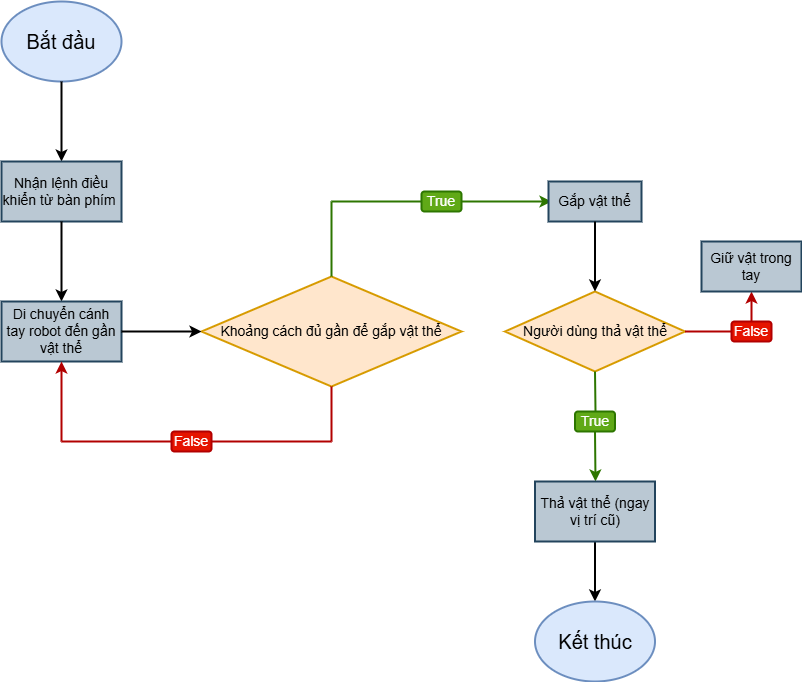
### <fstream> //Thư viện để xử lý luồng

## Sơ đồ khối

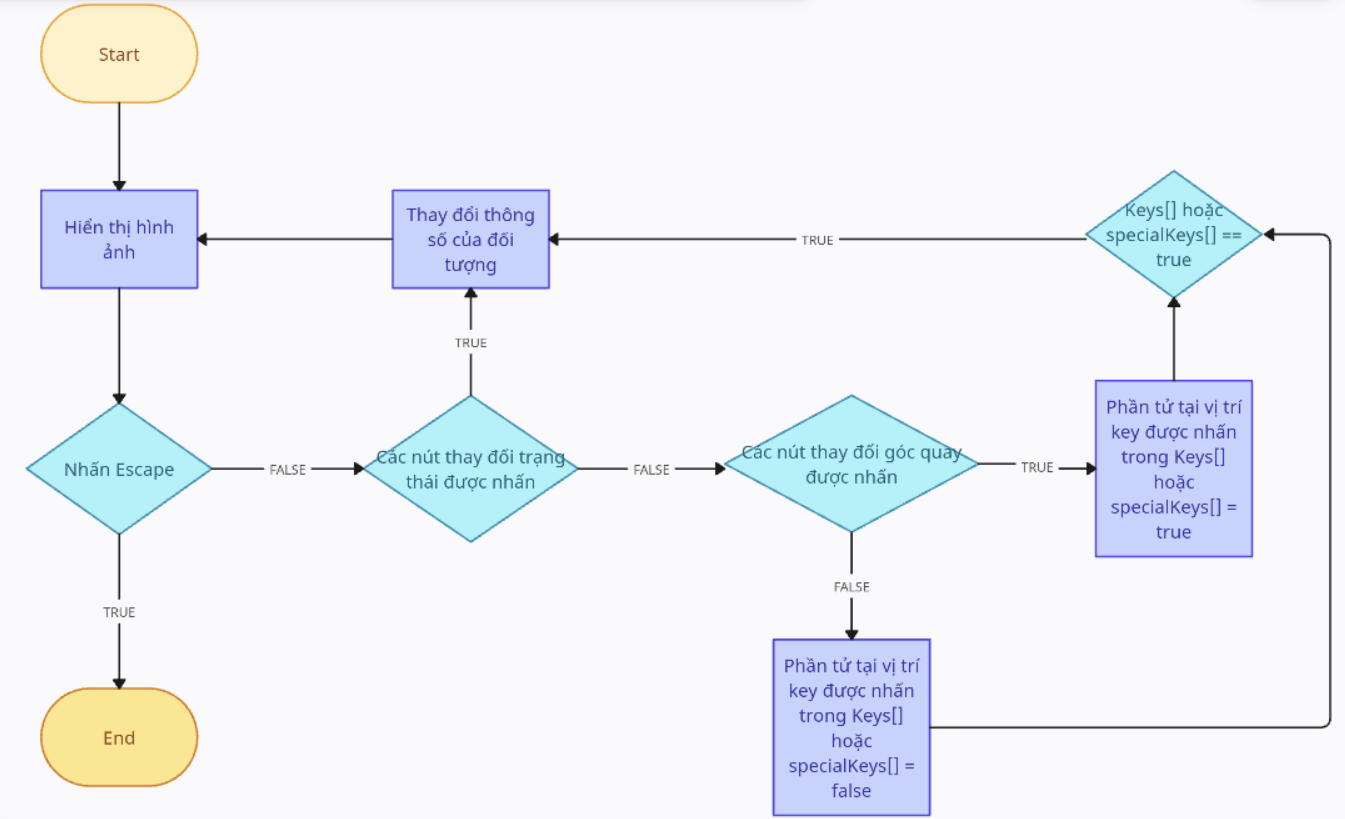
### Sơ đồ khối tổng thể



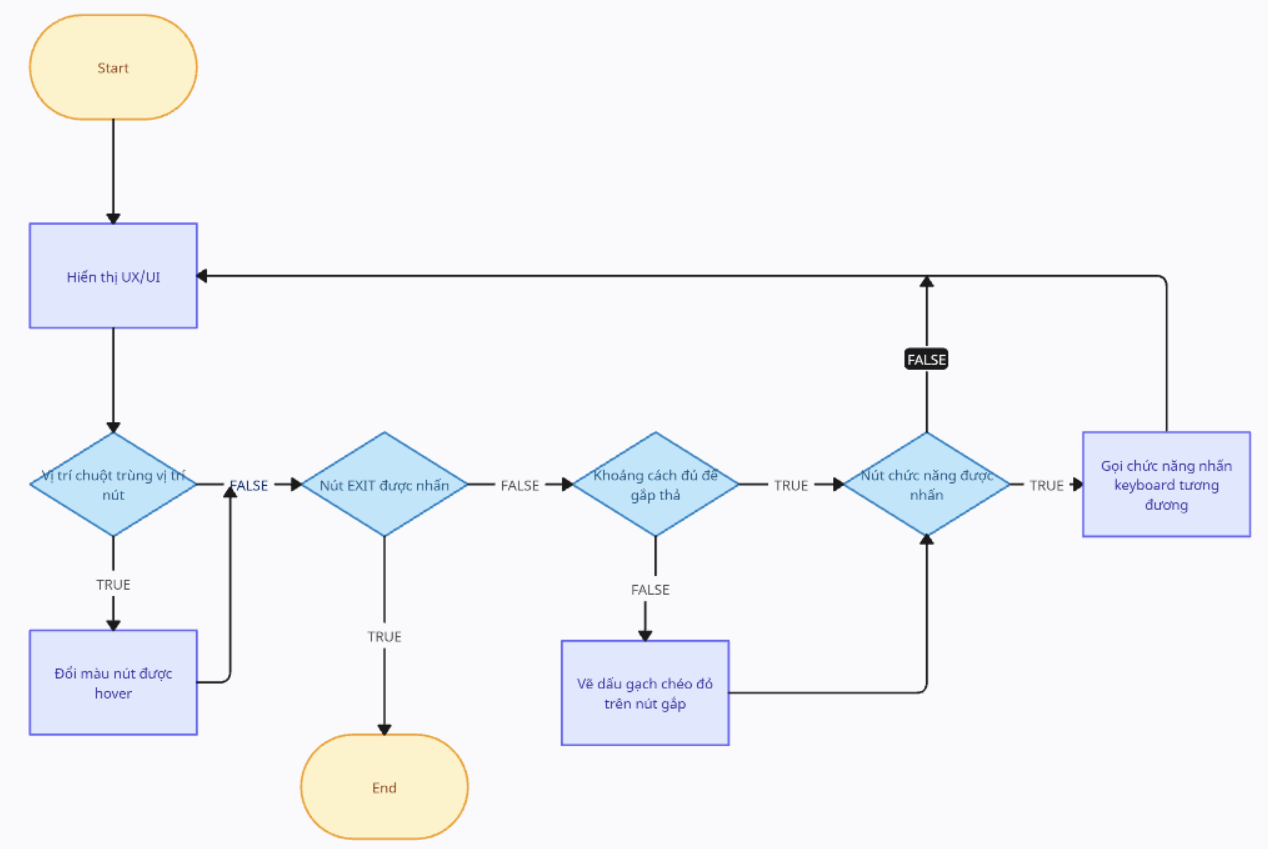
### Sơ đồ khối thực hiện chức năng gắp đồ vật



### Sơ đồ khối xử lý logic khi ấn bàn phím



### **2.2.4** Sơ đồ khối xử lí logic UX/UI



## Các module được xây dựng

### “include/Config.h” : Lưu trữ các const và enum

### “include/Console.h” : Chứa các hàm xử lí hiển thị Console

### “include/Utils.h” : Chứa các hàm tiện ích cơ sở

### “include/UI.h” : Chứa các class UX/UI

### “include/UI\_Attributes.h” : Chứa các thông số const của UI

### “include/RobotArm.h” : Chứa các class Robot

## Lớp, Enum

1. Class RobotArm //chứa các phương thức và trạng thái của Robot
2. Class UI\_Elements // Abstract class cho các lớp UX/UI khác
3. Class Label : public UI\_Elements // Các đối tượng là kí tự trên UI
4. Class Button : public UI\_Elements //Các nút bấm hình chữ nhật
5. Class Symbol : public UI\_Elements // các biểu tượng trên các nút bấm
6. Class UI\_Manager // Lớp chứa unordered\_map có value là con trỏ UI\_Elements để quản lý tất cả các UI
7. Enum UIObject //Tên các đối tượng UI
8. Enum TypeStatus //Các loại trạng thái của class RobotArm
9. Enum TypeAngle //Các loại góc của class RobotArm
10. Enum TypeVector //Các loại Vector của class RobotArm
11. Enum Color //Các màu được sử dụng trong chương trình

