CMC UNIVERSITY

BÀI TẬP LỚN
HỌC PHẦN ĐỒ HỌA MÁY TÍNH

Giảng viên: Ngô Trường Giang

Nhóm sinh viên:
1. Lê Thị Hương Ly BCS230054
2. Nguyễn Thị Quỳnh BCS230076
3. Nguyễn Trọng Đoàn BCS230025

Hà Nội, 2025

CMC UNIVERSITY

ĐỀ TÀI

MÔ PHỎNG HOẠT ĐỘNG CỦA ROBOT
TRONG KHÔNG GIAN 3D VỚI THƯ VIỆN OPENGL

Hà Nội, 2025

PHIẾU CHẤM ĐIỂM

Nhận xét chung :.....

.....

.....

.....

.....

.....

.....

.....

.....

Kết quả:

	Họ và tên	Mã sinh viên	Điểm số	Điểm chữ
1.	Lê Thị Hương Ly	BCS230054		
2.	Nguyễn Thị Quỳnh	BCS230076		
3.	Nguyễn Trọng Đoàn	BCS230025		

Giảng viên đánh giá:

Yêu cầu nội dung:

1. Mỗi nhóm (tối đa 3 thành viên) đề xuất một đề tài mô phỏng HOẠT ĐỘNG của đối tượng trong không gian 3D. Ý tưởng đề tài có thể giống nhau nhưng nội dung triển khai phải khác nhau. Ví dụ: Cùng ý tưởng xây dựng robot nhảy, nhưng mỗi robot thiet kế 1 kiểu, nhảy 1 kiểu...
2. Một số hướng đề tài: Mô phỏng các đối tượng: robot, máy bay, ô tô, con vật, mẫu vật, căn phòng ảo, các phương tiện giao thông, các hiện tượng khoa học.... *Không đăng ký đề tài Hệ mặt trời (mặt trăng mặt trời), game rắn săn mồi, rubik, đồ họa 2D.*
3. Nội dung thực hiện:
   - Phát biểu bài toán
   - Phương án giải quyết bài toán
   - Lập trình xử lý, demo kết quả
   - Kết luận
   - Tài liệu tham khảo
4. Các yêu cầu xử lý:
   - Xây dựng đối tượng (mô hình hoá): có thể sử dụng các đối tượng cơ bản đề xây dựng mô hình, hoặc có thể dùng các mô hình được tạo bởi các phần mềm đồ hoạ 3D như blender, 3Dmax...
   - Sử dụng kết hợp các phép biến đổi 3D để mô phỏng các hoạt động của đối tượng 3D.
   - Thiết lập chế độ quan sát (góc nhìn), hiệu ứng tô bóng, blending, texture mapping... cho đối tượng.

Yêu cầu trình bày quyển báo cáo

1. Tiểu luận/ Bài tập lớn được trình bày vi tính trên trang giấy A4 cụ thể như sau: chữ Times New Roman cỡ 14 (hoặc 13). dãn dòng 1.5, lề trái 3cm, lề phải 2cm, trên 2cm và dưới 2cm.
2. Đánh số trang canh giữa theo chiều ngang trong phần lề dưới của văn bản liên tục từ 1 cho đến hết (bắt đầu từ trang nội dung tiểu luận/ bài tập lớn), không dùng các ký tự khác để đánh số trang.
3. Trích dẫn tài liệu: trích dẫn theo kiểu APA hoặc IEEE.

4. Thứ tự các trang
    - Trang bìa:
    - Bìa lót: giấy trắng, trình bày như trang bìa cứng
    - Đề bài (trang riêng)
    - Phần chấm điểm/ nhận xét của giảng viên (trang riêng)
    - Mục lục (trang riêng, đánh tự động)
    - Danh mục từ viết tắt (xếp theo ABC trang riêng)
    - Danh mục bảng biểu, hình vẽ (nếu có), (trang riêng)
    - Nội dung tiểu luận/ bài tập lớn
    - Danh mục tài liệu tham khảo (trang riêng)

MỤC LỤC

I. PHÁT BIỂU BÀI TOÁN

Mô phỏng hoạt động của robot trong không gian 3D bằng lập trình C++ với thư viện đồ họa OpenGL và freeglut, cụ thể là:

- Vẽ mô hình 3D Robot dựa vào các đối tượng cơ bản trong thư viện đồ họa;
- Dựa vào các phép biến đổi 3D, tạo hoạt cảnh Robot thực hiện các hoạt động sau:
    - Chào cờ;
    - Nhảy múa;
    - Chạy tiến;
    - Chạy lùi;
    - Nhảy cao xoay vòng;
    - Nhảy xa;
    - Dừng lại trở về trạng thái đứng yên.
- Thiết lập tô màu và chiếu sáng cho Robot;
- Thay đổi góc nhìn, tăng giảm tốc độ hoạt động của Robot, tạo menu điều khiển.

II. PHƯƠNG ÁN GIẢI QUYẾT BÀI TOÁN

2.1. Thiết kế mô hình 3D Robot

2.1.1. Phương pháp thực hiện

Để vẽ mô hình 3D Robot, nhóm đã sử dụng các hàm tạo đối tượng cơ bản trong thư viện đồ họa, kết hợp với các phép biến đổi. Các hàm tạo đối tượng trong thư viện freeglut được sử dụng bao gồm:

- Hàm glutSolidCube(Gldouble size); // Vẽ hình lập phương với cạnh a = size
- Hàm glutSolidSphere(Gldouble radius, GLint slices, GLint stacks); //Vẽ hình cầu với bán kính, số lượng slice và stack
- Hàm glutSolidCone(Gldouble base, Gldouble height, GLint slices, GLint stacks); //Vẽ hình chóp cụt với bán kính ở mặt đáy, chiều cao, số lượng slice và stack
- Hàm gluCylinder(GLUquadric *qobj, Gldouble baseRadius, Gldouble topRadius, GLdouble height, GLint slices, GLint stacks); //Vẽ hình trụ tròn với đối tượng quadric (được tạo bằng gluNewQuadric), bán kính mặt đáy, bán kính mặt đỉnh, chiều cao, số lượng slice và số lượng stack

Các đối tượng trên được kết hợp với các phép biến đổi để tạo ra thân mình, chân tay,... của Robot, cụ thể là:

- Hàm glScalef(GLfloat x, GLfloat y, GLfloat z); //Biến đổi tỷ lệ kích thước các đối tượng cơ bản với tỷ lệ x theo trục X, tỷ lệ y theo trục Y và tỷ lệ z theo trục Z với tâm tỷ lệ tại gốc tọa độ;
- Hàm glTranslatef(GLfloat x, GLfloat y, GLfloat z); //Tịnh tiến hệ trục tọa độ theo véc tơ tịnh tiến nối từ gốc tọa độ đến điểm (x, y, z);
- Hàm glRotatef(GLfloat angle, GLfloat x, GLfloat y, GLfloat z); //Quay 1 góc angle (angle > 0 thì quay ngược chiều kim đồng hồ) quanh trục nối gốc tọa độ với điểm (x, y, z).

