//999999999999999999999999999999999999999999999999999999999999999999999999999999999999999//9999999999

#include <windows.h>

#include <stdio.h>

#include <iostream>

#include <gl/glut.h>

#include<conio.h>

//#include "opencv2/imgcodecs.hpp"

#include "opencv2/core/core.hpp"

#include "opencv2/highgui/highgui.hpp"

#include "opencv2/imgproc/imgproc.hpp"

#include<math.h>

using namespace cv;

using namespace std;

//Variables for OpenGl

// actual vector representing the camera's direction

float lx = 0.0f, lz = -1.0f;

// XZ position of the camera

float x = 0.0f, z = 15.0f;

float ly = 0.0f, y = 1.0f;

//float robox = x + lx;

GLdouble wx = 1.0, wy, wz; /\* returned world x, y, z coords \*/

float angle = 0.0f;

double dAreaThumb;

int DistForRotX = 0;

GLdouble wxForThumb, wyForThumb, wzForThumb;

//GLdouble AtTheTimeOfSelectionX, AtTheTimeOfSelectionY;

//GLint realy2;

//int mrv = 5;//mill rotation varibale

short int AbtX = 0, AbtY = 0, AbtZ = 0;

// all variables initialized to 1.0, meaning

// the triangle will initially be white

//float red = 1.0f, blue = 1.0f, green = 1.0f;

// angle for rotating triangle

//float angle = 0.0f;

float RoboRotateAngleForTurn = 0.0f;

//End Variable for OpenGl

// variables for image processing ...........

//For Green Color .......

//Of Green Below Of Index

int GreeniLowH = 50;//37;// 42;// 49;// 43;// 25;// 65;// 25;// 44;// 47;// 44;// 56;// 70;// 43;// 22;// 77;// 86;// 63;// 59;// 42;//72;//170//

int GreeniHighH = 71;// 59;// 60;// 79;// 72;// 76;// 85;// 76;// 94;// 57;// 94;// 76k;// 96;// 103;// 78;// 96;// 107;// 99;// 86;// 109;// 98;// 101;// 74;// 90;//179//

int GreeniLowS = 110;// 93;// 164;// 56;// 131;// 93;// 143;// 93;// 100;// 160;// 100;// 84k;// 91k; //108;// 108;// 87;// 104;// 143;// 112;// 102;//102;//33K;// 102k;// 129;// 98;// 147;// 117;// 115;//150,,

int GreeniHighS = 255;// 189;// 255;// 189;// 176;// 255;//255

int GreeniLowV = 96;// 0;// 66;// 171;// 56;// 126;// 105;// 126;// 0;// 102;//;// 51k;// 0k;// 58;// 84;// 112;// 204;// 117;// 101;// 101;// 1;// 1;// 136K;// 1k;// 154;// 37;// 79;// 114;//60,

int GreeniHighV = 255;//255

//int GreeniLowH = 32;//37;// 42;// 49;// 43;// 25;// 65;// 25;// 44;// 47;// 44;// 56;// 70;// 43;// 22;// 77;// 86;// 63;// 59;// 42;//72;//170//

//int GreeniHighH = 179;// 59;// 60;// 79;// 72;// 76;// 85;// 76;// 94;// 57;// 94;// 76k;// 96;// 103;// 78;// 96;// 107;// 99;// 86;// 109;// 98;// 101;// 74;// 90;//179//

//int GreeniLowS = 173;// 93;// 164;// 56;// 131;// 93;// 143;// 93;// 100;// 160;// 100;// 84k;// 91k; //108;// 108;// 87;// 104;// 143;// 112;// 102;//102;//33K;// 102k;// 129;// 98;// 147;// 117;// 115;//150,,

//int GreeniHighS = 255;// 189;// 255;// 189;// 176;// 255;//255

//int GreeniLowV = 100;// 0;// 66;// 171;// 56;// 126;// 105;// 126;// 0;// 102;//;// 51k;// 0k;// 58;// 84;// 112;// 204;// 117;// 101;// 101;// 1;// 1;// 136K;// 1k;// 154;// 37;// 79;// 114;//60,

//int GreeniHighV = 255;//255

//End For Green Color....

//Of Blue/Other Thumb Colr Below For Thumb

int BlueiLowH = 17;// 154;// 139;// 156;// 170;// 166;// 156;// 166;// 125 this is ok the 2nd;// 146;// 159;// 147;// 157k;// 22;// 86;// 96;// 102;// 99;// 78;// 78;// 82K;// 78k;// 110;// 54;// 91;// 99;// 91;// 102;// 42;// 105;// 2;//96

int BlueiHighH = 32;// 179;// 174;// 174;// 179;// 169;// 179;// 179;// 34;// 129;// 162;// 128;// 124;// 119;// 132;// 114;// 123;// 115;// 132;// 115;// 128;// 121;// 7;//137

int BlueiLowS = 150;// 75;// 65;// 105;// 91;// 89;// 91;// 107;// 87;// 87;// 77;// 98;// 108;// 50k;// 109k;// 114;// 94;// 104;// 56;// 147; //107; //123;// 123;// 31K;// 123k;// 80;// 28;// 122;// 122;// 89;// 9;//189

int BlueiHighS = 255;// 255;// 199;// 5;//255

int BlueiLowV = 98;// 166;// 150;// 192;// 153;// 201;// 119;// 201;// 124;// 122;// 97;// 180;// 227;// 182;// 114;// 180;// 171k;// 105k;// 159;// 157;// 201;// 128;// 170;// 133;// 110;// 110;// 154K;// 110k;//;// 176;// 112;// 70;// 112;// 45;// 1;//31

int BlueiHighV = 255;// 5;//255

//int BlueiLowH = 162;// 139;// 156;// 170;// 166;// 156;// 166;// 125 this is ok the 2nd;// 146;// 159;// 147;// 157k;// 22;// 86;// 96;// 102;// 99;// 78;// 78;// 82K;// 78k;// 110;// 54;// 91;// 99;// 91;// 102;// 42;// 105;// 2;//96

//int BlueiHighH = 179;// 174;// 174;// 179;// 169;// 179;// 179;// 34;// 129;// 162;// 128;// 124;// 119;// 132;// 114;// 123;// 115;// 132;// 115;// 128;// 121;// 7;//137

//int BlueiLowS = 96;// 75;// 65;// 105;// 91;// 89;// 91;// 107;// 87;// 87;// 77;// 98;// 108;// 50k;// 109k;// 114;// 94;// 104;// 56;// 147; //107; //123;// 123;// 31K;// 123k;// 80;// 28;// 122;// 122;// 89;// 9;//189

//int BlueiHighS = 255;// 255;// 199;// 5;//255

//int BlueiLowV = 141;// 150;// 192;// 153;// 201;// 119;// 201;// 124;// 122;// 97;// 180;// 227;// 182;// 114;// 180;// 171k;// 105k;// 159;// 157;// 201;// 128;// 170;// 133;// 110;// 110;// 154K;// 110k;//;// 176;// 112;// 70;// 112;// 45;// 1;//31

//int BlueiHighV = 255;// 5;//255

//int ThumbUpCount = 0;

//int TransCount = 0;

int ThumbIndexDiffX = 0;//Thumb Index Difference x-axis

int ThumbIndexDiffY = 0;// same above for y

bool SelectEnable = 0;

bool ScaleEnable = 0;

//bool LoadUnload = 0;

bool ForSelect = 1;//By defualt selected for select/Translate

//bool ForScale = 0;

bool ThumbVisible = 0;

bool IndexVisible = 0;

bool MakePositionForScaleFixed = 0;

bool Gate4Scale = 0;

int Gate4Stop = 0;

bool TeapotRotateEnable = 0;

bool SelectA = 0;

bool SelectAA = 0;

bool SelectB = 0;

Mat imgHSVThumb;

//Mat WholeImg;

Mat WholeImgHSV;

Mat imgThreshold0;

Mat imgThresholded;

Mat imgHSVThumbLeft;

Mat imgThresholdedThumb;

Mat imgThreshold0Thumb;

