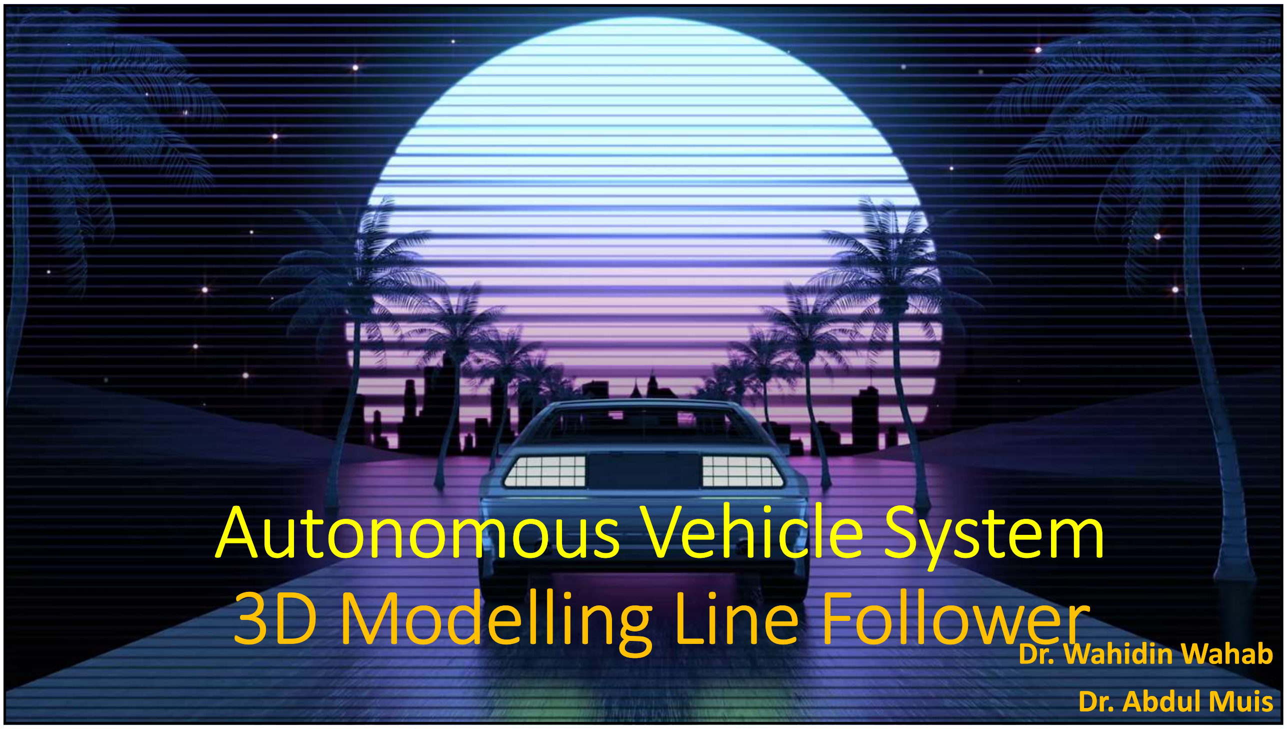
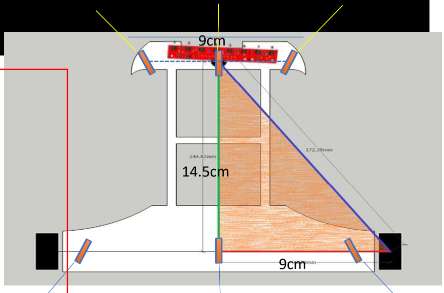


# Autonomous Vehicle System
# 3D Modelling Line Follower

Dr. Wahidin Wahab
Dr. Abdul Muis

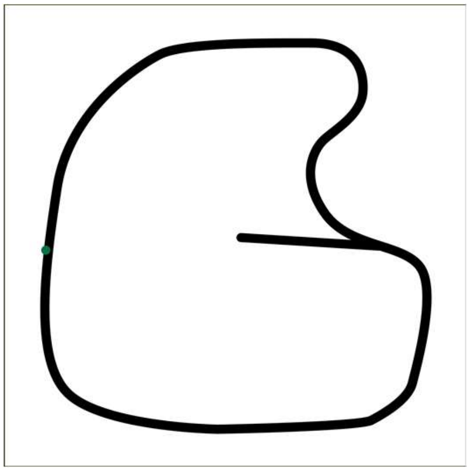
## Last Exercise

- Draw a mobile robot with two wheel
  - Mobile robot (on the right)
  - Ground clearance 2cm
  - Wheel diameter 2.5cm
  - Wheel thickness 1cm
- Draw rectangle as line sensor with length 10cm x 2cm
- Draw 6 laser range sensor on the top of mobile robot
  - with 3 on the front and 3 on the back.
  - The front corner is 4cm from middle, while the back corner is about 8cm from middle
  - the sensor at corner has 30degree orientation from the middle
  - draw a line to illustrate the measured distance of those sensor
- Animate mobile robot path trajectory with linear motion
  - go forward 1m and then turn right and then go forward 1m
  - go straight right-forward with angle 30 degree
  - go right-forward on slope with radius 2m