2.1.2. Minh họa cách thực hiện

Để minh họa cách thực hiện vẽ mô hình Robot 3D, ta lấy ví dụ khi vẽ thân mình, cổ và đầu Robot.

- Các hàm tạo đối tượng cơ bản kết hợp với glScaled() như sau:
  - Thân robot được vẽ bằng các hàm:

    glScaled(1., 1., .75); //Thay đổi tỷ lệ cube: 1*x, 1*y, 0.75*z

    glutSolidCube(1.25); //Vẽ cube với a=1.25

    Khi đó ta sẽ được hình hộp chữ nhật với kích thước (1*1.25, 1*1.25, 0.75*1.25) theo 3 chiều x, y, z tương ứng.
  - Cổ robot được vẽ bằng hàm:

    glScaled(1.5, 1.5, 1.5); //Thay đổi tỷ lệ Sphere

    glutSolidSphere(.24, 16, 16); //Vẽ Sphere: r=0.24, 16 slices, 16 stacks

    Khi đó ta vẽ được cổ là hình cầu với bán kính r=1.5*0.24. Ở đây có thể gộp 2 hàm trên bằng glutSolidSphere(.36, 16, 16).
  - Đầu robot được vẽ bằng hàm:

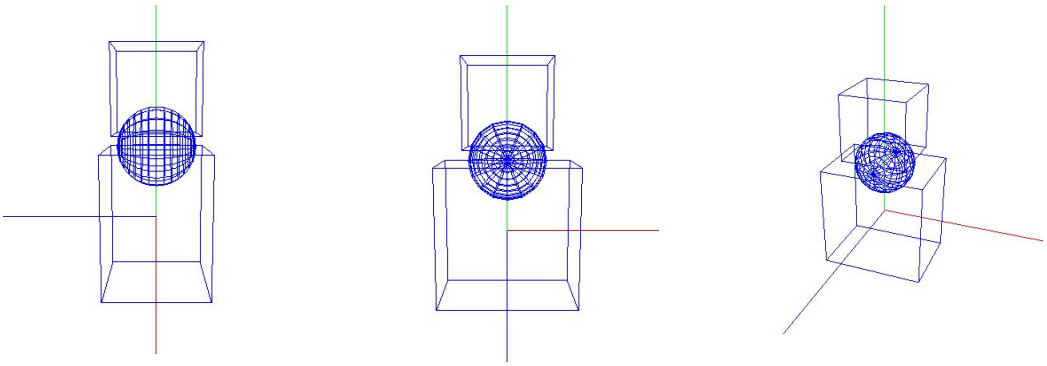
    glutSolidCube(.77); //Vẽ hình lập phương với a=0.77

    Khi đó ta vẽ được đầu là hình lập phương với cạnh a=0.77.
- Các hình trên được vẽ với tâm là gốc tọa độ hiện hành. Nếu không có phép chuyển trục thì thân mình, cổ, đầu robot sẽ bị trùng nhau. Do đó cần kết hợp với các phép biến đổi, cụ thể như sau:

```
/*Thân mình*/
glPushMatrix();              //Lưu lại ma trận hiện hành
glScaled(1., 1., .75);       //Thay đổi tỷ lệ cube: 1*x, 1*y, 0.75*z
glutSolidCube(1.25);         //Vẽ thân mình
glPopMatrix();               //Phục hồi ma trận hiện hành
```

```
/*Cổ*/
glPushMatrix();                 //Lưu lại ma trận hiện hành
glTranslated(0, .65, 0);        //Tịnh tiến 0.65 theo trục y
glutSolidSphere(.36, 16, 16);   //Vẽ cổ
glPopMatrix();                  //Phục hồi ma trận hiện hành

/*Đầu*/
glPushMatrix();                 //Lưu lại ma trận hiện hành
glTranslated(0, 1.15, 0);        //Tịnh tiến 1.15 theo trục y
glutSolidCube(.77);             //Vẽ đầu
glPopMatrix();                  //Phục hồi ma trận hiện hành
```



(a) (b) (c)

Hình 1. Hình vẽ thân mình, cổ, đầu robot ở chế độ đường nét (wire): (a) nhìn từ trục X; (b) nhìn từ trục Z; (c) nhìn nghiêng từ trên xuống

2.2. Tạo các hoạt cảnh hoạt động của Robot

2.2.1. Phương pháp thực hiện

Trước hết, để robot quay đầu, nghiêng người, hoạt động tay, chân thì cần tạo ra các "khớp quay" cho nó. Ví dụ muốn quay đầu thì cần tạo "khớp cổ", quay cả cánh tay thì phải có "khớp bả vai", quay cẳng tay thì phải có "khớp khuỷu tay", nâng hạ chân thì phải có "khớp háng", "khớp gối"… Như vậy, tại các vị trí này, ta cần tạo các biến góc quay theo trục X, theo trục Y và theo trục Z, đồng thời sử dụng hàm biến đổi glRotatef(GLfloat angle, GLfloat x, GLfloat y, GLfloat z); //Quay 1 góc angle (angle > 0 thì quay ngược chiều kim đồng hồ) quanh trục nối gốc tọa độ với điểm (x, y, z).

Tiếp đến, để thực hiện di chuyển tiến lùi, lên xuống, trái phải thì ta cần tạo các biến dịch chuyển theo trục X, trục Y và trục Z, đồng thời sử dụng hàm biến đổi tịnh tiến glTranslatef(GLfloat x, GLfloat y, GLfloat z).

