1. ФИНАЛЬНЫЙ ЭТАП

Задача командного тура

1.1. Легенда

С каждым днем в нашем мире задействуется все больше современных технологий. Еще 150 лет назад не существовало машин, а сегодня мы встречаем беспилотные автомобили и роботов доставщиков. Такие технологии позволяют передать роботам некоторые задачи выполняемые людьми, которые требуют большого внимания или могут быть опасны для здоровья. Таким образом, будет минимизироваться человеческий фактор, а вслед за этим повышаться скорость и точность работы.

Уже сейчас существуют роботы -работники в разных сферах, например, робот хирург, спасающий жизни людей, робот - доставщик, робот - помощник и другие. В промышленной среде широко распространены роботы-манипуляторы. Манипулятор — устройство для перемещения объектов без прямого воздействия человека. Но для каждого такого манипулятора нужны разработчики, способные "оживить"его и дать ему указания для работы. В основе задач профиля «Интеллектуальные робототехнические системы» лежит работа с манипулятором.

Для решения задачи участникам потребуются знания линейной алгебры и аналитической геометрии, умение работать с компьютерным зрением, а также навыки взаимодействия с манипулятором.

Задача финала заключается в следующем: На Заводе №6 занимаются изготовлением различных металлических деталей путем литья. Процесс устроен таким образом, что за одну итерацию изготавливаются несколько различных типов изделий, которые перемешаны между собой и хранятся все вместе. Сортировка деталей вручную возможна, но этот процесс занимает слишком много времени и Завод рискует не успеть сделать планируемое количество в срок, а другие методы слишком дороги или неэффективны. По стечению обстоятельств, на складе Завода лежит подозрительно неприспособленный к работе манипулятор, и он достался вам вместе с задачей по сортировке деталей. К счастью для вас, конфигурация манипулятора известна, а детали, которые вам поручили сортировать, оказались маленькими блоками двух разных цветов. Однако, места на складах вам выделили немного, а среди блоков периодически проскакивает брак.

Ваша задача — при помощи известного манипулятора распределить кучу блоков двух разных цветов в заранее известные места. При сортировке блоки желательно складывать в виде башен, чтобы оптимизировать пространство.

1.2. Набор заданий

Решение командной задачи разбито на четыре этапа. Первые три этапа итеративно подводят участников к решению полной финальной задачи, осуществляемому во время последнего четвертого этапа. На каждом этапе в проверку решения заданий данного этапа входят:

- способность проверить гипотезу о работоспособности алгоритма через демонстрацию решения в симуляторе;
- полнота решения задания конкретного этапа;
- воспроизводимость результатов — робототехническое устройство участников должно неоднократно выполнить требуемые действия.

Первый этап

Задача: Так как систему сортировки построить сразу крайне сложно, начинать следует с простого. Уже имеется робот с известной конфигурацией и в начальной зоне есть два блока разных цветов. Известно где блоки будут расположены, но не их ориентация.

Позиция места хранения красных блоков: $X : -0.76453, Y : 0.2541, Z : 0.48366, RX : 1.912, RY : 2.421, RZ : -0.009$

Позиция места хранения синих блоков: $X : -0.94841, Y : 0.2557, Z : 0.49056, RX : 1.918, RY : 2.429, RZ : -0.011$

Позиция №1 для блока: $X : -0.93917, Y : -0.32313, Z : 0.34318, RX : 1.918, RY : 2.429, RZ : -0.011$

Позиция №2 для блока: $X : -0.76379, Y : -0.0601, Z : 0.33589, RX : 0.377, RY : 3.075, RZ : 0.028$

Для каждого цвета обозначено свое место хранения. Место хранения имеет форму квадрата с стороной 5см. Места хранения имеют постоянные координаты. Блоки надо перенести из изначального места расположения в соответствующие их цветам места хранения.

Включая содержательные задачи:

- Калибровка камеры роботехнического устройства.
- Определение изменения системы координат робота и камеры.
- Реализация алгоритмов определения объектов и их характеристик в пространстве.
- Реализация алгоритмов планирования перемещения робота для выполнения поставленной задачи.