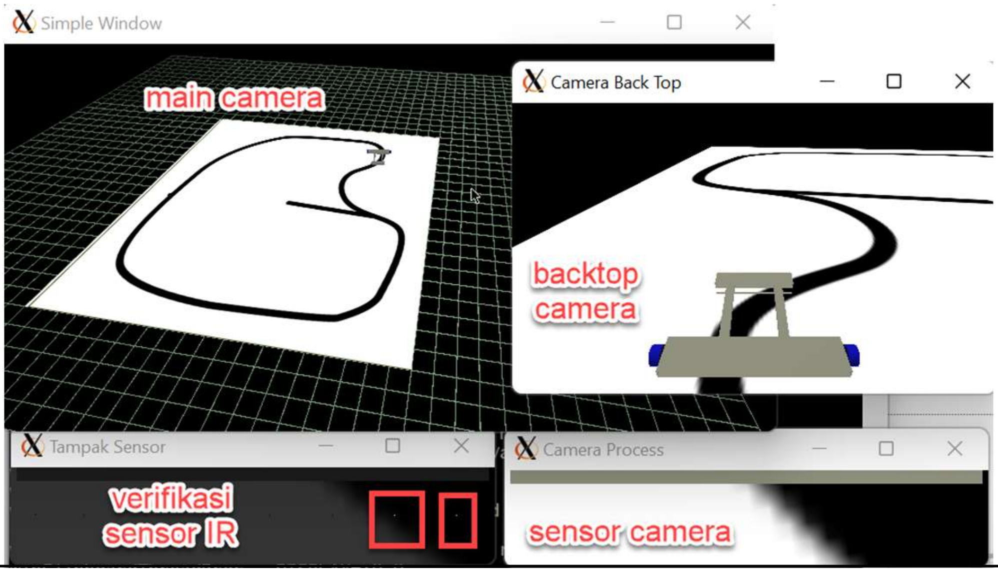


# Persyaratan lain

- Buat track dengan inkscape/krita 500x500pixels save sebagai ppm



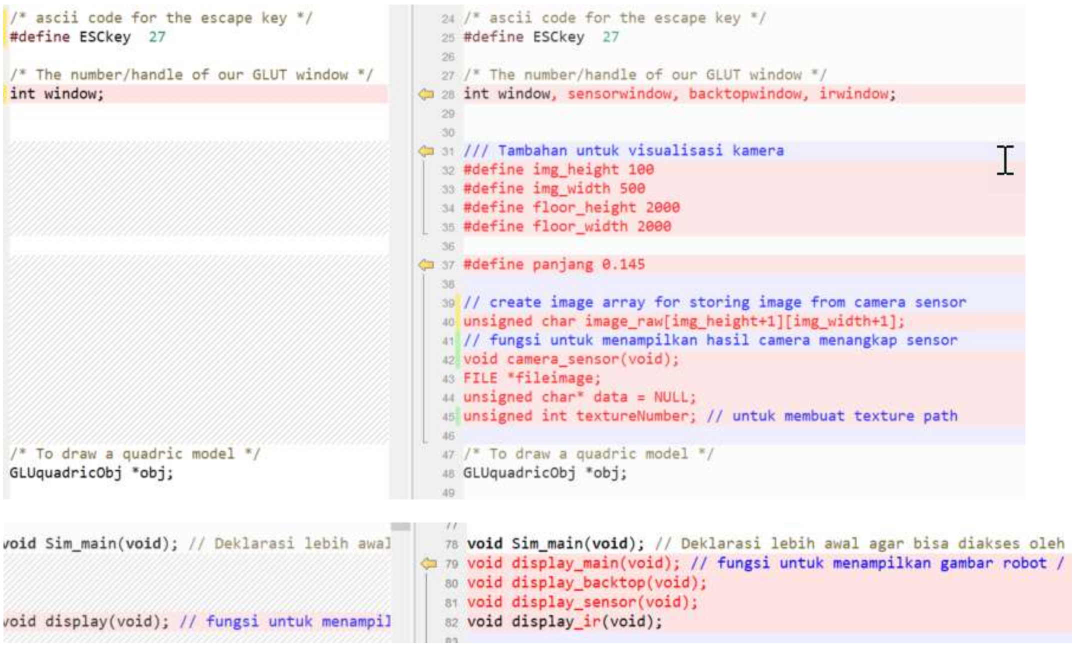
# Kebutuhan Window



## Step 1: Create sensor’s camera (camera_sensor) and result (window_sensor)

```
#define img_height 100
#define img_width 500

// declare variables
Int window, sensorwindow, irwindow;

// create image array for storing image from camera sensor
unsigned char image_raw[img_height+1][img_width+1];
// fungsi untuk menampilkan hasil camera menangkap sensor
void camera_sensor(void);
FILE *fileimage;
unsigned char* data = NULL;
unsigned int textureNumber; // untuk membuat texture path

void Sim_main(void); // Deklarasi lebih awal agar bisa diakses oleh fungsi sebelumnya
void display_main(void); // fungsi untuk menampilkan gambar awal (sebelumnya display())
void display_sensor(void); // fungsi untuk membuat emulasi sensor window / camera sensor
void display_ir(void); // fungsi untuk menampilkan hasil IR
```