2.2.2. Minh họa cách thực hiện

Để minh họa cách thực hiện tạo "khớp thân mình" và "khớp cổ" cho Robot 3D, ta lấy ví dụ khi vẽ thân mình, cổ và đầu Robot. Cụ thể như sau:

```
void robot_body(){
        glPushMatrix();                 /*Lưu lại ma trận biến đổi hiện hành 0*/

        //Phép biến đổi để thực hiện góc xoay toàn bộ thân, đầu, cổ robot
        glTranslated(0, -.625, 0);      //Tịnh tiến gốc tọa độ đến thân dưới của robot
        glRotated(RbodyX, 1, 0, 0);     //Xoay quanh trục X
        glRotated(RbodyY, 0, 1, 0);     //Xoay quanh trục Y
        glRotated(RbodyZ, 0, 0, 1);     //Xoay quanh trục Z
        glTranslated(0, +.625, 0);      //Tịnh tiến trở về gốc tọa độ cũ

        //Thân mình
        glPushMatrix();                 //Lưu lại ma trận hiện hành
        glScaled(1., 1., .75);          //Thay đổi tỷ lệ cube: 1*x, 1*y, 0.75*z
        glutSolidCube(1.25);            //Vẽ thân mình
        glPopMatrix();                  //Phục hồi ma trận hiện hành

        /* Đầu, cổ*/
        glPushMatrix();             //Lưu ma trận hiện hành 1
        //Phép biến đổi để thực hiện góc xoay khớp cổ
        glTranslated(0, .65, 0);        //Tịnh tiến gốc tọa độ đến tâm cổ của robot
        glRotated(RNeckX, 1, 0, 0);     //Xoay quanh trục X
        glRotated(RNeckY, 0, 1, 0);     //Xoay quanh trục Y
        glRotated(RNeckZ, 0, 0, 1);     //Xoay quanh trục Z
        glTranslated(0, -.65, 0);       //Tịnh tiến trở về gốc tọa độ cũ
        //Cổ
        glPushMatrix();                 //Lưu lại ma trận hiện hành
        glTranslated(0, .65, 0);        //Tịnh tiến 0.65 theo trục y
        glutSolidSphere(.36, 16, 16);   //Vẽ cổ
        glPopMatrix();                  //Phục hồi ma trận hiện hành
        //Đầu
        glPushMatrix();                 //Lưu lại ma trận hiện hành
        glTranslated(0, 1.15, 0);       //Tịnh tiến 1.15 theo trục y
        glutSolidCube(.77);             //Vẽ đầu
        glPopMatrix();                  //Phục hồi ma trận hiện hành
        glPopMatrix();              //Phục hồi ma trận hiện hành 1
        glPopMatrix();          /*Phục hồi ma trận hiện hành 0*/
}
```

Giả sử Robot đang đứng tại gốc tọa độ (biến bodyX=0, bodyY=0, bodyZ=0), hướng nhìn của Robot dọc theo trục Z. Khi muốn cho Robot tiến về trước và nghiêng người 1 góc 10^o (quanh trục X), đầu vẫn nhìn thẳng về phía trước thì ta làm như sau:

```
        void display() {
```

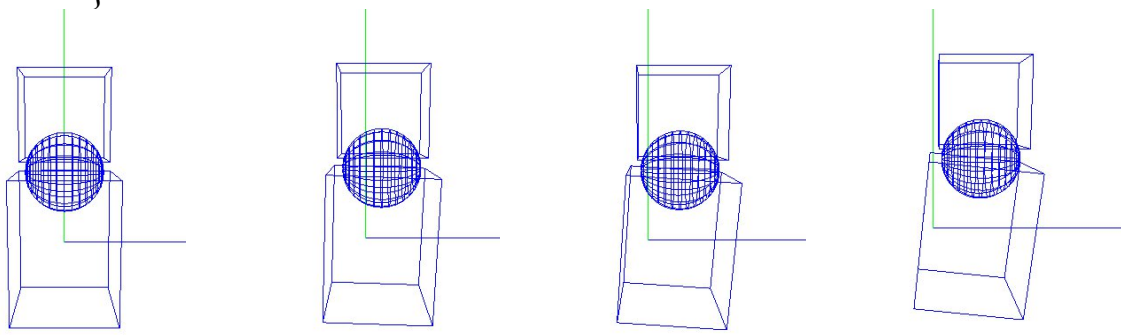
```
//Xóa mọi pixel (color buffer và depth buffer)
glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);

//Khởi tạo ma trận biến đổi hiện hành
glMatrixMode(GL_MODELVIEW);
glLoadIdentity();

//Robot tiến về trước và nghiêng 10º quanh trục X
glPushMatrix();     //Lưu lại ma trận hiện hành
glTranslated(bodyX, bodyY, bodyZ); //Dịch chuyển theo 3 trục x, y, z
robot_body();       //Vẽ thân, cổ, đầu của robot
glPopMatrix();      //Phục hồi lại ma trận hiện hành cũ

bodyZ = bodyZ + 0.1;        //Robot di chuyển về phía trước theo trục Z
//Khi góc quay <=10o thì RbodyX cộng 2º và RNeckX trừ 2º
if (RbodyX <=10) {
        RbodyX = RbodyX + 2;
        RNeckX = RNeckX - 2;
}

glutSwapBuffers();  //Thực hiện việc hoán đổi 2 buffer
glutPostRedisplay();//Bắt vẽ lại
}
```



Frame 0	Frame 1	Frame 2	Frame 3
Ban đầu	Nghiêng 2º Di chuyển 0.1	Nghiêng 4º Di chuyển 0.2	Nghiêng 6º Di chuyển 0.3

Hình 2. Quá trình chuyển động của Robot

2.2.3. Hoạt cảnh Robot Chào cờ

```
//Hàm hoạt động của robot
void robot_animation() {
   switch (option) {
   case 4: //Robot chào
      if (animation != option) robot_reset();
      else {
         if (RLeftArmTopZ == -135) leftArmStart = 1;
         if (RLeftArmTopZ == 0) leftArmStart = 0;
```

```
        if (leftArmStart == 0) {
           RLeftArmTopZ = RLeftArmTopZ - 1;          //góc quay ở bả vai quanh trục Z
           RLeftArmBottomZ = RLeftArmBottomZ - 0.7;  //góc quay ở khuỷu tay theo trục Z
           RNeckX = RNeckX - 0.05;
        }
      }
      break;
   }
}
//Hàm trở về trạng thái ban đầu khi chuyển option khác
void robot_reset() {
   switch (animation) {
   case 4: //Trả về trạng thái đứng yên khi Robot chào
      if (RLeftArmTopZ != 0)
      {
         RLeftArmTopZ = RLeftArmTopZ + 1.0;
         RLeftArmBottomZ = RLeftArmBottomZ + 0.7;
         RNeckX = RNeckX + 0.05;
      }
      else animation = option;
      break;

   default:
      animation = option;
   }
}
```

