Санкт-Петербургский политехнический университет Петра Великого
Институт машиностроения, материалов и транспорта
Высшая школа автоматизации и робототехники

Отчёт

по лабораторной работе №1

Дисциплина: Техническое зрение

Тема: Моделирование движения робота с использованием библиотеки OpenCV

Студент гр. 3331506/70401 Кученов Д.С.

Преподаватель Варлашин В.В.

« »_____2020 г.

Санкт-Петербург

2020

Задание

Необходимо с помощью OpenCV реализовать движение роботелеги по заданному полю, его поворот на месте, а также учитывать выезд за пределы поля.

Ход работы

Общий алгоритм программы:

1) Ожидание нажатия кнопки;

2) Вычисление координат всех точек роботелеги, в зависимости от того, какая кнопка нажата;

3) Проверка на пересечение роботом границ заданного поля;

4) Вывод нового изображения робота на экран.

Весь процесс находится в цикле *while()*, пока не будет нажата клавиша «*esc*», после чего выполнение будет завершено.

Для того, чтобы на экране не отображалось предыдущее изображение робота, создан отдельный объект класса Mat — общий фон. Главное изображение становится клоном общего фона, таким образом стирается предыдущее изображение робота, а затем уже на пустом экране рисуется следующее положение роботелеги.

Класс *MyRobot*

Для создания робота был создан класс *MyRobot.* Данный класс обладает параметрами, показанными на рисунке 1.

```
float m_speed;
float m_angularSpeed;
cv::Point2f m_center;
cv::Size2i m_area;
float m_angle;
float m_width;
float m_height;
float m_wheelWidth;
float m_wheelDiameter;
float m_rotation;
```

Рисунок 1 — Параметры класса

Координаты центра устанавливаются с помощью функции *setCenter(Mat image)*, которая в начальный момент времени устанавливает центр робота в центр заданного поля. Эта функция так же помогает переместить роботелегу на противоположную сторону поля при пересечении границы.

Область движения устанавливается с помощью функции *setArea(Mat image)*.

Все остальные параметры устанавливаются при вызове конструктора.

Основные функции

1) Функция *move()*

Сканирование нажатия клавиш продольного и поперечного движений и одновременно с этим вычисление новых координат центра робота осуществлено в функции *move()*.

При нажатии какой-либо из клавиш вычисляются новые координаты для центра роботелеги.

Функция *move()* представлена на рисунке 2:

```
void MyRobot::move(float x, float y)
{
    m_center.x = m_center.x + (x*cos(m_angle*Pi / 180) - y * sin(m_angle*Pi / 180))*m_speed;
    m_center.y = m_center.y + (x*sin(m_angle*Pi / 180) + y * cos(m_angle*Pi / 180))*m_speed;
}
```

Рисунок 2 — Функция *move()*

2) Функция *rotate()*

Сканирование нажатия клавиш поворота и одновременно с этим вычисление нового угла поворота робота осуществлено в функции *rotate()*.

Функция показана на рисунке 3.

```
int MyRobot::rotate(float angle)
{
    m_angle = m_angle + angle * m_angularSpeed;
    return 0;
}
```

Рисунок 3 — Функция *rotate()*

3) Функция *draw()*

Вывод нового изображения робота на экран производится функцией *draw()*.

Модель робота с колесами представляет из себя 16 линий. 4 у корпуса и по 3 на каждое колесо. В функции *draw()* вычисляются координаты точек каждой линии.

Пример для вычисления точек корпуса приведен на рисунке 4:

```
//Левая верхняя точка
corpus[0].x = m_center.x - (m_width / 2 * cos(m_angle*Pi / 180) + m_height / 2 * -sin(m_angle*Pi / 180));
corpus[0].y = m_center.y - (m_width / 2 * sin(m_angle*Pi / 180) + m_height / 2 * cos(m_angle*Pi / 180));
//Правая верхняя точка
corpus[1].x = m_center.x + (m_width / 2 * cos(m_angle*Pi / 180) + m_height / 2 * sin(m_angle*Pi / 180));
corpus[1].y = m_center.y - (m_width / 2 * -sin(m_angle*Pi / 180) + m_height / 2 * cos(m_angle*Pi / 180));
//Левая нижняя точка
corpus[2].x = m_center.x - (m_width / 2 * cos(m_angle*Pi / 180) + m_height / 2 * sin(m_angle*Pi / 180));
corpus[2].y = m_center.y + (m_width / 2 * -sin(m_angle*Pi / 180) + m_height / 2 * cos(m_angle*Pi / 180));
//Правая нижняя точка
corpus[3].x = m_center.x + (m_width / 2 * cos(m_angle*Pi / 180) + m_height / 2 * -sin(m_angle*Pi / 180));
corpus[3].y = m_center.y + (m_width / 2 * sin(m_angle*Pi / 180) + m_height / 2 * cos(m_angle*Pi / 180));
```

Рисунок 4 — Вычисление точек корпуса

Затем происходит вывод всех линий на экран с помощью функции *line()*. Пример приведен на рисунке 5.

```
line(ioImage, corpus[0], corpus[1], lineColor, 2, 8, 0);
line(ioImage, corpus[1], corpus[3], lineColor, 2, 8, 0);
line(ioImage, corpus[3], corpus[2], lineColor, 2, 8, 0);
line(ioImage, corpus[2], corpus[0], lineColor, 2, 8, 0);
```

Рисунок 5 — Отрисовка корпуса

Границы

Проверка на пересечение роботом границ заданного поля происходит следующим образом:

1) Проверяется, пересек ли центр робота одну из границ ;

2) Если да, то робот появляется с противоположной стороны поля;

Пример представлен на рисунке 6:

```
if ((robot.getCenterY()) < 0)
{
    robot.setCenter(robot.getCenterX(), robot.getCenterY() + 600);
}

if ((robot.getCenterX()) < 0)
{
    robot.setCenter(robot.getCenterX() + 800, robot.getCenterY());
}

if ((robot.getCenterY()) > 600)
{
    robot.setCenter(robot.getCenterX(), robot.getCenterY() - 600);
}

if ((robot.getCenterX()) > 800)
{
    robot.setCenter(robot.getCenterX() - 800, robot.getCenterY());
}
```

Рисунок 5 — Проверка границ

Вывод

В ходе роботы проведено успешно создана роботелега, умеющая двигаться по заданной области по нажатию клавиш. Успешно изучены и использованы необходимые алгоритмы *OpenCV*.