5

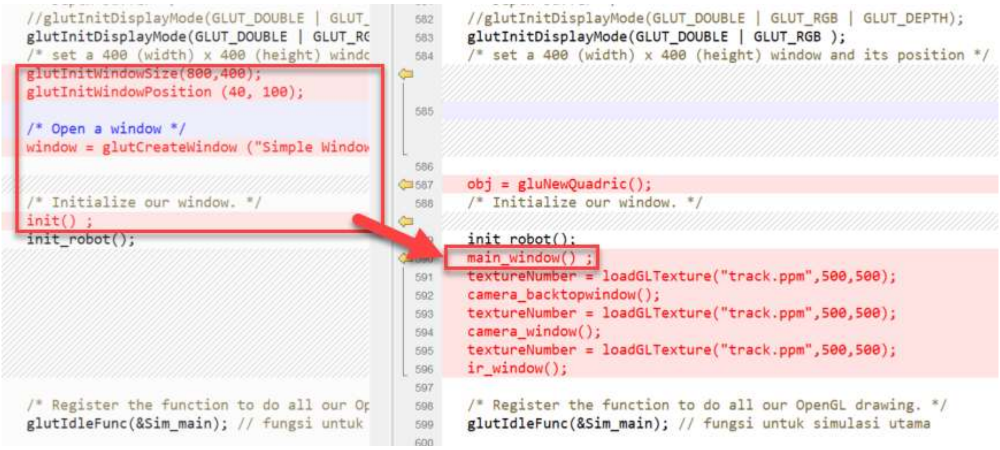


```
/* ascii code for the escape key */
#define ESCkey  27

/* The number/handle of our GLUT window */
int window;

/* To draw a quadric model */
GLUquadricObj *obj;

void Sim_main(void); // Deklarasi lebih awal

void display(void); // fungsi untuk menampil
```

```
/* ascii code for the escape key */
#define ESCkey  27

/* The number/handle of our GLUT window */
int window, sensorwindow, backtopwindow, irwindow;


/// Tambahan untuk visualisasi kamera
#define img_height 100
#define img_width 500
#define floor_height 2000
#define floor_width 2000

#define panjang 0.145

// create image array for storing image from camera sensor
unsigned char image_raw[img_height+1][img_width+1];
// fungsi untuk menampilkan hasil camera menangkap sensor
void camera_sensor(void);
FILE *fileimage;
unsigned char* data = NULL;
unsigned int textureNumber; // untuk membuat texture path

/* To draw a quadric model */
GLUquadricObj *obj;

void Sim_main(void); // Deklarasi lebih awal agar bisa diakses oleh
void display_main(void); // fungsi untuk menampilkan gambar robot /
void display_backtop(void);
void display_sensor(void);
void display_ir(void);
```

6

## Setup Main

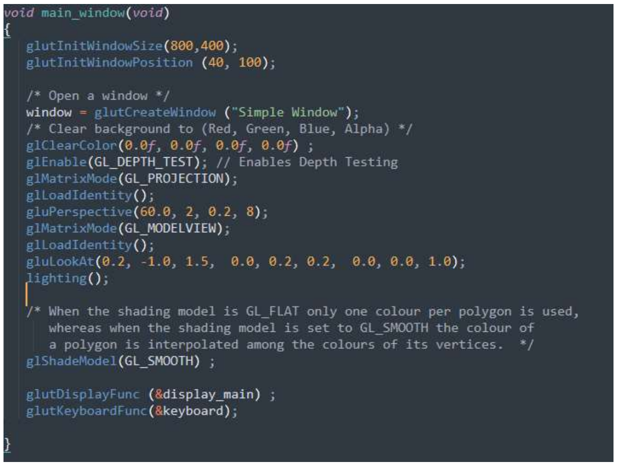


```
//glutInitDisplayMode(GLUT_DOUBLE | GLUT_
glutInitDisplayMode(GLUT_DOUBLE | GLUT_R(
/* set a 400 (width) x 400 (height) windc
glutInitWindowSize(800,400);
glutInitWindowPosition (40, 100);

/* Open a window */
window = glutCreateWindow ("Simple Windov


/* Initialize our window. */
init() ;
init_robot();


/* Register the function to do all our Op
glutIdleFunc(&Sim_main); // fungsi untuk
```

```
//glutInitDisplayMode(GLUT_DOUBLE | GLUT_RGB | GLUT_DEPTH);
glutInitDisplayMode(GLUT_DOUBLE | GLUT_RGB );
/* set a 400 (width) x 400 (height) window and its position */


obj = gluNewQuadric();
/* Initialize our window. */

init_robot();
main_window() ;
textureNumber = loadGLTexture("track.ppm",500,500);
camera_backtopwindow();
textureNumber = loadGLTexture("track.ppm",500,500);
camera_window();
textureNumber = loadGLTexture("track.ppm",500,500);
ir_window();

/* Register the function to do all our OpenGL drawing. */
glutIdleFunc(&Sim_main); // fungsi untuk simulasi utama
```

