Посвящается моей матери,
Jancy Joseph и моему отцу C. G. Joseph
которые поддержали меня в создании этой книги

Содержание

Составители

Об авторе

Лентин Джозеф – автор этой книги, предприниматель-робототехник из Индии. Он руководит компанией Qbotics Labs. Эта организация находится в Индии, разрабатывает программное обеспечение для роботов и имеет 7-летний опыт работы в сфере робототехники, прежде всего в ROS, OpenCV и PCL.

Лентин Джозеф является автором четырех книг по ROS: «Изучение робототехники с помощью Python», «Освоение ROS для программирования робота», «Проекты робототехники ROS» и «Операционная система робота (ROS) для абсолютных новичков».

В настоящее время автор проводит магистерскую программу по робототехнике в Индии, а также в институте робототехники, CMU, США.

О рецензенте

Ruixiang Du – кандидат технических наук Политехнического университета Вустера. Работает в лаборатории исследования автономии, контроля и оценки и специализируется на планировании движения и управления автономными мобильными роботами в динамичной среде. В 2011 г. получил степень бакалавра автоматизации в Северном Китайском электроэнергетическом университете, а в 2013 г. – степень магистра в области робототехники WPI. Занимался проектами с роботизированными платформами, начиная от медицинских роботов и беспилотных воздушных/наземных транспортных средств и до человекоподобных роботов.

Введение

Книга «Изучение робототехники с помощью Python» состоит из девяти глав, в которых рассказывается, как создать с нуля автономный мобильный робот и, используя язык программирования Python, запрограммировать его. Это устройство разрабатывалось как обслуживающий робот, с помощью которого можно подавать еду в квартире, гостинице и ресторане. В книге представлены подробные пошаговые инструкции, выполняя которые, вы этого робота построите. Сначала рассматриваются основы робототехники. Затем будет создана 3D-модель. После этого будут рассмотрены аппаратные компоненты, необходимые для создания прототипа устройства.

В основном программная часть реализована на языке программирования Python, метаоперационной системе Robot Operating System (ROS) и библиотеке алгоритмов компьютерного зрения, обработки изображений и численных алгоритмов общего назначения OpenCV. Показано использование Python как для проектирования робота, так и для создания пользовательского интерфейса. Симулятор Gazebo используется для моделирования робота, работы с библиотекой машинного зрения OpenCV, OpenNI и PCL и обработки 2D- и 3D-изображений. Каждая глава начинается с теории, позволяющей понять прикладную часть. Книга анализировалась специалистами в области робототехники.

Для кого предназначена эта книга

Книга «Изучение робототехники с помощью Python» предназначена для инженеров-робототехников, желающих углубить свои знания в этой области и усовершенствовать созданные ранее роботы, предпринимателей, использующих сервисных роботов в своем деле, студентов, изучающих робототехнику, и людей, увлеченных созданием роботов. Книга содержит легковыполнимые пошаговые инструкции.

Краткое содержание

Глава 1 «Начало работы с операционной системой для робота (ROS)»: объясняются основные понятия ROS, являющейся основной платформой для программирования роботов.

Глава 2 «Основные понятия роботов с дифференциальным приводом»: рассматриваются основные принципы роботов с дифференциальным приводом. Обсуждаются фундаментальные концепции дифференциального привода мобильного робота, понятия кинематики и обратной кинематики дифференци-

ального привода. Знания этих принципов помогут вам правильно настроить программное обеспечение регулятора дифференциальной передачи.

Глава 3 «Моделирование робота с дифференциальным приводом»: рассчитываем конструкцию робота и создаем чертежи мобильного робота в 2D и 3D. Создание 2D- и 3D-чертежей является одним из условий разработки мобильного робота. После завершения проектирования и моделирования робота читатель получит расчетные параметры и чертежи, которые будут использованы для создания модели робота.