Второй этап

Задача: На данном этапе, как и на первом, всё еще робот в начальной зоне которого располагаются блоки двух цветов. Аналогично же есть два места хранения двух цветов соответствующих цветам блоков. Однако на данном этапе блоков 8 и располагаются они, как при реальном производстве навалом. Вы не знаете их место-

положение или ориентацию в изначальной зоне, более того, обьекты могут располагаться друг на друге. Блоки как и ранее требуется распределить в места хранения на основе соответствии цветов.

Включая содержательные задачи:

- Реализация алгоритмов определения отдельных объектов при их наложении.
- Реализация алгоритмов планирования определения последовательности захвата объектов на основе их пространственного расположения.
- Реализация алгоритмов захвата обьектов роботом при усложненном пространственном расположении обьектов.

Третий этап

Задача: Аналогично предыдущему этапу в наличии робот в начальной зоне которого располагается навал из 8 блоков двух разных цветов. Блоки всё еще надо рассортировать в два места хранения двух цветов соответствующих цветам блоков. Однако теперь на производстве произошел брак и в навале присутствует обьект отличающийся от остальных по форме. Данный отличающийся обьект требуется оставить в изначальной зоне.

Включая содержательные задачи:

- Реализация алгоритмов для определения формы обьектов
- Реализация алгоритмов определения отдельных объектов при их наложении.
- Реализация алгоритмов планирования определения последовательности захвата объектов на основе их пространственного расположения.
- Реализация алгоритмов захвата обьектов роботом при усложненном пространственном расположении обьектов.

Четвертый этап

Задача: Ситуация схожа с предыдущим этапом в наличии робота в начальной зоне которого располагается навал из 8 блоков двух разных цветов. На сей раз известно, что в навале помимо блоков присутсвует 2 бракованных объекта, но они всё еще отличаются от обычных блоков формой. Также, теперь для экономии места блоки необходимо сложить так, чтобы как можно меньше из них соприкасалось с изначальной зоной (например в виде башни). При этом складывать блоки всё еще необходимо по цветам (т.е. так, чтобы обьекты разных цветов не соприкасались.)

Включая содержательные задачи:

- Реализация алгоритмов для определения формы обьектов
- Реализация алгоритмов определения отдельных объектов при их наложении.
- Реализация алгоритмов планирования определения последовательности захвата объектов на основе их пространственного расположения.
- Реализация алгоритмов захвата обьектов роботом при усложненном пространственном расположении обьектов.
- Реализация высокоточного и аккуратного перемещения робота
- Реализация алгоритмов составления обьектов

1.3. Описание системы

Робот - манипулятор, способный выполнять вращательные и поступательные движения. Манипулятор оснащен камерой, для распознавания объектов расположенных на столе. В начальной позиции робот имеет координаты

$$X : -0.82, Y : -0.1723, Z : 0.68, RX : 1.487, RY : 3.536, RZ : -0.669$$

где X, Y, Z в м, а RX, RY, RZ в рад. Гарантируется что из начальной позиции начальная зона видна полностью.

Рабочая / достижимая область — пространство стола, в котором манипулятор способен взаимодействовать с объектами. Непосредственно в этой зоне производятся все действия манипулятора, и ограничиваются ей. В рабочей области расположены начальная зона и места сортировки.

Блок — объект, который располагается внутри рабочей области робота и имеет форму прямоугольного параллелепипеда $50 \times 25 \times 20$ мм. Существует два вида блоков: красные блоки и синие блоки.