Mat imgThreshold1Thumb;

GLdouble FiexdCenterPositionForRotationY, FiexdCenterPositionForRotationX;

GLdouble ThumbFiexdCenterPositionForRotationY, ThumbFiexdCenterPositionForRotationX;

int posXThumb, posYThumb;

int posXThumbLeft, posYThumbLeft;

bool ThumbUp = 0;

float Tpx = 0;//TeaPot x

//float TpxPre; 0, -.028, -0.15

float Tpy = -0.028;//TeaPot y

float Tpz = -0.14;//TeaPot z

float FixedTpyForScale;//

float TpxForScale = 0.130;//For Scalng increase

float FixedTpxForScale;//

float TpyForScale = 0.15;

float FixedTpzForScale;//

float TpzForScale = 0.1;

bool TeapotTranslEnable = 0;

bool TeapotScalelEnable = 0;

//Mat ForBlueImg;

//Mat BlueHueImage;

bool Start = 0;

//bool Phase1 = 1;

//bool Phase2 = 0;

int CountSelectA = 0;

//bool start = 1;

Mat image;// , image2;

Mat NearIndexImage, NearIndexImageLeft;// , NearIndexImageLeftThreshold;

static bool NavigateEnable = 0;

bool OnceGate = 1;

bool Gate4Rotate = 0;

bool SelectAndNavigate = 0;

bool GateFromCentralArea = 0;

bool Gate4WheelRotate = 0;

bool ForRotate = 0;

bool OnceGate4Rotat = 0;

bool Detected = 0;

bool LeftTurn = 0, RightTurn = 0;

bool GateForUp = 0;

int IndexGreenposX, IndexGreenposY;

int IndexGreenposXNScale, IndexGreenposYNScale;

int posYUpLimit;

//int PreposX, PreposY;

long int Arr[5];

int cnt = 0;

int centerXArr[5];

int centerYArr[5];

int centerX;

int centerY;

int WheelRotateAng = 0;

Mat imgHSV;

Mat ForGreenImg;

Mat WholeImg;

Mat GreenimgThresholded;

Mat GreenimgThresholde0;

Mat BlueimgThresholded;

int Aavg;

//End for variable in image processing........

cv::VideoCapture cap(0);

//....bool data

void reset()

{

lx = 0.0f, lz = -1.0f;

// XZ position of the camera

x = 0.0f, z = 15.0f;

ly = 0.0f, y = 1.0f;

AbtX = 0;

AbtY = 0;

AbtZ = 0;

DistForRotX = 0;

//robox = x + lx;

//mrv = 5;//mill rotation varibale

// all variables initialized to 1.0, meaning

// the triangle will initially be white

//red = 1.0f, blue = 1.0f, green = 1.0f;

// angle for rotating triangle

//angle = 0.0f;

//End Variable for OpenGl

ThumbIndexDiffX = 0;//Thumb Index Difference x-axis

ThumbIndexDiffY = 0;// same above for y

SelectEnable = 0;

ScaleEnable = 0;

//LoadUnload = 0;

ForSelect = 1;//By defualt selected for select/Translate

//ForScale = 0;

ThumbVisible = 0;

IndexVisible = 0;

MakePositionForScaleFixed = 0;

Gate4Scale = 0;

Gate4Stop = 0;

ThumbUp = 0;

Tpx = 0;//TeaPot x

//float TpxPre; 0, -.028, -0.15

Tpy = -0.028;//TeaPot y

Tpz = -0.14;//TeaPot z

FixedTpyForScale;//

TpxForScale = 0.130;//For Scalng increase

FixedTpxForScale;//

TpyForScale = 0.15;

FixedTpzForScale;//

TpzForScale = 0.1;

TeapotTranslEnable = 0;

TeapotScalelEnable = 0;

//Mat ForBlueImg;

//Mat BlueHueImage;

Start = 0;

//Phase1 = 1;

//Phase2 = 0;

//bool start = 1;

NavigateEnable = 0;

OnceGate = 1;

SelectAndNavigate = 0;

GateFromCentralArea = 0;

Gate4WheelRotate = 0;

Detected = 0;

LeftTurn = 0, RightTurn = 0;

GateForUp = 0;

IndexGreenposX, IndexGreenposY;

IndexGreenposXNScale, IndexGreenposYNScale;

posYUpLimit;

//int PreposX, PreposY;

//End For Green Color....

NavigateEnable = 0;

OnceGate = 1;

GateFromCentralArea = 0;

Gate4WheelRotate = 0;

IndexGreenposX = 0, IndexGreenposY = 0;

//PreposX = 0, PreposY = 0;

//Arr[5];

cnt = 0;

centerX = 0;

centerY = 0;

WheelRotateAng = 0;

//Mat ForGreenImg;

//Mat GreenimgThresholded;

Aavg = 0;

Detected = 0;

GateForUp = 0;

SelectA = 0;

SelectAA = 0;

SelectB = 0;

//dArea = 0;

}

//Start for Scale Fun

//End for Scale Fun

void detectImg(cv::Mat scene)

{

float fraction = 0.1f;// for camera ratio

/\*Mat src;

src = scene;\*/

//imshow()

//imshow("poll", scene);

//if (Phase1 == 1)

{

//Mat ForThumbWholeScene = scene;

// Start For Green Thumb Detection.....

ForGreenImg = scene;

//imshow("poll", scene);

//ForBlueImg = scene;

//imshow("ForGrenImg", ForGreenImg);

cvtColor(ForGreenImg, imgHSV, COLOR\_BGR2HSV); //Convert the captured frame from BGR to HSV

//blur(ForGreenImg, ForGreenImg, Size(10, 10));

cv::inRange(imgHSV, Scalar(GreeniLowH, GreeniLowS, GreeniLowV), Scalar(GreeniHighH, GreeniHighS, GreeniHighV), GreenimgThresholded);//Green (50 - rng, 70, 70), cv::Scalar(70 + rng, 255, 255)

// N Lab (60 - rng, 100, 100), cv::Scalar(80 + rng, 255, 255)

// GreenimgThresholded = GreenimgThresholde0;

erode(GreenimgThresholded, GreenimgThresholded, getStructuringElement(2, Size(10, 10)));

dilate(GreenimgThresholded, GreenimgThresholded, getStructuringElement(2, Size(10, 10)));

//imshow("For Green Thresholded", GreenimgThresholded);

//Calculate the moments of the thresholded image

Moments oMoments = moments(GreenimgThresholded);

double dM01 = oMoments.m01;

double dM10 = oMoments.m10;

double dArea = oMoments.m00;

//cout << "\nM0Index=" << dM10 << "M1Index" << dM01 << "AreaIndex" << dArea;

if (dArea > 1080 && dM10 > 8 && dM01 > 8)//of Index green of almost greater/equal size is found

{

//IndexVisible = 1;

IndexGreenposX = dM10 / dArea;

IndexGreenposY = dM01 / dArea;

//if (dArea > Aavg - 52000)//not to detect if very small points

{

Detected = 1;//on the bool i.e. if IndexCap is detected....

