#include<windows.h>

#include<MMSystem.h>

#include<GL/glut.h>

#include<GL/gl.h>

#include<GL/glu.h>

#include<math.h>

#include<iostream>

static double time5 = 0; // military\_time 변수

GLfloat R\_Arm\_x = 0; // 오른쪽 어깨

GLfloat R\_Arm\_y = 0; // 오른쪽 팔

GLfloat L\_Arm\_x = 0; // 왼쪽 어깨

GLfloat L\_Arm\_y = 0; // 왼쪽 팔

GLfloat R\_Leg\_x = 0; // 오른 허벅지

GLfloat R\_Leg\_y = 0; // 오른 종아리

GLfloat L\_Leg\_x = 0; // 왼 허벅지

GLfloat L\_Leg\_y = 0; // 왼 종아리

GLfloat R = 0; // 왼쪽 로켓 펀치 움직임 변수

GLfloat R2 = 0;// 오른쪽 로켓 펀치 움직임 변수

GLUquadricObj\* cyl;// 실린더 객체 선언

int a = 0; // x축 기준(값이 1일 때 x축을 기준으로 회전)

int b = 0; // y축 기준(값이 1일 때 y축을 기준으로 회전)

int c = 0; // z축 기준(값이 1일 때 z축을 기준으로 회전)

static int flag = 0; // wireframe 모드와 solid rendering 모드 구분 변수

static int key = 7; // 동작 모드와 색상 변경 변수//키 값을 7로 줘버림 해병대 체조로

int Width, Height; //화면 크기 가로,세로 변수

#define MILITARY 7 //군대 변수를 MILITARY 7이라고 둔다

void glInit(void) { // 지엘 초기화 함수

glEnable(GL\_DEPTH\_TEST); // 깊이 테스팅 사용

glEnable(GL\_NORMALIZE); // 정규화

glEnable(GL\_SMOOTH); // 각 정점의 색상을 부드럽게 연결하여 칠해지도록 하므로, 각정점에 적용된 색상이 혼합되어 적용된다.

glEnable(GL\_LIGHTING); // 지엘 조명. 빛을 받는 각도에 따라 로봇 색상이 변화

// 빛을 받는 각도에 따라 로봇 색깔의 변화를 위한 설정

GLfloat ambientLight[] = { 0.3f,0.3f,0.3f,1.0f };

GLfloat diffuseLight[] = { 0.7f,0.7f,0.7f,1.0f };

GLfloat specular[] = { 1.0f,1.0f,1.0f,1.0f };

GLfloat specref[] = { 1.0f,1.0f,1.0f,1.0f };

GLfloat position[] = { 400.0,300.0,-700.0,1.0 };

glLightfv(GL\_LIGHT0, GL\_AMBIENT, ambientLight);

glLightfv(GL\_LIGHT0, GL\_DIFFUSE, diffuseLight);

glLightfv(GL\_LIGHT0, GL\_SPECULAR, specular);

glLightfv(GL\_LIGHT0, GL\_POSITION, position);

glEnable(GL\_LIGHT0);

glEnable(GL\_COLOR\_MATERIAL);

glColorMaterial(GL\_FRONT, GL\_AMBIENT\_AND\_DIFFUSE);

glMateriali(GL\_FRONT, GL\_SHININESS, 128);

glClearColor(1.0, 1.0, 1.0, 1.0); // 배경 흰색

glMatrixMode(GL\_PROJECTION); // 모드 설정

glLoadIdentity(); // CTM 값 초기화

glOrtho(-2.0, 2.0, -2.0, 2.0,0.5 , 5.0); // 가시 범위 설정

glMatrixMode(GL\_MODELVIEW);

glLoadIdentity();

}

void Draw\_Color(int i) { // 색상을 그리는 함수

glColor3f(0.0, 0.7, 0.7); //연두색 군대체조 애니메이션

}

/\*

로봇 왼팔을 그리는 함수

\*/

void DrawL\_Arm(int x, int a, int b, int c) { //로봇의 왼쪽 팔을 그리는 함수

glPushMatrix(); //메트릭스로 하나의 변환행렬 설정

Draw\_Color(key); // 색 설정 key: 동작모드와 색상 변경 변수

cyl = gluNewQuadric(); // 실린더 객체 생성, Quadric 객체 속성 설정 함수

//Quadrics는 공간상에 기하학적인 모양을 생성하는 것이다

glRotatef(x, a, b, c); // 파라미터 값만큼 회전

//DrawL\_Arm 함수 호출시 피라미터 값을 정해줄 것이며 그 값 만큼 회전 할 것이다

glRotatef(90.0, 1.0, 0.0, 0.0); // 실린더를 x축 기준으로 90도 회전(실린더 생성 시 각도 변경을 위해)

