

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

УНИВЕРСИТЕТ (РОСБИОТЕХ)

Федеральное государственное бюджетное образовательное учреждение высшего образования
«РОССИЙСКИЙ БИОТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ (РОСБИОТЕХ)»

ФОНД ОЦЕНОЧНЫХ СРЕДСТВ ПО МДК 03.02

«Программирование робототехнических систем»

Уровень образования:	Среднее профессиональное образование
Специальность	15.02.10 Мехатроника и робототехника (по отраслям)
Квалификация	специалист по мехатронике и робототехнике
Форма обучения	Очная
Срок освоения образовательной программы в соответствии с ФГОС (очная форма)	2 г. 10 м. (на базе среднего общего образования)
Год начала подготовки	2026 г.
Период освоения дисциплины	4, 5 семестр
Форма контроля	Зачёт, экзамен.

1. Область применения.

Фонд оценочных средств (ФОС) является неотъемлемой частью программы дисциплины при реализации программы подготовки специалистов среднего звена (ППСЗ) среднего профессионального образования (СПО) по специальности:

15.02.10 МЕХАТРОНИКА И РОБОТОТЕХНИКА (ПО ОТРАСЛЯМ)

Оценочные фонды разрабатываются для проведения оценки степени соответствия фактических результатов обучения при изучении дисциплины запланированным результатам обучения, соотнесенных с установленными в программе индикаторами достижения компетенций, а также сформированности компетенций, установленных программой подготовки специалистов среднего звена.

Таблица 1
Паспорт фонда оценочных средств

Компетенции	Знать:	Уметь:	Владеть навыками (иметь практический опыт):
ПК 2.1. Выявлять внешние дефекты узлов и агрегатов мехатронных устройств и систем в результате их внешнего осмотра	- виды и признаки внешних дефектов модулей и узлов мехатронных устройств и систем; - правила приемки и сдачи выполненных работ; меры безопасности при подготовке к работе узлов, агрегатов и электронных модулей мехатронных устройств и систем;	- выявлять внешние дефекты узлов и агрегатов мехатронных устройств и систем в результате их внешнего осмотра; - поддерживать состояние рабочего места при подготовке к работе узлов, агрегатов и электронных модулей мехатронных устройств и систем и проведении контроля их технического состояния в соответствии с требованиями электробезопасности, охраны труда, промышленной, экологической и пожарной безопасности;	- выявлять внешние дефекты узлов и агрегатов мехатронных устройств и систем в результате их внешнего осмотра; -проводить периодический контроль технического состояния механических узлов, электронных устройств управления, приводов, датчиков и кабелей мехатронных устройств и систем;
ПК 2.2. Проверять соответствие диагностируемых параметров узлов, агрегатов и электронных модулей мехатронных устройств и систем требованиям эксплуатационной документации	- способы и технические средства проверки работоспособности механических частей мехатронных устройств и систем; - способы и технические средства проверки работоспособности электронных модулей и устройств управления мехатронных устройств и систем;	- проверять соответствие рабочих характеристик узлов, агрегатов и электронных модулей мехатронных устройств и систем с применением измерительных приборов требованиям, указанным в эксплуатационной документации;	- проводить текущий контроль технического состояния механических узлов, электронных устройств управления, приводов, датчиков и кабелей мехатронных устройств и систем;
ПК 2.3. Проводить контроль работоспособности программного обеспечения электронных устройств управления, приводов и датчиков мехатронных устройств и систем	- способы и технические средства проверки работоспособности датчиков мехатронных устройств и систем; - способы и технические средства проверки работоспособности исполнительных двигателей мехатронных устройств и систем;	- проверять соответствие рабочих характеристик узлов, агрегатов и электронных модулей мехатронных устройств и систем с применением измерительных приборов требованиям, указанным в эксплуатационной документации;	- составлять ведомости выявленных дефектов; - проверять соответствия диагностируемых параметров узлов, агрегатов и электронных модулей мехатронных устройств и систем требованиям эксплуатационной документации;
ПК 2.4. Выявлять отработавшие ресурс или вышедшие из строя компоненты мехатронных устройств и систем	- способы и технические средства проверки работоспособности датчиков мехатронных устройств и систем; - способы и технические средства проверки работоспособности исполнительных двигателей мехатронных устройств и систем; - CADсистемы: классы,	- проверять соответствие рабочих характеристик узлов, агрегатов и электронных модулей мехатронных устройств и систем с применением измерительных приборов требованиям, указанным в эксплуатационной документации;	- проверять соответствия диагностируемых параметров узлов, агрегатов и электронных модулей мехатронных устройств и систем требованиям эксплуатационной документации;

<p>ПК 2.5. Заменять отработавшие ресурс или вышедшие из строя компоненты мехатронных устройств и систем</p> <p>ПК 2.6. Проводить контроль корректности работы и обновление программного обеспечения мехатронных устройств и систем</p> <p>ПК 2.7. Проводить текущее техническое обслуживание узлов и агрегатов мехатронных устройств и систем</p>	<p>наименования, - возможности и порядок работы в них; содержание эксплуатационной документации на узлы и агрегаты мехатронных устройств и систем, руководств по установке программного обеспечения; - специализированное программное обеспечение, применяемое для чтения журналов параметров состояния программного обеспечения узлов, агрегатов и электронных модулей мехатронных устройств и систем; - способы определения отработавших ресурс или вышедших из строя составных частей мехатронных устройств и систем классификацию и виды отказов оборудования; алгоритмы поиска неисправностей; - виды и методы контроля и испытаний, методику их проведения и сопроводительную документацию; - стандарты, положения, методические и другие нормативные материалы по аттестации, испытаниям, эксплуатации и ремонту оборудования мехатронных систем; - понятие, цель и функции технической диагностики; методы диагностирования, неразрушающие методы контроля; - физические принципы работы, конструкцию, технические характеристики, области применения, правила эксплуатации оборудования мехатронных систем; - порядок проведения стандартных и</p>	<p>внесение изменений в очередность работ, отмечать выполнение работ, готовить отчеты о выполненных работах с использованием прикладных программ управления проектами; - читать файловые отчеты о параметрах работы программного обеспечения электронных устройств управления, приводов и датчиков мехатронных устройств и систем; - проверять соответствие параметров работы программного обеспечения электронных устройств управления, приводов и датчиков мехатронных устройств и систем требованиям, указанным в эксплуатационной документации; выявлять вышедшие из строя составные части мехатронных устройств и систем; - поддерживать состояние рабочего места при проведении технического обслуживания в соответствии с требованиями электробезопасности, охраны труда, промышленной, экологической и пожарной безопасности; - разрабатывать мероприятия по устранению причин отказов и обнаружению дефектов оборудования мехатронных систем; - применять соответствующие методики контроля, испытаний и диагностики оборудования мехатронных систем; - обнаруживать неисправности мехатронных систем; производить диагностику оборудования мехатронных</p>	<p>управления, приводов и датчиков мехатронных устройств и систем; - Проводить текущий контроль работоспособности программного обеспечения электронных устройств управления, приводов и датчиков мехатронных устройств и систем; - выявлять отработавшие ресурс или вышедшие из строя детали механических узлов и агрегатов мехатронных устройств и систем; - выявлять отработавшие ресурс или вышедшие из строя блоки и модули электронных устройств управления; - выявлять отработавшие ресурс или вышедшие из строя компоненты приводов мехатронных устройств и систем; - выявлять отработавшие ресурс или вышедших из строя кабелей; заменять отработавшие ресурс или вышедшие из строя детали механических узлов и агрегатов мехатронных устройств и систем; - заменять отработавшие ресурс или вышедших из строя блоки и модули электронных устройств управления; - заменять отработавшие ресурс или вышедших из строя компоненты приводов мехатронных устройств и систем; - замена отработавшие ресурс или вышедших из строя кабели; контролировать корректности работы программного обеспечения мехатронных устройств и систем;</p>
---	---	--	---