## Hàm

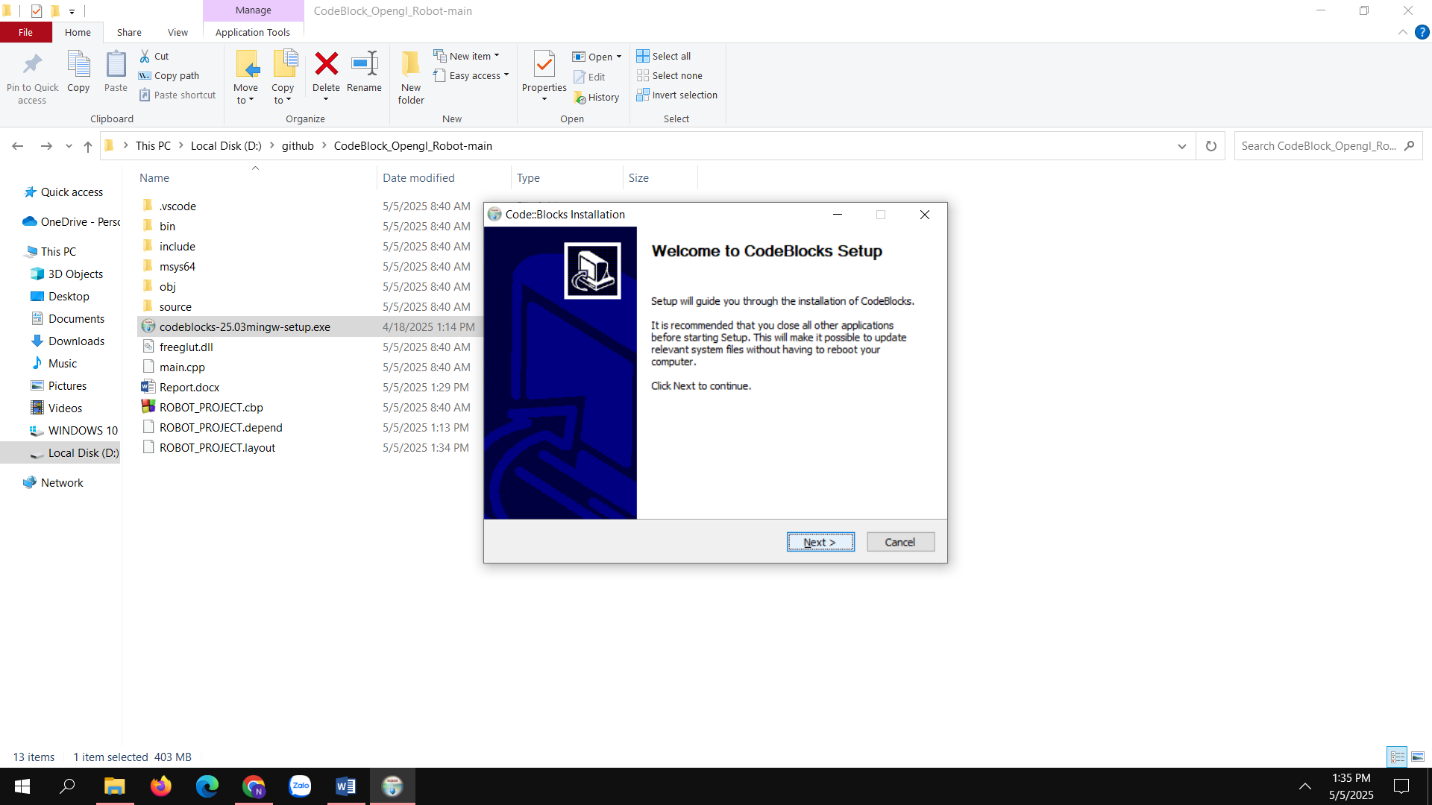
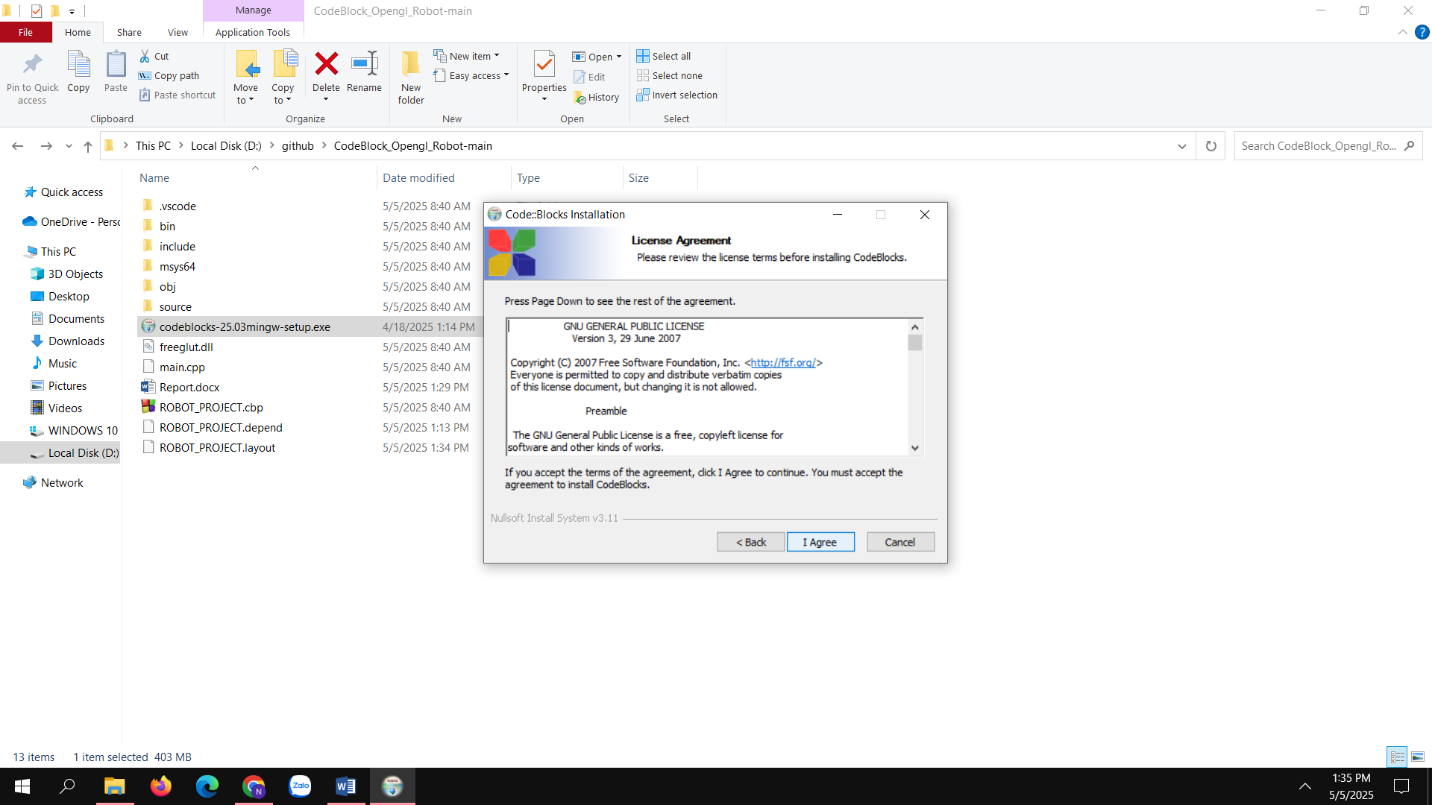
1. Void buttonOver(int mouseX, int mouseY); //Kiểm tra nút được hover
2. Bool getStatusHovered(); //Lấy trạng thái hovered của đối tượng
3. Boid setStatusHovered(bool status); //Gán trạng thái hovered cho đối tượng
4. Bool getDisable(); // Lấy trạng thái disable của đối tượng
5. Void setDisable(bool status); //Gán trạng thái disable cho đối tượng
6. Void setPosition(int x, int y); //Gán vị trí x,y cho đối tượng
7. Void draw(); //Hàm vẽ sẽ gọi tất cả các hàm vẽ bộ phận khác
8. Void drawWheel(); // Hàm vẽ bánh xe
9. Void drawRobotBody(); //Hàm vẽ thân Robot
10. Void drawRobotJoint(); //Hàm vẽ khuỷu tay Robot
11. Void drawRobotShortArm(); //Hàm vẽ cánh tay ngắn của Robot
12. Void drawRobotLongArm(); //Hàm vẽ cánh tay dài của Robot
13. Void drawRobotHand(); //Hàm vẽ bàn tay Robot
14. Void drawContainer(); //Hàm vẽ vật gắp
15. Void drawDirection(); //Hàm vẽ hướng Vector
16. Void printAttributes(); //Hàm in thông số của Robot
17. Bool checkHandClaw(); //Hàm kiểu tra xem bàn tay có đang gắp vật không
18. Void checkMinValueAngle(); // Giúp cánh tay chỉ quay một khoảng cụ thể
19. Void checkMaxValueAngle(); //Giúp cánh tay chỉ quay một khoảng cụ thể
20. Void update(); // Hàm check lại chỉ số khi có một sự kiện xảy ra
21. Void rotateAngle(TypeAngle angle, float rotate); //Quay các vector một góc
22. Void changeStatus(TypeStatus status); //Thay đổi trạng thái
23. Void move(float); //Di chuyển robot
24. Bool getStatus(TypeStatus status); //Trả về trạng thái hiện tại
25. Point getCenterPoint(); //Trả về Struct Point hiện tại
26. Point getCenterHeadPoint();
27. Vector getVector(TypeVector v); //Trả về Struct Vector hiện tại