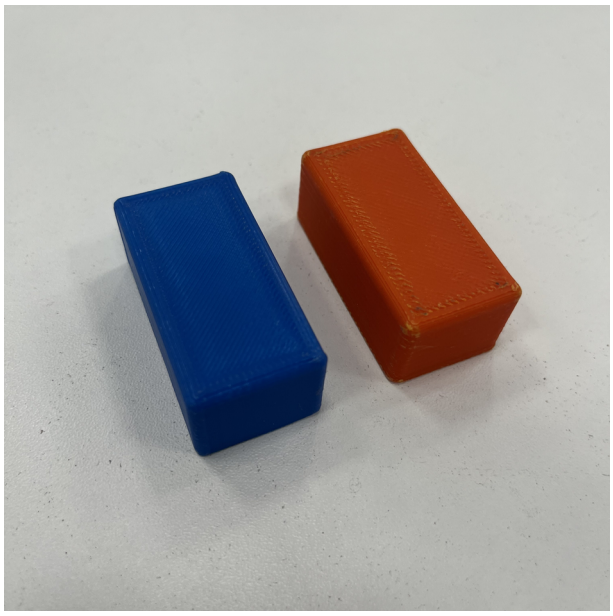

Рис. 1.1: Примеры блоков

Начальная зона — прямоугольная зона 40×30 см, расположенная внутри рабочей области. В данной зоне изначально располагаются все блоки с которыми манипулятор взаимодействует. Внутри стартовой зоны блоки могут располагаться навалом: блоки имеют случайную ориентацию и местоположение внутри данной области. Также допускается их расположение друг на друге.

Место хранения — квадратная зона 10×10 см, расположенная внутри рабочей области. Всего на столе расположено два таких места. Места хранения имеют постоянные координаты.

1.4. Описание робота

На финальном туре команды используют реального робота. Робот представляет собой манипулятор UR10e, оснащенный камерой realsense.

Манипулятор UR10e имеет 6 степеней свободы. Полную официальную документацию манипулятора можно найти здесь (источник на английском): https://s3-eu-west-1.amazonaws.com/ur-support-site/27484/UR10_User_Manual_en_E67ON_Global-3.4.5.pdf

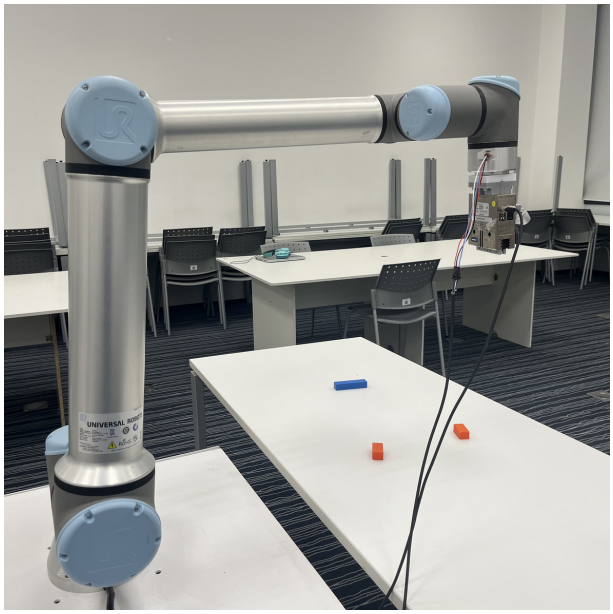

Рис. 1.2: Манипулятор UR10e

На манипулятор установлен двухпалый мягкий захват, который имеет два состояния — открыт и закрыт. Гарантируется, что при помощи данного захвата можно захватить любые представленные в задачах объекты тем или иным образом.

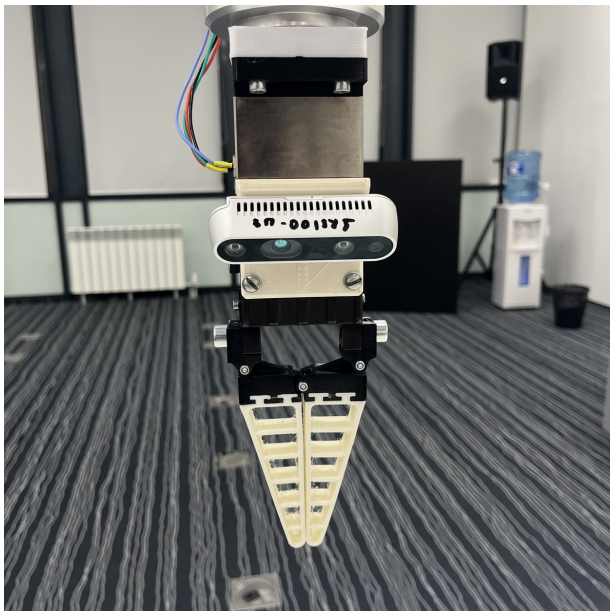

Рис. 1.3: Захват

Робот управляется при помощи python - библиотеки `python-urx` (https://github.com/jkur/python-urx), а для работы с камерой помогает python - библиотека `open3d` (http://www.open3d.org/docs/0.9.0/tutorial/Basic/python_interface.html). Однако для удобства участников существуют специальные скрипты - прослойки, упрощающие работу.