//실린더 생설 할때 각도 변경을 위해 90도 회전하여 설정

glTranslatef(0.25, 0.0, 0.0); // 왼쪽 어깨 시작점으로 이동한다

glRotatef(15.0, 0.0, 1.0, 0.0); // 왼쪽 어깨를 y축을 기준으로 15도 회전

gluCylinder(cyl, 0.05, 0.05, 0.2, 50, 1); // 실린더를 그림

//원통형을 그린다 실린더= 원통형

}

/\*

로봇 왼쪽 손을 그리는 함수

\*/

void DrawL\_Hand(int y, int a, int b, int c) { //로봇 왼쪽 손을 그리는 함수

glPushMatrix(); //매트릭스를 이용해 하나의 변환행렬 설정

Draw\_Color(key); // 색 설정

cyl = gluNewQuadric(); // 실린더 객체 생성

glTranslatef(0.0, 0.0, 0.22); // 왼손 시작점을 설정

glRotatef(y, a, b, c); // 파라미터 값만큼 회전

//DrawL\_Hand의 피라미터 값을 받아 그만큼 회전

gluCylinder(cyl, 0.05, 0.05, 0.2, 15, 1); // 실린더를 그림

//연아 모드에서는 깔때기 모양을 한 팔 아래쪽으로 갈수록 좁아짐 //그 외의 모드는 일정한 원통 모양

glPopMatrix();

glPopMatrix();

}

/\*

로봇 오른팔을 그리는 함수

\*/

void DrawR\_Arm(int x, int a, int b, int c) { //로봇 오른팔을 그리는 함수이다

glPushMatrix();

Draw\_Color(key); // 색 설정

cyl = gluNewQuadric(); // 실린더 객체 생성

glRotatef(x, a, b, c); // 파라미터 값만큼 회전

glRotatef(90.0, 1.0, 0.0, 0.0); // 실린더를 x축을 기준으로 90도 회전(실린더 생성 시 각도 변경을 위해)

//왼쪽 팔과 마찬가지로 오른팔도 x축(1.0)기준으로 90도 회전하여 초기 설정

glTranslatef(-0.25, 0.0, 0.0); //우측 어깨 시작점(-0.25)으로 이동

glRotatef(-15.0, 0.0, 1.0, 0.0); //우측 어깨 -15도 y축기준 회전

//왼쪽 어깨는 +15도 와 다르게 우측어깨는 +15도로 y축 기준(1.0) 회전

gluCylinder(cyl, 0.05, 0.05, 0.2, 50, 1);// 실린더 그림

//해당 오른팔 원통형 그리기

}

/\*

로봇 오른쪽 손을 그리는 함수

\*/

void DrawR\_Hand(int y, int a, int b, int c) { //로봇 오른쪽 손(y)부분 그리는 함수

glPushMatrix();

Draw\_Color(key); // 색 설정 //동작 모드에 따라 색상도 다름

cyl = gluNewQuadric();//실린더 객체 생성

glTranslatef(0.0, 0.0, 0.22);//오른쪽 손 시작점 , 외손 시작점과 동일한 z값 시작점

glRotatef(y, a, b, c);// 파라미터값 만큼 회전

gluCylinder(cyl, 0.05, 0.05, 0.2, 50, 1);// 실린더 그림

//연아 모드 제외하고는 그냥 평범한 원통형 실린더

glPopMatrix();

glPopMatrix();

}

/\*

로봇 몸체를 그리는 함수

\*/

void DrawBody(int x, int a, int b, int c) { //로봇 몸통 그리는 함수

glPushMatrix();

Draw\_Color(key);

cyl = gluNewQuadric();//실린더 객체 생성

glRotatef(90.0, 1.0, 0.0, 0.0);//실린더 90도 x축기준 회전 (실린더 생성시 각도 변경을 위하여)