## Main Window

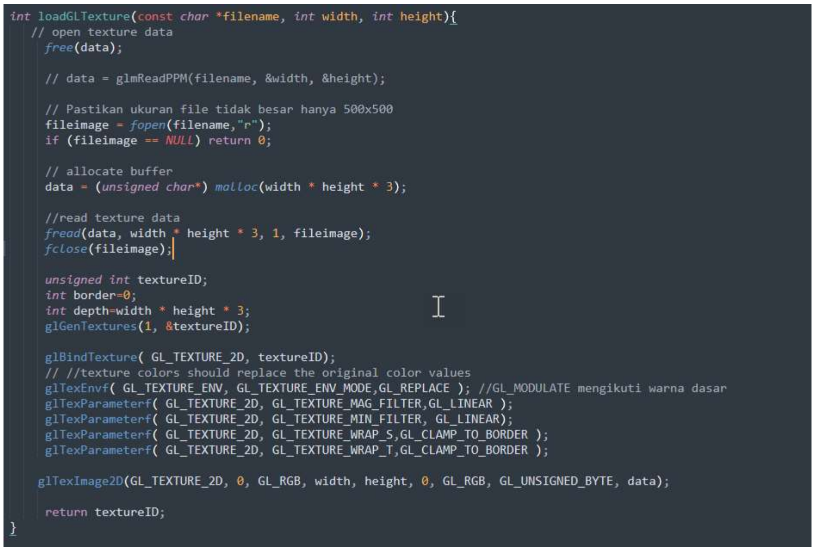
- The main window is formed in a function main_window which declare the window size and position
- The camera view and initialization which are stated in init() are embedded in this function
    - gluPerspective
    - gluLookAt



```
void main_window(void)
{
    glutInitWindowSize(800,400);
    glutInitWindowPosition (40, 100);

    /* Open a window */
    window = glutCreateWindow ("Simple Window");
    /* Clear background to (Red, Green, Blue, Alpha) */
    glClearColor(0.0f, 0.0f, 0.0f, 0.0f) ;
    glEnable(GL_DEPTH_TEST); // Enables Depth Testing
    glMatrixMode(GL_PROJECTION);
    glLoadIdentity();
    gluPerspective(60.0, 2, 0.2, 8);
    glMatrixMode(GL_MODELVIEW);
    glLoadIdentity();
    gluLookAt(0.2, -1.0, 1.5,  0.0, 0.2, 0.2,  0.0, 0.0, 1.0);
    lighting();

    /* When the shading model is GL_FLAT only one colour per polygon is used,
       whereas when the shading model is set to GL_SMOOTH the colour of
       a polygon is interpolated among the colours of its vertices.  */
    glShadeModel(GL_SMOOTH) ;

    glutDisplayFunc (&display_main) ;
    glutKeyboardFunc(&keyboard);

}
```

## Loading Texture

- The texture comes from *.ppm which has 3 channel (R,G,B). The picture is applied to image2D
- The function return texture handleid (textureNumber)

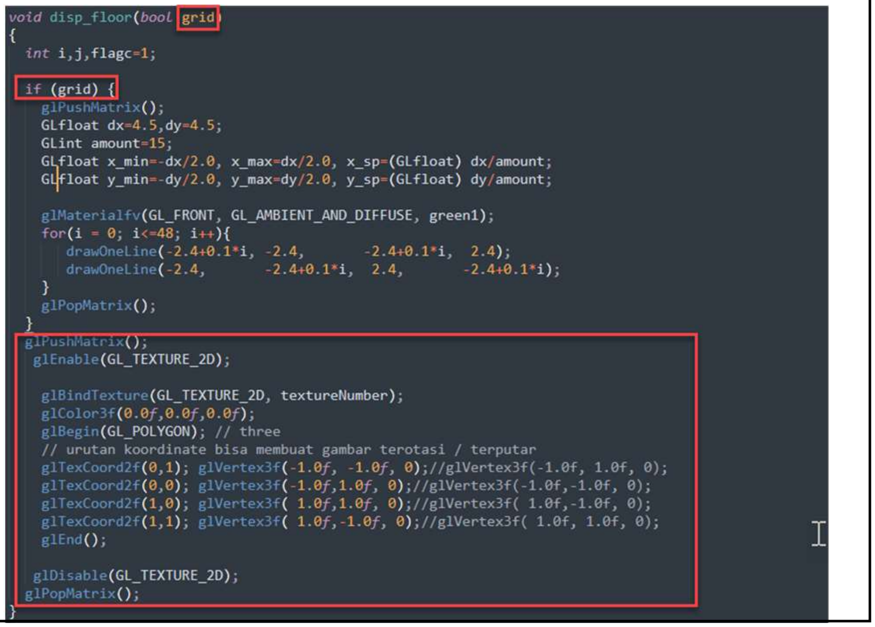


```
int loadGLTexture(const char *filename, int width, int height){
   // open texture data
     free(data);

     // data = glmReadPPM(filename, &width, &height);

     // Pastikan ukuran file tidak besar hanya 500x500
     fileimage = fopen(filename,"r");
     if (fileimage == NULL) return 0;

     // allocate buffer
     data = (unsigned char*) malloc(width * height * 3);

     //read texture data
     fread(data, width * height * 3, 1, fileimage);
     fclose(fileimage);

     unsigned int textureID;
     int border=0;
     int depth=width * height * 3;
     glGenTextures(1, &textureID);

     glBindTexture( GL_TEXTURE_2D, textureID);
     // //texture colors should replace the original color values
     glTexEnvf( GL_TEXTURE_ENV, GL_TEXTURE_ENV_MODE,GL_REPLACE ); //GL_MODULATE mengikuti warna dasar
     glTexParameterf( GL_TEXTURE_2D, GL_TEXTURE_MAG_FILTER,GL_LINEAR );
     glTexParameterf( GL_TEXTURE_2D, GL_TEXTURE_MIN_FILTER, GL_LINEAR);
     glTexParameterf( GL_TEXTURE_2D, GL_TEXTURE_WRAP_S,GL_CLAMP_TO_BORDER );
     glTexParameterf( GL_TEXTURE_2D, GL_TEXTURE_WRAP_T,GL_CLAMP_TO_BORDER );

    glTexImage2D(GL_TEXTURE_2D, 0, GL_RGB, width, height, 0, GL_RGB, GL_UNSIGNED_BYTE, data);

     return textureID;
}
```

9

## Drawing floor with additional texture

- We add additional grid variable so that we can show/hide floor grid
- The texture from track.ppm is loaded based on texture handleid (textureNumber)