2.2.4. Hoạt cảnh Robot Chạy tiến

```
//Hàm hoạt động của robot
void robot_animation() {
   switch (option) {
   case 1:
     //nếu chuyển sang hoạt động khác thì reset hoạt động cũ của robot rồi mới bắt đầu hđ mới
      if (animation != option) robot_reset();
      else
      {
        if (RLeftArmTopX == -90) leftArmStart = 1; //khi tay nâng lên 90o thì: check = bắt đầu
hạ xuống
         if (RLeftArmTopX == 0) leftArmStart = 0; //khi tay ở 0o thì: check = bắt đầu nâng lên

         //Khi tay trái duỗi thẳng thì bắt đầu nâng tay trái, chân phải lên
         if (leftArmStart == 0)
         {
            RLeftArmTopX = RLeftArmTopX - 2;          //góc quay ở bả vai
            RLeftArmBottomX = RLeftArmBottomX - 1.5;  //góc quay ở khuỷu tay theo trục x
            RLeftArmBottomZ = RLeftArmBottomZ + 1.5;  //góc quay ở khuỷu tay theo trục z

            RRightLegTopX = RRightLegTopX - 1.75;      //góc quay ở khớp háng
            RRightLegBottomX = RRightLegBottomX + 2.5; //góc quay ở đầu gối
```

```
        }
        //Khi tay trái quay đến 90o thì bắt đầu hạ tay trái, chân phải xuống
        if (leftArmStart == 1) {
           RLeftArmTopX = RLeftArmTopX + 2;            //góc quay ở bả vai
           RLeftArmBottomX = RLeftArmBottomX + 1.5;  //góc quay ở khuỷu tay theo trục x
           RLeftArmBottomZ = RLeftArmBottomZ - 1.5;    //góc quay ở khuỷu tay theo trục z

           RRightLegTopX = RRightLegTopX + 1.75;       //góc quay ở khớp háng
           RRightLegBottomX = RRightLegBottomX - 2.5; //góc quay ở đầu gối

        }
        i = i + 1; //mỗi lần nâng góc 2o thì i+1
        if (i >= 45) i = 45; //cộng đến khi góc nâng 90o thì i=45
        //Khi i=45 tức là tay trái nâng lên đến 90o và bắt đầu hạ xuống thì mới nâng tay phải
        if (i == 45) {
           if (RRightArmTopX == -90) rightArmStart = 1;
           if (RRightArmTopX == 0) rightArmStart = 0;

           if (rightArmStart == 1) {
              RRightArmTopX = RRightArmTopX + 2;
              RRightArmBottomX = RRightArmBottomX + 1.5;
              RRightArmBottomZ = RRightArmBottomZ + 1.5;

              RLeftLegTopX = RLeftLegTopX + 1.75;
              RLeftLegBottomX = RLeftLegBottomX - 2.5;
           }
           if (rightArmStart == 0) {
              RRightArmTopX = RRightArmTopX - 2;
              RRightArmBottomX = RRightArmBottomX - 1.5;
              RRightArmBottomZ = RRightArmBottomZ - 1.5;

              RLeftLegTopX = RLeftLegTopX - 1.75;
              RLeftLegBottomX = RLeftLegBottomX + 2.5;
           }
        }
        //Di chuyển về phía trước
        if (bodyZ < 10) {
           bodyZ += 0.05 / speed;
        }
        //Người hơi nghiêng về trước
        RbodyX = 5;
        RNeckX = -5;
     }
     break;
  }
}
//Hàm trở về trạng thái ban đầu khi chuyển option khác
void robot_reset() {
  switch (animation) {
  case 1: //Trả về trạng thái đứng yên khi robot chạy tiến
     if (RLeftArmTopX != 0) {
```

```
            RLeftArmTopX = RLeftArmTopX + 2;            //góc quay ở bả vai
            RLeftArmBottomX = RLeftArmBottomX + 1.5;    //góc quay ở khuỷu tay theo trục x
            RLeftArmBottomZ = RLeftArmBottomZ - 1.5;    //góc quay ở khuỷu tay theo trục z

            RRightLegTopX = RRightLegTopX + 1.75;       //góc quay ở khớp háng
            RRightLegBottomX = RRightLegBottomX - 2.5; //góc quay ở đầu gối
            i = 0;
            RbodyX = 0;
            RNeckX = 0;
        }
        else if (RRightArmTopX != 0) {
            RRightArmTopX = RRightArmTopX + 2;
            RRightArmBottomX = RRightArmBottomX + 1.5;
            RRightArmBottomZ = RRightArmBottomZ + 1.5;

            RLeftLegTopX = RLeftLegTopX + 1.75;
            RLeftLegBottomX = RLeftLegBottomX - 2.5;
            RbodyX = 0;
            RNeckX = 0;
        }
        else animation = option;
        break;

    default:
        animation = option;
    }
}
```

2.2.5. Hoạt cảnh Robot Chạy lùi

Tương tự như hoạt cảnh robot chạy tiến, nhưng thay '+' bằng '-' biến dịch chuyển bodyZ (giảm bodyZ).

2.2.6. Hoạt cảnh Robot Nhảy múa

```
//Hàm hoạt động của robot
void robot_animation() {
    switch (option) {
    case 6: //Trả về trạng thái đứng yên khi robot nhún nhảy
        if (animation != option) robot_reset();
            else {
                if (RLeftLegTopX == 0) LegStart = 0;
                if (RLeftLegTopX == -60) LegStart = 1;

                if (LegStart == 0)
                {
                    RLeftLegTopX = RLeftLegTopX - 1;
                    RRightLegTopX = RRightLegTopX - 1.5;
                    RRightLegTopZ = RRightLegTopZ + .2;
```

```
                RLeftLegBottomX = RLeftLegBottomX + 1.5;
                RRightLegBottomX = RRightLegBottomX + 2;

                RLeftArmTopX = RLeftArmTopX - 2;
                RLeftArmTopZ = RLeftArmTopZ - 1;
                RLeftArmBottomX = RLeftArmBottomX - 2;

                RRightArmTopX = RRightArmTopX - 1.5;
                RRightArmBottomZ = RRightArmBottomZ - 2;

                bodyY = bodyY - 0.003;
            }
            if (LegStart == 1)
            {
                RLeftLegTopX = RLeftLegTopX + 1;
                RRightLegTopX = RRightLegTopX + 1.5;
                RRightLegTopZ = RRightLegTopZ - .2;

                RLeftLegBottomX = RLeftLegBottomX - 1.5;
                RRightLegBottomX = RRightLegBottomX - 2;

                RLeftArmTopX = RLeftArmTopX + 2;
                RLeftArmTopZ = RLeftArmTopZ + 1;
                RLeftArmBottomX = RLeftArmBottomX + 2;

                RRightArmTopX = RRightArmTopX + 1.5;
                RRightArmBottomZ = RRightArmBottomZ + 2;

                bodyY = bodyY + 0.003;
            }
          }
          break;
        }
}
//Hàm trở về trạng thái ban đầu khi chuyển option khác
void robot_reset() {
   switch (animation) {
    case 6: //Trả về trạng thái đứng yên khi robot nhún nhảy
       if (RRightLegTopX != 0)
          {
            RLeftLegTopX = RLeftLegTopX + 1;
            RRightLegTopX = RRightLegTopX + 1.5;
            RRightLegTopZ = RRightLegTopZ - .2;

            RLeftLegBottomX = RLeftLegBottomX - 1.5;
            RRightLegBottomX = RRightLegBottomX - 2;

            RLeftArmTopX = RLeftArmTopX + 2;
            RLeftArmTopZ = RLeftArmTopZ + 1;
            RLeftArmBottomX = RLeftArmBottomX + 2;
```

```
            RRightArmTopX = RRightArmTopX + 1.5;
            RRightArmBottomZ = RRightArmBottomZ + 2;

            bodyY = bodyY + 0.003;
         }
         else animation = option;
         break;

      default:
         animation = option;
      }
}
```