Start = 1;

}

//..............\*\*\*\*\*\*\*\*\*\*

//Start modificatn for Blue

if (IndexGreenposX - 5 > 0 && IndexGreenposY - 30 > 0)//if in limit

{

//if (posY - 53 < 0)//f indexFngr is near the top

//{

// NearIndexImage = scene(Rect(0, 0, scene.cols - 2, 50));

//}

//else

// ForScaleFun(scene);

if (Detected == 1)

if (ForSelect == 1)

{

//ForBlueImg = scene(Rect(0, 0, scene.cols - 2, IndexGreenposY - 30));//For Thumb for SelectAndTranslate

NearIndexImage = scene;// (Rect(0, 0, scene.cols - 2, IndexGreenposY - 20));//For Thumb for SelectAndTranslate

//imshow("NearIndex", NearIndexImage);

cvtColor(NearIndexImage, imgHSVThumb, COLOR\_BGR2HSV); //Convert the captured frame from BGR to HSV

cv::Mat BlueHueImage;

cv::inRange(imgHSVThumb, Scalar(BlueiLowH, BlueiLowS, BlueiLowV), Scalar(BlueiHighH, BlueiHighS, BlueiHighV), BlueHueImage);//Blue

//In DigLab Scalar(100, 100, 100), cv::Scalar(120, 255, 255) Or (105, 110, 110), cv::Scalar(125, 255, 255)

erode(BlueHueImage, BlueHueImage, getStructuringElement(2, Size(5, 5)));

dilate(BlueHueImage, BlueHueImage, getStructuringElement(2, Size(5, 5)));

// blur(BlueHueImage, BlueHueImage, Size(5, 5));

// imshow("ForBluNearIndex", BlueHueImage);//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*this is to sh thumb thrshld img

Moments oMomentsThumb = moments(BlueHueImage);

double dM01Thumb = oMomentsThumb.m01;

double dM10Thumb = oMomentsThumb.m10;

dAreaThumb = oMomentsThumb.m00;

//cout << "\nM0=" << dM10Thumb << "M1" << dM01Thumb << "AreaTh" << dAreaThumb;

if (dAreaThumb > 1016 && dM10Thumb > 8 && dM01Thumb > 8)

{

posXThumb = dM10Thumb / dAreaThumb;

posYThumb = dM01Thumb / dAreaThumb;

}//if(dAreaThumb...) i.e when thumb is found viible

}//End of if(ForSelect==1)

}//End if(PosX...) In limits

//End modificatn for Blue

//.............\*\*\*\*\*\*\*\*\*\*\*

if (IndexGreenposX - 20 > 2 && IndexGreenposY - 20 > 2 && SelectAndNavigate == 1) // F n limits and happens to start Navigatn

{

if (cnt <= 2)//first it was 5

{

centerXArr[cnt] = (IndexGreenposX + posXThumb) / 2;

centerYArr[cnt] = (IndexGreenposY + posYThumb) / 2;

Arr[cnt] = (dArea + dAreaThumb) / 2;// area at center i.e main positn

cnt = cnt + 1;

//centerX = posX;//x at center positn

//centerY = posY;

//Beep(900, 200);

}

else if (OnceGate == 1)

{

Aavg = (Arr[0] + Arr[1] + Arr[2]) / 3;// + Arr[3] + Arr[4]) / 5;

centerX = (centerXArr[0] + centerXArr[1] + centerXArr[2]) / 3;// +centerXArr[3] + centerXArr[4]) / 5;//x at center positn

centerY = (centerYArr[0] + centerYArr[1] + centerYArr[2]) / 3;// +centerYArr[3] + centerYArr[4]) / 5;//x at center positn;

OnceGate = 0;

GateFromCentralArea = 1;//Make it off and no navigtn

}

// if (SelectAndNavigate == 1)//This if is new modifctn for condition to first selectng and then navigate etc

{ //cout << "\nSelectAnd=" << SelectAndNavigate << "Detec=" << Detected << "Davera=" << dArea << "GateFromCentrlaAr=" << GateFromCentralArea << "IndexxGreenPosy=" << IndexGreenposY;

//Start Below is if for going inside screen

// cout << "\nCenterX==" << centerX << "Center Y==" << centerY << " \nggOf Index-X" << IndexGreenposX << "gggOf Index-Y" << IndexGreenposY << "\n\n";

if (SelectA == 0 && (Detected == 1) && (dArea / 2) > (Aavg / 2) + 35000 && (dAreaThumb / 2) > (Aavg / 2) + 35000 && GateFromCentralArea == 1 && (abs(IndexGreenposX - posXThumb)<40))

{//Here first conditn was SelectAndNavigate == 1 //>Aavg+170000 //have disabled the above to chk efficency...

//cout << " >>>>>> INCREASING...";

x += lx \* 2 \* fraction;//for going inside screen

z += lz \* 2 \* fraction;

Gate4WheelRotate = 1;

//SelectA = 0;

//SelectAA = 0;

}//End if ( dArea>avg...) if for going inside screen

else

Gate4WheelRotate = 0;

if (SelectA == 0 && (Detected == 1) && (dAreaThumb / 2) < (Aavg / 2) - 25000 && (dArea / 2) < (Aavg / 2) - 25000 && GateFromCentralArea == 1 && (abs(IndexGreenposX - posXThumb)<40))

//Here first conditn was SelectAndNavigate == 1 <Aavg-40000

{//if for going away from screen

//cout << " <<<<<<< DECREASING...";

x -= lx \* 2 \* fraction;//for going away from screen

z -= lz \* 2 \* fraction;

Gate4WheelRotate = 1;

}//end if(dAvrea<avg) for going away frm screen

if (SelectA == 0 && (Detected == 1) && IndexGreenposY + 100 < centerY && GateFromCentralArea == 1 && !((dArea / 2) < (Aavg / 2) - 8000) && !((dArea / 2) > (Aavg / 2) + 80000) && !(IndexGreenposX + 150 < centerX) && (abs(IndexGreenposX - posXThumb)<40))

{ //<40000 //+170000

//cout << " U U UP Going...";

y = y + 0.1;// for hauring over the scene

GateForUp = 1;

posYUpLimit = IndexGreenposY;

//cout << "\n...wx=" << wx << "and wy..=" << wy<<"y="<<y;

centerY = y;

}

if (SelectA == 0 && (Detected == 1) && GateFromCentralArea == 1 && !((dArea / 2) < (Aavg / 2) - 5000) && !((dArea / 2) > (Aavg / 2) + 8000) && (IndexGreenposY - posYUpLimit >= 40) && (abs(IndexGreenposX - posXThumb)<40))

{ //<aavg-40000 && !(IndexGreenposX + 50 < centerX) //120 Comment abve to see efficency

if (!(y <= 1))

{

//cout << " D D Down Going...";

//cout << " U U UP Going...";

y = y - 0.1;// for hauring over the scene

//cout << "\n....y=" << y;

//Gate4WheelRotate = 0;

}

//GateForUp = 0;

}

if (SelectA == 0 && (Detected == 1) && IndexGreenposX <50 && GateFromCentralArea == 1 && (abs(IndexGreenposX - posXThumb)<40))

{

//cout << " L L Left Going...";

//..........

//angle += 0.08f;// turnung camera to Right

//lx = sin(angle);

//lz = -cos(angle);

//RightTurn = 1;

//............

//angle += 0.05f;//Turnng camera to Left

//lx = sin(angle);

//lz = -cos(angle);

//RightTurn = 1;

}

if (SelectA == 0 && (Detected == 1) && IndexGreenposX >260 && GateFromCentralArea == 1 && (abs(IndexGreenposX - posXThumb)<40))

{

//cout << " R R Right Going...";

//..........

//angle -= 0.05f;//Turnng camera to Left

//lx = sin(angle);

//lz = -cos(angle);

//RightTurn = 1;

//........

}

}//End if SlectAndNavig for new modifictn to conditn navg etc

}// End if (posX...)

//else// if NOT posx....and not navigation in or out stop wheeel rotatn

//Gate4WheelRotate = 0;

//cout << "\n Difference BTWn y=" << (wy - wyForThumb);

//if (ForScale == 1)vvvvvvvvvvvvvvvddwqqasssxbn kpoiuytrew

/\*{

cout << "\n Index y=" << IndexGreenposY<< "and Thumb y=" <<posYThumb ;

}\*/

//Start for Select-A region...............

if (SelectB == 0 && Gate4Rotate == 0 && SelectEnable == 0 && TeapotTranslEnable == 0 && abs(wx - wxForThumb) < 0.05 && abs(wy - wyForThumb) < 0.03 &&abs(wx - Tpx) < 0.03 && abs(wxForThumb - Tpx) < 0.03&&abs(wy - Tpy) < 0.03 && abs(wyForThumb - Tpy) < 0.03)//Check for Selection

{

SelectA = 1;

SelectAndNavigate = 1;//To stop selection at the beging first virtual hand have to touch central locatn

//SelectAA = 0;

//\_getch();

//cout << "Selectn A";

if (CountSelectA>6)

{

SelectAA = 1;

ForRotate = 1;

// OnceGate4Rotat = 0;

CountSelectA = 0;

Beep(500, 50);

// CountSelectA = 0;

}

/\*else

SelectAA = 0;\*/

}

else if (abs(IndexGreenposX - posXThumb)>50)//Cross gesture for Deselection..

