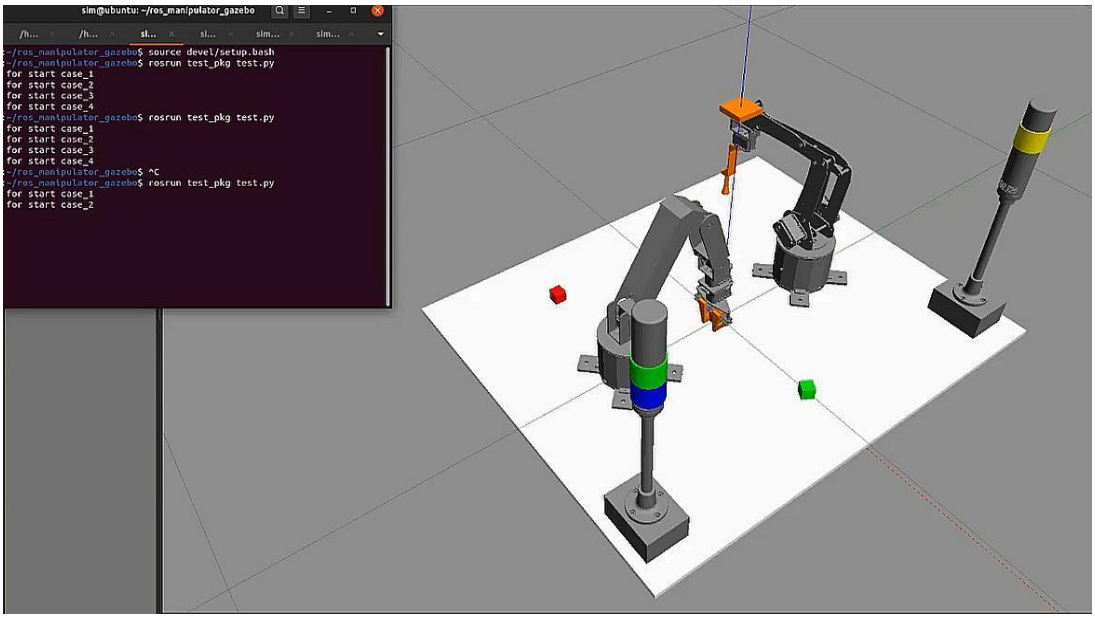
ЛАБОРАТОРНАЯ РАБОТА N 12

«Разработка управляющей модели манипулятора с системой технического зрения»

Автор: Владислав Бакаев

Цель работы: получение практических навыков разработки симуляционных моделей манипуляционных РТК в среде Gazebo фреймворка ROS.



1. Теоретическая часть

В лабораторной работе создаются программные средства симуляции движения двух манипуляционных роботов выполняющих тестовую задачу в составе РТК. За задания лабораторной работы взят регламент соревнований Робофест 2020 в номинации “Манипуляционные РТК”. Данные соревнования нацелены на введение участников в принципы построения многоагентных систем и управления манипуляционными роботами. В качестве хард скилов участники получают навыки разработки программного обеспечения на базе фреймворка ROS, а также программирования на языке Python.

1.1. Основная задача

В данных соревнованиях в качестве объекта для настройки автономной работы рассматривается учебная производственная ячейка, состоящая из нескольких манипуляторов, сигнальных ламп, системы технического зрения и объектов манипулирования. Набор подобных элементов ячейки позволяет моделировать задачи складирования, упаковки, перемещения различных объектов.

Соревновательное задание разделяется на несколько этапов. Различные этапы подразумевают использование различных инструментов. В конечном итоге все соревновательные задания сводятся к написанию последовательности действий для управления отдельными роботами и устройствами.

Примером такой задачи может служить следующая формулировка:

Разработать систему управления для решения задачи перемещения объекта кубического вида из известной точки 1 в известную точку 2 с помощью манипулятора определенного типа. После чего необходимо переместить этот объект из точки 2 в точку 3 с помощью манипулятора другого типа. При этом необходимо соблюдать цветовую индикацию сигнальных ламп для отображения действий, выполняемых каждым манипулятором.

1.2. Состав РТК

РТК состоит из следующих компонентов:

- двух манипуляторов: одного с плоскопараллельной кинематической схемой (паллетайзер), другого – с ангулярной (углового), имеющего пять степеней свободы

- СТЗ на базе телекамеры и специально разработанного для него программного обеспечения;
- два сигнальных фонаря, один из которых синхронизирован с работой манипулятора паллетайзера, другой – манипулятора углового типа;
- рабочего поля (полигона) с тестовыми объектами (разноцветными кубиками).

2. Практическая часть

В лабораторной работе необходимо создать комплекс программных средств симулирующих выполнение задачи складирования объектов в среде Gazebo фреймворка ROS.

ROS (Robot Operating System) — это экосистема для программирования роботов, предоставляющий функциональность для распределённой работы. ROS обеспечивает стандартные службы операционной системы, такие как: аппаратную абстракцию, низкоуровневый контроль устройств, реализацию часто используемых функций, передачу сообщений между процессами, и управление пакетами. ROS основан на архитектуре графов, где обработка данных происходит в узлах, которые могут получать и передавать сообщения между собой.

Для моделирования движения роботов и их визуализации используется программный комплекс Gazebo. Программный комплекс Gazebo представляет собой среду для симулирования работы виртуальных роботов с различными сенсорами в окружении всевозможных объектов. Приложение состоит из графической части и части по имитированию взаимодействия твердых объектов, позволяя моделировать динамику и кинематику механизмов роботов (включая моменты взаимодействия с телами внешней среды) и формировать физически правдоподобные показания виртуальных датчиков.

2.1. Задание на лабораторную работу

1. Запустить симуляционный полигон РТК.
2. Запустить симуляционные модели роботов.
3. Изучить и опробовать функционал роботов.
4. Изучить работу сигнальных ламп полигона.
5. Реализовать задачу манипулирования объектами по известным координатам.
6. Реализовать задачу манипулирования объектами с использованием СТЗ.

2.2. Основные программные средства

В состав ПО лабораторной работы входят пакеты среды ROS для управления роботами и их визуализации.