	<p>сертифицированных испытаний;</p> <ul style="list-style-type: none"> - методы повышения долговечности оборудования; -технологические процессы ремонта и восстановления деталей и оборудования мехатронных систем; -технологическую последовательность разборки, ремонта и сборки узлов и механизмов мехатронных систем; - CAD-системы: классы, наименования, возможности и порядок работы в них; - прикладные программы управления проектами: наименования, возможности и порядок работы в них; - принципы работы и обновления программного обеспечения узлов, агрегатов, блоков и модулей мехатронных устройств и систем; -контрольно-измерительные приборы для определения технического состояния узлов, агрегатов, блоков и модулей мехатронных устройств и систем; 	<p>систем и определение его ресурсов;</p> <ul style="list-style-type: none"> - оформлять документацию по результатам диагностики мехатронных систем; - заменять вышедшие из строя составные части мехатронных устройств и систем на исправные; - контролировать и обеспечивать надежность закрепления механических узлов и агрегатов мехатронных устройств и систем; - производить разборку и сборку гидравлических, пневматических, электромеханических устройств мехатронных систем; - выявлять необходимость в обновлении и обновлять программное обеспечение мехатронных устройств и систем; - читать эксплуатационную документацию на мехатронные устройства и системы и их программное обеспечение; -контролировать соответствие условий эксплуатации мехатронных устройств и систем; - чистить и смазывать механические узлы и агрегаты мехатронных устройств и систем; - контролировать и обеспечивать надежность закрепления механических узлов и агрегатов мехатронных устройств и систем; -обеспечивать безопасность работ при ремонте, техническом обслуживании, контроле и испытаниях оборудования мехатронных систем; - применять технологии бережливого производства при организации и выполнении работ по 	<ul style="list-style-type: none"> - обновлять программное обеспечение мехатронных устройств и систем; - вести журнал учета технического обслуживания узлов и агрегатов мехатронных устройств и систем, обновления программного обеспечения; -проводить периодический контроль соблюдения условий эксплуатации мехатронных устройств и систем; - проводить текущее техническое обслуживание узлов и агрегатов мехатронных устройств и систем; - вести журнал учета технического обслуживания узлов и агрегатов мехатронных устройств и систем, обновления программного обеспечения.
--	---	--	--

		техническому обслуживанию, контролю и испытаниям мехатронных систем.	
--	--	--	--

Цели и задачи фонда оценочных средств.

Целью ФОС является установление соответствия уровня подготовки обучающихся требованиям федерального государственного образовательного стандарта ФГОС СПО по ОПОП.

ФОС предназначен для решения задач контроля достижения целей реализации ОПОП СПО и обеспечения соответствия результатов обучения области, сфере, объектам профессиональной деятельности, области знаний и типам задач профессиональной деятельности.

СТРУКТУРА И СОДЕРЖАНИЕ УЧЕБНОЙ ДИСЦИПЛИНЫ

Распределение часов дисциплины по семестрам

Семестр (<Курс>.<Семестр на курсе>)	4(2.2)		5(3.1)		Итого	
Вид занятий	УП	РП	УП	РП	УП	РП
Лекции	16	16	16	16	32	32
Лабораторные	20	20	20	20	40	40
Итого ауд.	36	36	36	36	72	72
Контактная работа	36	36	36	36	72	72
Сам. работа	12	12	4	4	16	16
Итого	50	50	42	42	92	92

Задания для промежуточной аттестации с ключами ответов

№ вопроса	Формулировка тестовых заданий	Варианты ответов	Правильный ответ
1.	Какой тип данных чаще всего используется в ROS (Robot Operating System) для представления одометрии робота?	A) std::string B) sensor_msgs/Image C) nav_msgs/Odometry D) geometry_msgs/Twist	C) nav_msgs/Odometry
2.	Что такое обратная кинематика в робототехнике?	A) Определение скоростей двигателей для достижения заданной скорости B) Расчет требуемых положений сочленений для достижения заданной позиции и ориентации рабочего органа C) Расчет сил и моментов, действующих на звенья манипулятора D) Определение траектории движения рабочего органа во времени	B) Расчет требуемых положений сочленений для достижения заданной позиции и ориентации рабочего органа
3.	Для чего в управлении	A) Для планирования траектории	B) Для