//원통형을 x축 기준으로 90도 회전하여 설정

glRotatef(x, a, b, c);//파라미터값 만큼 회전

//Change\_Wire\_Or\_Solid(flag);

gluCylinder(cyl, 0.2, 0.2, 0.45, 50, 1);// 실린더 그림

glPopMatrix();

}

/\*

로봇 왼쪽 허벅지를 그리는 함수

\*/

void DrawL\_Legs(int x, int a, int b, int c) { //로봇 왼쪽 허벅지를 그리는 함수

glPushMatrix();

Draw\_Color(key); //해당 모드당 색깔이 다르며 그 변수 key를 설정

cyl = gluNewQuadric();//실린더 객체 생성

glRotatef(90.0, 1.0, 0.0, 0.0);//실린더 90도 x축기준 회전 (실린더 생성시 각도 변경을 위하여)

//초기 설정 x축으로 90도 회전하여 설정

glTranslatef(0.1, 0.0, 0.42);//왼쪽 허벅지 시작점

//왼쪾 허벅지의 시작점은 x축으로 0.1 만큼 이동 z축으로 0.42만큼 이동하여 초기 설정

glRotatef(x, a, b, c);//파라미터만큼 회전

//Change\_Wire\_Or\_Solid(flag);

gluCylinder(cyl, 0.05, 0.05, 0.15, 50, 1);// 실린더 그림

//객체, 밑부분 반지름 , 윗부분 반지름, 높이, 얇은 정도(slice), 스택

}

/\*

로봇 왼쪽 종아리를 그리는 함수

\*/

void DrawL\_foot(int y, int a, int b, int c) { //로봇 왼쪽 종아리 부분

glPushMatrix();

Draw\_Color(key);

cyl = gluNewQuadric();//실린더 객체 생성

glTranslatef(0.0, 0.0, 0.18);//왼쪽 종아리 시작점

glRotatef(y, a, b, c);//파라미터만큼 회전

//Change\_Wire\_Or\_Solid(flag);

gluCylinder(cyl, 0.05, 0.05, 0.2, 50, 1);// 실린더 그림

//연아 모드 제외하고는 일반적인 원통형 모형

glPopMatrix();

glPopMatrix();

}

/\*

로봇 오른쪽 허벅지를 그리는 함수

\*/

void DrawR\_Legs(int x, int a, int b, int c) {

glPushMatrix();

Draw\_Color(key);

cyl = gluNewQuadric();//실린더 객체 생성

glRotatef(90.0, 1.0, 0.0, 0.0);//실린더 90도 x축기준 회전 (실린더 생성시 각도 변경을 위하여)

glTranslatef(-0.1, 0.0, 0.42);//오른쪽 허벅지 시작점

glRotatef(x, a, b, c);//파라미터만큼 회전

//Change\_Wire\_Or\_Solid(flag);

gluCylinder(cyl, 0.05, 0.05, 0.15, 50, 1);// 실린더 그림

}

/\*

로봇 오른쪽 종아리를 그리는 함수

\*/

void DrawR\_foot(int y, int a, int b, int c) {

glPushMatrix();

Draw\_Color(key);

cyl = gluNewQuadric();//실린더 객체 생성

glTranslatef(0.0, 0.0, 0.18);//오른쪽 종아리 시작점

glRotatef(y, a, b, c);//파라미터만큼 회전

//Change\_Wire\_Or\_Solid(flag);

gluCylinder(cyl, 0.05, 0.05, 0.2, 15, 1);// 실린더 그림

glPopMatrix();

glPopMatrix();

}

void Drawneck() { //로봇 목부분 그리는 함수이다

glPushMatrix();

glColor3f(1.0, 1.0, 1.0);//색 지정(흰색)

cyl = gluNewQuadric();//실린더 객체 생성

glRotatef(90.0, 1.0, 0.0, 0.0);//실린더 90도 x축기준 회전 (실린더 생성시 각도 변경을 위하여)

glTranslatef(0.0, 0.0, -0.045); // 목 시작점

gluCylinder(cyl, 0.2, 0.2, 0.025, 100, 1);// 실린더 그림

glPopMatrix();