Все пакеты *управления и визуализации* роботов располагаются в папке workspace. На данной виртуальной машине папка workspace называется ros_polygon_simulations. Там располагаются папки, отвечающие за сборку проекта (devel и build), а также папка src, в которой непосредственно находятся пакеты визуализации и управления. Структура пакетов в данном случае имеет следующий вид (рис. 1):

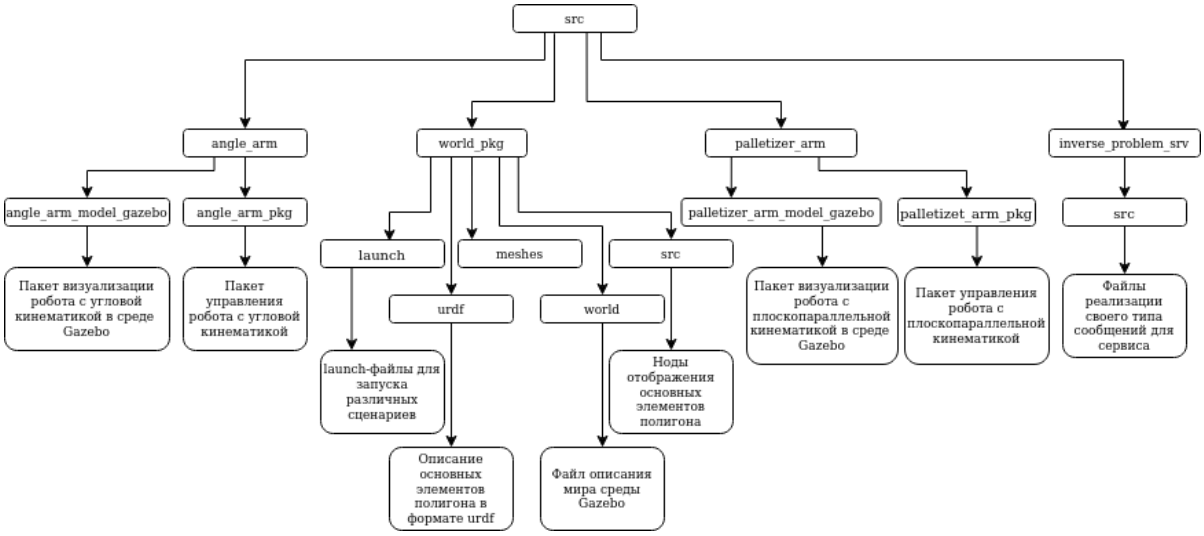


Рис.1. Структура пакетов управления и визуализации

Из рис. 1 следует, что всего в папке src и во вложенных в нее папках расположено в общей сложности 6 пакетов, а именно:

- angle_arm_model_gazebo - пакет, необходимый для визуализации манипулятора углового типа в среде Gazebo;
- angle_arm_pkg - пакет управления роботом с угловой кинематикой;
- world_pkg - пакет, содержащий инструменты для запуска заданий, создания полигона, формирования мира в среде Gazebo, работы с камерой;
- palletizer_arm_model_gazebo - пакет, необходимый для визуализации манипулятора с плоскопараллельной кинематиков в среде Gazebo;
- palletizer_arm_pkg - пакет управления роботом с плоскопараллельной кинематикой;
- inverse_problem_srv - пакет, описывающий специальный созданный тип сообщений сервиса.

2.3. Основы работы с полигоном в симуляции

Для начала работы с полигоном необходимо запустить «Мир». Для этого требуется перейти в папку ros_polygon_simulations и выполнить команду

source devel/setup.bash.

После чего нужно выполнить запуск «Мира» с помощью команды

roslaunch world_pkg world.launch.

После выполнения данной команды появится окно симулятора Gazebo (рис. 2).

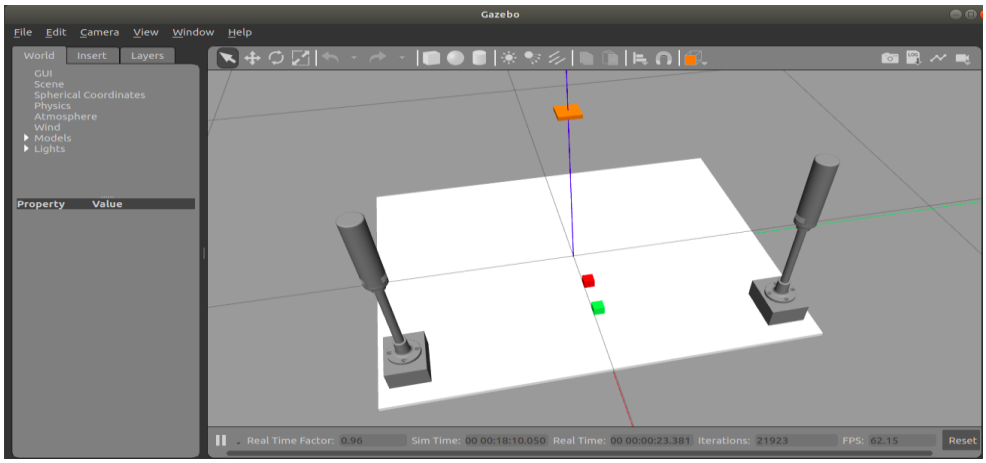


Рис.2. Окно симулятора РТК с компонентами

На поле располагается набор объектов (макет полигона, два фонаря, объекты для выполнения кейсов и камера, изображенная в виде прямоугольника).

2.3.1. Управление фонарями

Первой задачей является управление фонарями, необходимые для отображения текущих задач, выполняемых определенным роботом. Для этого существует набор сервисов. У каждого фонаря (один для паллетайзера, другой – для углового манипуляционного робота) имеется 4 цветовых индикатора (рис. 3). Таким образом, имеется 8 сервисов для задания состояния всех цветовых индикаторов каждого фонаря:

- /angle_robot/blue_light
- /angle_robot/green_light
- /angle_robot/red_light
- /angle_robot/yellow_light
- /palletizer_robot/blue_light
- /palletizer_robot/green_light
- /palletizer_robot/red_light
- /palletizer_robot/yellow_light

Каждый сервис имеет тип «SetBool» и включает свет при подаче сигнала «True», выключает при «False».