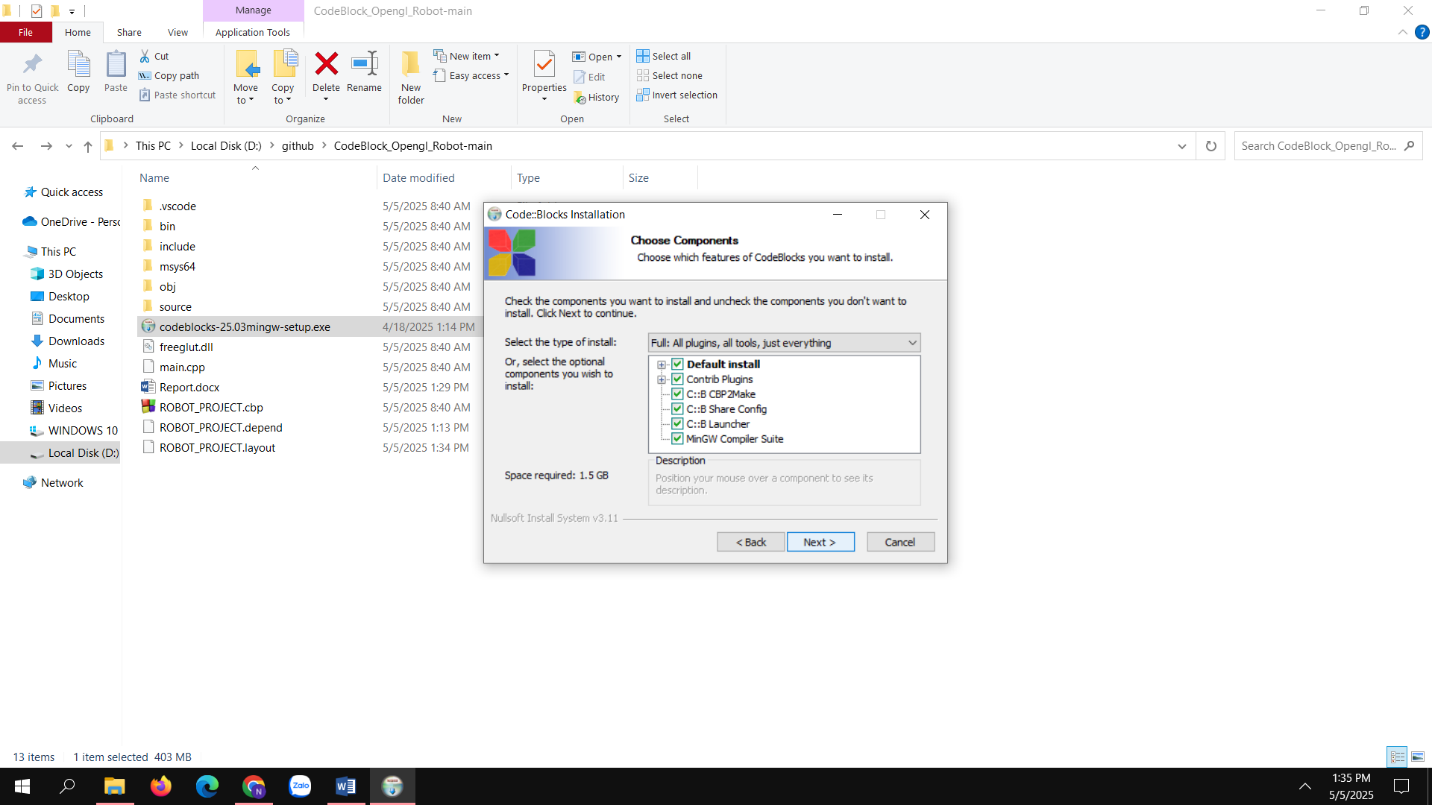
# 3. CÀI ĐẶT VÀ SỬ DỤNG

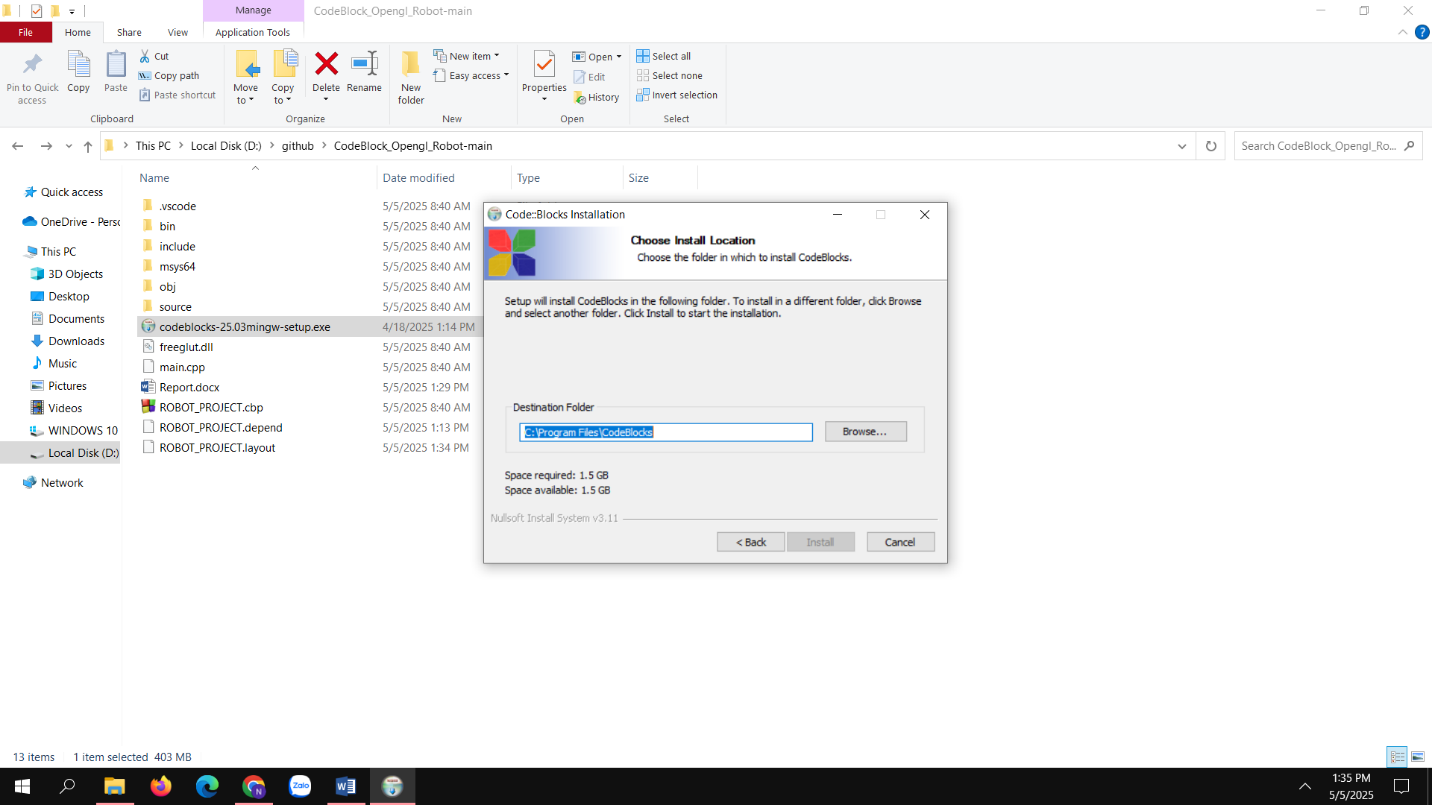
## Hướng dẫn cài đặt CodeBlock và cài thư viện OpenGL trên CodeBlock

**Bước 1:** Download dự án từ Internet [link](https://github.com/Sune27/CodeBlock_Opengl_Robot.git) và download file setup tại đây [link(2)](https://drive.google.com/file/d/10ZQTJ9i0O24y2x4TXWIq8Cj7AWfa1QSN/view?usp=sharing)

