**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра МО ЭВМ**

**КУРСОВАЯ РАБОТА**

**по дисциплине «Системы реального времени на основе LINUX»**

Тема: Использование симулятора GAZEBO

|  |  |  |
| --- | --- | --- |
| Студент гр. 1303 |  | Плохой Н.А. |
| Преподаватель |  | Филатов А.Г. |

Санкт-Петербург

2017

**Исходная постановка задачи**

Построить траекторию (и пройти по ней), которая покрывает всю территорию карты (пропылесосит весь пол). Необходимо реализовать возможность контролирования робота (пуск, пауза, остановка).

**Исходные данные**

Робот появляется в неизвестном мире и начинает движение. Робот знает габариты своего чистящего устройства, исходя из этого должен построить траекторию очистки всей территории. Окружающий мир – произвольная замкнутая область. Робот оснащен лазерным дальномером.

**Проектирование решения**

Решение поставленной задачи можно разделить на два пункта:

* Реализация программного модуля для связи ROS и Gazebo
* Реализация алгоритма построения пути

При проектировании программного модуля было принято решение сделать его максимально конфигурируемым и расширяемым. Чтобы при необходимости имелась возможность доработки / быстрого внедрения нового функционала в систему. Результаты проектирования представлены в виде диаграммы классов на рисунке 1.

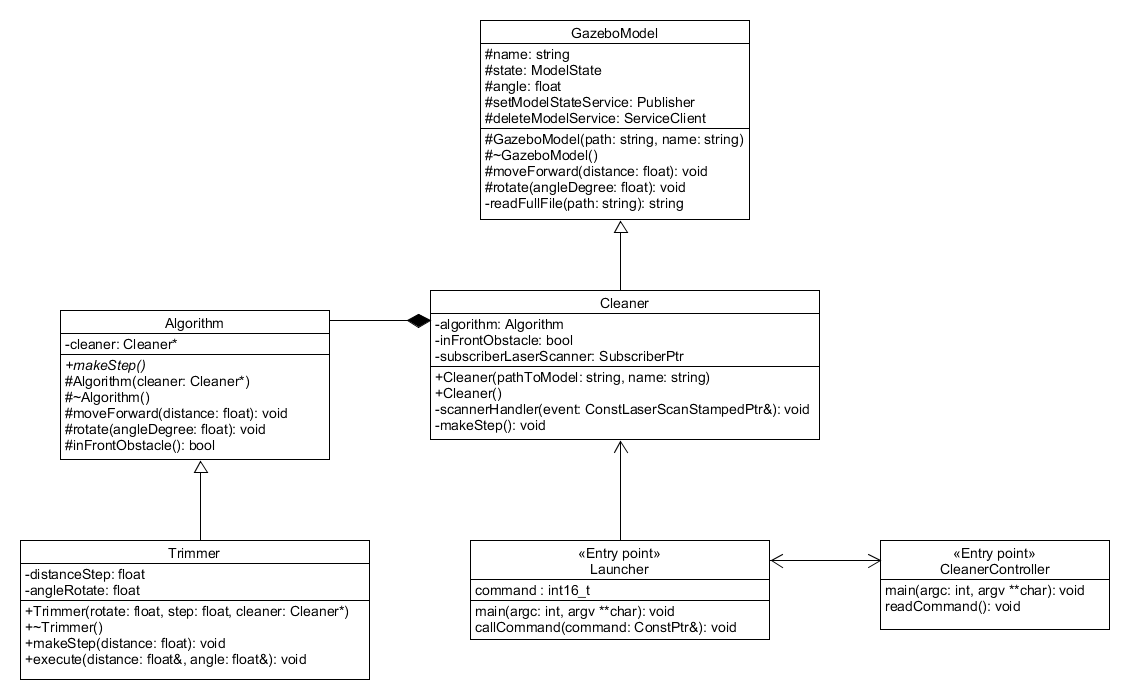


Рисунок 1. Диаграмма классов программного модуля.

Базовым классом модели является класс *GazeboModel*. Он позволяет загружать модель из XML файла, динамически создавать объект с заданными параметрами в среде Gazebo и осуществлять базовые операции (перемещение, поворот).

Класс *Cleaner* расширяет базовый функционал, добавляя взаимодействие с лазерным дальномером, установленным на модели. Помимо этого, в классе также предусмотрено поле для хранения реализации конкретного алгоритма для построения пути.

В качестве базового класса алгоритма используется *Algorithm*. На данный момент, единственной реализацией является класс *Trimmer*.