{

CountSelectA = 0;

SelectA = 0;

SelectAA = 0;

Gate4Rotate = 0;

TeapotRotateEnable = 0;

OnceGate4Rotat = 0;

ForRotate = 0;

SelectB = 0;

SelectEnable = 0;

TeapotTranslEnable = 0;

//DeSelectGate = 1;

}

//End for Select-A region.................

//Start for Selection B..............

if (SelectA == 1 && Gate4Rotate == 0 && SelectEnable == 0 && TeapotTranslEnable == 0 && abs(wx - wxForThumb) < 0.05 && abs(wy - wyForThumb) < 0.01 &&abs(wx - Tpx) < 0.04 && abs(wxForThumb - Tpx) < 0.04&&abs(wy - Tpy) < 0.04 && abs(wyForThumb - Tpy) < 0.04)//Check for Selection

{

SelectB = 1;

SelectEnable = 1;

TeapotTranslEnable = 1;

SelectA = 0;

SelectAA = 0;

Gate4Rotate = 0;

TeapotRotateEnable = 0;

OnceGate4Rotat = 0;

ForRotate = 0;

//\_getch();

}

else if (abs(wx - wxForThumb) >0.06 && abs(wy - wyForThumb) > 0.02)

{

SelectB = 0;

SelectEnable = 0;

TeapotTranslEnable = 0;

//cout << "\n Deselct SELECTBBBB";

}

//End for Selection B...............

//Start for Rotation if Gate4Rotate is Enabled.................................

if (ForRotate == 1 && OnceGate4Rotat == 0)

{

/\*AbtX = 0;

AbtY = 0;

AbtZ = 0;\*/

OnceGate4Rotat = 1;

//Beep(500, 90);

//Sleep(9000);

Gate4Rotate = 1;

FiexdCenterPositionForRotationX = IndexGreenposX;// (wx);// - wxForThumb)/2;// (IndexGreenposX + posXThumb) / 2;

FiexdCenterPositionForRotationY = IndexGreenposY;// (wy);// -wyForThumb) / 2; //(IndexGreenposX + posXThumb) / 2;

ThumbFiexdCenterPositionForRotationY = posYThumb;

ThumbFiexdCenterPositionForRotationX = posXThumb;

//cout << "GateForRota=" << Gate4Rotate;

//ForRotate = 0;

//\_getch();

}

//cout << "\nIndexCentl y" << FiexdCenterPositionForRotationY << "Index currnt " << IndexGreenposY << "ThumbFixedxx==" << ThumbFiexdCenterPositionForRotationY << "Thumb posx" << posYThumb;

//cout << "\nIndexCentl x" << FiexdCenterPositionForRotationY << "Index currnt " << IndexGreenposY << "ThumbFixedxx==" << ThumbFiexdCenterPositionForRotationX << "Thumb posx" << posYThumb;

if (Gate4WheelRotate == 0 && Gate4Rotate == 1 && (FiexdCenterPositionForRotationX > IndexGreenposX + 20) && (ThumbFiexdCenterPositionForRotationX > posXThumb + 20) && (FiexdCenterPositionForRotationY > IndexGreenposY + 30) && (ThumbFiexdCenterPositionForRotationY > posYThumb + 30))

{

//cout << "Going Left........";

AbtX = 0;

AbtY = 0;

AbtZ = 1;

DistForRotX = 3 \* abs(IndexGreenposY - FiexdCenterPositionForRotationY);// 2000 \* abs((wx));// + wxForThumb) / 2 - FiexdCenterPositionForRotationX);

//cout << "Distance For R="<<DistForRotX;

//Beep(500, 100);

}

else{

if (Gate4WheelRotate == 0 && Gate4Rotate == 1 && ((abs(FiexdCenterPositionForRotationY - IndexGreenposY) < 30 && abs(ThumbFiexdCenterPositionForRotationY - posYThumb) < 30) || (abs(FiexdCenterPositionForRotationX - IndexGreenposX) < 30 && abs(ThumbFiexdCenterPositionForRotationX - posXThumb) < 30)))

{

TeapotRotateEnable = 1;

SelectEnable = 0;

TeapotTranslEnable = 0;

TeapotScalelEnable = 0;

//cout << "\nIndexCentl x" << FiexdCenterPositionForRotationX << "Index currnt " << IndexGreenposX << "ThumbFixedxx==" << ThumbFiexdCenterPositionForRotationX<<"Thumb posx"<<posXThumb;

//for z-axis rotn below

//if ((FiexdCenterPositionForRotationX > IndexGreenposX) && (ThumbFiexdCenterPositionForRotationX > posXThumb) && (FiexdCenterPositionForRotationY > IndexGreenposY) && (ThumbFiexdCenterPositionForRotationY > posYThumb))

//{

// //cout << "Going Left........";

// AbtX = 0;

// AbtY = 0;

// AbtZ = 1;

// DistForRotX = 2 \* abs(IndexGreenposY - FiexdCenterPositionForRotationY);// 2000 \* abs((wx));// + wxForThumb) / 2 - FiexdCenterPositionForRotationX);

//

// //cout << "Distance For R="<<DistForRotX;

// //Beep(500, 100);

//}

//For z-ais rotn ends..

//else

{

if (FiexdCenterPositionForRotationX + 30 < IndexGreenposX && (ThumbFiexdCenterPositionForRotationX + 30 < posXThumb))//&& abs(FiexdCenterPositionForRotationY - IndexGreenposY) <200)

{

//cout << "Going Left........";

AbtX = 0;

AbtZ = 0;

AbtY = -1;

DistForRotX = 3 \* abs(IndexGreenposX - FiexdCenterPositionForRotationX);// 2000 \* abs((wx));// + wxForThumb) / 2 - FiexdCenterPositionForRotationX);

//cout << "Distance For R="<<DistForRotX;

//Beep(500, 90);

}

else if (FiexdCenterPositionForRotationX > IndexGreenposX + 30 && (ThumbFiexdCenterPositionForRotationX > posXThumb + 30))//abs(FiexdCenterPositionForRotationY - IndexGreenposY) <200)

{

//cout << "Going Left........";

AbtX = 0;

AbtZ = 0;

AbtY = 1;

DistForRotX = 3 \* abs(FiexdCenterPositionForRotationX - IndexGreenposX);// 2000 \* abs((wx));// + wxForThumb) / 2 - FiexdCenterPositionForRotationX);

//cout << "Distance For R="<<DistForRotX;

//Beep(500, 90);

}

else if ((FiexdCenterPositionForRotationY > IndexGreenposY + 20) && (ThumbFiexdCenterPositionForRotationY > posYThumb + 10))// && abs(FiexdCenterPositionForRotationX -IndexGreenposX) <200)// Abtx Goinh Up rotn

{

//cout << "Going up........";

AbtY = 0;

AbtZ = 0;

AbtX = 1;

DistForRotX = 3 \* abs(FiexdCenterPositionForRotationY - IndexGreenposY);// +wyForThumb) / 2 - FiexdCenterPositionForRotationY);

// Beep(500, 90);

//cout << "Distance For R=";

}

else if ((FiexdCenterPositionForRotationY + 40 < IndexGreenposY) && (ThumbFiexdCenterPositionForRotationY + 40 < posYThumb))// && abs(FiexdCenterPositionForRotationX -IndexGreenposX) <200)// Abtx Goinh Up rotn

{

//cout << "Going Left........";

AbtY = 0;

AbtZ = 0;

AbtX = 1;

DistForRotX = 3 \* abs(FiexdCenterPositionForRotationY - IndexGreenposY);// +wyForThumb) / 2 - FiexdCenterPositionForRotationY);

// Beep(500, 90);

//cout << "Distance For R="<<DistForRotX;

}

}

}

//Start Below start for scaling conditions...............