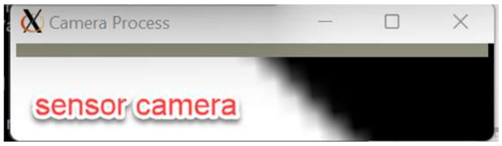


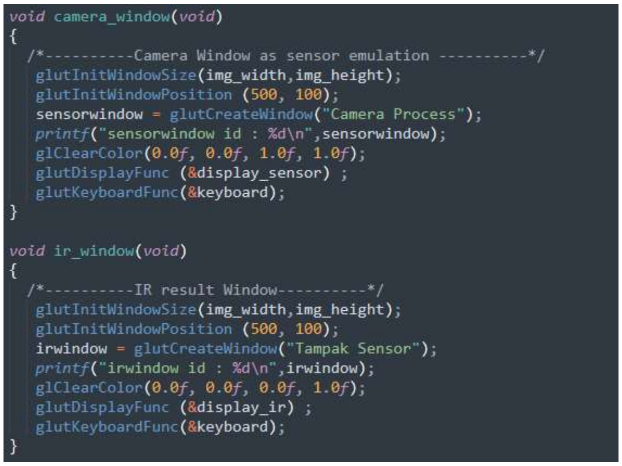
```
void disp_floor(bool grid)
{
  int i,j,flagc=1;

  if (grid) {
    glPushMatrix();
    GLfloat dx=4.5,dy=4.5;
    GLint amount=15;
    GLfloat x_min=-dx/2.0, x_max=dx/2.0, x_sp=(GLfloat) dx/amount;
    GLfloat y_min=-dy/2.0, y_max=dy/2.0, y_sp=(GLfloat) dy/amount;

    glMaterialfv(GL_FRONT, GL_AMBIENT_AND_DIFFUSE, green1);
    for(i = 0; i<=48; i++){
       drawOneLine(-2.4+0.1*i, -2.4,       -2.4+0.1*i,  2.4);
       drawOneLine(-2.4,       -2.4+0.1*i,  2.4,       -2.4+0.1*i);
    }
    glPopMatrix();
  }
  glPushMatrix();
   glEnable(GL_TEXTURE_2D);

    glBindTexture(GL_TEXTURE_2D, textureNumber);
    glColor3f(0.0f,0.0f,0.0f);
    glBegin(GL_POLYGON); // three
    // urutan koordinate bisa membuat gambar terotasi / terputar
    glTexCoord2f(0,1); glVertex3f(-1.0f, -1.0f, 0);//glVertex3f(-1.0f, 1.0f, 0);
    glTexCoord2f(0,0); glVertex3f(-1.0f,1.0f, 0);//glVertex3f(-1.0f,-1.0f, 0);
    glTexCoord2f(1,0); glVertex3f( 1.0f,1.0f, 0);//glVertex3f( 1.0f,-1.0f, 0);
    glTexCoord2f(1,1); glVertex3f( 1.0f,-1.0f, 0);//glVertex3f( 1.0f, 1.0f, 0);
    glEnd();

   glDisable(GL_TEXTURE_2D);
  glPopMatrix();
}
```

10

## Step 2: Setup Sensor : camera window & IR window

- Dipanggil di main program (main) dan di setiap loop (sim_main)

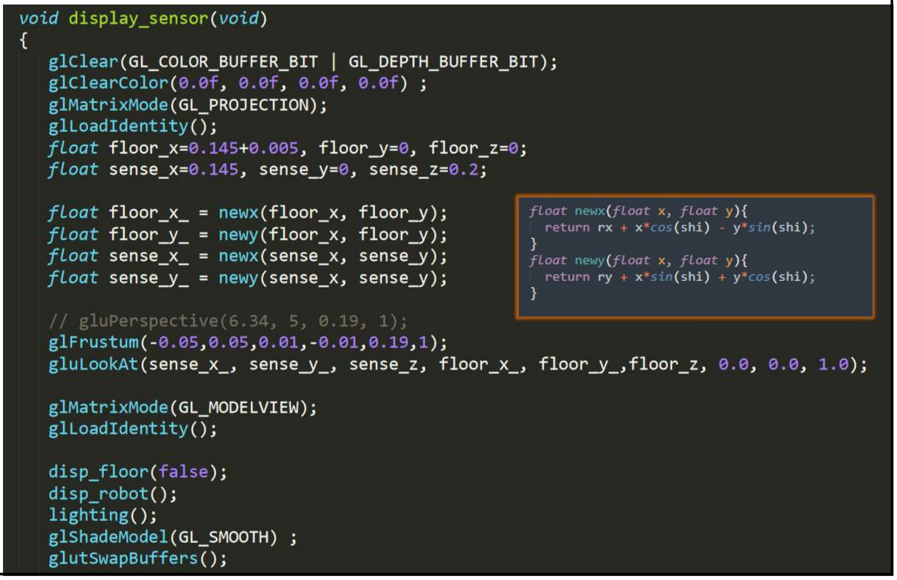




```
void camera_window(void)
{
  /*----------Camera Window as sensor emulation ----------*/
   glutInitWindowSize(img_width,img_height);
   glutInitWindowPosition (500, 100);
   sensorwindow = glutCreateWindow("Camera Process");
   printf("sensorwindow id : %d\n",sensorwindow);
   glClearColor(0.0f, 0.0f, 1.0f, 1.0f);
   glutDisplayFunc (&display_sensor) ;
   glutKeyboardFunc(&keyboard);
}

void ir_window(void)
{
  /*----------IR result Window----------*/
   glutInitWindowSize(img_width,img_height);
   glutInitWindowPosition (500, 100);
   irwindow = glutCreateWindow("Tampak Sensor");
   printf("irwindow id : %d\n",irwindow);
   glClearColor(0.0f, 0.0f, 0.0f, 1.0f);
   glutDisplayFunc (&display_ir) ;
   glutKeyboardFunc(&keyboard);
}
```