}

void DrawGround1() { //움직이지 않는 땅 표면을 그리는 함수

cyl = gluNewQuadric();

glPushMatrix();

glColor3f(1.0, 1.0, 0.0);//색 지정(흰색)

glTranslatef(0.0, -2.73, 0.0);//땅 시작점

glRotatef(45.0, 0, 1, 0);//45도만큼 y축 방향으로 회전

//Change\_Wire\_Or\_Solid(flag);

gluSphere(cyl, 2.5, 30, 90);

glPopMatrix();

}

void DrawHead() { //로봇 머리 그리는 함수이다

glTranslatef(0.0, 0.02, 0.0);//머리 시작점 , 머리시작점 정해 놓고 변한 행렬 설정하기

glPushMatrix();// 처음 저장 좌표 다시 저장

cyl = gluNewQuadric();

//Change\_Wire\_Or\_Solid(flag);

Draw\_Color(key);//색깔 설정

gluSphere(cyl, 0.20, 30, 10);//머리 그리기

/\*왼쪽 뿔 그리기\*/

glRotatef(90.0, 1.0, 0.0, 0.0);//실린더 90도 x축기준 회전 (실린더 생성시 각도 변경을 위하여)

glTranslatef(-0.16, 0.0, -0.22);//왼쪽 뿔 시작점

glRotatef(35.0, 0.0, 1.0, 0.0);//35도 y축 기준 회전

gluCylinder(cyl, 0.005, 0.008, 0.1, 3, 1);//실린더 그림

glPopMatrix(); // 처음 저장 좌표로 돌아 간다.

glPushMatrix();// 처음 저장 좌표 다시 저장

/\*오른쪽 뿔 그리기\*/

glRotatef(90.0, 1.0, 0.0, 0.0);//실린더 90도 x축기준 회전 (실린더 생성시 각도 변경을 위하여)

glTranslatef(0.16, 0.0, -0.22);//오른쪽 뿔 시작점, 왼쪽 뿔과 정 반재로 0.16으로 시작

glRotatef(-35.0, 0.0, 1.0, 0.0);//-35도 y축 기준 회전

gluCylinder(cyl, 0.005, 0.008, 0.1, 3, 1);//실린더 그림

glPopMatrix(); // 처음 저장 좌표로 돌아 간다.

glPushMatrix();// 처음 저장 좌표 다시 저장

/\*왼쪽 눈 그리기\*/

glTranslatef(-0.1, 0.1, 0.13);//왼쪽 눈 시작점

glColor3f(0.0, 0.0, 0.0);// 눈 컬러 설정(흑색)

gluSphere(cyl, 0.03, 10, 10);

glPopMatrix(); // 처음 저장 좌표로 돌아 간다.

glPushMatrix();// 처음 저장 좌표 다시 저장

/\*오른쪽 눈 그리기\*/

glTranslatef(0.1, 0.1, 0.13);//오른쪽 눈 시작점

gluSphere(cyl, 0.03, 10, 10);

glPopMatrix();// 처음 저장 좌표 다시 저장

}

void MilitaryExercise() { //로봇이 해병대 만세체조

/\*로봇 움직임 가속도는 sin, cos함수를 이용하였고 관전 제한을 abs()함수를 이용하였다\*/

L\_Arm\_x = -abs(sin(time5) \* 180); //왼쪽,오른쪽 어깨를 180도정도 회전하게 하였고

R\_Arm\_x = L\_Arm\_x; //abs함수를 사용하여 뒤쪽으로는 꺽이지 않게 하였다

R\_Leg\_x = abs(sin(time5) \* 30) - 30; //왼쪽,오른쪽 허벅지는 -30에서 시작으로 30범위를 왔다갔다 움직이게 하였다

L\_Leg\_x = R\_Leg\_x; //물론 뒤로 넘어가지 않게 abs함수를 사용하였으며

R\_Leg\_y = abs(cos(time5) \* 30); //허벅지와 달리, 상단쪽이 30도로 왔다갔다해야 하므로

L\_Leg\_y = R\_Leg\_y; //cos함수를 이용하였으며 반대로 꺽이지 않게 abs함수를 사용하였다

//애니메이션 그리기//

glClear(GL\_COLOR\_BUFFER\_BIT | GL\_DEPTH\_BUFFER\_BIT); //초기화

//GL\_COLOR\_BUFFER\_BIT: 컬러를 쓰기 위해 현재 활성화된 버퍼

//GL\_DEPTH\_BUFFER\_BIT: 깊이 버퍼

//로봇이 체조를 하면서 상,하 움직이는 것을 표현

float i = 0;