//TpyForScale = TpyForScale + 0.02;

if (abs(FiexdCenterPositionForRotationY - IndexGreenposY)>10 && abs(ThumbFiexdCenterPositionForRotationY - posYThumb)>10 && (FiexdCenterPositionForRotationY >IndexGreenposY + 5 && ThumbFiexdCenterPositionForRotationY + 5<posYThumb))

{

TpyForScale = TpyForScale + 0.02;

// Beep(500, 40);

//cout << "\n\tScaling" << "ThumbFiexdCenterPositionForRotationY=" << ThumbFiexdCenterPositionForRotationY<<"Index green y="<<IndexGreenposY;

}

else if (abs(FiexdCenterPositionForRotationY - IndexGreenposY)>10 && abs(ThumbFiexdCenterPositionForRotationY - posYThumb)>2 && (FiexdCenterPositionForRotationY<IndexGreenposY) && (ThumbFiexdCenterPositionForRotationY >posYThumb + 5))//&& ThumbFiexdCenterPositionForRotationY>posYThumb))

{

TpyForScale = TpyForScale - 0.02;

// Beep(500, 40);

//cout << "Down Scaling";

}

else if (abs(FiexdCenterPositionForRotationX - IndexGreenposX)>10 && abs(ThumbFiexdCenterPositionForRotationX - posXThumb)>10 && (FiexdCenterPositionForRotationX >IndexGreenposX + 5 && ThumbFiexdCenterPositionForRotationX + 5<posXThumb) && SelectAA == 1)

{ //Along X-ais Scaling Up

TpxForScale = TpxForScale + 0.04;

// Beep(500, 40);

//cout << "\n\tScaling" << "ThumbFiexdCenterPositionForRotationY=" << ThumbFiexdCenterPositionForRotationY<<"Index green y="<<IndexGreenposY;

}

else if (abs(FiexdCenterPositionForRotationX - IndexGreenposX)>10 && abs(ThumbFiexdCenterPositionForRotationX - posXThumb)>10 && (FiexdCenterPositionForRotationX + 6<IndexGreenposX && ThumbFiexdCenterPositionForRotationX >posXThumb + 6) && SelectAA == 1)

{//Along X-ais Scaling Down

TpxForScale = TpxForScale - 0.04;

//Beep(500, 40);

//cout << "Down Scaling";

}

//End For Scaling -..............................

//else if (abs(FiexdCenterPositionForRotationY - IndexGreenposY)<10 && abs(FiexdCenterPositionForRotationX - IndexGreenposX)<10 && posXThumb>FiexdCenterPositionForRotationX)//For Abt Z-axis ths tme based on Thumb i.e yellow colr

//{

// //cout << "Going for z rotn........";

// AbtY = 0;

// AbtX = 0;

// //DistForRotX = 4 \* abs(FiexdCenterPositionForRotationY - IndexGreenposY);// +wyForThumb) / 2 - FiexdCenterPositionForRotationY);

//// Beep(500, 90);

// //cout << "Distance For R="<<DistForRotX;

//}

}

//End for Rotation ...........

}//End if(Area...) green of almost equal size

}//End if(Phase1=1)

}//End of function detect...

void plant()

{//glPushMatrix();

glLineWidth(4);

glPushMatrix();

glTranslatef(8, 1, -10);

glColor3f(0.2, 0.5, 0.2);

glScalef(0.5, 0.4, 0.5);//scale this to apply to all plants

glPushMatrix();

glBegin(GL\_LINE\_STRIP);//first from below gros part

glVertex3f(-2, 1, 0);

glVertex3f(-1, 2, 0);

glVertex3f(0, -1, 0);

glVertex3f(1, 2, 0);

glVertex3f(2, 1, 0);

// glVertex3f(0,-1.5,);

glEnd();

glPopMatrix();

glTranslatef(0, 1.8, 0);

glScalef(0.6, 0.6, 0.6);

glBegin(GL\_LINE\_STRIP);//2nd from below gros part

glVertex3f(-2, 1, 0);

glVertex3f(-1, 2, 0);

glVertex3f(0, -1, 0);

glVertex3f(1, 2, 0);

glVertex3f(2, 1, 0);

// glVertex3f(0,-1.5,);

glEnd();

glTranslatef(0, 2.5, 0);

glScalef(0.6, 0.6, 0.6);

glBegin(GL\_LINE\_STRIP);//3rd from below gros part

glVertex3f(-2, 1, 0);

glVertex3f(-1, 2, 0);

glVertex3f(0, -1, 0);

glVertex3f(1, 2, 0);

glVertex3f(2, 1, 0);

// glVertex3f(0,-1.5,);

glEnd();

//glTranslatef(0,1.8,0);

//glScalef(0.6,0.6,0.6);

glColor3f(1, 1, 0);

glBegin(GL\_LINES);//stem of the plant.

glVertex3f(0, 4, 0);

glVertex3f(0, -15, 0);

// glVertex3f(0,-1.5,);

glEnd();

//glPopMatrix();

glPopMatrix();

}//end for the plant function

void pointer()

{

//if (ThumbUp)

{

GLint viewportForThumb[4];

GLdouble mvmatrixForThumb[16], projmatrixForThumb[16];

GLint realyForThumb; /\* OpenGL y coordinate position \*/

{

glGetIntegerv(GL\_VIEWPORT, viewportForThumb);

glGetDoublev(GL\_MODELVIEW\_MATRIX, mvmatrixForThumb);

glGetDoublev(GL\_PROJECTION\_MATRIX, projmatrixForThumb);

realyForThumb = viewportForThumb[3] - (GLint)(posYThumb \* 2);

// realy2 = viewport[3] - (GLint)((IndexGreenposY+160) \* 2);

//if (minLoc.x > (int)coreimg2.cols / 2)

{

gluUnProject((GLfloat)posXThumb \* 4, (GLfloat)(realyForThumb), 0.0,

mvmatrixForThumb, projmatrixForThumb, viewportForThumb, &wxForThumb, &wyForThumb, &wzForThumb);

}

wxForThumb = -(2 \* wxForThumb);// +wx;// +0.5;

//wy = wy;// +wy;// +0.5;

//printf("World coords at z=0.0 are (%f, %f, %f)\n",wx, wy, wz);

}

}

//if (Start == 1)//f image has been seeen

{

GLint viewport[4];

GLdouble mvmatrix[16], projmatrix[16];

GLint realy; /\* OpenGL y coordinate position \*/

{

glGetIntegerv(GL\_VIEWPORT, viewport);

glGetDoublev(GL\_MODELVIEW\_MATRIX, mvmatrix);

glGetDoublev(GL\_PROJECTION\_MATRIX, projmatrix);

realy = viewport[3] - (GLint)(IndexGreenposY \* 2);

// realy2 = viewport[3] - (GLint)((IndexGreenposY+160) \* 2);

//if (minLoc.x > (int)coreimg2.cols / 2)

{

gluUnProject((GLfloat)IndexGreenposX \* 4, (GLfloat)(realy), 0.0,

mvmatrix, projmatrix, viewport, &wx, &wy, &wz);

}

wx = -(2 \* wx);// +wx;// +0.5;

//wy = wy;// +wy;// +0.5;

//printf("World coords at z=0.0 are (%f, %f, %f)\n",wx, wy, wz);

}

//For First finger

glPushMatrix();

glColor3f(1, 1, 1);

//glTranslated(122, 3, 0);

glTranslatef(wx, wy, -0.13);//z is 0.4

//if (!(TeapotScalelEnable == 1 && TeapotTranslEnable == 0))//Disable show of Pointer f scaling is ON

{

//cout << "\n\t\*\*\*\*\*\*\*\*\*\*";

glScalef(0.18, 0.22, 0.1);

glutSolidCube(0.05);

glTranslatef(-0.018, 0.038, 0);//index n hand shape

glScalef(0.7, 1.4, 0.1);

glutSolidCube(0.02);

glTranslatef(0.02, -0.008, 0);//2nd 2 index n hand shape

glScalef(0.7, 0.9, 0.1);

glutSolidCube(0.02);

glTranslatef(0.02, -0.005, 0);//2nd 2 index n hand shape

glScalef(0.7, 0.8, 0.1);

glutSolidCube(0.02);

glTranslatef(-0.09, -0.005, 0);//2nd 2 index n hand shape

glScalef(0.7, 0.1, 0.1);

glutSolidCube(0.02);