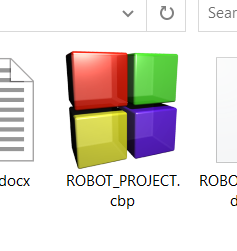
**Bước 2:** Mở cài đặt file setup Codeblocks

1. Nhấn Next.
2. Nhấn I Agree.
3. Nhấn Next.

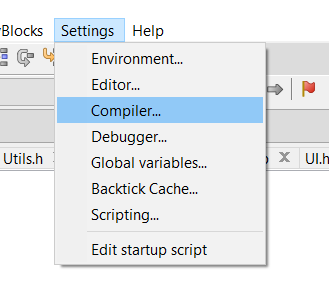


1. Chọn ổ và nhấn Install để hoàn tất down dự án về.

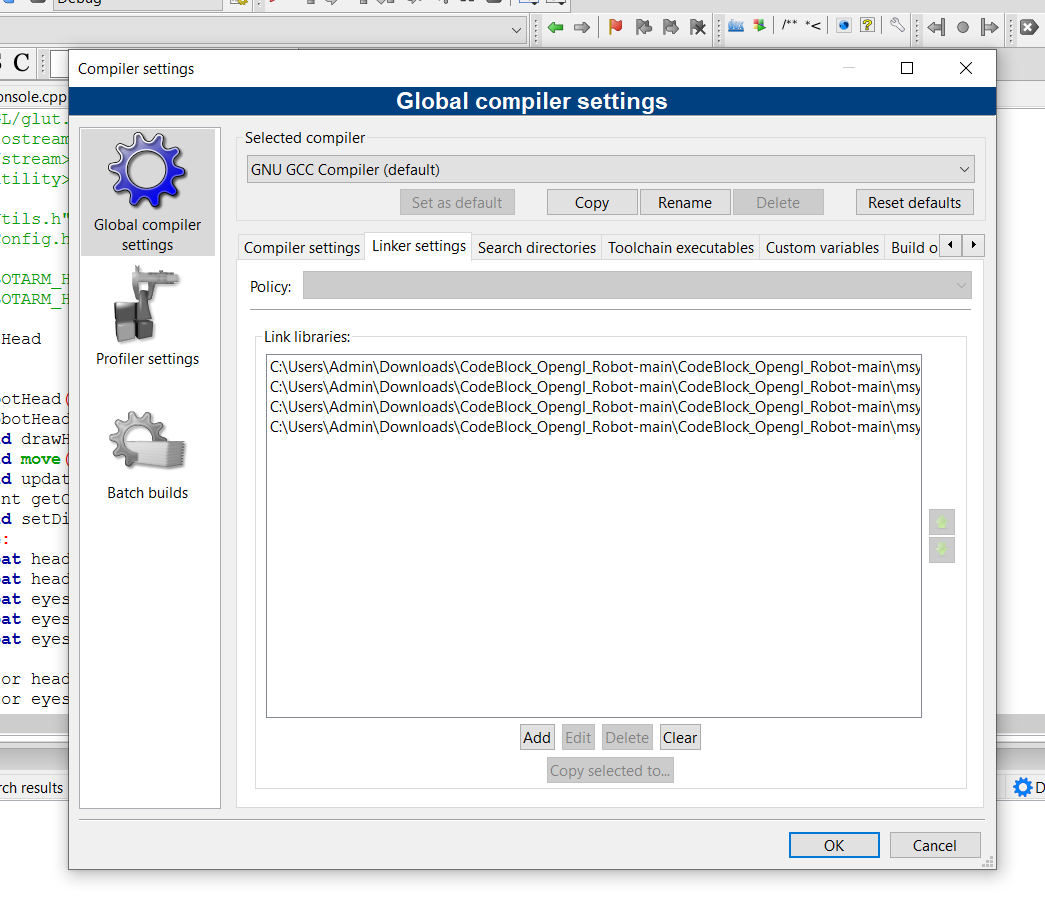
**Bước 3** : Chọn vào file .cbp



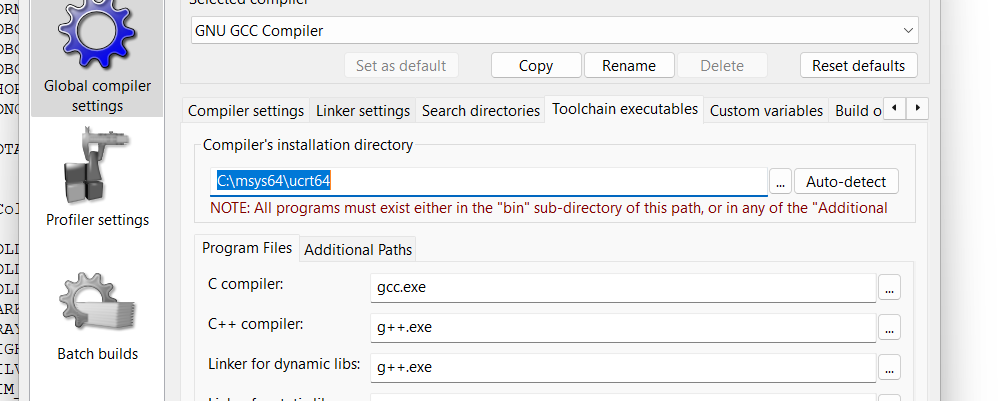
**Bước 4:** Chọn Setting -> Compiler



**Bước 5:** Thêm thư viện trong Linker settings

**Tab:** Linker settings

**Add các thư viện sau:** libopengl32.a, libglu32.a, libglut32.a, libfreeglut.a

**Tại Toolchain executables, chuyển trình biên dịch thành thư mục ucrt64 (vừa tải về từ link)**

## Hướng dẫn các thao tác với chương trình.

### Thoát khỏi chương trình

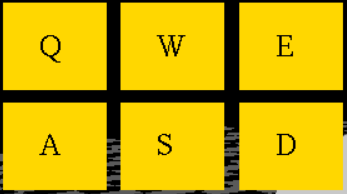
Nhấn nút EXIT trên màn hình hoặc nút escape trên bàn phím