	роботом используется ПИД-регулятор?	В) Для стабилизации системы и точного достижения целевого состояния С) Для обработки изображений с камеры D) Для построения карты окружающей среды	стабилизации системы и точного достижения целевого состояния
4.	Что описывает URDF-файл в ROS??	A) Алгоритм навигации B) Протокол обмена данными C) Модель робота (геометрию, кинематические и динамические свойства) D) Графический интерфейс	C) Модель робота (геометрию, кинематические и динамические свойства)
5.	Какой алгоритм SLAM является одним из самых распространенных для построения карты по данным с лидара?	A) A B) D C) Gmapping D) Dijkstra	C) Gmapping
6.	Какая структура данных оптимальна для хранения разряженной карты занятости (occupancy grid) в ROS?	A) Одномерный массив B) Двумерный массив C) OctoMap D) Квиздерево	C) OctoMap
7.	Что такое "трансформ" (transform) в контексте ROS?	A) Переход от одной системы координат к другой B) Изменение типа сообщения C) Фильтрация данных с датчика D) Процедура калибровки камеры	A) Переход от одной системы координат к другой
8.	Какой пакет ROS предоставляет функционал для движения робота к цели с учетом динамических препятствий?	A) amcl B) move_base C) gmapping D) roscore	B) move_base
9.	Для чего используется алгоритм RRT (Rapidly-exploring Random Tree) в робототехнике?	A) Для распознавания объектов B) Для планирования пути в пространстве конфигураций C) Для сжатия данных с датчиков D) Для управления усилием в схвате	B) Для планирования пути в пространстве конфигураций
10.	Какой тип контроллера чаще всего используется для управления сервомоторами (например, Dynamixel)?	A) Пропорциональный (P) B) Пропорционально-интегральный (PI) C) Пропорционально-дифференциальный (PD) D) Пропорционально-интегрально-дифференциальный (PID)	D) Пропорционально-интегрально-дифференциальный (PID)
11.	Что делает нода ROS с параметром "respawn=true" в launch-файле?	A) Запускается только при наличии свободных ресурсов B) Запускается с пониженным приоритетом C) Автоматически	C) Автоматически перезапускается при завершении работы

		перезапускается при завершении работы D) Отправляет диагностические сообщения	
12.	Какая библиотека C++ является стандартом для линейной алгебры в робототехнических расчетах (работа с матрицами, векторами)?	A) OpenCV B) PCL C) Eigen D) Boost	C) Eigen
13.	Что такое "дрейф" в контексте инерциальных измерительных систем (IMU)?	A) Резкое изменение показаний B) Постепенное накопление ошибки при интегрировании сигнала C) Потеря связи с датчиком D) Калибровка в нулевое положение	B) Постепенное накопление ошибки при интегрировании сигнала
14.	Для чего используется метод Монте-Карло локализации (AMCL)?	A) Для калибровки камеры и лидара B) Для оценки позы робота на известной карте C) Для обучения с подкреплением D) Для фильтрации шума с дальнометров	B) Для оценки позы робота на известной карте
15.	Что такое "обратное распространение ошибки" в нейросетях?	A) Алгоритм прямого прохода сигнала B) Метод настройки весов сети на основе градиента функции потерь C) Способ визуализации работы сети D) Метод снижения размерности данных	B) Метод настройки весов сети на основе градиента функции потерь
16.	Соотнесите датчик робота с типом данных, которые он предоставляет:	1. Одометрия колес 2. Инерциальный измерительный блок (IMU) 3. 2D Лидар 4. Стереокамера A) Точки облака в полярных координатах (дальность, угол) B) Смещение и угол поворота робота относительно начальной точки C) Угловая скорость, линейное ускорение, ориентация D) Глубина изображения (disparity map)	1-B, 2-C, 3-A, 4-D
17.	Соотнесите концепцию ROS с ее определением:	1. Нода (Node) 2. Топик (Topic) 3. Сервис (Service)	1-C, 2-B, 3-D, 4-A

		<p>4. Действие (Action)</p> <p>A) Асинхронный вызов с возможностью отмены и обратной связью о ходе выполнения</p> <p>B) Канал для однонаправленной потоковой передачи сообщений</p> <p>C) Исполняемый файл, который общается с другими нодами через ROS</p> <p>D) Синхронный запрос-ответ между двумя нодами</p>	
18.	Соотнесите компонент робота-манипулятора с его описанием:	<p>1. Звено (Link)</p> <p>2. Сочленение (Joint)</p> <p>3. Рабочий орган (End-Effector)</p> <p>4. Привод (Actuator)</p> <p>A) Механическое соединение, обеспечивающее относительное движение между двумя звеньями</p> <p>B) Устройство, преобразующее электрическую/гидравлическую энергию в механическое движение</p> <p>C) Ригидная часть манипулятора, обладающая массой и инерционными свойствами</p> <p>D) Инструмент, закрепленный на конечном звене (схват, сварочная горелка и т.д.)</p>	1-C, 2-A, 3-D, 4-B
19.	Соотнесите алгоритм с областью его применения в робототехнике:	<p>1. Алгоритм Дейкстры</p> <p>2. Алгоритм RANSAC</p> <p>3. Алгоритм ICP (Iterative Closest Point)</p> <p>4. Алгоритм обратной кинематики для руки человека (например, CCD)</p> <p>A) Поиск кратчайшего пути на графе</p> <p>B) Совмещение двух облаков точек</p> <p>C) Выделение геометрических примитивов из данных датчиков</p> <p>D) Поиск углов суставов для приближения к цели в декартовом пространстве</p>	1-A, 2-C, 3-B, 4-D

20.	Соотнесите проблему робототехники с типовым способом ее решения:	<p>1. Проблема локального минимума при планировании пути</p> <p>2. Неточность модели робота и окружения</p> <p>3. Вычислительная сложность планирования в пространстве высокой размерности</p> <p>4. Зашумленность данных с датчиков</p> <p>А) Использование рандомизированных алгоритмов (RRT, PRM)</p> <p>В) Применение рекурсивных фильтров (Калмана, частиц)</p> <p>С) Использование эвристик и техник вроде "виртуальных полей отталкивания"</p> <p>Д) Введение адаптивных и robust-алгоритмов управления</p>	1-C, 2-D, 3-A, 4-B
21.	Опишите основную идею и основные шаги работы алгоритма А для планирования пути на дискретной карте.		<p>А: Комбинация стоимости пройденного пути (g) и эвристической оценки до цели (h). Работает на графе/сетке, поддерживает список открытых и закрытых вершин, выбирает вершину с минимальным $f = g + h$, обновляет соседей.</p>
22.	Объясните разницу между управлением роботом по положению (position control) и по усилию (force control). Приведите по одному практическому примеру использования каждого подхода.		<p>Position control: Точное следование заданной траектории в пространстве конфигураций (например, сварка роботом). Force control: Управление усилием взаимодействия с объектом (например, полировка, сборка с натягом).</p>

23.	Что такое система координат Денавита-Хартенберга (DH) и для чего она применяется при программировании роботов-манипуляторов?		DH-параметры: Стандартный способ задания систем координат на звеньях манипулятора (4 параметра: a , α , d , θ) для упрощения вычисления прямой кинематики.
24	Перечислите и кратко опишите основные компоненты (ноды), из которых состоит типичный стек навигации move_base в ROS.		move_base: global_planner (планировщик по глобальной карте), local_planner (локальная траектория с учетом препятствий), global_costmap, local_costmap, recovery_behaviors.
25.	Объясните, что такое "карта затрат" (costmap) в навигационном стеке ROS и как она используется для обхода препятствий.		Costmap: Растровая карта, где каждая ячейка имеет "стоимость" прохождения. Использует данные датчиков. Планировщик строит путь, минимизирующий суммарную стоимость, обходя зоны с высокой стоимостью (препятствия).