}

glPopMatrix();//End of First Fingr

glPushMatrix();//Start of second Fingr

glColor3f(1, 0, 1);

glTranslatef(wxForThumb, wyForThumb, -0.13);//z is 0.4

//if (!(TeapotScalelEnable == 1 && TeapotTranslEnable == 0))//Disable show of Pointer f scaling is ON

{ //cout << "\n\t\*\*\*\*\*\*\*\*\*\*";

glScalef(0.18, 0.22, 0.1);

glutSolidCube(0.05);

glTranslatef(-0.018, 0.038, 0);//index n hand shape

glScalef(0.7, 1.4, 0.1);

glutSolidCube(0.02);

glTranslatef(0.02, -0.008, 0);//2nd 2 index n hand shape

glScalef(0.7, 0.9, 0.1);

glutSolidCube(0.02);

glTranslatef(0.02, -0.005, 0);//2nd 2 index n hand shape

glScalef(0.7, 0.8, 0.1);

glutSolidCube(0.02);

glTranslatef(-0.09, -0.005, 0);//2nd 2 index n hand shape

glScalef(0.7, 0.1, 0.1);

glutSolidCube(0.02);

}

glPopMatrix();//End of Second Fing

}

}//End of Pointer

void changeSize(int w, int h)

{

// Prevent a divide by zero, when window is too short

// (you cant make a window of zero width).

if (h == 0)

h = 1;

float ratio = w \* 1.0 / h;

// Use the Projection Matrix

glMatrixMode(GL\_PROJECTION);

// Reset Matrix

glLoadIdentity();

// Set the viewport to be the entire window

glViewport(0, 0, w, h);

// Set the correct perspective.

gluPerspective(45.0f, ratio, 0.1f, 100.0f);

// Get Back to the Modelview

glMatrixMode(GL\_MODELVIEW);

}//End of Reshape/ChangeSize Func

void drawTree()

{

glPushMatrix();

//glColor3f(1.0f, 1.0f, 1.0f);

glColor3f(0, 0.8, 0);

// Draw Body

//glTranslatef(0.0f, 0.75f, 0.0f);

glTranslatef(0.0f, 3.6f, 0.0f);

//glutSolidSphere(0.75f, 20, 20);

glBegin(GL\_TRIANGLES);//green part of tree

glVertex3f(-4, 2, 0);

glVertex3f(-5, -2, 0);

glVertex3f(-3, -2, 0);

glEnd();

glPushMatrix();

glTranslatef(-3.9, -2.4, 0.1);

glColor3f(0.6, 0, 0);//for stem of tree

glScalef(0.4, 1.0, 0);

glutSolidCube(1);

glPopMatrix();

glPopMatrix();

}//End of Snowman Func

void display()

{// Clear Color and Depth Buffers

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

glClearColor(0.50f, 0.50f, 0.90f, 0.0f); // clear to black

// Reset transformations

glLoadIdentity();

//pointer();

//Start Static Scene...................

pointer();

glPushMatrix();

glColor3f(0, 0.6, 0);

//For Rotation Starts......

if (TeapotRotateEnable == 1)//&& SelectEnable == 1)

{

//SSCount = 0;

//TransCount = TransCount + 1;

//Tpx = (wx + wxForThumb) / 2;

//Tpy = (wy + wyForThumb) / 2;

glTranslatef(Tpx, Tpy, Tpz);

glRotatef(DistForRotX, AbtX, AbtY, AbtZ);

glScalef(TpxForScale, TpyForScale, TpzForScale);

//glutSolidTeapot(0.1);

//cout << "\nRotate";// << TransCount;

}

//For Rotatioin Ends.......

if (TeapotTranslEnable == 1)//&& SelectEnable == 1)

{

//SSCount = 0;

//TransCount = TransCount + 1;

Tpx = (wx + wxForThumb) / 2;

Tpy = (wy + wyForThumb) / 2;

glTranslatef(Tpx, Tpy, Tpz);

glScalef(TpxForScale, TpyForScale, TpzForScale);

glRotatef(DistForRotX, AbtX, AbtY, AbtZ);

//glutSolidTeapot(0.1);

//cout << "\nTransCount=" << TransCount;

}

if (TeapotRotateEnable == 0 && TeapotTranslEnable == 0 && TeapotScalelEnable == 0 && MakePositionForScaleFixed == 0)

{

// \_getch();

glTranslatef(Tpx, Tpy, Tpz);// Teapotx=0, Teapty=0.2,0

glScalef(TpxForScale, TpyForScale, TpzForScale);

glRotatef(DistForRotX, AbtX, AbtY, AbtZ);

//glutSolidTeapot(0.1);

//\_getch();

}

if (TeapotScalelEnable == 1 && TeapotTranslEnable == 0)//if Scaling is on

{

//TpxForScale = TpxForScale + 0.1;

//TpxPre = wx;

FixedTpxForScale = (float)Tpx;

FixedTpyForScale = (float)Tpy;

FixedTpzForScale = (float)Tpz;

glTranslatef(FixedTpxForScale, FixedTpyForScale, FixedTpzForScale);

//cout << "x=" << FixedTpxForScale << "y=" << FixedTpxForScale;

glScalef(TpxForScale, TpyForScale, TpyForScale);

glRotatef(DistForRotX, AbtX, AbtY, AbtZ);

//glutSolidTeapot(0.1 );

//TeapotScalelEnable = 0;

}

glutSolidTeapot(0.1);

glPopMatrix();

//Start For Circle...........

if (SelectA == 1)

{

glPushMatrix();

glColor3f(1, 0, 0);

if (SelectAA == 1)

glColor3f(0, 1, 0);

glTranslatef(Tpx, Tpy, Tpz + 0.01);

glutWireCube(0.03);

glPopMatrix();

CountSelectA += 1;

}

if (SelectB == 1)

{

glPushMatrix();

//glColor3f(1, 0, 0);

//if (SelectAA == 1)

glColor3f(0, 1, 0);

glTranslatef(Tpx, Tpy, Tpz + 0.01);

glutWireCube(0.02);

glPopMatrix();

// CountSelectA += 1;

}

//End For Circle.............

if (Detected == 1)//F Index Cap is visible to the camera for text

{ //..Start...For Up Line...

glPushMatrix();

glColor3f(1, 0, 0);

glTranslatef(0, 1.7, -5.9);

//glutSolidCube(0.5);//at Righttt wheel

//glRotatef(65, 0, 1, 0);

glScalef(0.005, 0.0052, 0.0012);

glutStrokeCharacter(GLUT\_STROKE\_ROMAN, 'D');

glutStrokeCharacter(GLUT\_STROKE\_ROMAN, 'E');

glutStrokeCharacter(GLUT\_STROKE\_ROMAN, 'T');

glutStrokeCharacter(GLUT\_STROKE\_ROMAN, 'E');

glutStrokeCharacter(GLUT\_STROKE\_ROMAN, 'C');

glutStrokeCharacter(GLUT\_STROKE\_ROMAN, 'T');

glutStrokeCharacter(GLUT\_STROKE\_ROMAN, 'E');

glutStrokeCharacter(GLUT\_STROKE\_ROMAN, 'D');

glPopMatrix();

if (GateForUp == 1 && Gate4WheelRotate == 0)

{

glPushMatrix();

glTranslatef(-2.5, 0.9, -5.9);

glColor3f(1, 1, 1);