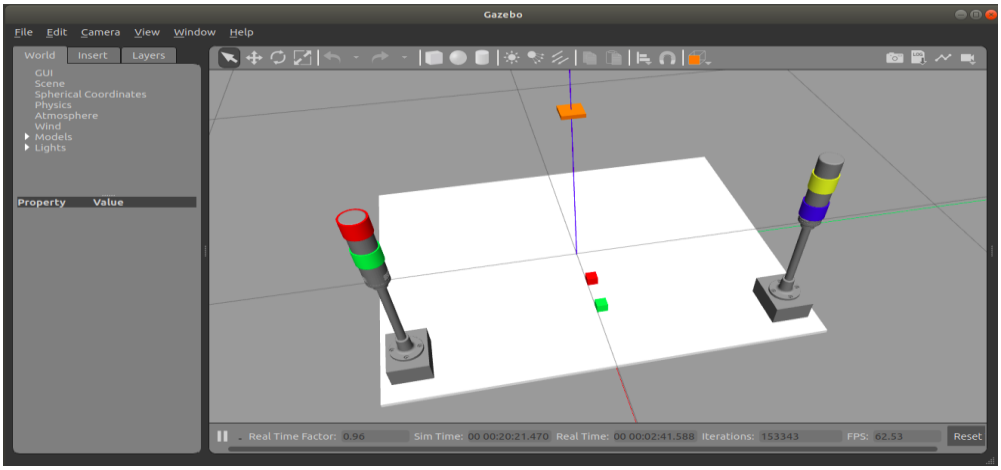


Рис. 3. Окно симулятора РТК с зажженными фонарями

Пример команды для включения красного сигнала фонаря углового манипулятора в терминале:

rosservice call /angle_robot/red_light "data: true"

Для обновления поля используется сервис /polygon/restart_cubs, который перезагружает все перемещаемые объекты и устанавливает их в начальное положение. Сервис имеет тип «Empty». Пример команды в терминале:

rosservice call /polygon/restart_cubs "{ }"

2.3.2. Подключение роботов

Для загрузки манипулятора с угловой кинематикой в открытый мир среды Gazebo необходимо открыть новый терминал, выполнить команду

source devel/setup.bash,

а затем

roslaunch angle_arm_model_gazebo arngle_arm_model_control.launch.

Далее нужно открыть новый терминал, в котором в этот же мир необходимо загрузить второй манипулятор с плоскопараллельной кинематикой, и выполнить команду

source devel/setup.bash,

а после запустить

roslaunch palletizer_arm_model_gazebo palletizer_arm_model_control.launch.

В результате выполнения данных команд на полигоне появятся два манипулятора, доступных для управления (рис. 4).

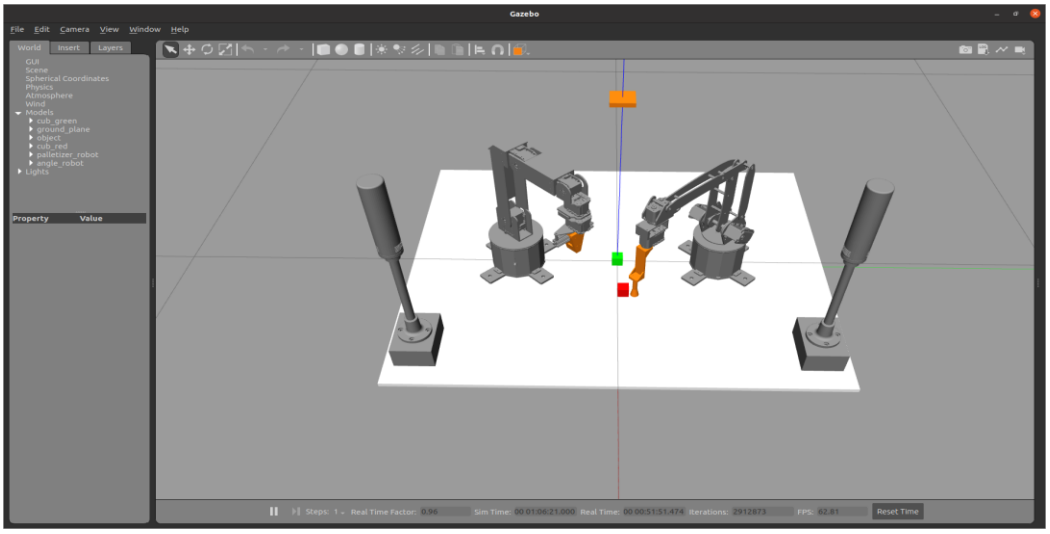


Рис. 4. Полигон с фонарями и манипуляторами

Для непосредственного управления движениями роботов используются следующие сервисы.

- Для задания управляющего воздействия угловому манипулятору применяется сервис /angle_robot/cmd_point
  Данный сервис принимает команду в виде строки, имеющей формат «x y z roll», где первые три координаты являются линейными координатами декартова пространства в миллиметрах, последняя координата – угол поворота схвата манипулятора относительно оси z манипулятора (т.е. относительно вертикали). Все координаты разделены пробелами. В качестве ответа сервис возвращает «True», если точка успешно достигнута и «False», если манипулятор не может ее достичь.