Глава 4 «Моделирование дифференциального привода робота, управляемого операционной системой ROS»: знакомимся со средой моделирования робота Gazebo и помогаем читателям с помощью Gazebo создать модель собственного робота.

Глава 5 «Проектирование оборудования и схем ChefBot»: выбираем аппаратные компоненты, необходимые для сборки ChefBot.

Глава 6 «Согласование приводов и датчиков с контроллером робота»: рассматриваем взаимодействие регулятора, привода и датчиков робота. Обсуждаем, как взаимодействуют приводы и датчики робота с регулятором Launchpad Tiva C.

Глава 7 «Согласование датчиков зрения с ROS»: обсуждается, как согласовать датчики зрения Kinect и Orbecc Astra, которые могут быть использованы в Chefbot для автономной навигации с ROS.

Глава 8 «Сборка робота ChefBot и интеграция программного обеспечения»: рассказывается, как установить электронное оборудование мобильного робота и настроить его программное обеспечение.

Глава 9 «Разработка графического интерфейса для робота с использованием Qt и Python»: обсуждается, как с помощью GUI передавать команды роботу для его перемещения к столам в гостинице.

Получаем максимальную отдачу от этой книги

Книга содержит подробную информацию, позволяющую создать робот. Для этого вам понадобятся электронные компоненты и материалы (пластик, металл, фанера). Робот может быть построен с нуля, или вы можете купить готовое устройство, оснащенное дифференциальным приводом и энкодером. Для управления роботом вам понадобится плата контроллера, например Texas instruments LaunchPad. Для разработки, моделирования и программирования робота необходим ноутбук/нетбук. В этой книге в качестве бортового компьютера мы используем высокопроизводительный мини-компьютер Intel NUC. Это очень эффективный высокопроизводительный ПК размером 10×10 см. Для компьютерного 3D-зрения используется 3D-датчик, например лазерный сканер Kinect или Orbbec Astra.

Что касается программного обеспечения, вам понадобится знание команд GNU/Linux и Python. Для работы с примерами понадобится операционная си-

стема Ubuntu 16.04 LTS. Быстрее создать робот вам поможет хорошее знание ROS, OpenCV, OpenNI и PCL. Кроме того, вам потребуется установить Ros Kinect Melodic с примерами.

Загрузка файлов с примерами программного кода

Вы можете скачать файлы с примерами программного кода по адресу www.packtpub.com со своего аккаунта. Если вы приобрели эту книгу в другом месте, зарегистрируйтесь на сайте www.packtpub.com/support. Файлы вам будут отправлены по электронной почте.

Чтобы загрузить файлы кода, выполните следующие действия:

1) если вы уже зарегистрированы, зайдите под своим аккаунтом на сайт www.packtpub.com. Если у вас нет учетной записи, сначала зарегистрируйтесь;
2) выберите вкладку **SUPPORT**;
3) щелкните мышью на ссылке на Code Downloads & Errata;
4) введите название книги в поле поиска и следуйте инструкциям на экране.

После того как файл будет загружен, пожалуйста, убедитесь, что вы сможете его распаковать. Это можно сделать с помощью программ:

- WinRAR/7-Zip for Windows;
- Zipeg/iZip/UnRarX for Mac;
- 7-Zip/PeaZip for Linux.

Пакет кодов для этой книги также размещен на GitHub по адресу: https://github.com/PacktPublishing/Learning-Robotics-using-Python-Second-Edition.

Другие программные пакеты и видео для нашего богатого каталога книг вы найдете по адресу: https://github.com/PacktPublishing/.

Загрузка цветных изображений

Вы также можете загрузить PDF-файл с цветными изображениями скриншотов/диаграмм, используемых в этой книге. Данный файл находится по адресу: https://www.packtpub.com/sites/default/files/downloads/LearningRoboticsusingPythonSecondEdition_ColorImages.pdf.

Условные обозначения, принятые в этой книге

В этой книге используется ряд типографических условных обозначений.