Примерные контрольные вопросы для зачёта и экзамена:

1. Что такое ROS и каковы её основные преимущества перед традиционными подходами к программированию роботов?
2. Объясните разницу между топиками (topics) и сервисами (services) в ROS. В каких случаях использовать каждый подход?
3. Для чего используется URDF-файл в робототехнике? Какие основные компоненты он описывает?
4. Что такое система трансформаций (tf) в ROS и почему она критически важна для навигации робота?

5. Опишите жизненный цикл разработки ПО для робототехнической системы от проектирования до тестирования.
6. Объясните разницу между прямой и обратной кинематикой манипулятора. Приведите практический пример использования каждой.
7. Что такое ПИД-регулятор? Опишите физический смысл каждой составляющей и где она применяется в управлении роботом.
8. Какие основные типы приводов используются в робототехнике? Сравните их по точности, мощности и сложности управления.
9. Что такое дифференциальный привод и как рассчитывается одометрия для мобильного робота с такой кинематикой?
10. Объясните понятие "особенностей" (singularities) в кинематике манипулятора и как они влияют на управление.
11. Классифицируйте основные типы датчиков в робототехнике. Приведите по 2 примера каждого типа и их назначение.
12. Как работает ультразвуковой дальномер HC-SR04? В каких задачах он эффективен, а в каких — нет?
13. Что такое сенсорная интеграция (sensor fusion) и зачем она нужна? Приведите пример использования фильтра Калмана.
14. Опишите базовый конвейер обработки изображений для задачи обнаружения объекта по цвету.
15. Как лидар используется для построения карты занятости (occupancy grid)? Опишите основные шаги.
16. Дайте определение SLAM. Почему эта задача фундаментальна для автономных роботов?
17. Объясните алгоритм A* для планирования пути. В чём его сильные и слабые стороны?
18. Что такое глобальный и локальный планировщики? Как они взаимодействуют в навигационном стеке ROS move_base?
19. Опишите метод динамического окна (DWA) для локального планирования. Какие ограничения робота он учитывает?
20. Как работает адаптивная Монте-Карло локализация (AMCL)? В чём преимущество фильтра частиц перед другими методами?
21. Разработайте архитектуру ПО для мобильного робота-курьера внутри здания. Какие модули необходимы?
22. Как организовать взаимодействие между высокоуровневым планировщиком и низкоуровневым контроллером двигателей?
23. Что такое конечный автомат для управления поведением робота? Приведите пример для робота, выполняющего задание "найти и принести объект".
24. Опишите методы отладки робототехнических систем. Какие инструменты ROS вы бы использовали?
25. Как обеспечить безопасность при программировании коллаборативного робота (кобота), работающего рядом с человеком?
26. Проанализируйте ошибки в данном фрагменте кода управления двигателями и предложите исправления.
27. Разработайте алгоритм следования по линии для робота с 3 датчиками. Опишите логику для каждого состояния.
28. Как реализовать объезд статического препятствия роботом с ультразвуковым датчиком? Опишите алгоритм.

29. Предложите сенсорную комплектацию для робота-пылесоса. Обоснуйте выбор каждого датчика.
30. Опишите полный цикл задачи "захват объекта манипулятором" от обнаружения до выполнения.

Критерии и шкалы оценивания.

Текущий контроль по дисциплине

Оценивание обучающегося на занятиях осуществляется в соответствии с локальным актом университета (положением), регламентирующим проведение текущего контроля успеваемости и промежуточной аттестации обучающихся и организации учебного процесса с применением балльно-рейтинговой системы оценки качества обучения.

Промежуточная аттестация по дисциплине

Форма промежуточной аттестации – Зачёт и Экзамен.

Оценка *«отлично»* выставляется обучающемуся, если дан полный, развернутый ответ на поставленный вопрос, системно показана совокупность освоенных знаний об объекте, проявляющаяся в свободном оперировании понятиями, умении выделить существенные и несущественные его признаки, причинно-следственные связи. Ответ формулируется при помощи научного категориально-понятийного аппарата, изложен последовательно, логично, доказательно, демонстрирует авторскую позицию студента.

Оценка *«хорошо»* выставляется обучающемуся, если дан полный, развернутый ответ на поставленный вопрос, показана совокупность осознанных знаний об объекте, доказательно раскрыты основные положения темы; в ответе прослеживается четкая структура, логическая последовательность, отражающая сущность раскрываемых понятий, теорий, явлений. Ответ изложен последовательно, логично и доказательно, однако допущены недочеты в определении понятий, исправленные студентом самостоятельно в процессе ответа.

Оценка *«удовлетворительно»* выставляется обучающемуся, если дан полный, но недостаточно последовательный ответ на поставленный вопрос, но при этом показано умение выделить существенные и несущественные признаки и причинно-следственные связи. Ответ логичен и изложен научным языком. Могут быть допущены две-три ошибки в определении основных понятий, которые студент затрудняется исправить самостоятельно.

Оценка *«неудовлетворительно»* выставляется обучающемуся, если дан неполный ответ, представляющий собой разрозненные знания по теме вопроса с существенными ошибками в определениях. Присутствуют фрагментарность, нелогичность изложения. Студент не осознает связи между понятиями, концептуальные пересечения, структурные закономерности между различными объектами дисциплины. Отсутствуют выводы, конкретизация и доказательность изложения. Речь неграмотная. Дополнительные и уточняющие вопросы преподавателя не приводят к коррекции ответа студента не только на поставленный вопрос, но и на другие вопросы дисциплины.

Результат обучения по дисциплине считается достигнутым при получении обучающимся оценки «зачтено», «удовлетворительно», «хорошо», «отлично» по каждому из контрольных мероприятий, относящихся к данному результату обучения.