Участникам предоставляется библиотека **“OperateRobot.py”**, в которой лежит класс OperateRobot, содержащий следующие методы:

- `movel` — перемещает робота в точку с координатой относительно основания робота и заданными углами.
- `getl` — получает координаты робота относительно основания робота
- `getj` — получает координаты робота в пространстве звеньев манипулятора (joint space)
- `close` — закрывает соединение с роботом
- `open_gripper` — открывает захват
- `close_gripper` — закрывает захват

Также участникам предоставляется библиотека **“OperateCamera.py”**, в которой лежит класс OperateCamera, содержащий следующие методы:

- `catch_frame` — получает один снимок с камеры
- `save` — сохраняет последний снимок полученный `catch_frame` в файл с указанным именем
- `stop` — завершает работу с камерой

Файл **“tutorial.py”** позволяет познакомиться с основными методами и способами работы с роботом и камерой.

1.5. Условия проведения

1. Участники во время командного этапа финального тура могут использовать интернет и заранее подготовленные библиотеки для решения задачи.
2. В качестве языка программирования робота используется `Python 3.9`.
3. Участники могут использовать любые библиотеки указанные организаторами в заранее выданном участникам файле `requirements.txt`, а также стандартные.
4. Для управления роботом и камерой основным методом считается использование библиотек `OperateRobot.py` и `OperateCamera.py`, в случае необходимости участники могут использовать иные методы управления роботом по договоренности с организаторами.
5. Не допускается использование скорости робота больше $0.2\frac{m}{c}$ и ускорения больше $0.2\frac{m}{c^2}$. В случае если участники будут использовать или попытаются использовать большую скорость они будут дисквалифицированны.
6. Участники обязаны бережно относиться к оборудованию, в частности роботу, блокам и т.д. В случае подозрений на преднамеренную порчу или опасное управление оборудованием организаторы оставляют за собой право оштрафовать команду нарушителей на баллы или вовсе снять их с соревнований.
7. В случае возникновения опасности повреждения оборудования организаторы имеют право экстренно прервать исполнение программы участников. Выпол-

нение попытки будет остановлено в тот же момент, на этом она завершится. Баллы за такую попытку присваиваются в соответствии с правилами, но на усмотрение судей.

8. Проверка решений будет происходить как на реальном роботе, так и в отдельной виртуальной системе или в системе с подобными зависимостями, например, с такими же версиями используемых библиотек.
9. Участники не могут использовать помощь тренера, сопровождающего лица или привлекать третьих лиц для решения задачи.
10. Финальная задача формулируется участникам в первый день финального тура, но участники выполняют решение задачи поэтапно. Критерии прохождения каждого этапа формулируются для каждого дня финального тура. За подзадачи, решенные в конкретном этапе, начисляются баллы. Баллы за подзадачи можно получить только в день, закреплённый за конкретным этапом.
11. Во время рабочего времени команды могут проводить испытания путём записи в очередь при помощи известного участникам канала связи с организаторами. Организаторы по просьбе участников могут предоставить участникам какие-либо артефакты оставшиеся после испытаний. Во время испытаний допускается вывод произвольных данных в консоль, в отличие от приемочных запусков.
12. Каждый день финального тура в 15:00 по МСК (может варьироваться в зависимости от расписания) команда должна загрузить свое решение на gitlab.com в соответствующий репозиторий дня, внутри своей группы, доступ к которой участники получат в начале олимпиады. Время может изменяться и зависит от количества команд и сложности подзадач, принимаемых в конкретный этап.
13. До истечения указанного времени команды могут изменять файл с решением сколько угодно раз. Проверяться будет всегда только последняя доступная версия.
14. Необходимо зафиксировать последнее решение - создать Merge Request (MR).
15. При создании Merge Request в качестве ответственного (assignee) укажите того, кто будет отвечать за приемку результатов.
16. После момента, когда все отправили свои решения на проверку, судьи приступают к проверке отправленных решений и подзадач, закрепленных за этапом конкретного дня финального тура.
17. Может быть предусмотрено до двух попыток сдачи решения одной и той же подзадачи в симуляторе. Конкретное количество попыток определяется в конкретных подзадачах.
18. После прохождения приемочных запусков, баллы, набранные командой, заносятся судьями в протокол.
19. Робот должен выполнять задание полностью автономно. Удаленное управление не допускается.
20. Во время сдачи не допускается вывод в консоль ничего, кроме информации, требуемой в задаче. В случае вывода иной информации, она может быть расценена как часть ответа.
21. Если во время приемочных запусков у судьи возникает ситуация, когда он не может однозначно решить выполняются ли критерии решения подзадачи, он вправе принять решение не в пользу команды.
22. В случае, если возникает техническая проблема, независящая от участников, то по решению судей может быть предоставлена возможность перезапуска.

