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 Куликов М.М.

Преподаватель Варлашин В.В.

« »_____2020 г.

Санкт-Петербург

2020

Задание

С помощью методов *OpenCV* реализовать движение робота по заданному полю, его поворот на месте, а также ограничение на выезд за пределы.

Ход работы

Общий алгоритм

Общий алгоритм выполнения программы:

1. Задание начальных значений для класса MyRobot;

Следующие операции зациклены с помощью while для постоянного их повторения. Цикл прекращается при нажатии клавиши Esc (27).

2. Вывод пустого изображения (фона);
3. Отрисовка робота;
4. Ожидание нажатия клавиши;
5. Вычисление новых координат робота при движении или повороте.

Класс MyRobot

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

```
MyRobot::MyRobot(float width, float height,
                 float wheelWidth, float wheelDiameter,
                 float speed, float angularSpeed):
    m_width(width),
    m_height(height),
    m_wheelWidth(wheelWidth),
    m_wheelDiameter(wheelDiameter),
    m_speed(speed),
    m_angularSpeed(angularSpeed)
```

```
private:
    cv::Point2f m_center;
    cv::Size2i m_area;
    float m_angle = 0;
    float m_width;
    float m_height;
    float m_wheelWidth;
    float m_wheelDiameter;
    float m_speed;
    float m_rotation;

    float m_angularSpeed;
```

Рисунок 1 – Поля класса

Некоторые параметры задаются при объявлении объекта класса, к остальным доступ осуществляется только с помощью методов.

Основные методы

1) Метод move

Реализация метода представлена на рисунке 2.

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

Рисунок 2 — Метод move

Данный метод осуществляет продольное и поперечное движение робота. Его координаты умножены на матрицу поворота для того, чтобы робот двигался в сторону, куда направлена его передняя часть.

Пример использования данного метода представлен ниже на рисунке 3.

```
switch(key)
{
case 'w':
    robot.move(0, -1);
    break;
```

Рисунок 3 — Пример движения вперёд

2) Метод rotate

Реализация метода представлена на рисунке 4.

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

Рисунок 4 — Метод rotate

Данный метод осуществляет поворот робота вокруг своей оси

Пример использования данного метода представлен ниже на рисунке 5.

```
case 'y':
    robot.rotate(-1);
    break;
```

Рисунок 5 — Пример поворота влево

3) Метод draw

Метод draw создает все необходимые точки для построения робота, а также рисует линии между ними. Все точки робота заранее умножены на матрицу поворота, для реализации поворотов робота вокруг своей оси

Так же, метод draw проверяет робота на выход за границы окна с помощью дополнительного метода checkBorders. Проверяются крайние точки робота, при пересечении границ, робот возвращается в заданную точку центра.

Вывод

В ходе роботы проведено успешное моделирование движения робота по заданной области. Успешно изучены и использованы необходимые алгоритмы *OpenCV*.