`Моноширинный текст`: указывает кодовые слова в тексте, имена таблиц базы данных, названия папок, имена файлов, расширения файлов, имена путей сохранения файлов, фиктивные URL-адреса, пользовательский ввод и маркеры Twitter. Например: «первая процедура – создать файл и сохранить его с расширением `.world`».

Пример блока кода:

```
<xacro:include filename="$(find
  chefbot_description)/urdf/chefbot_gazebo.urdf.xacro"/>
  <xacro:include filename="$(find
  chefbot_description)/urdf/chefbot_properties.urdf.xacro"/>
```

Любой ввод или вывод из командной строки записывается следующим образом:

```
$ roslaunch chefbot_gazebo chefbot_empty_world.launch
```

Так будут оформляться предупреждения и важные примечания.

Так будут оформляться советы или рекомендации.

Отзывы и пожелания

Мы всегда рады отзывам наших читателей. Расскажите нам, что вы думаете об этой книге – что понравилось или, может быть, не понравилось. Отзывы важны для нас, чтобы выпускать книги, которые будут для вас максимально полезны.

Вы можете написать отзыв прямо на нашем сайте www.dmkpress.com, зайдя на страницу книги и оставив комментарий в разделе «Отзывы и рецензии». Также можно послать письмо главному редактору по адресу dmkpress@gmail.com, при этом напишите название книги в теме письма.

Если есть тема, в которой вы квалифицированы, и вы заинтересованы в написании новой книги, заполните форму на нашем сайте по адресу http://dmkpress.com/authors/publish_book/ или напишите в издательство по адресу dmkpress@gmail.com.

Список опечаток

Хотя мы приняли все возможные меры, для того чтобы удостовериться в качестве наших текстов, ошибки все равно случаются. Если вы найдете ошибку в одной из наших книг — возможно, ошибку в тексте или в коде, — мы будем очень благодарны, если вы сообщите нам о ней. Сделав это, вы избавите других читателей от расстройств и поможете нам улучшить последующие версии данной книги.

Если вы найдете какие-либо ошибки в коде, пожалуйста, сообщите о них главному редактору по адресу dmkpress@gmail.com, и мы исправим это в следующих тиражах.

Нарушение авторских прав

Пиратство в интернете по-прежнему остается насущной проблемой. Издательства «ДМК Пресс» и Packt очень серьезно относятся к вопросам защиты авторских прав и лицензирования. Если вы столкнетесь в интернете с неза-

конно выполненной копией любой нашей книги, пожалуйста, сообщите нам адрес копии или веб-сайта, чтобы мы могли принять меры.

Пожалуйста, свяжитесь с нами по адресу электронной почты dmkpress@gmail.com со ссылкой на подозрительные материалы.

Мы высоко ценим любую помощь по защите наших авторов, помогающую нам предоставлять вам качественные материалы.

Глава 1

Начало работы с операционной системой для робота (ROS)

Основная цель этой книги – помочь вам создать автономный мобильный робот. Робот оснащается операционной системой ROS, а все операции будут смоделированы с помощью программы-симулятора, называющейся Gazebo. Также в этой главе вы найдете теоретический расчет, проект робота, чертежи деталей, внутреннее программное обеспечение и программное обеспечение высокого уровня, интегрированное с ROS.

Глава начинается с основных понятий о ROS, таких как установка, написание основных программ с использованием ROS и Python. Далее идет знакомство с основами работы в Gazebo. Эта глава станет базовой в вашем проекте автономного робота. Если вы уже знакомы с основами ROS и эта операционная система у вас установлена, то эту главу можете пропустить. Однако вы всегда можете вернуться, чтобы освежить память об основах ROS.

В этой главе будут рассмотрены следующие темы:

- введение в ROS;
- установка Ros Kinetic на Ubuntu 16.04.3;
- знакомство, установка и тестирование Gazebo.

Начинаем программировать робота с помощью Python и операционной системы робота (ROS).