//glutSolidCube(0.5);//at Righttt wheel

//glRotatef(65, 0, 1, 0);

glScalef(0.002, 0.0022, 0.0002);

glutStrokeCharacter(GLUT\_STROKE\_ROMAN, 'U');

glutStrokeCharacter(GLUT\_STROKE\_ROMAN, 'P');

glutStrokeCharacter(GLUT\_STROKE\_ROMAN, ' ');

glutStrokeCharacter(GLUT\_STROKE\_ROMAN, 'G');

glutStrokeCharacter(GLUT\_STROKE\_ROMAN, 'O');

glutStrokeCharacter(GLUT\_STROKE\_ROMAN, 'I');

glutStrokeCharacter(GLUT\_STROKE\_ROMAN, 'N');

glutStrokeCharacter(GLUT\_STROKE\_ROMAN, 'G');

glPopMatrix();

}

if (Gate4WheelRotate == 1)

{

glPushMatrix();

glTranslatef(-2.5, 0.9, -5.9);

glColor3f(1, 1, 1);

//glutSolidCube(0.5);//at Righttt wheel

//glRotatef(65, 0, 1, 0);

glScalef(0.002, 0.0022, 0.0002);

glutStrokeCharacter(GLUT\_STROKE\_ROMAN, 'N');

glutStrokeCharacter(GLUT\_STROKE\_ROMAN, 'A');

glutStrokeCharacter(GLUT\_STROKE\_ROMAN, 'V');

glutStrokeCharacter(GLUT\_STROKE\_ROMAN, 'I');

glutStrokeCharacter(GLUT\_STROKE\_ROMAN, 'G');

glutStrokeCharacter(GLUT\_STROKE\_ROMAN, 'T');

glutStrokeCharacter(GLUT\_STROKE\_ROMAN, 'I');

glutStrokeCharacter(GLUT\_STROKE\_ROMAN, 'O');

glutStrokeCharacter(GLUT\_STROKE\_ROMAN, 'N');

glPopMatrix();

}

if (SelectEnable == 1 || SelectAA == 1)//RightTurn == 1)

{

glPushMatrix();

glTranslatef(-3.5, 2, -5.9);

glColor3f(1, 1, 1);

//glutSolidCube(0.5);//at Righttt wheel

//glRotatef(65, 0, 1, 0);

glScalef(0.002, 0.0022, 0.0002);

glutStrokeCharacter(GLUT\_STROKE\_ROMAN, 'S');

glutStrokeCharacter(GLUT\_STROKE\_ROMAN, 'E');

glutStrokeCharacter(GLUT\_STROKE\_ROMAN, 'L');

glutStrokeCharacter(GLUT\_STROKE\_ROMAN, 'E');

glutStrokeCharacter(GLUT\_STROKE\_ROMAN, 'C');

glutStrokeCharacter(GLUT\_STROKE\_ROMAN, 'T');

//glutStrokeCharacter(GLUT\_STROKE\_ROMAN, ' ');

//glutStrokeCharacter(GLUT\_STROKE\_ROMAN, 'O');

//glutStrokeCharacter(GLUT\_STROKE\_ROMAN, 'N');

glPopMatrix();

}

if (TeapotTranslEnable == 1)//RightTurn == 1)

{

glPushMatrix();

glTranslatef(-2, 2, -5.9);

glColor3f(1, 1, 1);

//glutSolidCube(0.5);//at Righttt wheel

//glRotatef(65, 0, 1, 0);

glScalef(0.002, 0.0022, 0.0002);

glutStrokeCharacter(GLUT\_STROKE\_ROMAN, 'T');

glutStrokeCharacter(GLUT\_STROKE\_ROMAN, 'R');

glutStrokeCharacter(GLUT\_STROKE\_ROMAN, 'A');

glutStrokeCharacter(GLUT\_STROKE\_ROMAN, 'N');

glutStrokeCharacter(GLUT\_STROKE\_ROMAN, 'S');

glutStrokeCharacter(GLUT\_STROKE\_ROMAN, 'L');

glutStrokeCharacter(GLUT\_STROKE\_ROMAN, 'T');

//glutStrokeCharacter(GLUT\_STROKE\_ROMAN, ' ');

//glutStrokeCharacter(GLUT\_STROKE\_ROMAN, 'O');

//glutStrokeCharacter(GLUT\_STROKE\_ROMAN, 'N');

glPopMatrix();

}

if (Gate4Rotate == 1 && (AbtX != 0 || AbtY != 0 || AbtZ != 0))//RightTurn == 1)

{

glPushMatrix();

glTranslatef(-2, 2.2, -5.9);

glColor3f(1, 1, 1);

//glutSolidCube(0.5);//at Righttt wheel

//glRotatef(65, 0, 1, 0);

glScalef(0.002, 0.0022, 0.0002);

glutStrokeCharacter(GLUT\_STROKE\_ROMAN, 'R');

glutStrokeCharacter(GLUT\_STROKE\_ROMAN, 'O');

glutStrokeCharacter(GLUT\_STROKE\_ROMAN, 'T');

glutStrokeCharacter(GLUT\_STROKE\_ROMAN, 'A');

glutStrokeCharacter(GLUT\_STROKE\_ROMAN, 'T');

glutStrokeCharacter(GLUT\_STROKE\_ROMAN, 'E');

//glutStrokeCharacter(GLUT\_STROKE\_ROMAN, 'T');

//glutStrokeCharacter(GLUT\_STROKE\_ROMAN, ' ');

//glutStrokeCharacter(GLUT\_STROKE\_ROMAN, 'O');

//glutStrokeCharacter(GLUT\_STROKE\_ROMAN, 'N');

glPopMatrix();

}

//glPopMatrix();

}

//End Static Scene...............................

// Set the camera

gluLookAt(x, y, z,

x + lx, y + ly, z + lz,

0.0f, 1.0f, 0.0f);

// Draw ground

glColor3f(0.7f, 0.9f, 0.4f);

glBegin(GL\_QUADS);

glVertex3f(-100.0f, 0.0f, -100.0f);

glVertex3f(-100.0f, 0.0f, 100.0f);

glVertex3f(100.0f, 0.0f, 100.0f);

glVertex3f(100.0f, 0.0f, -100.0f);

glEnd();

//for Robo

//glColor3f(1, 1, 0);

//glutSolidCube(1.2);

glColor3f(1, 0, 0.2);

glTranslatef((x + lx), y + ly, (z + lz) + 0.1);//+0.1//0.8

//End for Robo

//Start of New and Perhaps triavial modification.....

glLoadIdentity();

// Set the camera

gluLookAt(x, y, z,

x + lx, y + ly, z + lz,

0.0f, 1.0f, 0.0f);

//End of New modificatns

glPushMatrix();

glTranslatef(8.5, 0, -6);

//wmill();

//glPopMatrix();

glPushMatrix();

for (int r = 2; r <= 4; r++)

{

//for (int c = 3; c >= 0; c--)

//{

// plant();//first plant at leftest

glTranslatef(r, 0, 0);

//plant();//2nd from left

glPushMatrix();

glTranslatef(-2, 0, r + 10);//Front

//plant();//3rd from left

glPopMatrix();

glPushMatrix();

glTranslatef(-5, 0, -(r + 10));//Inside

//plant();//3rd from left

glPopMatrix();

glTranslatef(-10 + r, 0, -(r + 20));//Inside

//plant();//3rd from left

glPopMatrix();

//glTranslatef(2.5, 0, 0);

//plant();//4th from left

//}//End inner loop

}//End outer loop

glPopMatrix();

//glPopMatrix();

// Draw Some Trees

for (int i = -2; i < 1; i++)

{

for (int j = -2; j < 1; j++)

{

glPushMatrix();

glTranslatef(i\*10.0, 0, j \* 10.0);

//drawTree();

glPopMatrix();

}//End inner loop

}//End outer loop

//Start for House

glPushMatrix();

glColor3f(0.2f, 0.2f, 0.2f);

glTranslatef(-1, 2.5, 10);

glTranslatef(-2, -2, 5.1);//for first pillr at left front

glColor3f(0.1, 0.1, 0.1);

glScalef(0.1, 2, 0.1);

glutSolidCube(2);

///glPushMatrix();