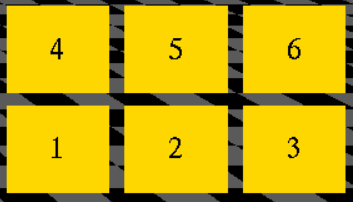
### **3.2.2** Điều chỉnh góc nhìn máy quay

hoặc nhấn các nút trên màn hình

### **3.2.3** Điều chỉnh hướng khớp tay

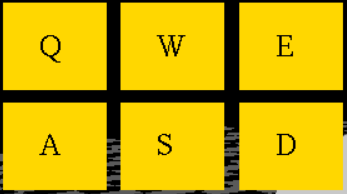
* **Tay trái:**
* Nhấn **A** để xoay khớp lên
* Nhấn **Q** để xoay khớp xuống



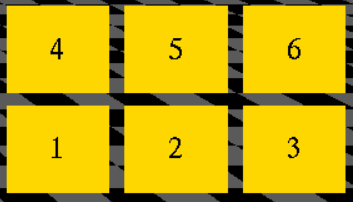
* **Tay phải:**
* Nhấn **1** để xoay khớp lên
* Nhấn **4** để xoay khớp xuống

### **3.2.4** Điều chỉnh hướng cánh tay

* **Tay trái:**
* Nhấn **E** để đưa cánh tay lên
* Nhấn **D** để hạ cánh tay xuống
* Nhấn **S** để xoay cánh tay sang trái
* Nhấn **W** để xoay cánh tay sang phải



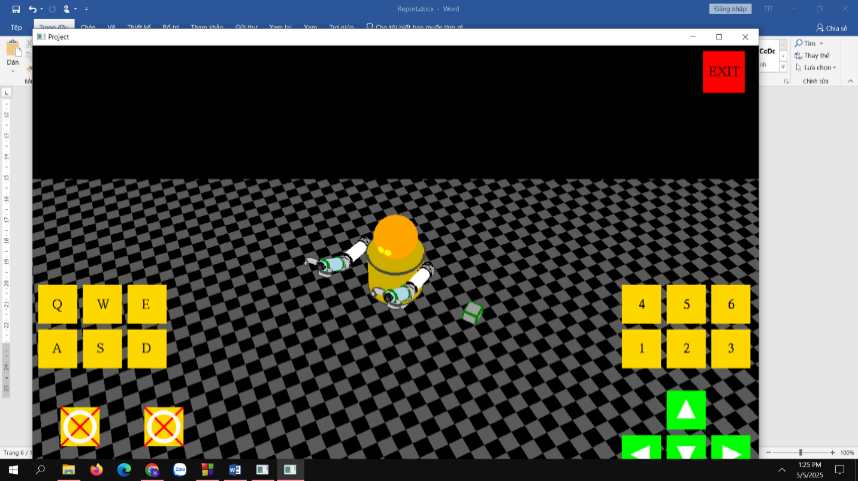
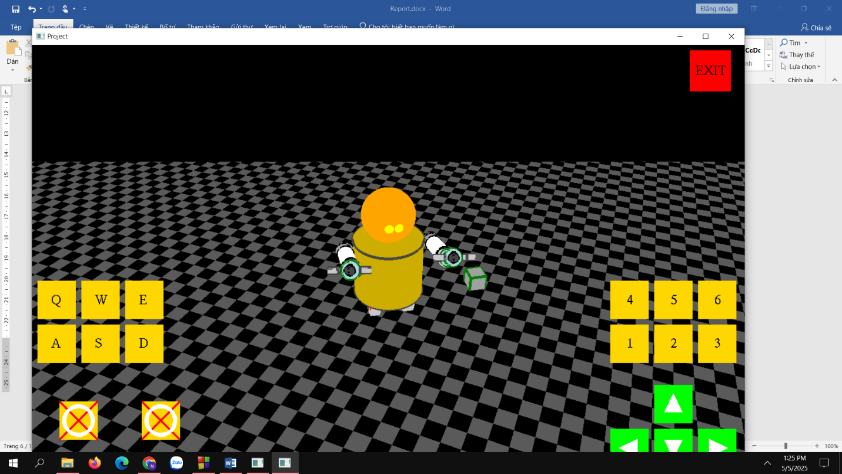
* **Tay phải:**
* Nhấn **6** để đưa cánh tay lên
* Nhấn **3** để hạ cánh tay xuống
* Nhấn **2** để xoay cánh tay sang trái
* Nhấn **5** để xoay cánh tay sang phải