## 3. Setup Camera Sensor #1

- Di panggil di setiap loop (Sim_main)
- Koordinat kamera (sense) dan titik focus kamera (floor) terhadap base robot harus di definisikan, selanjutnya di konversi terhadap coordinate world yang digunakan sebagai parameter gluLookAt dalam mode GL_PROJECTION
- Menggambar model floor, robot dan lighting
- Swap buffer untuk ditampilkan

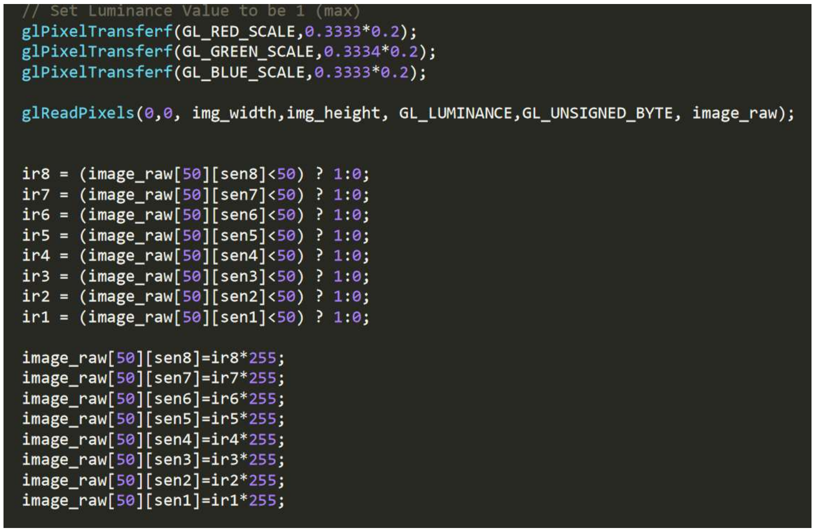


```
void display_sensor(void)
{
    glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);
    glClearColor(0.0f, 0.0f, 0.0f, 0.0f) ;
    glMatrixMode(GL_PROJECTION);
    glLoadIdentity();
    float floor_x=0.145+0.005, floor_y=0, floor_z=0;
    float sense_x=0.145, sense_y=0, sense_z=0.2;

    float floor_x_ = newx(floor_x, floor_y);
    float floor_y_ = newy(floor_x, floor_y);
    float sense_x_ = newx(sense_x, sense_y);
    float sense_y_ = newy(sense_x, sense_y);

    // gluPerspective(6.34, 5, 0.19, 1);
    glFrustum(-0.05,0.05,0.01,-0.01,0.19,1);
    gluLookAt(sense_x_, sense_y_, sense_z, floor_x_, floor_y_,floor_z, 0.0, 0.0, 1.0);

    glMatrixMode(GL_MODELVIEW);
    glLoadIdentity();

    disp_floor(false);
    disp_robot();
    lighting();
    glShadeModel(GL_SMOOTH) ;
    glutSwapBuffers();
```

```
float newx(float x, float y){
  return rx + x*cos(shi) - y*sin(shi);
}
float newy(float x, float y){
  return ry + x*sin(shi) + y*cos(shi);
}
```

## 3. Setup Camera Sensor #2

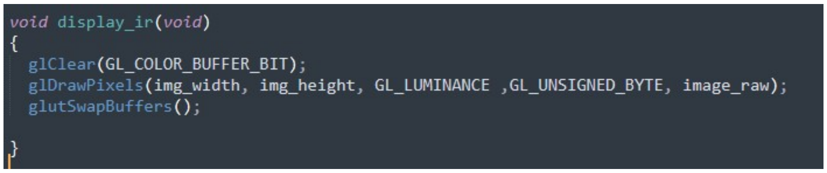
- Buat sebuah gambar Grayscale **image_raw** dimana tiap channel (warna) punya 1/3 kontribusi. Dan untuk membuat gambar agak gelap bisa dibuat setara 20%nya
- Deteksi keberadaan garis hitam (intensitas 0 atau boleh threshold <50)
- Untuk klarifikasi, beri tanda lokasi sensor yang dibaca diganti warna menjadi putih terang (255)



```
// Set Luminance Value to be 1 (max)
glPixelTransferf(GL_RED_SCALE,0.3333*0.2);
glPixelTransferf(GL_GREEN_SCALE,0.3334*0.2);
glPixelTransferf(GL_BLUE_SCALE,0.3333*0.2);

glReadPixels(0,0, img_width,img_height, GL_LUMINANCE,GL_UNSIGNED_BYTE, image_raw);


ir8 = (image_raw[50][sen8]<50) ? 1:0;
ir7 = (image_raw[50][sen7]<50) ? 1:0;
ir6 = (image_raw[50][sen6]<50) ? 1:0;
ir5 = (image_raw[50][sen5]<50) ? 1:0;
ir4 = (image_raw[50][sen4]<50) ? 1:0;
ir3 = (image_raw[50][sen3]<50) ? 1:0;
ir2 = (image_raw[50][sen2]<50) ? 1:0;
ir1 = (image_raw[50][sen1]<50) ? 1:0;

image_raw[50][sen8]=ir8*255;
image_raw[50][sen7]=ir7*255;
image_raw[50][sen6]=ir6*255;
image_raw[50][sen5]=ir5*255;
image_raw[50][sen4]=ir4*255;
image_raw[50][sen3]=ir3*255;
image_raw[50][sen2]=ir2*255;
image_raw[50][sen1]=ir1*255;
```