i = abs(sin(time5) \* 0.08); //i변수값 설정

gluLookAt(0.0, 0.0, 7.0, 0.0, 0.0, 0.0, 0.0, 1.0, 0.0);

glViewport(0, 0, Width / 2, Height / 2); //좌하단

DrawGround1(); //움직이지 않은 지면 호출

glPushMatrix();// 처음 저장 좌표를 저장한다

glTranslatef(0.0, i, 0); //변수 i만큼 로봇의 몸체가 Y축기준으로 움직인다

glTranslatef(0.0, 0.5, 0.0);//최초 위치로 다시 옮긴다

glPushMatrix(); // 처음 저장 위치

DrawBody(0, 0, 0, 0); // 몸통 함수 호출

Drawneck(); // 목함수 호출

DrawHead(); // 머리 호출

glPopMatrix(); // 처음 저장 좌표로 돌아 간다.

glPushMatrix();// 처음 저장 좌표 다시 저장//오른쪽 팔부분 호출

DrawR\_Arm(R\_Arm\_x, 1, 0, 0); //오른쪽팔호출

DrawR\_Hand(R\_Arm\_y, 1, 0, 0); //오른쪽팔뚝

glPopMatrix(); // 처음 저장 좌표로 돌아 간다.

glPushMatrix();// 처음 저장 좌표 다시 저장//왼쪽 팔 호출

DrawL\_Arm(L\_Arm\_x, 1, 0, 0); //왼쪽팔호출

DrawL\_Hand(L\_Arm\_y, 1, 0, 0); //왼족팔뚝

glPopMatrix(); // 처음 저장 좌표로 돌아 간다.

glPushMatrix();// 처음 저장 좌표 다시 저장//왼쪽 다리 호출

glTranslatef(-0.1, 0.0, -0.1);//미리 위치 조정후 회전

glRotatef(-90, 0, 1, 0); //y축 기준으로 회전

DrawL\_Legs(L\_Leg\_x, 1, 0, 0); //왼쪽허벅지

DrawL\_foot(L\_Leg\_y, 1, 0, 0); //왼쪽종아리

glPopMatrix(); // 처음 저장 좌표로 돌아 간다.

glPushMatrix();// 처음 저장 좌표 다시 저장//왼쪽 다리 호출

glTranslatef(0.1, 0.0, -0.1);//미리 위치 조정후 회전

glRotatef(90, 0, 1, 0); //y축 기준으로 회전

DrawR\_Legs(R\_Leg\_x, 1, 0, 0); //오른쪽허벅지

DrawR\_foot(R\_Leg\_y, 1, 0, 0); //오른쪽종아리

glPopMatrix(); // 처음 저장 좌표로 돌아 간다.

glPopMatrix(); // 처음 저장 좌표로 돌아 간다.

glMatrixMode(GL\_MODELVIEW);

glLoadIdentity(); //시점좌표를 초기화 후 재설정

gluLookAt(7.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0, 0.0); //우하단

glViewport(Width / 2, 0, Width / 2, Height / 2);

DrawGround1(); //움직이지 않은 지면 호출

glPushMatrix();// 처음 저장 좌표를 저장한다

glTranslatef(0.0, i, 0); //변수 i만큼 로봇의 몸체가 Y축기준으로 움직인다

glTranslatef(0.0, 0.5, 0.0);//최초 위치로 다시 옮긴다

glPushMatrix(); // 처음 저장 위치

DrawBody(0, 0, 0, 0); // 몸통 함수 호출

Drawneck(); // 목함수 호출

DrawHead(); // 머리 호출

glPopMatrix(); // 처음 저장 좌표로 돌아 간다.

glPushMatrix();// 처음 저장 좌표 다시 저장//오른쪽 팔부분 호출

DrawR\_Arm(R\_Arm\_x, 1, 0, 0); //오른쪽팔호출

DrawR\_Hand(R\_Arm\_y, 1, 0, 0); //오른쪽팔뚝

glPopMatrix(); // 처음 저장 좌표로 돌아 간다.

glPushMatrix();// 처음 저장 좌표 다시 저장//왼쪽 팔 호출

DrawL\_Arm(L\_Arm\_x, 1, 0, 0); //왼쪽팔호출

DrawL\_Hand(L\_Arm\_y, 1, 0, 0); //왼족팔뚝

glPopMatrix(); // 처음 저장 좌표로 돌아 간다.