- Для управления схватом манипулятора используется сервис /angle_robot/gripper_cmd. На вход сервис принимает строку, в которой указывается «0», если схват нужно закрыть и «1», если его нужно открыть. Пример задания команды манипулятору с помощью терминала:

rosservice call /angle_robot/cmd_point "point: '350 200 300 0'"

Пример задания команды схвату манипулятора:

rosservice call /angle_robot/gripper_cmd "point: '1'"

- Для управления паллетайзером используется сервис /palletizer_robot/cmd_point
  Данный сервис принимает команду в виде строки, имеющей формат «x y z», где первые три координаты также являются линейными координатами декартова пространства в миллиметрах. Все координаты разделены пробелами. В качестве ответа сервис возвращает «True», если точка успешно достигнута и «False», если манипулятор не может ее достичь. Пример команды для управления манипулятором с помощью терминала:

rosservice call /palletizer_robot/cmd_point "point: '150 150 0'"

- Для управления вакуумной присоской используется сервис /palletizer_robot/vacuum. В качестве управляющих воздействий сервис принимает «True», когда вакуумную присоску нужно включить и «False», когда нужно выключить. При приближении вакуумной присоски к объекту на определенное расстояние объект прикрепляется к схвату. Пример управления вакуумной присоской с помощью терминала:

rosservice call /palletizer_robot/vacuum "data: true"

После запуска всех инструментов в среде rqt_graph можно увидеть следующую взаимосвязь между нодами, топиками и сервисами (рис. 5):

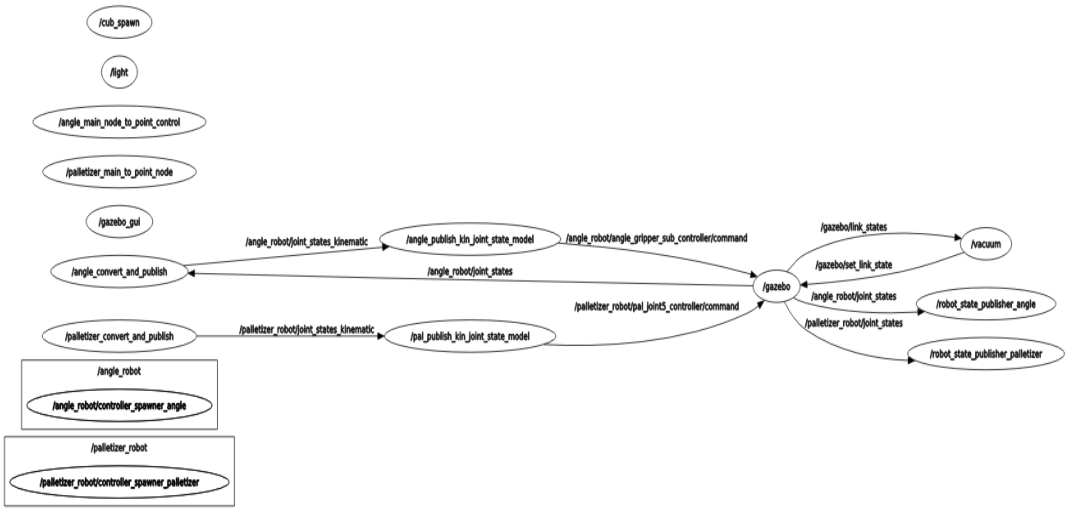


Рис. 5. Структура связей в программе

2.3.3. Работа с телекамерой

Для определения расположения различных объектов, над рабочим полем расположена телекамера. Для ее запуска в отдельном терминале, находясь в домашней директории, необходимо выполнить команду

./start_tracking_cam.sh.

Затем открыть новый терминал, перейти в workspace проекта и выполнить команды

source devel/setup.bash

и

rosrun world_pkg motocortex_proxy.py.

Далее необходимо выбрать цвет детектирования. Для этого в браузере необходимо перейти на страницу http://trackingcam3.local. При этом откроется окно (рис. 6):

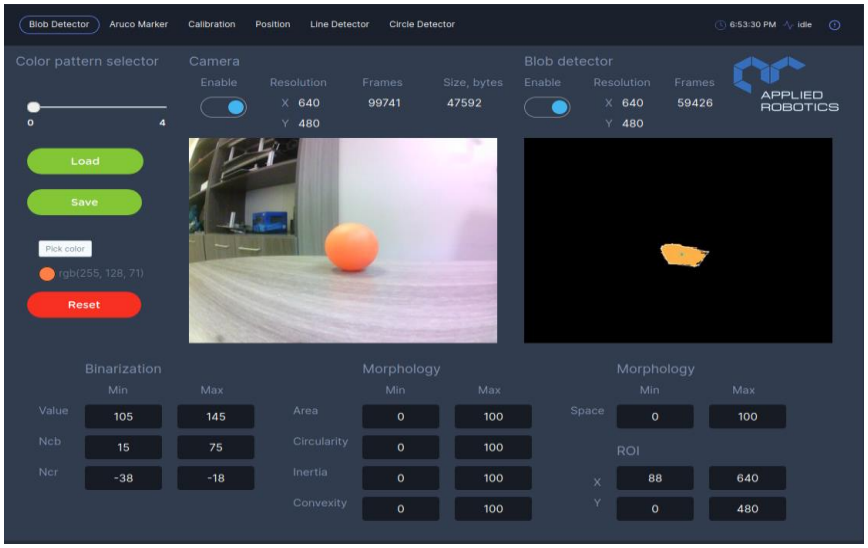


Рис. 6. GUI интерфейс настройки распознавания объектов по цвету

Для того чтобы выбрать объект для распознавания, необходимо нажать кнопку Pick color и указать курсором на объект, который нужно распознать. После выполнения этого действия программное обеспечение телекамеры само подберет набор цветовых оттенков для распознавания выбранного объекта. Ползунки Enable у полей Camera и Blob Detector отвечают за включение или отключение отображения исходного видео и распознанного объекта.

Данная телекамера позволяет определять координаты выбранного объекта в рабочем пространстве. В качестве результата камера возвращает координаты объекта на изображении в пикселях. Эти данные публикуются в топик /blob, который имеет тип сообщений geometry_msgs/Vector3. Данный тип сообщений имеет следующую структуру:

```
float64 x
float64 y
float64 z
```

где параметры x и y определяют координаты центра объекта в пикселях телекамеры, а координата z определяет класс объекта (классы отличаются по цветам).

Для создания алгоритмов перевода пиксельных координат в реальные расстояния будет предложен набор данных, таких как высота камеры над полем, координаты определенных маркеров на поле в пикселях и реальных координатах.