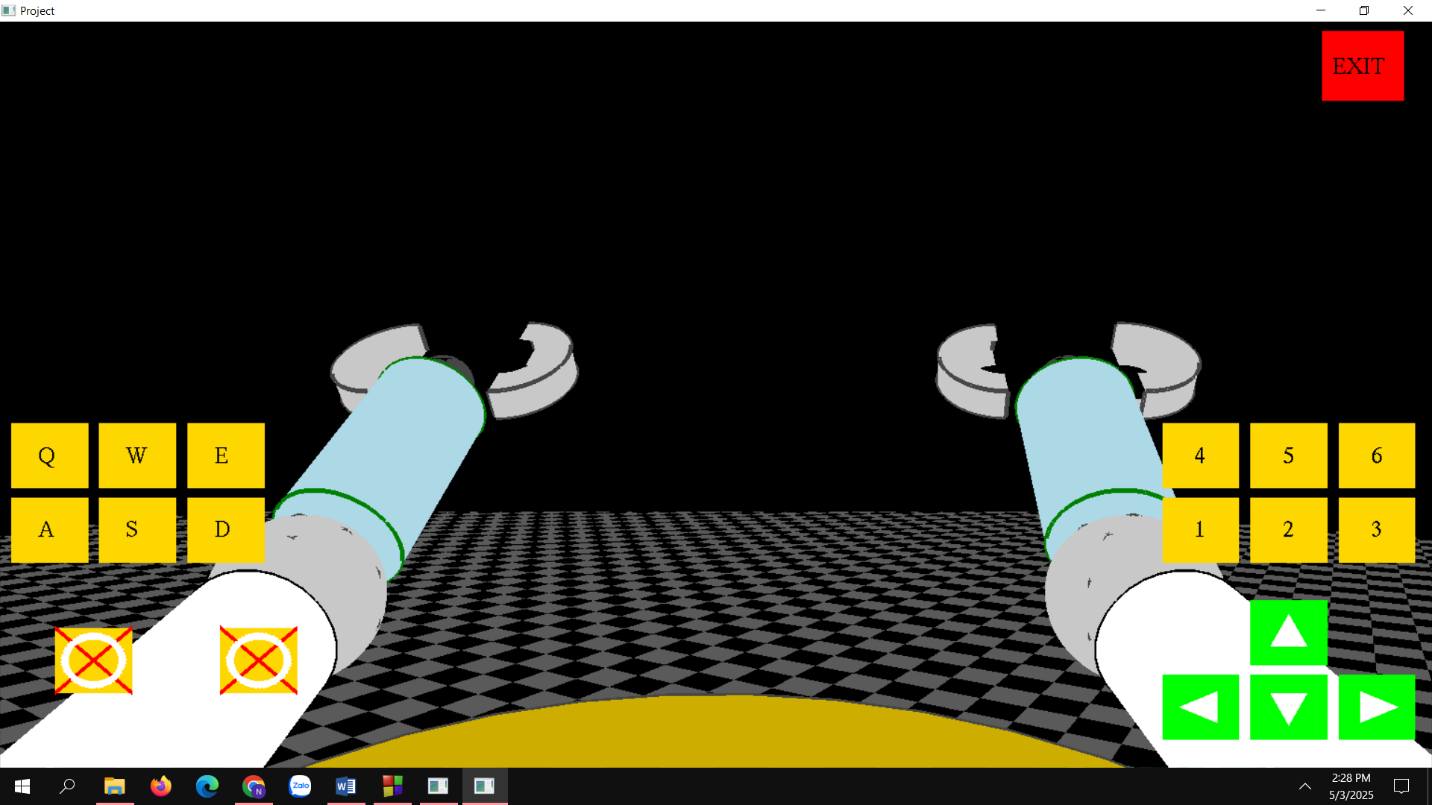
### **3.2.5** Gắp và thả vật thể

Di chuyển bàn tay đến gần vật thể, khi khoảng cách đủ gần (không có dấu X màu đỏ) tức là đã có thể gắp

### **3.2.6** Xoay góc nhìn

Dùng con trỏ để thay đổi.

### Chuyển đổi góc nhìn

Nhấn F1 để chuyển đổi giữa góc nhìn thứ nhất và góc nhìn thứ ba.

# ĐÁNH GIÁ VÀ PHƯƠNG HƯỚNG PHÁT TRIỂN

## ****Đánh giá****

Dự án đã hoàn thành việc mô phỏng cánh tay robot sử dụng ngôn ngữ C++ và thư viện OpenGL. Chương trình cho phép người dùng điều khiển các khớp tay robot thông qua bàn phím và các nút giao diện trực quan, thực hiện các thao tác cơ bản như xoay khớp và gắp/thả vật thể.

### **Ưu điểm:**

* Giao diện đơn giản, dễ sử dụng.
* Mô phỏng chuyển động chính xác, phản hồi điều khiển tốt.
* Sơ đồ thuật toán rõ ràng, dễ bảo trì và nâng cấp.

### **Hạn chế:**

* Điều khiển còn mang tính thủ công, chưa tích hợp động học nghịch (IK).
* Giao diện chưa hiển thị thông số trạng thái robot.
* Thiếu mô phỏng vật lý như va chạm hay trọng lực.

## ****Phương hướng phát triển****

Để nâng cao chất lượng mô phỏng và mở rộng khả năng ứng dụng, dự án có thể phát triển theo các hướng sau:

### **Cải thiện giao diện người dùng:**

* + Nâng cấp hệ thống nút điều khiển.
  + Bổ sung hiển thị thông số trạng thái như góc khớp, tọa độ đầu gắp, trạng thái gắp/thả.

### **Mô phỏng vật lý:**

* + Áp dụng các thư viện vật lý để mô phỏng va chạm, trọng lực và tương tác thực tế hơn với vật thể.

### **Mở rộng khả năng điều khiển:**

* + Kết nối với các thiết bị ngoại vi như joystick hoặc hệ thống điều khiển thời gian thực qua mạng.
  + Tích hợp cảm biến thực tế như từ Arduino hoặc camera để tương tác thông minh hơn.

### **Xuất dữ liệu mô phỏng:**

* + Ghi lại hình ảnh, video mô phỏng hoặc xuất thông số hoạt động phục vụ cho phân tích hoặc trình bày.