23. Результат тестирования участникам будет сообщен на следующий день после сдачи решения.
24. Команды могут до 10:00 МСК обратиться с апелляцией. Жюри вправе провести тестирование или видео воспроизведение попытки (если таковая имеется) и назначить баллы за соответствующие подзадачи.

1.6. Процедура проведения приемочных запусков и критерии оценки

Первый этап

1. В качестве задачи для симулятора участникам необходимо выполнить следующее:
   1.1. Необходимо определить цвета блоков с самой большой и самой маленькой координатой по оси Y любой части блока, получая на вход облако точек.
   1.2. *Входные данные:* через файл `input.ply` управляющей программе передается облако точек.
   1.3. *Ожидаемый результат:* В результате работы программы в консоль выведено 2 строки:
      1.3.1. Цвет блока с наибольшей координатой центра
      1.3.2. Цвет блока с наименьшей координатой центра
      В качестве цвета необходимо вывести “RED” или “BLUE”.
   1.4. Имя файла с управляющей программой для проверки решения в симуляторе: `sim.py`.
   1.5. Баллы за решение задачи:
      1.5.1. Алгоритм успешно решил задачу X раз из N тестов — $13\frac{X}{N}$ баллов.
2. Командам необходимо подготовить следующую задачу для робота:
   2.1. Робот с известной конфигурацией начинает свое движение. В достижимой для робота области расположены 2 блока в зоне начального расположения и 2 места хранения. Известно расположение объектов, но не их начальная ориентация. Место хранения — обозначенная область, имеющая форму квадрата с длинной стороны равной 5см и постоянные координаты, а также заранее известный цвет. Роботу необходимо распределить объекты по различным местам согласно их цвету.
   2.2. Имя файла с управляющей программой для проверки решения на роботе: `main.py`.
   2.3. Максимальное время выполнения одной попытки для задачи — 5 минут.
   2.4. На демонстрацию решения предоставляется 2 попытки.
   2.5. Баллы за решение подзадач этапа, связанных с роботом:
      2.5.1. Робот смог определить цвета объектов и вывел их в одной строке через пробел, начиная с дальнего от робота объекта. В качестве цвета необходимо вывести “RED” или “BLUE”. — 3 балла.
      2.5.2. Робот смог захватить и поднять первый объект. — 6 баллов.
      2.5.3. Робот успешно расположил первый объект в место любого цвета. — 5 баллов.

2.5.4. Робот успешно расположил первый объект в место верного цвета. — 6 баллов.

2.5.5. Робот смог захватить и поднять второй объект. — 6 баллов.

2.5.6. Робот успешно расположил второй объект в место любого цвета. — 5 баллов.

2.5.7. Робот успешно расположил второй объект в место верного цвета. — 6 баллов.

2.6. Баллы за все попытки в каждой подзадаче суммируются.

2.7. Выполнение всех критериев (не обязательно на максимум баллов) в каждой из двух попыток задачи на реальном роботе дает дополнительные 5 баллов.