- **Расположение систем координат и начальные данные**

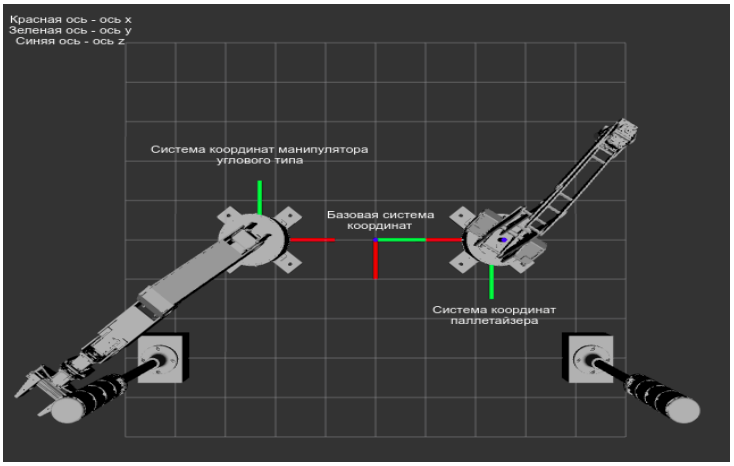


Рис. 7. Схема расположения объектов на полигоне с указанием направления их осей координат

- **Расположение систем координат роботов относительно базовой системы координат**
  Для манипулятора с угловой кинематикой:
  x = 0 мм
  y = -231 мм
  z = 0 мм

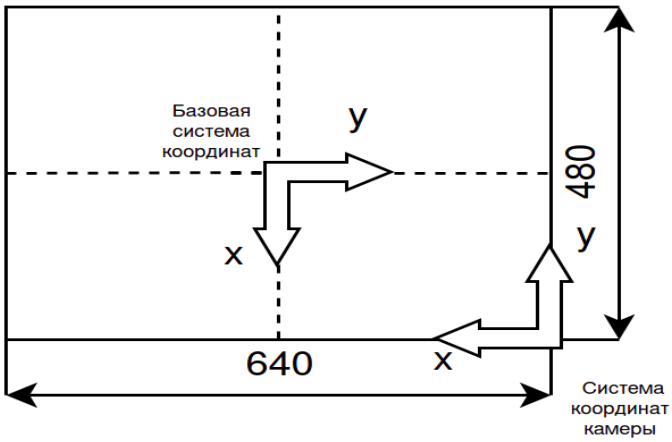


Рис. 8. Расположение базовой системы координат полигона относительно системы координат камеры

Для манипулятора с плоскопараллельной кинематикой (паллетайзера):
x = 0 мм
y = 231 мм
z = 0 мм

Центр базовой системы координат располагается точно по центру изображения телекамеры (x = 320, y = 240 в пикселях).

2.3.4 Реализация с помощью кода управления роботами и сигнальными лампами

Выше было предложено описание отдельных инструментов управления манипуляционными роботами и сигнальными фонарями и показан пример выполнения команд с помощью терминала. Но более практично написать специальный исполняемый файл, в котором будет написана последовательность действий. Пример такого файла предложен ниже:

```
#!/usr/bin/env python3
import rospy
from inverse_problem_srv.srv import point_cmd,point_cmdResponse
from std_srvs.srv import SetBool

point_cmd_srv_client_angle = rospy.ServiceProxy("/angle_robot/cmd_point", point_cmd)
point_cmd_srv_client_pall = rospy.ServiceProxy("/palletizer_robot/cmd_point", point_cmd)

gripper_cmd_srv_client = rospy.ServiceProxy("/angle_robot/gripper_cmd", SetBool)
light_red_srv_client = rospy.ServiceProxy("/palletizer_robot/red_light", SetBool)

rospy.init_node("test_node")

point_cmd_srv_client_angle("0 250 50 -1.57")
point_cmd_srv_client_pall("-150 200 50")

light_red_srv_client(True)
rospy.sleep(2)
light_red_srv_client(False)
```

```
gripper_cmd_srv_client(True)
gripper_cmd_srv_client(False)
```

Данный код запускает базовые функции для управления манипуляторами. Если рассматривать последовательность кода, то он выполняет следующие действия.

В строке 1 указывается версия языка Python, который будет использоваться для запуска кода, в строках 2-4 подтягиваются определенные библиотеки для работы кода. Строки 5-8 производят подключение к соответствующим сервисам для управления руками и рабочими инструментами манипуляторов. Строка 9 инициализирует данный исполняемый файл как часть среды ROS. В строках 10-11 продемонстрирован пример формирования и отправки команды манипуляторам с угловой кинематикой и паллетайзеру. Важным является замечание, что программа не продолжит свое выполнение после отправки команды в строке 10 до тех пор, пока команда не будет полность выполнена манипулятором. Только после достижения роботом желаемой позиции программа перейдет к строке 11. В строке 12 показан пример включения красного цвета сигнальной лампы манипулятора паллетайзера. Подобным образом производится управление и остальными сигналами. В строке 13 продемонстрирован пример команды, вызывающей паузу выполнения программы в секундах (в данном примере 2 секунды). В строках 15-16 показан пример управления рабочими инструментами манипуляторов.

2.4. Представление отчета по лабораторной работе

В отчет по лабораторной работе должны входить:

- скриншоты запущенной симуляции и визуализации;
- описание автомата решения задачи манипулирования объектами при известных координатах, листинг программы;
- описание автомата решения задачи манипулирования объектами с использованием СТЗ, листинг программы;
- видеозапись работы алгоритмов.

3. ПРИЛОЖЕНИЕ

3.1. Установка и настройка ПО

Для полноценной работы с фреймворком ROS необходим какой либо дистрибутив операционной системы с ядром linux, для этого можно использовать виртуальную машину. То есть, необходимо установить операционную систему внутри операционной системы, для этого понадобится программа virualBox, которую можно установить и скачать по ссылке: https://www.virtualbox.org/.

Также необходимо скачать сам образ операционной системы с предустановленным ROS, который можно будет открыть с помощью программы virtualBox. Для этого необходимо скачать архив по ссылке:

https://drive.google.com/file/d/1MH_DJYtO-E9kRmVtC8P6VGMzYNiT4jE_/view?usp=sharing

Возможно, потребуется сохранить файл на личный Яндекс диск для скачивания.