## Display IR

Draw Pixels from Display Sensor



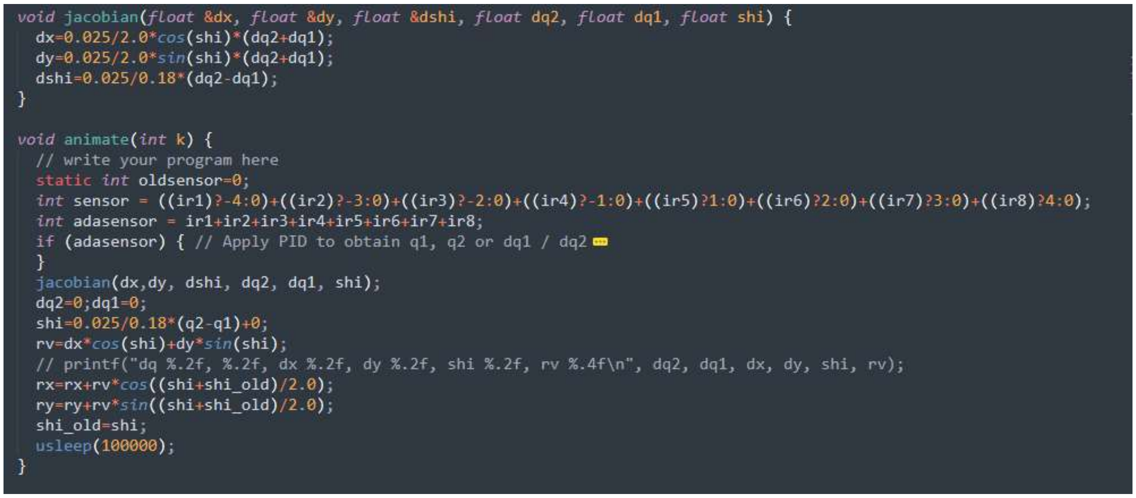
```
void display_ir(void)
{
  glClear(GL_COLOR_BUFFER_BIT);
  glDrawPixels(img_width, img_height, GL_LUMINANCE ,GL_UNSIGNED_BYTE, image_raw);
  glutSwapBuffers();

}
```

# Loop

- static int count=0;
- glutSetWindow(window);
- animate(count); // control robot
- display_main();
- glutSetWindow(backtopwindow);
- display_backtop();
- glutSetWindow(sensorwindow);
- display_sensor();
- glutSetWindow(irwindow);
- display_ir();
- usleep(xxx); // delay

15

# Animate Robot



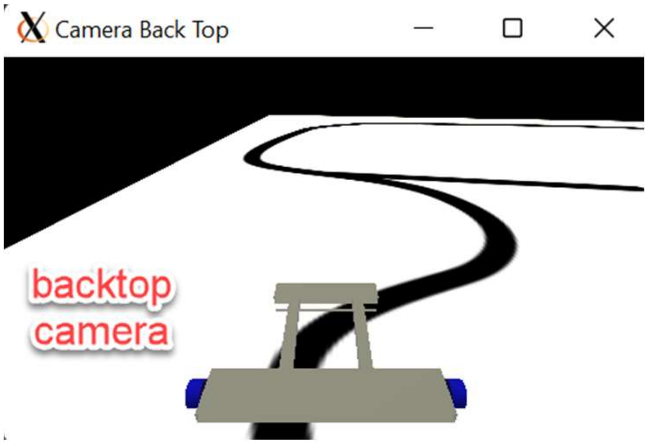
```
void jacobian(float &dx, float &dy, float &dshi, float dq2, float dq1, float shi) {
  dx=0.025/2.0*cos(shi)*(dq2+dq1);
  dy=0.025/2.0*sin(shi)*(dq2+dq1);
  dshi=0.025/0.18*(dq2-dq1);
}

void animate(int k) {
  // write your program here
  static int oldsensor=0;
  int sensor = ((ir1)?-4:0)+((ir2)?-3:0)+((ir3)?-2:0)+((ir4)?-1:0)+((ir5)?1:0)+((ir6)?2:0)+((ir7)?3:0)+((ir8)?4:0);
  int adasensor = ir1+ir2+ir3+ir4+ir5+ir6+ir7+ir8;
  if (adasensor) { // Apply PID to obtain q1, q2 or dq1 / dq2 ⋯
  }
  jacobian(dx,dy, dshi, dq2, dq1, shi);
  dq2=0;dq1=0;
  shi=0.025/0.18*(q2-q1)+0;
  rv=dx*cos(shi)+dy*sin(shi);
  // printf("dq %.2f, %.2f, dx %.2f, dy %.2f, shi %.2f, rv %.4f\n", dq2, dq1, dx, dy, shi, rv);
  rx=rx+rv*cos((shi+shi_old)/2.0);
  ry=ry+rv*sin((shi+shi_old)/2.0);
  shi_old=shi;
  usleep(100000);
}
```