3. Все попытки осуществляются 15 марта.
4. Merge Request необходимо назвать следующим образом: “код команды_day1”. Например: “irs202200_day1”.
5. Максимальное количество баллов за этап — 92.

Второй этап

1. В качестве задачи для симулятора участникам необходимо выполнить следующее:
   1.1. Необходимо определить количество синих и красных объектов. Гарантируется, что все блоки видны с камеры. Однако допускается их частичное перекрытие, иначе говоря, с камеры объекты могут быть видны не полностью.
   1.2. *Входные данные:* через файл `input.ply` управляющей программе передается облако точек.
   1.3. *Ожидаемый результат:* В результате работы программы в консоль выведено 2 целых числа через пробел:
      1.3.1. Количество синих объектов
      1.3.2. Количество красных объектов
   1.4. Имя файла с управляющей программой для проверки решения в симуляторе: `sim.py`.
   1.5. Баллы за решение задачи:
      1.5.1. Алгоритм успешно решил задачу X раз из N тестов —$15\frac{X}{N}$ баллов.
2. Командам необходимо подготовить следующую задачу для робота:
   2.1. Робот с известной конфигурацией начинает свое движение. В начальной зоне, достижимой роботом, располагается 8 объектов двух цветов. Данные объекты располагаются навалом, что означает что у них может быть случайная ориентация и местоположение в данной зоне. Также допускается их расположение друг на друге. Необходимо распределить данные объекты в два заранее известных места. Распределение необходимо выполнить на основании цвета объектов.
   2.2. Имя файла с управляющей программой для проверки решения на роботе: `main.py`.
   2.3. Максимальное время выполнения одной попытки для задачи — 5 минут.
   2.4. На демонстрацию решения предоставляется 2 попытки.

2.5. Баллы за решение подзадач этапа, связанных с роботом:
   2.5.1. Робот смог переместить один объект в любое из заранее известных мест — 2 балла за объект. Максимальное количество баллов за подзадачу — 16 баллов.
   2.5.2. Робот смог переместить один объект в место верного цвета — 2 балла за объект. Максимальное количество баллов за подзадачу — 16 баллов.
   2.5.3. Робот смог убрать все объекты из начальной зоны — 3 балла.
2.6. Баллы за все попытки в каждой подзадаче суммируются.
2.7. Выполнение всех критериев (не обязательно на максимум баллов) в каждой из двух попыток задачи на реальном роботе дает дополнительные 6 баллов.

3. Все попытки осуществляются 16 марта.
4. Merge Request необходимо назвать следующим образом: “код команды_day2”. Например: “irs202200_day2”.
5. Максимальное количество баллов за этап — 91.

Третий этап

1. В качестве задачи для симулятора участникам необходимо выполнить следующее:
   1.1. Необходимо определить в какой координатной четверти плоскости XY находится новый объект. Гарантируется что данный объект находится полностью в этой четверти.
   1.2. *Входные данные:* через файл `input.ply` управляющей программе передается облако точек.
   1.3. *Ожидаемый результат:* В результате работы программы в консоль выведено целое число — номер координатной четверти.
   1.4. Имя файла с управляющей программой для проверки решения в симуляторе: `sim.py`.
   1.5. Баллы за решение задачи:
      1.5.1. Алгоритм успешно решил задачу X раз из N тестов —$15\frac{X}{N}$баллов.
2. Командам необходимо подготовить следующую задачу для робота:
   2.1. Робот с известной конфигурацией начинает свое движение. В начальной зоне, достижимой роботом, располагается 8 объектов двух цветов. В начальной зоне также появляется объект отличающийся от остальных по форме, необходимо оставить его в первоначальной зоне. Данные объекты располагаются навалом, что означает что у них может быть случайная ориентация и местоположение в данной зоне. Также допускается их расположение друг на друге. Необходимо распределить данные объекты в два заранее известных места. Распределение необходимо выполнить на основании цвета объектов.
   2.2. Имя файла с управляющей программой для проверки решения на роботе: `main.py`.
   2.3. Максимальное время выполнения одной попытки для задачи — 5 минут.
   2.4. На демонстрацию решения предоставляется 2 попытки.
   2.5. Баллы за решение подзадач этапа, связанных с роботом:

2.5.1. Робот смог переместить один объект в любое из заранее известных мест — 2 балла за объект. Максимальное количество баллов за подзадачу — 16 баллов.