glPushMatrix();// 처음 저장 좌표 다시 저장//왼쪽 다리 호출

glTranslatef(-0.1, 0.0, -0.1);//미리 위치 조정후 회전

glRotatef(-90, 0, 1, 0); //y축 기준으로 회전

DrawL\_Legs(L\_Leg\_x, 1, 0, 0); //왼쪽허벅지

DrawL\_foot(L\_Leg\_y, 1, 0, 0); //왼쪽종아리

glPopMatrix(); // 처음 저장 좌표로 돌아 간다.

glPushMatrix();// 처음 저장 좌표 다시 저장//왼쪽 다리 호출

glTranslatef(0.1, 0.0, -0.1);//미리 위치 조정후 회전

glRotatef(90, 0, 1, 0); //y축 기준으로 회전

DrawR\_Legs(R\_Leg\_x, 1, 0, 0); //오른쪽허벅지

DrawR\_foot(R\_Leg\_y, 1, 0, 0); //오른쪽종아리

glPopMatrix(); // 처음 저장 좌표로 돌아 간다.

glPopMatrix(); // 처음 저장 좌표로 돌아 간다.

glMatrixMode(GL\_MODELVIEW);

glLoadIdentity(); //시점좌표를 초기화 후 재설정

gluLookAt(0.0, 7.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, -1.0); //좌상단

glViewport(0, Height / 2, Width / 2, Height / 2);

DrawGround1(); //움직이지 않은 지면 호출

glPushMatrix();// 처음 저장 좌표를 저장한다

glTranslatef(0.0, i, 0); //변수 i만큼 로봇의 몸체가 Y축기준으로 움직인다

glTranslatef(0.0, 0.5, 0.0);//최초 위치로 다시 옮긴다

glPushMatrix(); // 처음 저장 위치

DrawBody(0, 0, 0, 0); // 몸통 함수 호출

Drawneck(); // 목함수 호출

DrawHead(); // 머리 호출

glPopMatrix(); // 처음 저장 좌표로 돌아 간다.

glPushMatrix();// 처음 저장 좌표 다시 저장//오른쪽 팔부분 호출

DrawR\_Arm(R\_Arm\_x, 1, 0, 0); //오른쪽팔호출

DrawR\_Hand(R\_Arm\_y, 1, 0, 0); //오른쪽팔뚝

glPopMatrix(); // 처음 저장 좌표로 돌아 간다.

glPushMatrix();// 처음 저장 좌표 다시 저장/

DrawL\_Arm(L\_Arm\_x, 1, 0, 0); //왼쪽팔호출

DrawL\_Hand(L\_Arm\_y, 1, 0, 0); //왼족팔뚝

glPopMatrix(); // 처음 저장 좌표로 돌아 간다.

glPushMatrix();// 처음 저장 좌표 다시 저장

glTranslatef(-0.1, 0.0, -0.1);//미리 위치 조정후 회전

glRotatef(-90, 0, 1, 0); //y축 기준으로 회전

DrawL\_Legs(L\_Leg\_x, 1, 0, 0); //왼쪽허벅지

DrawL\_foot(L\_Leg\_y, 1, 0, 0); //왼쪽종아리

glPopMatrix(); // 처음 저장 좌표로 돌아 간다.

glPushMatrix();// 처음 저장 좌표 다시 저장

glTranslatef(0.1, 0.0, -0.1);//미리 위치 조정후 회전

glRotatef(90, 0, 1, 0); //y축 기준으로 회전

DrawR\_Legs(R\_Leg\_x, 1, 0, 0); //오른쪽허벅지

DrawR\_foot(R\_Leg\_y, 1, 0, 0); //오른쪽종아리 리

glPopMatrix(); // 처음 저장 좌표로 돌아 간다.

glPopMatrix(); // 처음 저장 좌표로 돌아 간다.

glMatrixMode(GL\_MODELVIEW);

glLoadIdentity(); //시점좌표를 초기화 후 재설정

gluLookAt(5.0, 5.0, 5.0, 0.0, 0.0, 0.0, 0.0, 1.0, 0.0); //우상단

glViewport(Width / 2, Height / 2, Width / 2, Height / 2);