16

# Back-top Camera

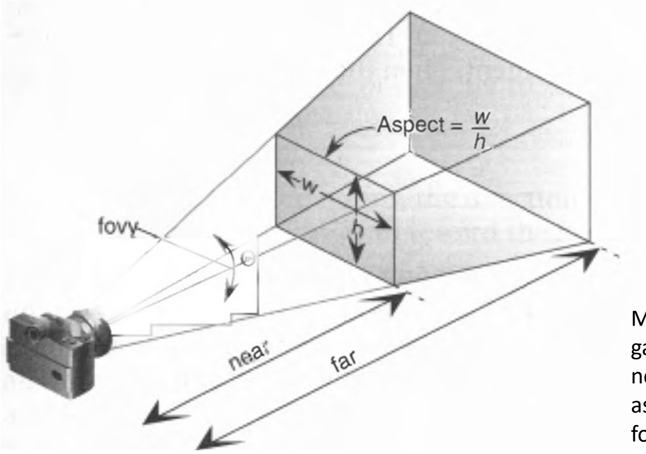
Koordinat kamera terhadap Robot Base
float sense_x=-0.2, sense_y=0, sense_z=0.3;
float floor_x=0.3, floor_y=0.0, floor_z=0.0;

Konversikan ke koordinate world



17

# gluPerspective

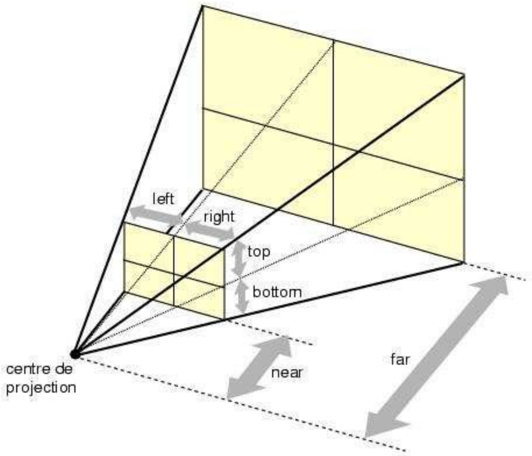


void **gluPerspective**( GLdouble *fovy* in y direction,
GLdouble *aspect*,
GLdouble *zNear*,
GLdouble *zFar*);

Mis : kamera 20cm diatas lantai dan ingin seluruh size gambar (500x100pixel) = 10x2cm di lantai, maka
near=0.19
aspect = 500/100 = 5
fovy = atan(0.02/0.19) =6.34 derajat

18

# glFrustum

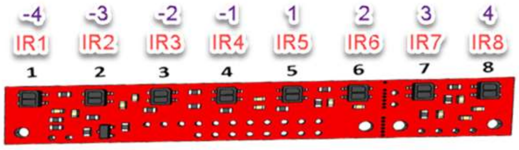


*) Vincent Nozick, "Video-Based Rendering and Virtual Reality", 2006

```
void glFrustum(    GLdouble left,
                   GLdouble right,
                   GLdouble bottom,
                   GLdouble top,
                   GLdouble nearVal,
                   GLdouble farVal);
```

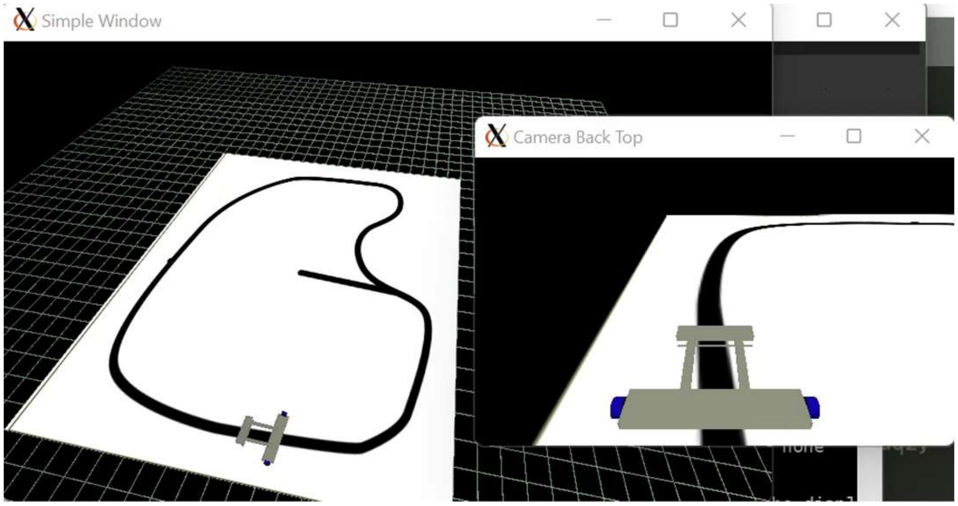
Mis : kamera 20cm diatas lantai dan ingin seluruh size gambar (500x100pixel) = 10x2cm = 0.1x0.02m di lantai, maka
near=0.19,
left = - 0.1/2 = -0.05
right = + 0.1/2 = 0.05
top = 0.02 / 2 = 0.01
bottom = - 0.02 / 2 = -0.01

# Konsep Pembacaan Sensor



- Bisa berdasarkan status sensor
  - Jika sensor terbaca di tengah (sum==0) maka roda kiri dan kanan maju
  - Jika sensor terbaca di kanan (sum > 0) maka roda kiri maju
  - Jika sensor terbaca di kiri (sum < 0) maka roda kanan maju

# Hasil (dengan delay 100ms)



Dari status