2.5.2. Робот смог переместить один объект в место верного цвета — 2 балла за объект. Максимальное количество баллов за подзадачу — 16 баллов.

2.5.3. Робот смог убрать все объекты из начальной зоны (не считая объекта нового вида) — 3 балла.

2.5.4. Отличающийся объект остался в начальной зоне. Баллы начисляются даже если 0 баллов за иное. — 6 баллов.

2.6. Баллы за все попытки в каждой подзадаче суммируются.

2.7. Выполнение всех критериев (не обязательно на максимум баллов) в каждой из двух попыток задачи на реальном роботе дает дополнительные 3 балла.

3. Все попытки осуществляются 17 марта.
4. Merge Request необходимо назвать следующим образом: “код команды_day3”. Например: “irs202200_day3”.
5. Максимальное количество баллов за этап — 100.

Четвёртый этап

1. В качестве задачи для симулятора участникам необходимо выполнить следующее:
   1.1. Необходимо посчитать количество объектов, которые не касаются основания. Гарантируется что все кубики видны на камере хотя бы частично.
   1.2. *Входные данные:* через файл `input.ply` управляющей программе передается облако точек.
   1.3. *Ожидаемый результат:* В результате работы программы в консоль выведено целое число — количество объектов, которые не касаются основания.
   1.4. Имя файла с управляющей программой для проверки решения в симуляторе: `sim.py`.
   1.5. Баллы за решение задачи:
      1.5.1. Алгоритм успешно решил задачу X раз из N тестов — $20\frac{X}{N}$ баллов.
2. Командам необходимо подготовить следующую задачу для робота:
   2.1. Робот с известной конфигурацией начинает свое движение. В начальной зоне, достижимой роботом, располагается 10 объектов двух цветов, из них 2 объекта отличаются по форме. Данные объекты располагаются навалом, что означает что у них может быть случайная ориентация и местоположение в данной зоне. Также допускается их расположение друг на друге. Необходимо распределить данные объекты в два заранее известных места. Распределение необходимо выполнить на основании цвета объектов, включая новые объекты. Необходимо сложить сортируемые объекты в виде башни.
   2.2. Имя файла с управляющей программой для проверки решения на роботе: `main.py`.
   2.3. Максимальное время выполнения одной попытки для задачи — 5 минут.

2.4. На демонстрацию решения предоставляется 2 попытки.

2.5. Баллы за решение подзадач этапа, связанных с роботом:

2.5.1. Робот смог переместить один объект в любое из заранее известных мест — 2 балла за объект. Максимальное количестиво баллов за подзадачу — 20 баллов.

2.5.2. В данном месте располагаются объекты одного цвета — 4 балла за место. Максимальное количество баллов за подзадачу — 8 баллов.

2.5.3. Робот смог переместить один объект в любое из заранее известных мест. Данный объект не касается основания, иначе говоря опирается только на другие объекты, своего цвета — 2 балла. Максимальное количество баллов за подзадачу — 16 баллов.

2.5.4. Робот смог убрать все объекты из начальной зоны — 3 балла.

2.6. Баллы за все попытки в каждой подзадаче суммируются.

2.7. Выполнение всех критериев (не обязательно на максимум баллов) в каждой из двух попыток задачи на реальном роботе дает дополнительные 3 балла.

3. Все попытки осуществляются 18 марта.
4. Merge Request необходимо назвать следующим образом: “код команды_day4”. Например: “irs202200_day4”.
5. Максимальное количество баллов за этап — 117.

1.7. Критерии определения команды-победителя командного тура

1. Максимальный балл за командный этап — 400 баллов.
2. Сумма баллов, набранных за решения подзадач всех этапов командной части финального этапа, определяет итоговую результативность команды (измеряемую в баллах).
3. Команды ранжируются по результативности.
4. Команда-победитель определяется как команда с максимальной результативностью.