Технические условия

Для получения полного кода, описанного в этой главе, перейдите по следующей ссылке: https://github.com/qboticslabs/learning_robotics_2nd_ed.

Введение в ROS

ROS – это программная платформа, назначение которой – написание программного обеспечения робота. Основная цель ROS – создание и использование робототехнического программного обеспечения по всему миру. ROS состоит из набора инструментов, библиотек и программных пакетов, упрощающих задачу создания программного обеспечения робота.

Официальное определение ROS

ROS – это метаоперационная система с открытым исходным кодом, разработанная для робота. Она предоставляет все необходимые для данной операционной системы службы, включая аппаратные средства визуализации, низкоуровневое управление устройством, реализацию функциональности для общего использования, передачу сообщений между процессами и пакетами управления. Она также предоставляет инструменты и библиотеки для получения, построения, написания и выполнения кода на нескольких компьютерах. ROS похожа в некоторых отношениях на «фреймворки робота», такие как Player, YARP, Orocos, CARMEN, Orca, MOOS и Microsoft Robotics Studio.

Более подробная информация о ROS здесь: http://wiki.ros.org/ROS/Introduction.

Основные функции, которые обеспечивает ROS:

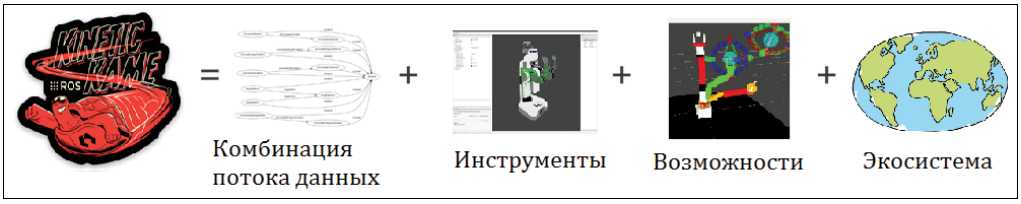
- **интерфейс передачи сообщений**. Это базовая функция ROS, позволяющая организовать межпроцессное взаимодействие. Используя эту функцию, ROS может обмениваться данными между связанными системами. Больше о технических терминах, связанных с обменом данными между ROS-программами/узлами, вы узнаете в этой главе;
- **аппаратная универсальность**. ROS обладает некоторой степенью универсальности, что позволяет разработчикам создавать роботизированные приложения. Они могут быть использованы с любым устройством. Разработчики должны лишь позаботиться о надлежащем аппаратном обеспечении робота;
- **управление пакетами**. Узлы ROS организованы в пакеты и называются пакетами ROS. Они состоят из исходных кодов, конфигурационных файлов типа «build» («строить») и т. д. Мы сначала разрабатываем пакет, собираем его и устанавливаем. В ROS предусмотрена система сборки, которая помогает создавать эти пакеты. Управление пакетами делает развитие ROS более упорядоченным и организованным;
- **дополнительные библиотеки**. В структуру ROS встроено несколько дополнительных библиотек, разработанных сторонними организациями, таких как Open-CV, PCL, OpenNI и др. Данные библиотеки помогают сделать ROS универсальной метаоперационной системой;
- **управление низкоуровневыми устройствами**. Во всех роботах присутствуют устройства низкого уровня, например устройства ввода/вы-

вода, контроллеры последовательных портов и др. Управление такими устройствами тоже осуществляется с помощью ROS;