2.2.7. Hoạt cảnh Robot Nhảy cao xoay vòng

```
//Hàm hoạt động của robot
void robot_animation() {
   switch (option) {
    case 7: //Robot nhảy cao xoay vòng
       if (animation != option) robot_reset();
          else {
             if (RLeftLegTopX == 0) LegStart = 0;
             if (RLeftLegTopX == -60) LegStart = 1;

             if (LegStart == 0)
             {
                if (check_jump == -1) {
                   RLeftLegTopX = RLeftLegTopX - 1;
                   RRightLegTopX = RRightLegTopX - 1;
                   RLeftLegBottomX = RLeftLegBottomX + 1.5;
                   RRightLegBottomX = RRightLegBottomX + 1.5;

                   RLeftArmTopX = RLeftArmTopX - 2.5;
                   RLeftArmTopZ = RLeftArmTopZ - .5;

                   RLeftArmBottomX = RLeftArmBottomX + 1.75;
                   RLeftArmBottomZ = RLeftArmBottomZ + 1.75;

                   RRightArmTopX = RRightArmTopX - 2.5;
                   RRightArmTopZ = RRightArmTopZ + .5;

                   RRightArmBottomX = RRightArmBottomX - 1.75;
                   RRightArmBottomZ = RRightArmBottomZ - 1.75;


                   bodyY = bodyY - 0.003;
                }
                if (check_jump == 1) {
                   bodyY = bodyY + 0.02;
                   //bodyZ = bodyZ + 0.02;
                   alpha += 3;
```

```
                if (bodyY > 3.6) check_jump = 2;
            }
            if (check_jump == 2) {
                bodyY = bodyY - 0.01;
                //bodyZ = bodyZ + 0.02;
                if (alpha >= 360) alpha = 0;
                else alpha += 3;
                if (bodyY <= 0) check_jump = -1;
            }
        }
        if (LegStart == 1)
        {
            if (bodyY <= 0) {
                RLeftLegTopX = RLeftLegTopX + 1;
                RRightLegTopX = RRightLegTopX + 1;
                RLeftLegBottomX = RLeftLegBottomX - 1.5;
                RRightLegBottomX = RRightLegBottomX - 1.5;

                RLeftArmTopX = RLeftArmTopX + 2.5;
                RLeftArmTopZ = RLeftArmTopZ + .5;

                RLeftArmBottomX = RLeftArmBottomX - 1.75;
                RLeftArmBottomZ = RLeftArmBottomZ - 1.75;

                RRightArmTopX = RRightArmTopX + 2.5;
                RRightArmTopZ = RRightArmTopZ - .5;

                RRightArmBottomX = RRightArmBottomX + 1.75;
                RRightArmBottomZ = RRightArmBottomZ + 1.75;

                bodyY = bodyY + 0.003;
                check_jump = 1;
            }
        }
      }
      break;
    }
}
//Hàm trở về trạng thái ban đầu khi chuyển option khác
void robot_reset() {
   switch (animation) {
    case 7: //Trả về trạng thái đứng yên khi robot nhảy cao
      if (bodyY > 0)
        {
          if (check_jump == 1) {
            bodyY = bodyY + 0.02;
            //bodyZ = bodyZ + 0.02;
            alpha += 3;
            if (bodyY > 3.6) check_jump = 2;
          }
          if (check_jump == 2) {
```

```
                bodyY = bodyY - 0.01;
                //bodyZ = bodyZ + 0.02;
                if (alpha >= 360) alpha = 0;
                else alpha += 3;
                if (bodyY <= 0) check_jump = -1;
            }
        }
        else animation = option;
        break;

    default:
        animation = option;
    }
}
```

2.2.8. Hoạt cảnh Robot Nhảy xa

```
//Hàm hoạt động của robot
void robot_animation() {
    switch (option) {
    if (animation != option) robot_reset();
        else {
            if (RLeftLegTopX == 0) LegStart = 0;
            if (RLeftLegTopX == -60) LegStart = 1;

            if (LegStart == 0)
            {
                if (check_jump == -1) {
                    RLeftLegTopX = RLeftLegTopX - 1;
                    RRightLegTopX = RRightLegTopX - 1;
                    RLeftLegBottomX = RLeftLegBottomX + 1.5;
                    RRightLegBottomX = RRightLegBottomX + 1.5;

                    RLeftArmTopX = RLeftArmTopX - 1.5;
                    RRightArmTopX = RRightArmTopX - 1.5;

                    bodyY = bodyY - 0.003;
                }
                if (check_jump == 1) {
                    bodyY = bodyY + 0.01;
                    bodyZ = bodyZ + 0.02;
                    if (bodyY > 2) check_jump = 2;
                }
                if (check_jump == 2) {
                    bodyY = bodyY - 0.01;
                    bodyZ = bodyZ + 0.02;
                    if (bodyY <= 0) check_jump = 3;
                }
                if (check_jump == 3 && i==0) {
                    RLeftLegTopX = RLeftLegTopX - 1;
                    RRightLegTopX = RRightLegTopX - 1;
```

```
                RLeftLegBottomX = RLeftLegBottomX + 1.5;
                RRightLegBottomX = RRightLegBottomX + 1.5;

                bodyY = bodyY - 0.003;
            }
        }
        if (LegStart == 1)
        {
            if (bodyY <= 0) {
                if (check_jump != 3) {
                    RLeftLegTopX = RLeftLegTopX + 1;
                    RRightLegTopX = RRightLegTopX + 1;
                    RLeftLegBottomX = RLeftLegBottomX - 1.5;
                    RRightLegBottomX = RRightLegBottomX - 1.5;

                    RLeftArmTopX = RLeftArmTopX + 1.5;
                    RRightArmTopX = RRightArmTopX + 1.5;

                    bodyY = bodyY + 0.003;
                    check_jump = 1;
                }
                if (check_jump == 3) {
                    RLeftLegTopX = RLeftLegTopX + 3;
                    RRightLegTopX = RRightLegTopX + 3;
                    RLeftLegBottomX = RLeftLegBottomX - 4.5;
                    RRightLegBottomX = RRightLegBottomX - 4.5;

                    bodyY = bodyY + 0.003;
                    i = 1;
                    if (bodyY >= 0) check_jump = 0;
                }
            }
        }
        break;
    }
}
//Hàm trở về trạng thái ban đầu khi chuyển option khác
void robot_reset() {
    switch (animation) {
    if (bodyY > 0)
        {
            bodyY = bodyY - 0.01;
            bodyZ = bodyZ + 0.02;
        }
        else if (RLeftLegTopX != 0) {
            RLeftLegTopX = RLeftLegTopX + 1;
            RRightLegTopX = RRightLegTopX + 1;
            RLeftLegBottomX = RLeftLegBottomX - 1.5;
            RRightLegBottomX = RRightLegBottomX - 1.5;
```

```
            bodyY = bodyY + 0.003;
            i = 0;
            check_jump = 0;
        }
        else animation = option;
        break;

    default:
        animation = option;
    }
}
```