Критерии оценки образовательных результатов обучающихся на зачете по дисциплине

Качество освоения ОПОП рейтинговые баллы	Оценка зачета с оценкой зачета (нормативная) в 5-балльной шкале	Уровень достижений компетенций	Критерии оценки образовательных результатов
85-100	Зачтено, 5, отлично	Высокий (продвинутый)	<p>ЗАЧТЕНО, ОТЛИЧНО заслуживает обучающийся, обнаруживший всестороннее, систематическое и глубокое знание учебно-программного материала на занятиях и самостоятельной работе. При этом, рейтинговая оценка (средний балл) его текущей аттестации по дисциплине входит в диапазон 85-100. При этом, на занятиях, обучающийся исчерпывающе, последовательно, чётко и логически стройно излагал учебно-программный материал, умел тесно увязывать теорию с практикой, свободно справлялся с задачами, вопросами и другими видами применения знаний, предусмотренные программой. Причем обучающийся не затруднялся с ответом при видоизменении предложенных ему заданий, правильно обосновывал принятое решение, демонстрировал высокий уровень усвоения основной литературы и хорошо знакомство с дополнительной литературой, рекомендованной программой дисциплины.</p> <p>Как правило, оценку «отлично» выставляют обучающемуся, усвоившему взаимосвязь основных понятий дисциплины в их значение для приобретаемой профессии, проявившему творческие способности в понимании, изложении и использовании учебно-программного материала.</p> <p>Рейтинговые баллы назначаются обучающемуся с учётом баллов текущей (на занятиях) и (или) рубежной аттестации (контроле).</p>

70-84	Зачтено, 4, хорошо	Хороший (базовый)	<p>ЗАЧТЕНО, ХОРОШО заслуживает обучающийся, обнаруживший осознанное (твердое) знание учебно-программного материала на занятиях и самостоятельной работе. При этом, рейтинговая оценка (средний балл) его текущей аттестации по дисциплине входит в диапазон 70-84.</p> <p>На занятиях обучающийся грамотно и по существу излагал учебно-программный материал, не допускал существенных неточностей в ответе на вопрос, правильно применял теоретические положения при решении практических вопросов и задач, владел необходимыми навыками и приёмами их выполнения, уверенно демонстрировал хороший уровень усвоения основной литературы и достаточное знакомство с дополнительной литературой, рекомендованной программой дисциплины.</p> <p>Как правило, оценку «хорошо» выставляют обучающемуся, показавшему систематический характер знаний по дисциплине и способным к их самостоятельному пополнению и обновлению в ходе дальнейшей учебной работы и профессиональной деятельности.</p> <p>Рейтинговые баллы назначаются обучающемуся с учётом баллов текущей (на занятиях) и (или) рубежной аттестации (контроле).</p>
-------	--------------------	----------------------	---

60-69	Зачтено, 3, удовлетворительно	Достаточный (минимальный)	<p>ЗАЧТЕНО, УДОВЛЕТВОРИТЕЛЬНО заслуживает обучающийся, обнаруживший минимальные (достаточные) знания учебно-программного материала на занятиях и самостоятельной работе. При этом, рейтинговая оценка (средний балл) его текущей аттестации по дисциплине входит в диапазон 60-69.</p> <p>На занятиях обучающийся демонстрирует знания только основного материала в объеме, необходимом для дальнейшей учебы и предстоящей профессиональной работы, слабое усвоение деталей, допускает неточности, в том числе в формулировках, нарушает логическую последовательность в изложении программного материала, испытывает затруднения при выполнении практических заданий и работ, знакомый с основной литературой, слабо (недостаточно) знаком с дополнительной литературой, рекомендованной программой.</p> <p>Как правило, оценку «удовлетворительно» выставляют обучающемуся, допускавшему погрешности в ответах на занятиях и при выполнении заданий, но обладающим необходимыми знаниями для их устранения под руководством преподавателя.</p> <p>Рейтинговые баллы назначаются обучающемуся с учётом баллов текущей (на занятиях) и (или) рубежной аттестации (контроле).</p>
-------	-------------------------------	---------------------------	--

Менее 60	Не зачтено, 2, неудовлетворительно	Недостаточный (ниже минимального)	НЕ ЗАЧТЕНО, НЕУДОВЛЕТВОРИТЕЛЬНО выставляется обучающемуся, который не знает большей части учебно-программного материала, допускает существенные ошибки, неуверенно, с большими затруднениями выполняет практические работы на занятиях и самостоятельной работе. Как правило, оценка «неудовлетворительно» ставится обучающемуся продемонстрировавшего отсутствие целостного представления по дисциплине, предмете, его взаимосвязях и иных компонентов. При этом, обучающийся не может продолжить обучение или приступить к профессиональной деятельности по окончании вуза без дополнительных занятий по соответствующей дисциплине. Компетенции, закреплённые за дисциплиной, сформированы на недостаточном уровне или не сформированы. Рейтинговые баллы назначаются обучающемуся с учётом баллов текущей (на занятиях) и (или) рубежной аттестации (контроле).
----------	------------------------------------	-----------------------------------	---

Промежуточная аттестация может проводиться в форме компьютерного тестирования. Обучающемуся отводится для подготовки ответа на один вопрос открытого и закрытого типа не менее 5 минут.

Итоговая оценка при проведении зачёта и экзамена выставляется с использованием следующей шкалы.

Оценка	Правильно решенные тестовые задания (%)
«отлично»	90-100
«хорошо»	66-89
«удовлетворительно»	50-65
«неудовлетворительно»	0-49

Примеры лабораторных работ

Варианты для лабораторных 1-4

Таблица вариантов			
№	№	Длина плеча	Плотность,
	L, м	Параметры Revolute_joint	кг/м ³
	1	2	3
			4
1	1	0.9	1) Min= 0, Range= 45; 2) Min= 110, Range= 70;
2	2	0.9	1) Min= -10, Range= 120; 2) Min= 120, Range= 60;
3	3	0.9	1) Min= -20, Range= 70; 2) Min= 90, Range= 90;
4	4	0.9	1) Min= -30, Range= 90; 2) Min= 100, Range= 80;
5	5	1.3	1) Min= 15, Range= 55; 2) Min= 120, Range= 60;
6	6	1.3	1) Min= 0, Range= 45; 2) Min= 110, Range= 70;
7	7	1.3	1) Min= -10, Range= 90; 2) Min= 90, Range= 90;
8	8	1.3	1) Min= -20, Range= 70; 2) Min= 100, Range= 80;
9	9	1.5	1) Min= 10, Range= 90; 2) Min= 110, Range= 70;
10	0	1.5	1) Min= 0, Range= 80; 2) Min= 120, Range= 60;
11	1	1.5	1) Min= -10, Range= 100; 2) Min= 80, Range= 100;
12	1	1.5	1) Min= -20, Range= 70; 2) Min= 100, Range= 80;

Лабораторная №1

1. Цель работы

- Научиться работать со сценой CoppeliaSim и объектами.
- Изучить работу с деревом объектов.
- Освоить ручное управление параметрами вращательного шарнира.