- **распределенные вычисления**. Объем вычислений, которые потребуются для обработки потока данных от многочисленных датчиков, очень высок. Используя ROS, мы можем легко распределить их в группе вычислительных узлов. Это позволит равномерно распределить вычислительные мощности, и данные будут обрабатываться быстрее, чем на одиночном компьютере;
- **многократное использование кода**. Основной целью ROS является многократное использование кода, способствующее развитию научно-исследовательского сообщества по всему миру. Исполняемые файлы ROS называются узлами. Они сгруппированы в единый объект, называемый пакетом ROS. Группа пакетов – стек. Стеки могут быть разделены и распределены;
- **независимость от языка**. В ROS можно программировать с помощью таких популярных языков, как Python, C++ и Lisp. Узлы могут быть написаны на любом из этих языков, и обмен данными между такими узлами через ROS не вызовет никаких проблем;
- **простота тестирования**. ROS имеет встроенный модуль интеграционного тестирования, называющийся rostest и предназначенный для тестирования пакетов ROS;
- **масштабирование**. ROS после некоторой модификации может выполнять сложные вычисления в роботах;
- **свободный и открытый исходный код**. Исходный код ROS открыт и абсолютно свободен в использовании. Основная часть ROS лицензирована BSD и может быть повторно использована в коммерческих и закрытых продуктах источника.

ROS – комбинация из потока данных (передачи сообщений), инструментов, возможностей и экосистемы. В ROS встроены мощные инструменты для ее отладки и визуализации. Робот, оснащенный ROS, обладает такими возможностями, как навигация, определение местонахождения, составление карты, манипуляции и т. д. Они помогают создать мощные приложения для роботов.

Приведенное ниже изображение показывает уравнение ROS:



Уравнение ROS

Более подробная информация о ROS здесь: http://wiki.ros.org/ROS/Introduction.

Концепции ROS

ROS состоит из трех основных уровней, таких как:

1) файловая система ROS;
2) вычислительный граф ROS;
3) сообщество ROS.

Файловая система ROS

Основная задача файловой системы ROS – организация файлов на диске.

Основные термины файловой системы ROS:

- **пакеты**. Пакеты ROS являются основной единицей в рамках программного фреймворка ROS. Пакет ROS может содержать исполняемые файлы, ROS-библиотеки, принадлежащие программному обеспечению сторонних производителей, конфигурационные файлы и т. д. Пакеты ROS можно использовать повторно и совместно;
- **описания пакета**. В описании пакетов (пакет .XML-файла) вы найдете всю детализацию пакетов, включая имя, описание, лицензию и зависимости;
- **типы сообщений** (msg). Все сообщения хранятся в отдельной папке, в файлах с расширением `.msg`. ROS-сообщения – это структурированные данные, проходящие через ROS;
- **типы служб** (SRV). Описания служб хранятся в папке SRV с расширением `.srv`. SRV-файлы определяют запрос и ответ для структуры данных в сервисе ROS.

Вычислительный граф ROS

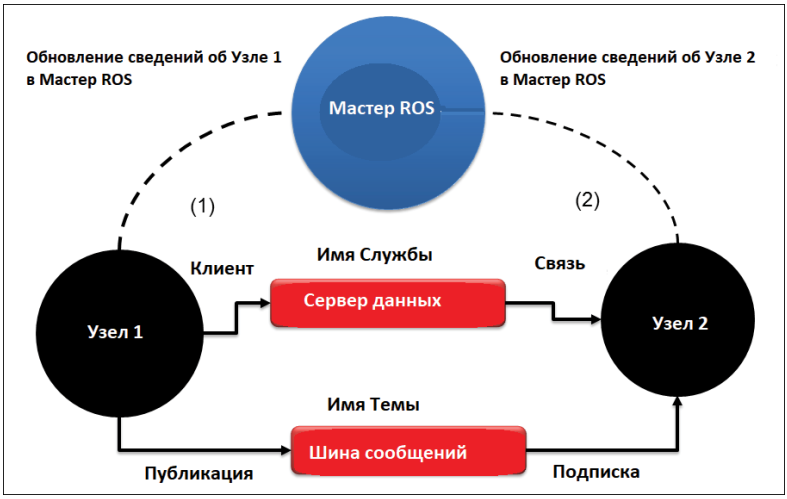
Вычислительный граф ROS – это одноранговая сеть систем ROS, назначение которой – обработка всех данных. Основными понятиями вычислительной графики ROS являются узлы, Мастер ROS, сервер параметров, сообщения и службы.