//glTranslatef(9, 1, 15);//for first pillr at left front

/\*glColor3f(0.1, 0.1, 0.1);

glScalef(0.1, 2, 0.1);\*/

glTranslatef(87, 0, 0.0);//2nd pillr at front right

glutSolidCube(2);

glTranslatef(0, 0, -70.0);//2nd pillr at front right

glutSolidCube(2);

glTranslatef(-80, 0, 0.0);//2nd pillr at front right

glutSolidCube(2);

glPopMatrix();

//End of Chat i.e Ground say first floor

//Start for chat second paer above the first flor

glPushMatrix();

glColor3f(0.2f, 0.2f, 0.2f);

glTranslatef(-1, 4, 10);

glBegin(GL\_QUADS);//Chat

glVertex3f(-3.0f, 0.0f, -7.0f);

glVertex3f(-3.0f, 0.0f, 7.0f);

glVertex3f(9.0f, 0.0f, 7.0f);

glVertex3f(9.0f, 0.0f, -7.0f);

glEnd();

glTranslatef(-2, -2, 5.1);//for first pillr at left front

glColor3f(0.1, 0.1, 0.1);

glScalef(0.1, 2, 0.1);

glutSolidCube(2);

glTranslatef(87, 0, 0.0);//2nd pillr at front right

glutSolidCube(2);

glTranslatef(0, 0, -70.0);//2nd pillr at front right

glutSolidCube(2);

glTranslatef(-80, 0, 0.0);//2nd pillr at front right

glutSolidCube(2);

glPopMatrix();

//End of Chat Upper portion second part

//222222222222222222222 Start of Table..

//Start for table at Right

//Start for table

glPushMatrix();

glColor3f(0.2f, 0.2f, 0.2f);

glTranslatef(5.5, 0.9, 30);

glScalef(0.51, 0.03, 0.4);

glutSolidCube(2);

//Below is table Leg

glPopMatrix();

glPushMatrix();

glTranslatef(5.6, 0.3, 30);//for first pillr at left front

glColor3f(0.1, 0.1, 0.1);

glScalef(0.04, 0.58, 0.1);

glutSolidCube(2);

glPopMatrix();//End of Table leg

//End of table at Right

//Start for table at left

//Start for table

glPushMatrix();

glColor3f(0.2f, 0.2f, 0.2f);

glTranslatef(-1, 0.9, 34);

glScalef(0.51, 0.03, 0.4);

glutSolidCube(2);

//Below is table Leg

glPopMatrix();

glPushMatrix();

glTranslatef(-1.1, 0.3, 34);//for first pillr at left front

glColor3f(0.1, 0.1, 0.1);

glScalef(0.04, 0.58, 0.1);

glutSolidCube(2);

glPopMatrix();//End of Table leg

//End of table

glPopMatrix();

//End of Table at left

//222222222222222222222 End of Table..

glPushMatrix();//For straight road

glColor3f(0.5f, 0.2f, 0.2f);

glTranslatef(-1, 2.5, 10);

glBegin(GL\_QUADS);//Road

glVertex3f(0.0f, -2.4f, -55.0f);

glVertex3f(0.0f, -2.4f, 25.0f);

glVertex3f(5.0f, -2.4f, 25.0f);

glVertex3f(5.0f, -2.4f, -55.0f);

glEnd();

glPopMatrix();//End of Straight Road

glPushMatrix();//For Stop board start

glColor3f(1.0f, 0.0f, 0.0f);

glTranslatef(1.3, 1.5, -40);

glScalef(1.9, 0.9, 0);

glutSolidCube(1);

glBegin(GL\_LINE\_STRIP);//For board leg/bar

glVertex3f(0, 0, 0);

glVertex3f(0, -2, 0);

glEnd();

glPopMatrix();//End of Stop board

glPushMatrix();

glTranslatef(0.5, 1.4, -39.5);

glColor3f(1.0f, 1.0f, 1.0f);

glScalef(0.0048, 0.0043, 0.0022);

glutStrokeCharacter(GLUT\_STROKE\_ROMAN, 'S');

glutStrokeCharacter(GLUT\_STROKE\_ROMAN, 'T');

glutStrokeCharacter(GLUT\_STROKE\_ROMAN, 'O');

glutStrokeCharacter(GLUT\_STROKE\_ROMAN, 'P');

glPopMatrix();

//glPopMatrix();//End of Stop board

glPushMatrix();//Start For Turn road at last negtive z axis

glColor3f(0.5f, 0.2f, 0.2f);

glTranslatef(3, 2.5, -28);

glBegin(GL\_QUADS);//Road

glVertex3f(1.0f, -2.4f, 27.0f);

glVertex3f(1.0f, -2.4f, 16.0f);

glVertex3f(7.0f, -2.4f, 16.0f);

glVertex3f(7.0f, -2.4f, 27.0f);

glEnd();

glPopMatrix();//End of turn road at last at negtv z axis

//Start for arrows

glPushMatrix();//Start for 1st arrow at Right

glColor3f(1, 0, 0);

glTranslatef(5, 0.1, 25);

//glRotatef(65, 0, 1, 0);

glScalef(0.0037, 0.0022, 0.0022);

glutStrokeCharacter(GLUT\_STROKE\_ROMAN, '-');

glutStrokeCharacter(GLUT\_STROKE\_ROMAN, '>');

glPopMatrix();//End of 1st arrow at rith

glPushMatrix();//start of 2nd arrowhed at right

glTranslatef(9, 0.1, 20);

glScalef(0.0037, 0.0022, 0.0022);

//glRotatef(45, 1, 0, 0);

glutStrokeCharacter(GLUT\_STROKE\_ROMAN, '^');

glPopMatrix();//End of 2nd arrowhed at right

//Start for arrow at Left

glPushMatrix();//Start for arrow at Left

glColor3f(1, 0, 0);

glTranslatef(-2.5, 0.3, 25);

//glRotatef(65, 0, 1, 0);

glScalef(0.0037, 0.0022, 0.0022);

glutStrokeCharacter(GLUT\_STROKE\_ROMAN, '<');

glutStrokeCharacter(GLUT\_STROKE\_ROMAN, '-');

glPopMatrix();//End of 1st arrow at left

glPushMatrix();//start of 2nd arrowhed at leftt

glTranslatef(-9, 0.1, 5);

glScalef(0.0037, 0.0022, 0.0022);

//glRotatef(45, 1, 0, 0);

glutStrokeCharacter(GLUT\_STROKE\_ROMAN, '^');

glPopMatrix();//End of 2nd arrowhed at left

glutSwapBuffers();

//glColor3f(1, 1, 0);

//glutSolidCube(1.2);

glutPostRedisplay();

//Gate4WheelRotate = 0;

//RightTurn = 0;

}

void idle()

{ // grab a frame from the camera

//(\*cap) >> image;

cap.retrieve(image);

//Sleep(100);

bool bSuccess = cap.read(image);

//image2 = image;

detectImg(image);

}

void keyboard(unsigned char key, int xx, int yy)

{

float fraction = 0.1f;// for camera ratio

if (key == 'r' || key == ' ')

{

//x += lx \* 2 \* fraction;//for going inside screen

//z += lz \* 2 \* fraction;

reset();

Beep(500, 200);

}

if (key == 'n')

{

x += 4 \* lx \* fraction;//for going inside screen

z += 4 \* lz \* fraction;

//reset();

//Beep(500, 900);

}

}

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

{

glutInit(&argc, argv);

glutInitDisplayMode(GLUT\_DOUBLE | GLUT\_RGB); // Setup display mode to double buffer and RGB color

//glutInitWindowSize(640, 480);

glutInitWindowPosition(10, 10);

glutInitWindowSize(1400, 900); // Set the screen size

glutCreateWindow("Opengl Window");

glutDisplayFunc(display);

glutReshapeFunc(changeSize);

//glutReshapeFunc(reshape);

//glutMouseFunc(mouse);

glutKeyboardFunc(keyboard);

glutIdleFunc(idle);

glEnable(GL\_DEPTH\_TEST);

// start GUI loop

glutMainLoop();

return 0;

}

//999999999999999999999999999999999999999999999999999999999999999999999999999999999999999//99999999999