2. Теоретическая часть

CoppeliaSim — система физического моделирования и автоматизации робототехнических систем. Любая сцена включает:

- *Объекты* (модели, примитивы, механизмы)
- *Joint (шарниры)* — позволяют задавать вращение/движение
- *Свойства объектов*: положение, ориентация, масса, инерция
- *Дерево сцены (Scene Hierarchy)*

Revolute Joint — вращательное соединение с параметрами:

- *Min/Range* — диапазон угла поворота
- *Position* — текущий угол
- *Target position / Target velocity* — управление
- *Torque/Force* — ограничение силы привода

Умение настраивать параметры шарнира — базовая компетенция для программирования робототехнических систем.

3. Задание:

- 1) Получить вариант к выполнению лабораторной работы
- 2) Открыть файл-сцену “Lab1.Shlagbaum_###m.ttt” в соответствии с вариантом
- 3) Освоить приемы перемещения камеры и выбора объекта сцены (ЛКМ, средняя КНМ)
- 4) Изучить структуру модели и назначение её составных частей
- 5) Познакомится с соединением поворотного типа (*Revolute_joint*)
- 6) Для каждой пары значений *Pos. min*, *Pos. rang* , согласно варианту (см.таблицу вариантов, 3 столбец), провести серию экспериментов:
 - a. Вычислить значение максимального углового положения (*Pos. max*) и записать в отчет
 - b. Ввести значения *Pos. min*, *Pos. range* в окно настроек *Revolute_joint*
 - c. Изобразить диапазон, в котором разрешено вращение, как сектор на тригонометрической окружности и записать в отчет
 - d. Зафиксировать в отчете снимки экрана модели шлагбаума в различных положениях (крайние, промежуточные) изменяя параметр *Position [deg]*

4. Ход выполнения работы

Открыть файл-сцену “Lab1.Shlagbaum_###m.ttt”. Для этого используйте пункт меню “File-> Open scene...” и выберите файл с соответствующим названием.

Файлы-сцены содержат в себе заранее расположение объекты окружения, модели роботов и программный код для управления ими.

При необходимости закрыть сцену используйте пункт меню “File->Close scene” или комбинацию CTRL+W

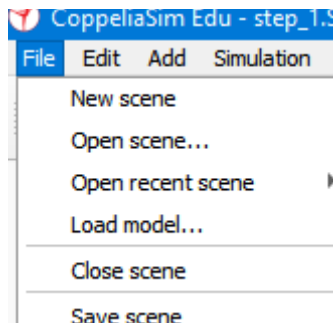





рисунок 1.1. Вид пунктов меню File

Для перемещения камеры по сцене нажмите кнопку  на горизонтальном меню:

- при зажатой ЛКМ движение мыши соответствует линейному перемещению относительно выбранной точки
- при зажатом СКМ движение мыши соответствуют вращению вокруг выбранной точки

Для изменения положения или вращения выбранного объекта нажмите кнопку  или  соответственно. При этом появится окно настройки режимов перемещения, которые будут рассмотрены позднее.

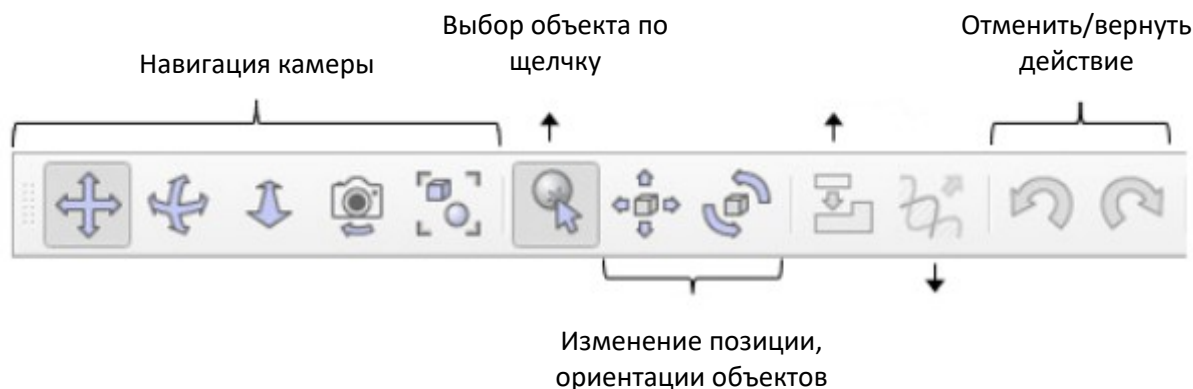


рисунок 1.2. Вид горизонтального меню

Внешний вид окна программы разделен на основные области

- Библиотека моделей - отображает готовые к использованию модели
- Иерархия объектов сцены - отображает и позволяет управлять структурой и взаимосвязью объектами и их составными частями
- Область 3d просмотра - отображает вид одной или нескольких “камер сцены”
- Горизонтальная панель инструментов
- Боковая вертикальная панель вызова окон
- Консоль вывода информации



рисунок 2. Основные элементы интерфейса

Структура модели и назначение составных частей

Открытая сцена содержит модель шлагбаума - устройства для преграждения и освобождения пути, при помощи изменения положения барьера. При этом стрела вращается вокруг горизонтально расположенной оси. Для корректной работы модели все её составные части должны быть соединены в особой последовательности, которая зависит от реализуемой механики.

Последовательность соединения отображена в окне иерархии сцены, которая имеет древовидный вид. Так объект *Base* является родительским для *Revolute_joint*. Это означает, что все изменения положений объекта *Base* будут соответственно влиять все дочерние.

Обратите внимание на строгое чередование типов: твердое тело - соединение - твердое тело - и т. д.