- **Узлы** – это вычислительные процессы в ROS. Каждый узел выполняет обработку определенных данных. Например, один узел обрабатывает данные с лазерного сканера, другой публикует эти данные и т. д. Узел ROS создается с помощью клиентской библиотеки ROS (например, roscpp и rospy). Более подробно мы познакомимся с этими библиотеками во время создания образца узла.
- **Мастер ROS.** Мастер ROS действует как служба имен в графе вычислений ROS. Он хранит темы и регистрационную информацию служб для узлов ROS. Узлы общаются с Мастером, сообщая свои регистрационные данные. Эти узлы, общаясь с мастером, получают информацию о других зарегистрированных узлах и соответственно устанавливают с ними соединения. Мастер также выполняет обратный вызов этих узлов при изменении регистрационной информации, позволяющей узлам динамически создать соединения при запуске новых узлов. Узлы подключа-

ются непосредственно к другим узлам. Мастер, как и DNS-сервер, обеспечивает только поиск информации. Узлы, подписывающиеся на тему, запрашивают соединения от узлов, публикующих эту тему, и позволяют установить связи по согласованному протоколу подключения. Наиболее распространенный протокол, используемый в ROS, называется TCPROS. Он использует стандартный протокол TCP/IP-сокетов.

- **Сервер параметров**. Статические данные ROS хранятся в общедоступном месте – на сервере данных, от которого узлы и получают доступ к этим данным. Мы можем установить объем сервера данных и определить его как частный или общественный, чтобы ограничить доступ к серверу одним узлом или разрешить доступ всем узлам.
- **Темы ROS**. Узлы обмениваются данными в виде сообщений через систему транспортировки в ROS с конкретным названием темы. Тема – это имя, используемое для идентификации содержания сообщения. Узел, заинтересованный в определенном виде данных, будет подписываться на соответствующую тему. В целом издатель и абонент не знают о существовании друг друга. Идея в том, чтобы отделить производство информации от ее потребления. Логически можно думать о теме как о шине строго типизированных сообщений. Каждая шина имеет имя, и любой может подключаться к ней для отправки или получения сообщений до тех пор, пока они имеют правильный тип.
- **Сообщения**. Узлы взаимодействуют друг с другом с помощью сообщений. Сообщение ROS состоит из примитивных типов данных, таких как целые числа, плавающие точки, логические значения («true» – «истина» или «false» – «ложь»). Сообщения ROS передаются через тему ROS. Тема может получать или посылать один тип сообщения только один раз. Мы можем создать собственное определение сообщения и передать его через тему.
- **Службы.** Выше было показано, что переслать/получить сообщение с использованием тем ROS – это очень простой метод общения между узлами. Этот способ коммуникации («один ко многим» – «one-to-many») позволяет подписываться на одну тему любому количеству узлов. В некоторых случаях может осуществляться частичное взаимодействие, обычно применяемое в распределенных системах. Используя определенное сообщение, можно послать запрос в другой узел, предоставляющий нужную услугу. Узел, пославший запрос другому узлу, обязан ожидать результат обработки данного запроса.
- **Bags** – это формат для сохранения и воспроизведения данных сообщений ROS. Bags – важный механизм для хранения данных, получаемых, например, от датчиков. Эти данные могут быть использованы позже для разработки и тестирования алгоритмов в автономном режиме.