После скачивания его нужно разархивировать в удобную вам папку. После разархивирования можно увидеть следующую структуру папок и файлов:

Ubuntu 64-bit
Ubuntu 64-bit
Файлы виртуальной машины

Данные файлы нельзя как-либо модифицировать и изменять во избежание всевозможных поломок ПО. В дальнейшем с помощью программы virtualBox будет открываться файл Ubuntu 64-bit с расширением Virtual Machine Disk Format (.vmdk. рис. 1)

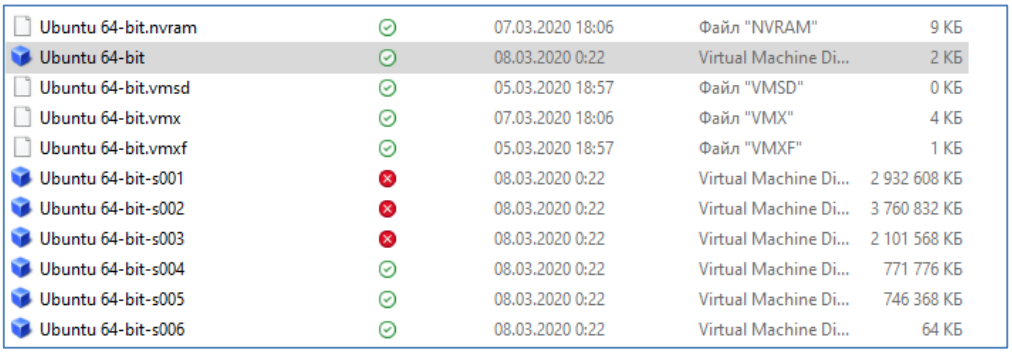


Рис. 1. Структура файлов операционной системы Ubuntu

3.2. Настройка

Для первого запуска виртуальной машины необходимо:

1. Запустить программу virtualBox
2. Нажать кнопку создать, чтобы создать новую виртуальную машину (рис. 2):

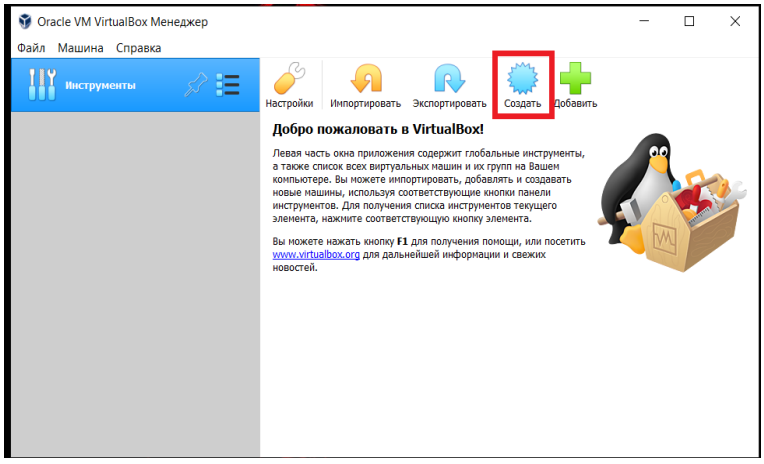


Рис. 2. Создание виртуальной машины

3. В высветившемся окне нажать кнопку экспертный режим (рис. 3)

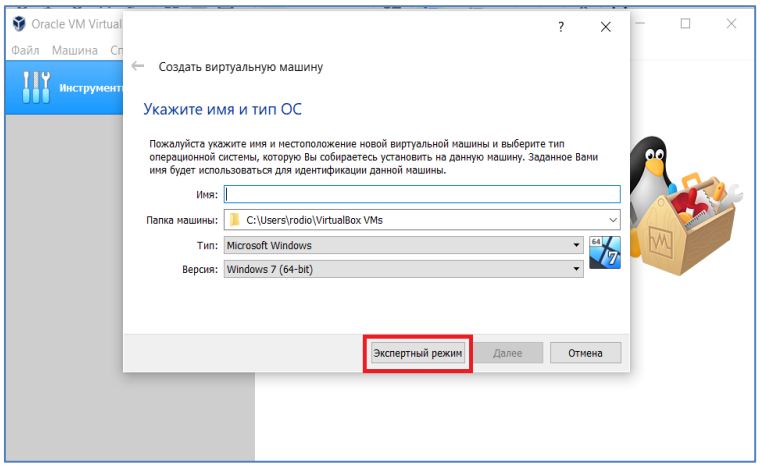
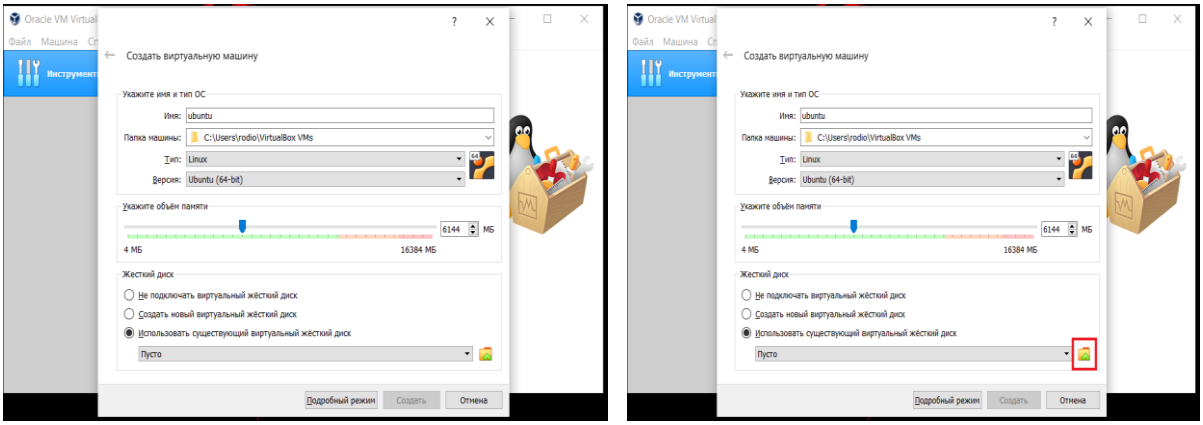