**Выбор алгоритма**

При анализе существующих алгоритмов построения траектории для движения роботов-пылесосов было выделено несколько реализаций:

1. Спиралевидное движение – относительно хорошо подходит для больших квадратных помещений без препятствий. Редко используется в качестве единственного применяемого алгоритма.
2. Свободное блуждание – позволяет роботу преодолевать почти все виды препятствий, но из-за случайности процесса нельзя назвать точное количество шагов для обхода всей требуемой территории.
3. «Плинтус» – позволяет обойти помещение по периметру. Редко используется в качестве единственного применяемого алгоритма.
4. «Газонокосилка» – сочетает в себе достоинства алгоритма плинтус с элементами свободного блуждания. Позволяет успешно обходить препятствия.

На основании этой краткой характеристики для реализации был выбран алгоритм «Газонокосилка». Ниже представлен псевдокод данного алгоритма, где:

inFrontObstacle – находится ли модель перед препятствием

travelledDistance – пройденное расстояние

minDistance – минимальное расстояние (шаг модели)

maxDistance – максимальное расстояние, которое проходит модель по прямой

|  |
| --- |
| if !inFrontObstacle:  distance = minDistance  move(distance)  travelledDistance += distance  else if distance > 0:  if travelledDistance < maxDistance:  if flag:  rotate(90)  else:  rotate(-90)  invert flag  else:  tmp = 180 - k2 / distance \* 57.4  if flag:  rotate(angle)  else:  rotate(-angle)  inver flag  distance = 0  else:  move(90) |

Листинг 1. Псевдокод алгоритма «Газонокосилка».

**Моделирование с использованием Gazebo**

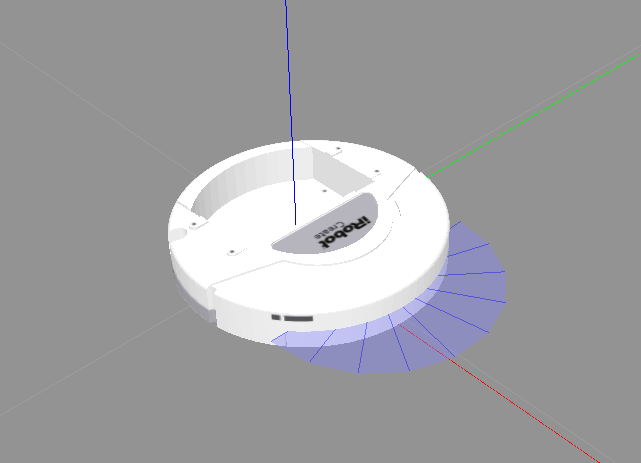
Gazebo – это программный пакет, позволяющий осуществлять симуляцию «мира», в котором реализованы некоторые из законов физики и имеется возможность наполнять его собственными и заимствованными моделями. Моделям можно задавать некоторое поведение. Физический движок позволяет моделировать такие явления как гравитацию, коллизии, трение, имитировать сенсоры, камеры и т.д.

Для использования собственной модели необходимо подготовить её, создав и соответствующим образом наполнив файл формата SDF. В качестве основы робота была использована модель с открытым исходным кодом IRobotCreate за авторством Nate Koenig.

Для использования лазера необходимо расположить компонент sensor на модели. После чего подключить плагин для сбора сообщений, генерируемых данных сенсором. Далее можно получать данные с лазера, подписавшись на соответствующий топик.

|  |
| --- |
| <sensor name="laser" type="ray">  <pose>0.110000 0.000000 0.040000 0 0 -0.000000</pose>  <visualize>true</visualize>  <update\_rate>30.0</update\_rate>  <ray>  <scan>  <horizontal>  <samples>10</samples>  <resolution>1.000000</resolution>  <min\_angle>-1.6</min\_angle>  <max\_angle>1.6</max\_angle>  </horizontal>  </scan>  <range>  <min>0.050000</min>  <max>0.150000</max>  <resolution>0.100000</resolution>  </range>  </ray>  </sensor> |

Листинг 2. Подключаемый к модели компонент «лазер».

Рисунок 2. Загруженная модель робота-пылесоса в среду Gazebo.

Взаимодействие с моделью осуществляется путем подписки и отправки данных на соответствующие топики (реализовано в классе *GazeboModel*).

**Выводы**

Основной задачей в рамках данной работы было знакомство со средством симуляции и визуализации Gazebo. Исходя из полученных знаний можно сказать, что данный пакет действительно позволяет довольно быстро сформировать требуемую среду и проверить работоспособность того или иного алгоритма. Однако, для этого требуется хорошее знание его функционала. Он имеет довольно высокий порог вхождения. Помимо этого, за счет того что это некоммерческий проект, разрабатываемый сообществом, нет полной стабильности и иногда система генерирует непредвиденные ошибки, возникшие не по вине пользователя, а из-за внутренних недоработок.