2.3. Thiết lập tô bóng và chiếu sáng

- Thiết lập tô màu và chiếu sáng được thực hiện như sau:

```
//Khai báo các véc tơ màu
GLfloat robot_color[4] = { .6, .7, .9, 1.0 };   //blue grey
GLfloat black_color[4] = { .0, .0, .0, 1. };    //black
GLfloat yellow_color[4] = { 1. ,1. ,.0 , 1. };  //yellow
GLfloat red_color[4] = { 1., .0, .0, 1. };      //red
GLfloat robot_shoes[4] = { .2, .2, 1., 1. };    //blue

//Hàm thực hiện các khởi tạo
void Init() {
    //Chọn màu xóa hình nền (phủ hình nền) bằng màu xanh dương
    glClearColor(.3, .5, .1, 0);

     /* Khai báo ánh sáng */
    //Bật chế độ chiếu sáng
    glEnable(GL_LIGHTING);
    glEnable(GL_LIGHT0);
    //Bật chức năng cho phép loại bỏ một phần của đối tượng bị che bởi đối tượng
khác
    glEnable(GL_DEPTH_TEST);

    //Thiết lập vị trí ánh sáng chiếu
    GLfloat light_position[] = { -.8, 1.513, 0.6, 0. }; //Véc tơ vị trí ánh sáng chiếu
    glLightfv(GL_LIGHT0, GL_POSITION, light_position);

    //Thiết lập chiếu sáng toàn phần cho đối tượng
    GLfloat light_ambient[] = { 1., 0., 0. }; //Véc tơ màu (red)
    glLightfv(GL_LIGHT0, GL_AMBIENT, light_ambient);

    //Tạo ánh sáng phản xạ trong đối tượng
    GLfloat specular[] = { 0.7, 0.7, 0.7, 1. };
    glMaterialfv(GL_FRONT_AND_BACK, GL_SPECULAR, specular);
```

```
    //Điều chỉnh cường độ điểm chiếu sáng phản xạ
    GLfloat shininess = 30.0f;
    glMateriali(GL_FRONT, GL_SHININESS, shininess);

    // Thiết lập chế độ shading (phủ màu) là smooth
    glShadeModel(GL_SMOOTH);
}
```

- Trước khi vẽ Robot, cần khai báo màu vẽ:

glMaterialfv(GL_FRONT_AND_BACK, GL_DIFFUSE, véc tơ màu);

với véc tơ màu robot_color, red_color,... đã khai báo ở trên.

2.4. Thay đổi góc nhìn, thiết lập tốc độ hoạt động của Robot, tạo điều khiển

- Thay đổi góc nhìn:

//Khai báo biến định vị camera

float gx = 4, gy = 3, gz = 11;

Trong hàm void display(), thiết lập view bằng hàm:

gluLookAt(gx, gy, gz, 0, 0, 0, 0, 1, 0);

Bằng việc thay đổi gx, gy, gz ta sẽ có những góc nhìn khác nhau.

- Thiết lập tốc độ hoạt động của Robot bằng cách khai báo biến speed. Trong hàm void display(), thiết lập tốc độ hoạt động bằng hàm Sleep(speed). Như vậy, khi speed càng tăng thì khoảng thời gian đổi 2 buffer càng chậm, tức là robot hoạt động càng chậm.
- Tạo điều khiển bằng bàn phím:

  Tạo hàm void processKeys(unsigned char key, int x, int y) như dưới đây; sau đó ở hàm main, gọi hàm glutKeyboardFunc(processKeys).

```
//Hàm điều khiển hoạt động bằng việc nhập từ bàn phím
void processKeys(unsigned char key, int x, int y)
{
   switch (key)
   {
   case 'f': //Chạy tiến
      option = 1;
      break;
   case 'b': //Chạy lùi
      option = 2;
      break;
   case 'h': //Chào
      option = 4;
      break;
   case 'j': //Nhảy xa
```

```
            option = 5;
            check_jump = -1;
            break;
        case 'd': //Nhảy múa
            option = 6;
            break;
        case 'k': //Nhảy cao
            option = 7;
            check_jump = -1;
            break;
        case 's': //Stop
            option = 3;
            break;
        case 'p': //Tăng tốc
            if (speed >= 3) speed -= 1;
            break;
        case 'o': //Giảm tốc
            if (speed <= 15) speed += 1;
            break;
        case 'v': //Thay đổi góc view
            if (gx == 4 && gz == 11) { gx = 0; gy = 3; gz = 15; }
            else if (gx == 0) { gx = -15; gy = 1; gz = 0; }
            else if (gz == 0) { gx = 3; gy = 15; gz = 3; }
            else { gx = 4; gy = 3; gz = 11; };
            break;
        case 'q': //Thoát chương trình bằng phím q
            exit(0);
            break;
        case 27: ////Thoát chương trình bằng phím Esc
            exit(0);
            break;
        default:
            break;
        }
    }
```

- Tạo điều khiển bằng chuột:

  Tạo hàm void menu(int id) như dưới đây;

```
    void menu(int id)
    {
        option = id;
        if (id == 5 || id == 7) check_jump = -1;
        else check_jump = 0;
}
```

  Sau đó, ở hàm main tạo menu như sau:

```
void main(){
…….
        glutCreateMenu(menu);
        glutAddMenuEntry(" Chao ", 4);
        glutAddMenuEntry(" Nhay mua ", 6);
        glutAddMenuEntry(" Chay tien ", 1);
        glutAddMenuEntry(" Chay lui ", 2);
        glutAddMenuEntry(" Nhay cao ", 7);
        glutAddMenuEntry(" Nhay xa ", 5);
        glutAddMenuEntry(" Dung lai ", 3);
        glutAttachMenu(GLUT_RIGHT_BUTTON);
……..
}
```

III. KẾT QUẢ THỬ NGHIỆM

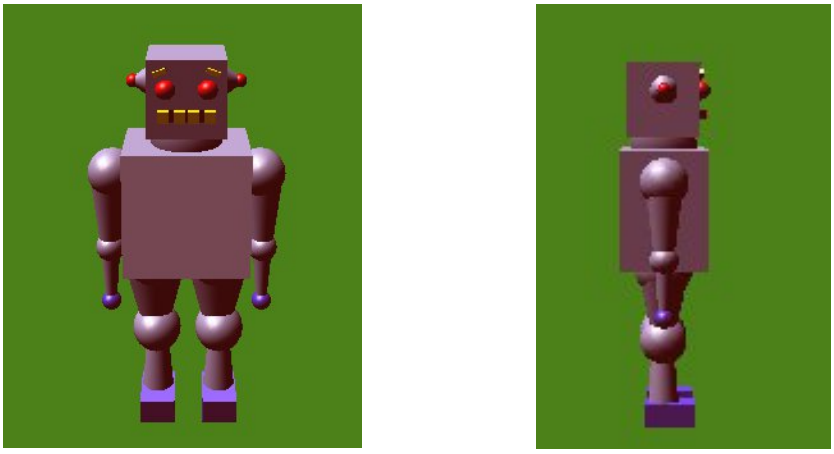
3.1. Môi trường cài đặt

Phần cứng:

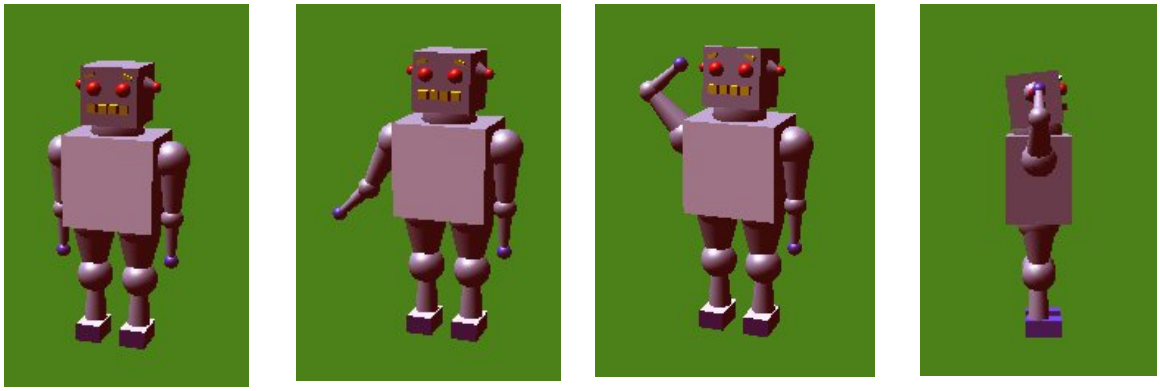
Phàn mềm và các thư viện:

3.2. Demo kết quả

- Kết quả vẽ robot 3D được thể hiện ở hình 3;
- Kết quả hoạt cảnh robot Chào cờ được thể hiện ở hình 4;
- Kết quả hoạt cảnh robot Chạy tiến, Chạy lùi được thể hiện ở hình 5;
- Kết quả hoạt cảnh robot Nhảy múa được thể hiện ở hình 6;
- Kết quả hoạt cảnh robot Nhảy cao xoay vòng được thể hiện ở hình 7;
- Kết quả hoạt cảnh robot Nhảy xa được thể hiện ở hình 8.

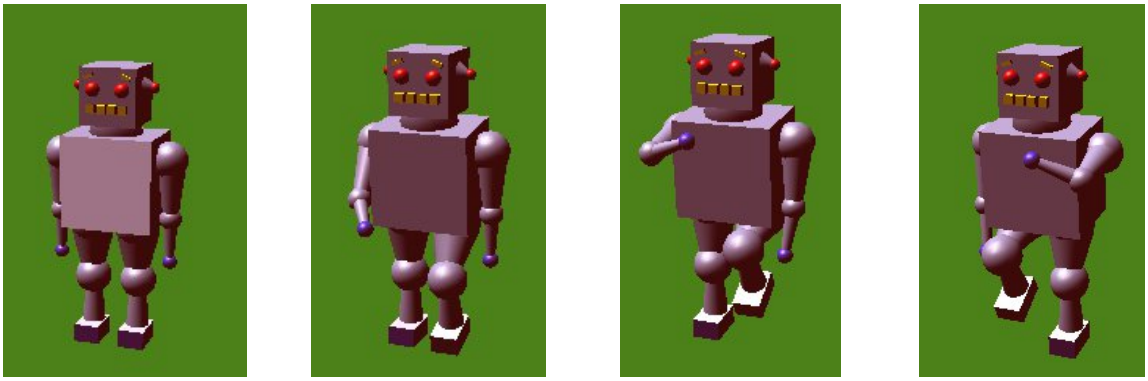


Hình 3. Kết quả vẽ mô hình 3D Robot (sau khi được tô màu và chiếu sáng)

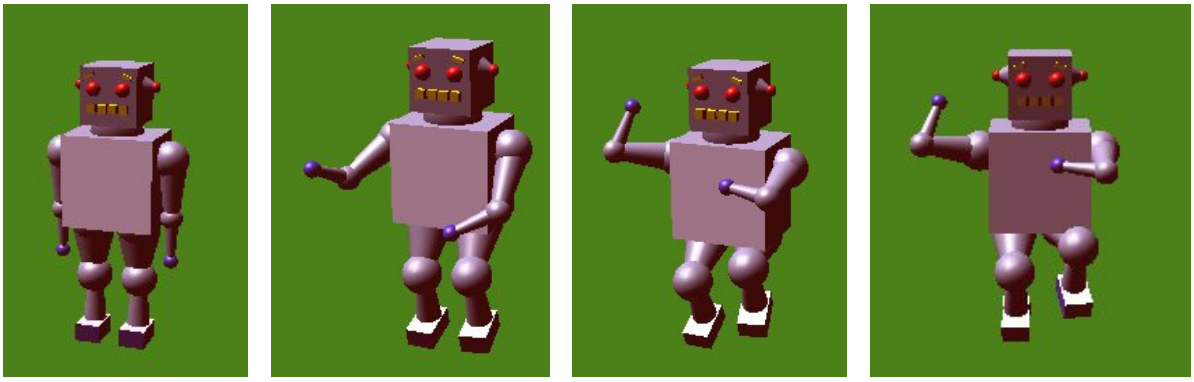


Frame 0 | Frame 45 | Frame 135 | Frame 135 Nhìn theo trục X

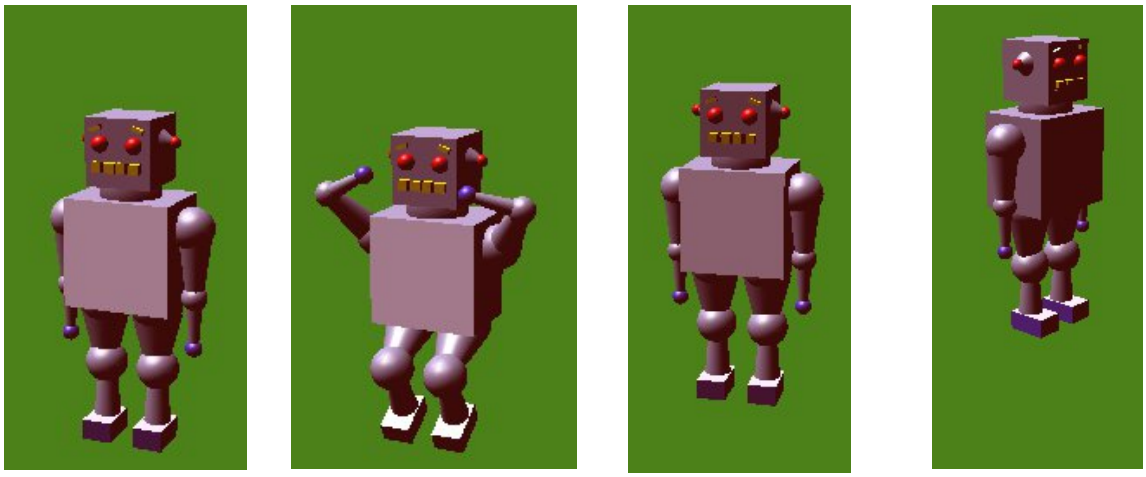
Hình 4. Hoạt cảnh Robot Chào cờ (sau khi được tô màu và chiếu sáng)