Рис. 3. Задание экспертного режима

4. Настраиваем виртуальную машину:
   4.1. Задать любое имя машины
   4.2. Выбрать папку, в которой будут храниться данные виртуальной машины, предложенную папку можно не изменять
   4.3. Тип машины - Linux, версия Ubuntu (64-bit)
   4.4. Выбрать количество оперативной памяти, которое будет доступно виртуальной машине. ДЛЯ КОРРЕКТНОЙ РАБОТЫ НЕОБХОДИМО НЕ МЕНЕЕ 6 ГБ (рис. 4, *а*)



а *б*

Рис. 4. Установка объема оперативной памяти

   4.5. В графе жесткий диск выбрать пункт **использовать существующий виртуальный жесткий диск**, и в качестве виртуального диска выбрать файл **Ubuntu 64-bit.vmdk** описанный в предыдущем пункте (рис. 4, *б*)

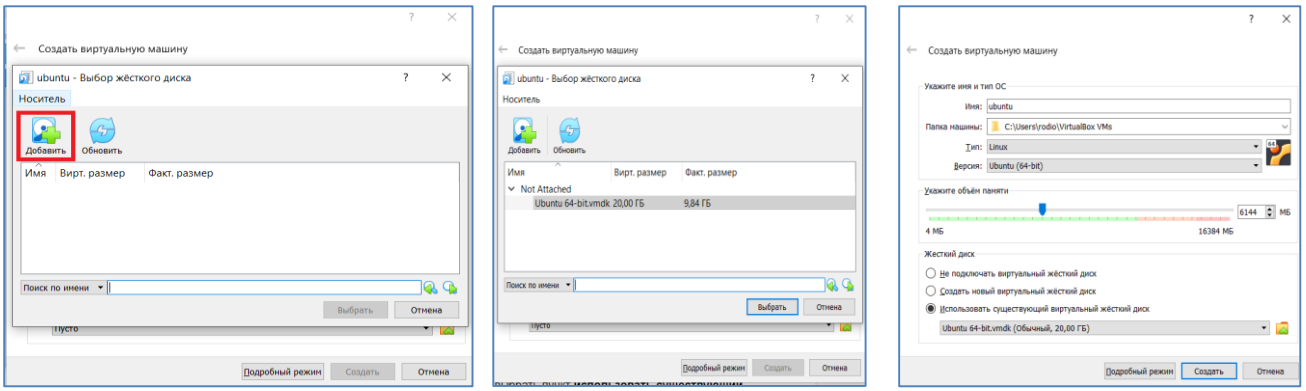
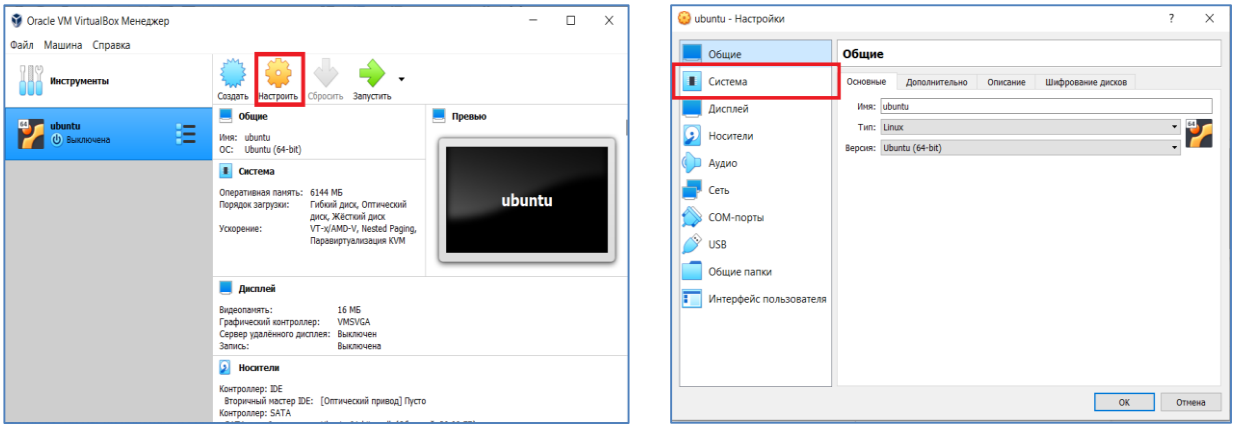


Рис. 5. Продолжение настройки

5. Когда все настройки выставлены в соответствии с пунктом 4, необходимо нажать кнопку «Создать» (рис. 5).

6. Перед запуском виртуальной машины, необходимо настроить количество **ядер процессора**, которые она сможет использовать (для комфортной работы рекомендуется не менее 4). Для этого:
   6.1. Необходимо нажать кнопку «Настроить» (рис. 6, *а*):



а *б*

Рис. 6. Настройка виртуальной машины

   6.2. Перейти в графу «Система» (рис. 6, *б*)
   6.3. Перейти во вкладку процессор и разрешить использование 4 или более ядер (количество разрешенных к использованию ядер зависит от общего количества ядер процессора, рис. 7):

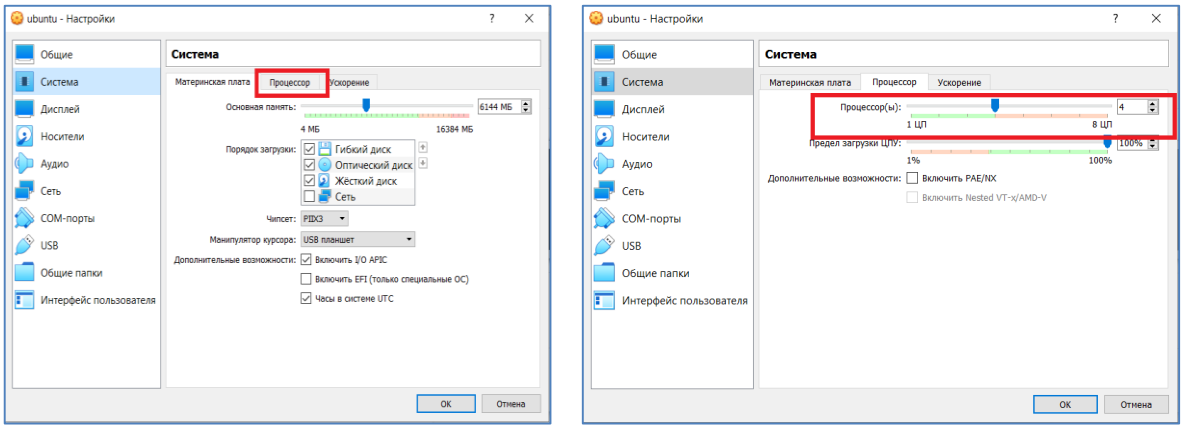


Рис. 7. Настройка процессора

7. После всех настроек можно запустить виртуальную машину, для этого необходимо:
   7.1. Нажать кнопку «Запустить» (рис. 8)