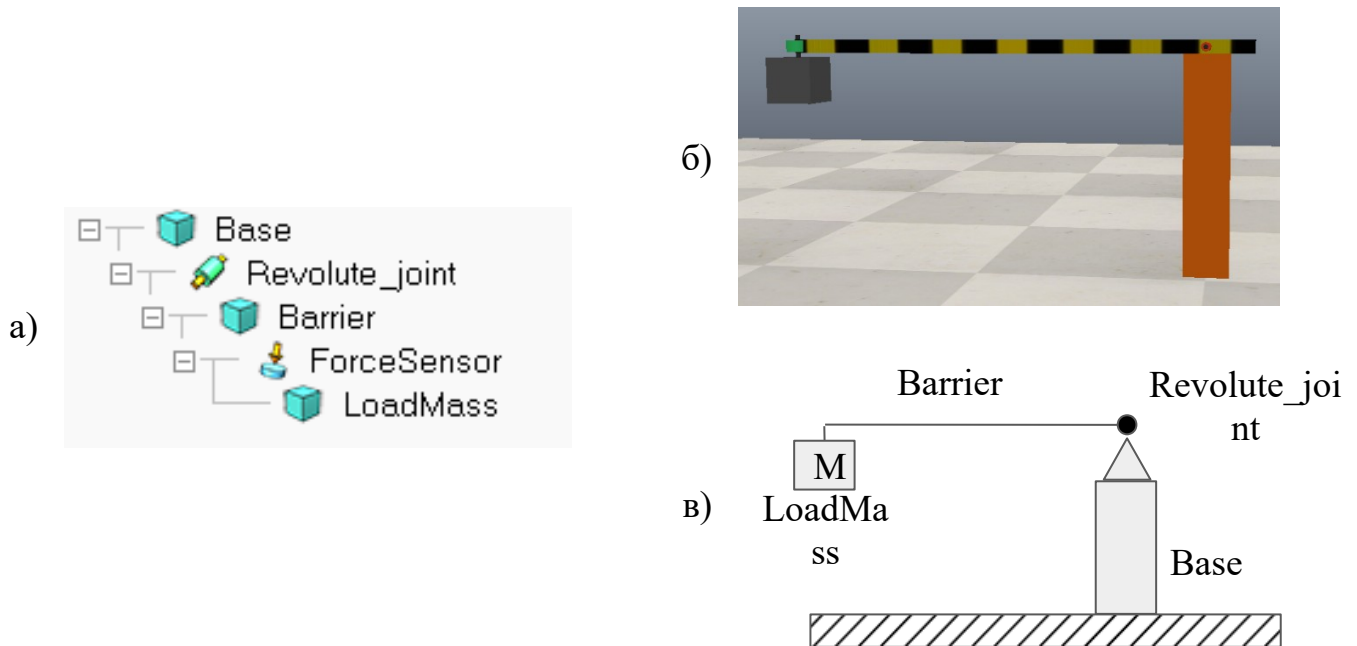


рисунок 3. Представление модели:

а - в окне иерархии сцены CoppeliaSim;

б - в окне 3d-сцены CoppeliaSim;

в - схема соединения составных частей.

Знакомство с подвижными соединениями (Joint)

Среди объектов сцены выделим три основных типа:

- твердые тела (Shape), различной степени сложности формы, которые обладают параметрами положения, ориентации, размера, формы, веса, массы и соответствующей инерции
- объекты подвижные соединения (Joint), которые обладают параметрами положения, ориентации, степени свободы, настройки параметров перемещения

- объект датчик силы (ForceSensor), в ряде случаев используемый для создания неподвижного соединения

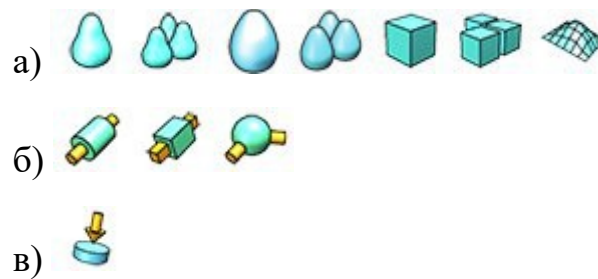


рисунок 4. Пиктограммы типов объектов в иерархии сцены:

а) твердые тела (Shape)

б) подвижные соединения (Joint)

в) датчик силы (ForceSensor)

Поворотное соединение (Joint)- это объект, который допускает относительное перемещение между родителем Joint и дочерним элементом Joint. Когда между соединением и объектом строится связь "родитель-потомок", объект присоединяется ко второй системе отсчета соединения, таким образом, изменение линейного/ углового положения соединения будет непосредственно отражаться на его дочерних элементах.

Поворотное соединение Revolute_joint - один из видов поворотного. Имеет единственную степень свободы и используется для описания вращательного движения.

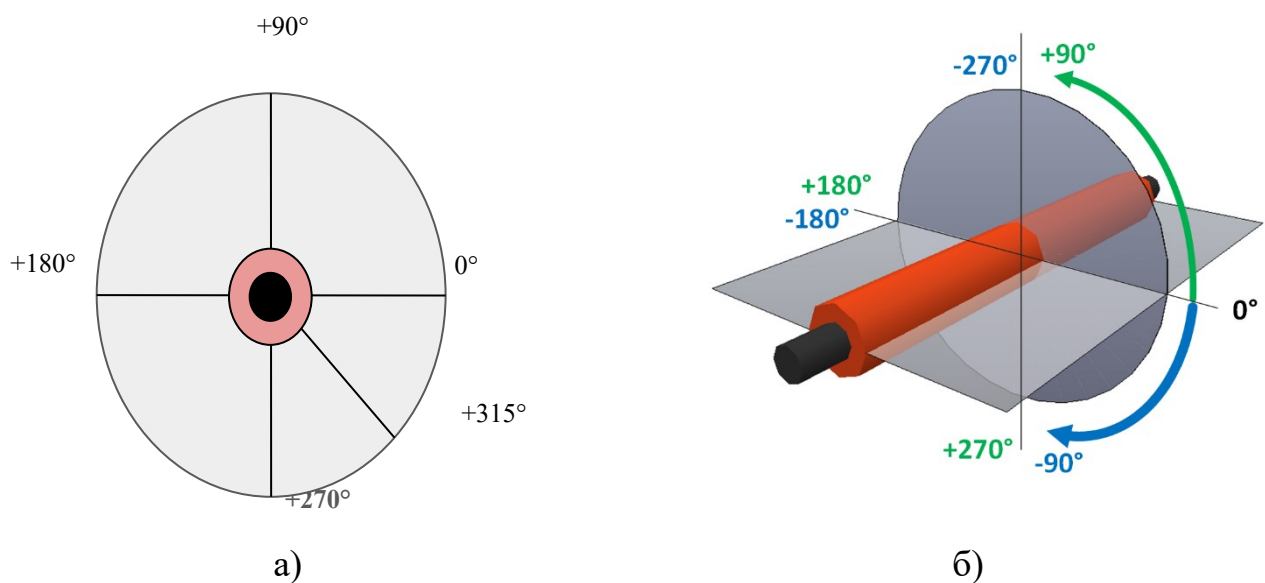


рисунок 5. Отсчет положений поворотного соединения (Revolute_joint)

Для настройки параметров объекта, необходимо открыть окно “Scene Object

Properties” путем двойного нажатия ЛКМ на пиктограмму    в окне иерархии сцены.