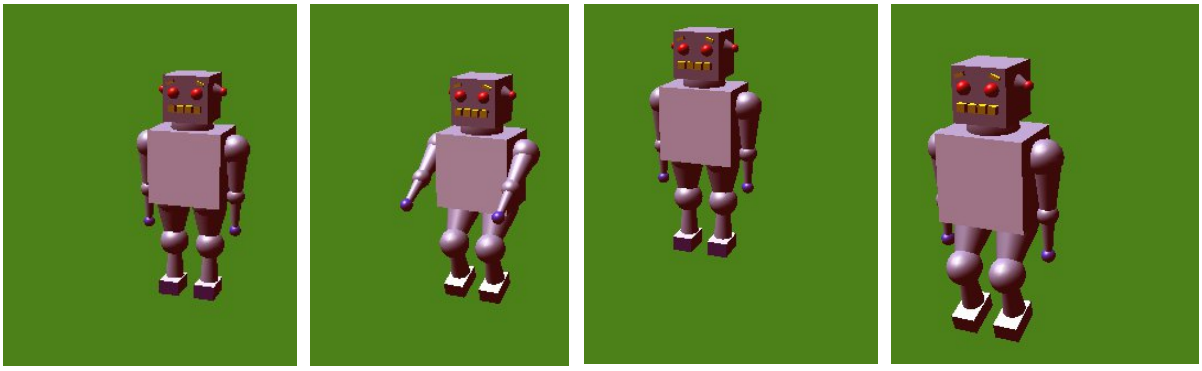
Hình 5. Hoạt cảnh Robot Chạy tiến, Chạy lùi (sau khi được tô màu và chiếu sáng)



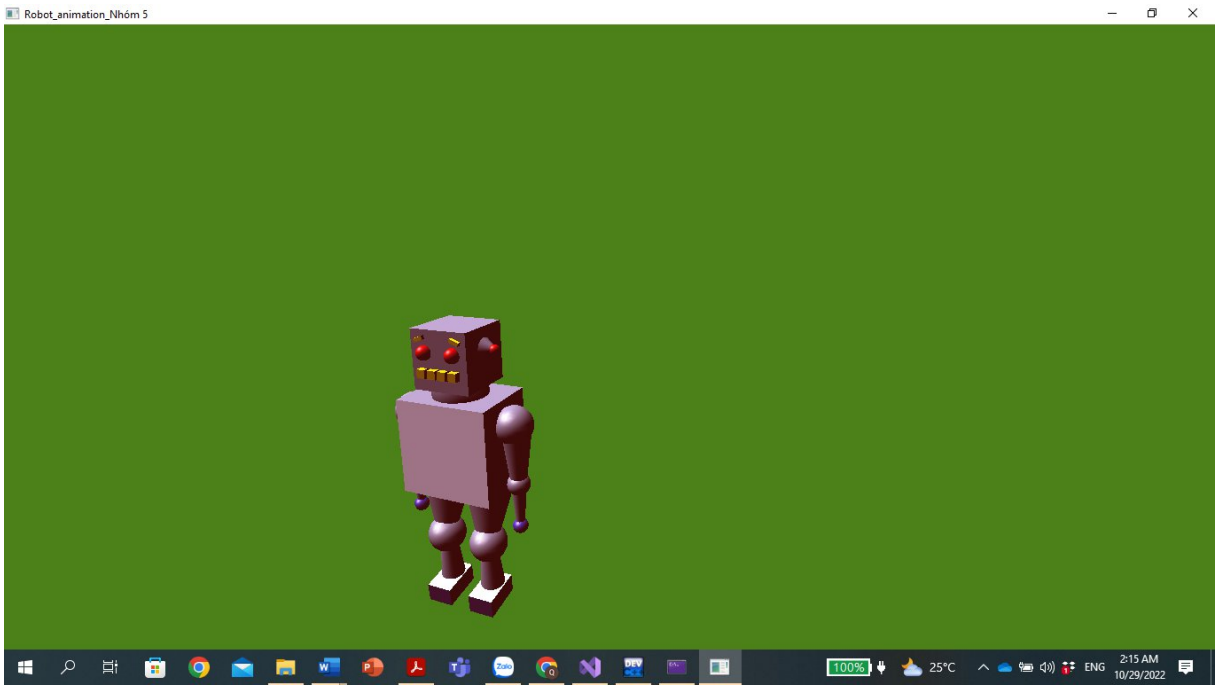
Hình 6. Hoạt cảnh Robot Nhảy múa (sau khi được tô màu và chiếu sáng)



Hình 7. Hoạt cảnh Robot Nhảy cao xoay vòng (sau khi được tô màu và chiếu sáng)



Hình 8. Hoạt cảnh Robot Nhảy xa (sau khi được tô màu và chiếu sáng)


Robot_animation_Nhóm 5
100%
25°C
ENG
2:15 AM
10/29/2022

IV. KẾT LUẬN

Qua quá trình nghiên cứu và thực hiện, về cơ bản bài tập lớn đã hoàn thành và đã đạt được những kết quả sau:

- Tổng kết được phương pháp vẽ, từ đó đã xây dựng được mô hình Robot 3D bằng ngôn ngữ C++ với các đối tượng và các phép biến đổi cơ bản trong thư viện đồ họa OPENGL và FREEGLUT.
- Tổng kết được phương pháp tạo hoạt động cho các đối tượng, từ đó đã thực hiện được các hoạt cảnh hoạt động của Robot 3D, bao gồm:
    - Chào cờ;
    - Nhảy múa;
    - Chạy tiến;
    - Chạy lùi;
    - Nhảy cao xoay vòng;
    - Nhảy xa;
    - Dừng lại trở về trạng thái đứng yên.
- Thiết lập tô màu và chiếu sáng cho Robot;
- Thay đổi góc nhìn, tăng giảm tốc độ hoạt động của Robot, tạo điều khiển bằng bàn phím và menu điều khiển bằng chuột.

Hạn chế:…..

Hướng phát triển:….

TÀI LIỆU THAM KHẢO

[1] Ngô Trường Giang, *"Bài giảng Đồ họa máy tính"*.

[2] Lê Phong, "Hướng dẫn lập trình OpenGL căn bản"

[3] Microsoft, https://learn.microsoft.com/en-us/windows/win32/opengl/opengl

[4] OpenGL Programming Guide, http://glprogramming.com/red/