DrawGround1(); //움직이지 않은 지면 호출

glPushMatrix();// 처음 저장 좌표를 저장한다

glTranslatef(0.0, i, 0); //변수 i만큼 로봇의 몸체가 Y축기준으로 움직인다

glTranslatef(0.0, 0.5, 0.0);//최초 위치로 다시 옮긴다

glPushMatrix(); // 처음 저장 위치

DrawBody(0, 0, 0, 0); // 몸통 함수 호출

Drawneck(); // 목함수 호출

DrawHead(); // 머리 호출

glPopMatrix(); // 처음 저장 좌표로 돌아 간다.

glPushMatrix(); // 처음 저장 위치

DrawR\_Arm(R\_Arm\_x, 1, 0, 0); //오른쪽팔호출

DrawR\_Hand(R\_Arm\_y, 1, 0, 0); //오른쪽팔뚝

glPopMatrix(); // 처음 저장 좌표로 돌아 간다.

glPushMatrix(); // 처음 저장 위치

DrawL\_Arm(L\_Arm\_x, 1, 0, 0); //왼쪽팔호출

DrawL\_Hand(L\_Arm\_y, 1, 0, 0); //왼족팔뚝

glPopMatrix(); // 처음 저장 좌표로 돌아 간다.

glPushMatrix(); // 처음 저장 위치

glTranslatef(-0.1, 0.0, -0.1);//미리 위치 조정후 회전

glRotatef(-90, 0, 1, 0); //y축 기준으로 회전

DrawL\_Legs(L\_Leg\_x, 1, 0, 0); //왼쪽허벅지

DrawL\_foot(L\_Leg\_y, 1, 0, 0); //왼쪽종아리

glPopMatrix(); // 처음 저장 좌표로 돌아 간다.

glPushMatrix(); // 처음 저장 위치

glTranslatef(0.1, 0.0, -0.1);//미리 위치 조정후 회전

glRotatef(90, 0, 1, 0); //y축 기준으로 회전

DrawR\_Legs(R\_Leg\_x, 1, 0, 0); //오른쪽허벅지

DrawR\_foot(R\_Leg\_y, 1, 0, 0); //오른쪽종아리

glPopMatrix(); // 처음 저장 좌표로 돌아 간다.

glPopMatrix(); // 처음 저장 좌표로 돌아 간다.

}

void MyDisplay() {

glMatrixMode(GL\_PROJECTION);

glPushMatrix();

glLoadIdentity();

gluPerspective(30, (GLdouble)Width / (GLdouble)Height, 3.0, 50.0);

glMatrixMode(GL\_MODELVIEW);

glPushMatrix();

//gluLookAt(0.0, 0.0, 7.0, 0.0, 0.0, 0.0, 0.0, 1.0, 0.0);

MilitaryExercise();

glPopMatrix();

glMatrixMode(GL\_PROJECTION);

glPopMatrix();

glutSwapBuffers();

}

void MyReshape(int w, int h) {

Width = w;

Height = h;

}

/\*

타이머 함수

각 장면마다 상태 표현을 다르게 두기 위해 대양한 시간 변수들이 쓰였다.

\*/

void MyTimer(int Value) {

time5 = time5 + 0.01; //군대체조 타이머

glutPostRedisplay();

glutTimerFunc(5, MyTimer, 1);

}

void MyIdle() {//아이들 콜백

time5 = time5 + 0.01; //군대체조 타이머

glutPostRedisplay();

}

int main(int argc, char\*\* argv) {

Width = 400; //초기설정값

Height = 400;

glutInit(&argc, argv);//초기화

glutInitDisplayMode(GLUT\_RGB | GLUT\_DOUBLE | GLUT\_DEPTH);//디스플레이 모드 설정

glutInitWindowSize(Width, Height);//윈도우 크기 설정

glutInitWindowPosition(0, 0);//윈도우 초기 위치 설정

glutCreateWindow("과제3 16011024 주수헌");//윈도우 생성

glInit(); // 조명 초기화

//glutTimerFunc(5, MyTimer, 1); //타이머 함수

glutIdleFunc(MyIdle); //아이들콜백함수로

glutDisplayFunc(MyDisplay); //디스플레이

glutReshapeFunc(MyReshape); //화면 크기 조정

glutMainLoop();

return 0;

}