На следующем рисунке показано взаимодействие между мастером, темами, службами и узлами:



Связь между узлами ROS и мастером ROS

На приведенной выше схеме показана связь, осуществляемая с помощью Мастера ROS (ROS Master) и двух узлов ROS: Узла 1 (Node 1) и Узла 2 (Node 2). Прежде чем начать обмен данными между этими двумя узлами, необходимо запустить Мастер ROS. Master ROS является посредником, позволяющим установить связь и обмениваться информацией между узлами. Например, Узел 1 (Node 1) хочет опубликовать тему/xyz с типом сообщения abc. Для этого он обращается к ROS Master и сообщает, что собирается опубликовать тему/xyz с сообщением вида abc и поделиться ею с другими узлами. А другой узел, например Узел 2 (Node 2), желает подписаться на эту тему/xyz с типом сообщения abc. Мастер сообщает Узлу 2, какой узел опубликовал запрашиваемую информацию, и выделяет порт для связи этих двух узлов напрямую, минуя Мастер ROS.

Таким образом, как уже упоминалось ранее, Master ROS представляет собой DNS-сервер, обеспечивающий поиск запрашиваемой узлами информации. Ранее уже упоминалось, что в ROS применяется протокол связи TCPROS (http://wiki.ros.org/ROS/TCPROS), использующий для связи в основном сокеты TCP/IP.

Уровень общения ROS

Сообщество ROS состоит из разработчиков и исследователей, создающих и поддерживающих пакеты ROS. Они обмениваются между собой информацией о существующих и недавно созданных пакетах и другими новостями, связанными со структурой ROS. Сообщество ROS предоставляет следующие услуги:

- **распространение**. В распространяемых дистрибутивах присутствует ряд пакетов, предназначенных для конкретных версий ROS. В этой книге используется дистрибутив ROS Kinetic. Доступны и другие версии дистрибутивов, например такие, как ROS Lunar или Indigo. Зная версию дис-

трибутива, легче подобрать к нему недостающие пакеты, которые будут стабильно работать;

- **репозитории**. Репозитории, или мета-пакеты ROS, находятся на удаленных серверах. Также в едином репозитории может храниться отдельный пакет;
- **Wiki-ROS**. На этом ресурсе доступна почти вся документация о ROS. Здесь вы можете получить информацию от фундаментальных понятиях до самого передового программирования. Любой желающий может зарегистрироваться и предоставить собственную документацию, исправления или обновления, написать учебники и т. д. Адрес ресурса: http://Wiki.ROS.org);
- **списки рассылки**. Для получения обновлений следует подписаться на рассылку ROS (http://lists.ros.org/mailman/listinfo/ros-users). Последние новости о ROS и форум, где обсуждается программное обеспечение ROS, находятся по адресу: https://discourse.ros.org;
- **ответы ROS**. Если у вас возникнут вопросы, связанные с ROS, обращайтесь по адресу: https://answers.ros.org/questions/. Здесь вы получите помощь и поддержку от разработчиков со всего мира.

Существует много функциональных возможностей ROS. Для получения дополнительных сведений о них обращайтесь на официальный сайт ROS по адресу: www.ros.org.

В следующем разделе мы рассмотрим процедуру установки ROS.

Установка ROS на Ubuntu

Как было сказано ранее, ROS – это метаоперационная система, которая устанавливается в ведущей операционной системе (host-системе). ROS полностью поддерживается такой операционной системой, как Linux Ubuntu. Проводились эксперименты с операционными системами и Windows, и OS X. Вот некоторые из последних ROS-дистрибутивов:

Дистрибутив	Дата выпуска
ROS Melodic Morenia	23 мая 2018
ROS Lunar Loggerhead	23 мая 2017
ROS Kinetic Kame	23 мая 2016
ROS Indigo Igloo	22 июля 2014

Теперь рассмотрим процедуру установки стабильной версии дистрибутива ROS Kinetic с долгосрочной поддержкой (LTS) на Ubuntu 16.04.3 LTS. Разработчики ROS Kinetic Kame в первую очередь ориентировались на Ubuntu 16.04 LTS. Вы также можете найти инструкции по настройке ROS LTS Melodic Morenia на Ubuntu 18.04 LTS. Если на вашем компьютере операционной системой является Windows или OS X, то сначала нужно установить виртуальную машину Virtual Box, а затем Linux Ubuntu 16.04 LTS.