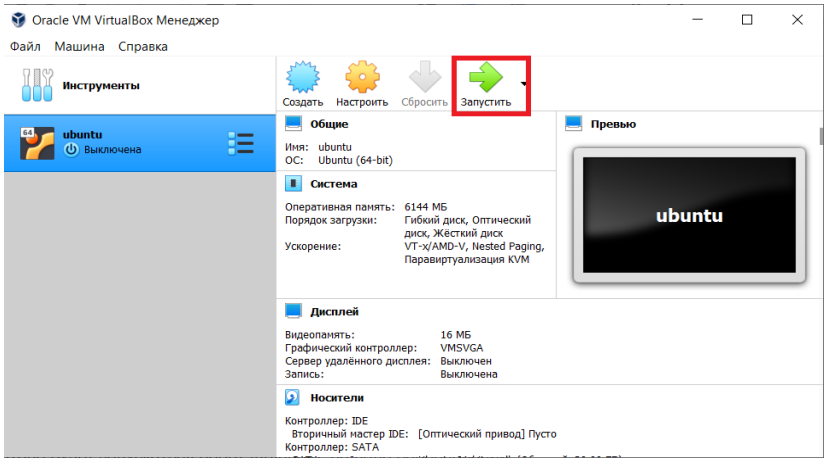


Рис. 8. Запуск виртуальной машины

   7.2. После загрузки системы появится следующее окно (рис. 9):

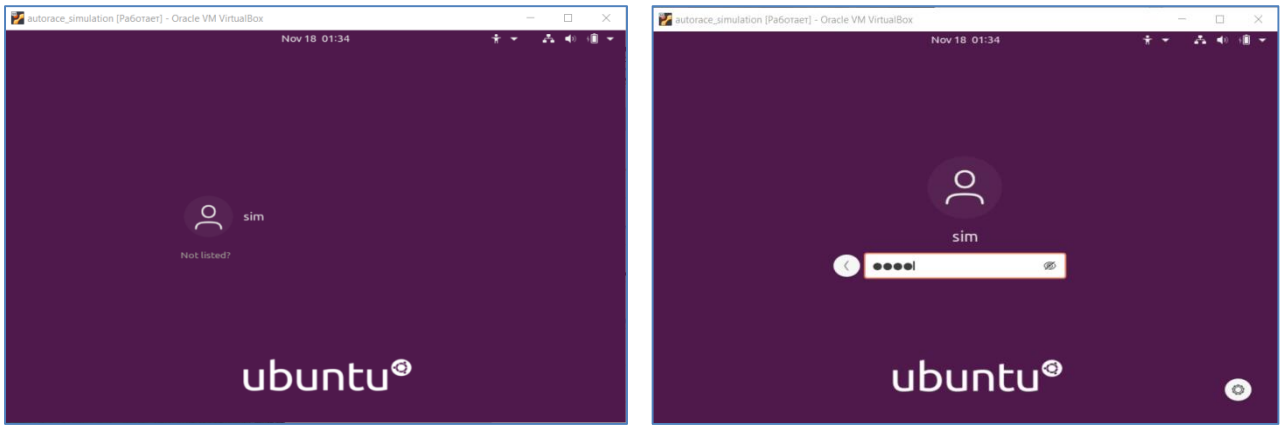
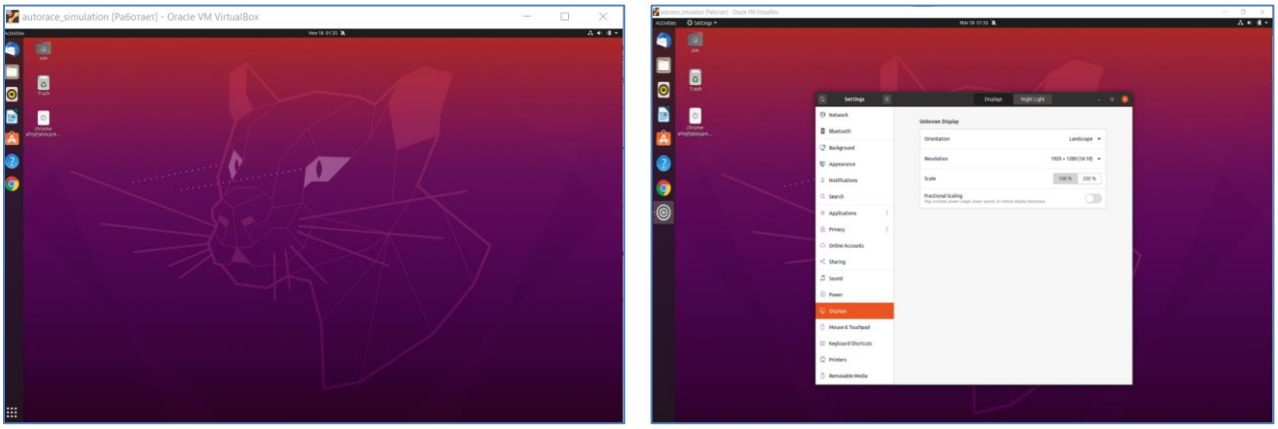


Рис. 9. Запуск операционной системы

7.3. В данном окне нажать на пользователя tb3, после чего высветится окно для ввода пароля:

7.4. Ввести пароль**1234**, после чего откроется окно системы (рис. 10):



а *б*

Рис. 10. Рабочий стол

Примечание

Если окно системы слишком мало для работы, то в настройках Ubuntu необходимо изменить разрешение экрана (рис. 10, *б*).

Теперь внутри данного окна можно вести полноценную работу с операционной системой Ubuntu.