На данном этапе нас будет интересовать часть окна со следующими параметрами (все значения указываются в градусах):

- “Pos. min [deg]” - минимальное угловое положение
- “Pos. range [deg]” - диапазон смещения относительно “Pos. min”
(только положительное значение)
- “Position [deg]” - текущая позиция

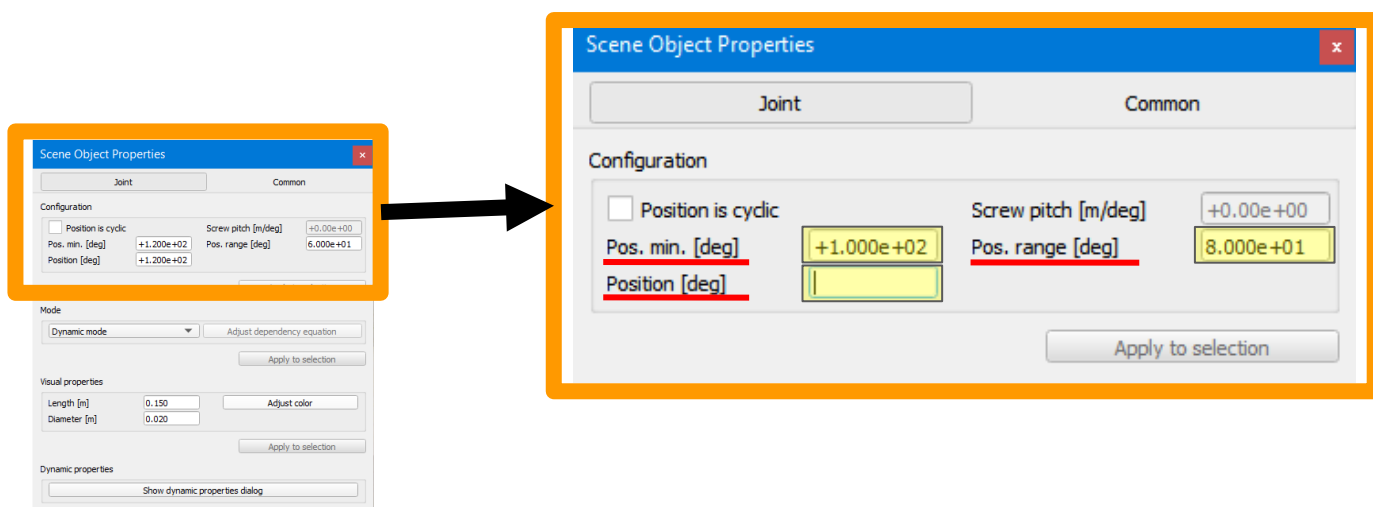


рисунок 6. Окно Scene Object Properties для соединенияповоротного типа (Revolute_joint)

Управление угловым положением поворотного соединения без симуляции. Осмотр и понимание устройства.

Имея исходные значения “Pos. min”, “Pos. range” можно рассчитать величину

$$\text{Pos. max} = \text{Pos. min} + \text{Pos. range},$$

а также изобразить на тригонометрической окружности диапазон, в котором будет разрешено вращение.

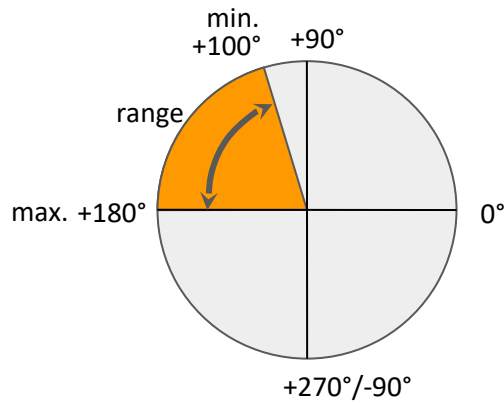
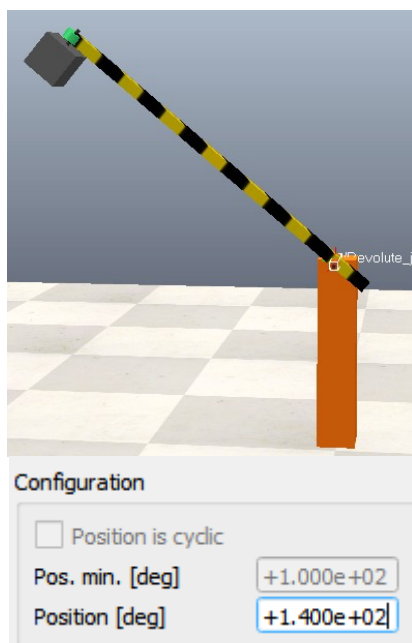
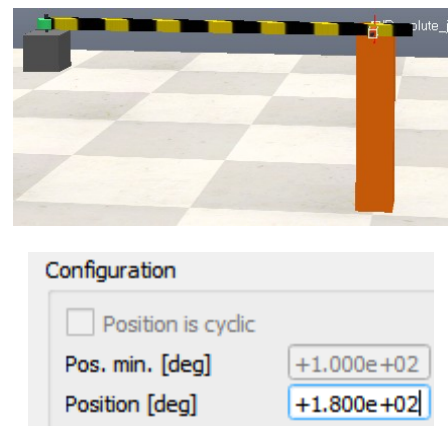


рисунок 7. Сектор диапазона разрешенных угловых положений

Для проверки расчетов используйте изменение параметра “Position [deg]” для объекта “Revolute_joint”, в окне “Scene Object Properties”, чтобы задать шлагбауму различные положения (крайние промежуточные, выходящие за разрешенный сектор). На данном этапе все изменения параметров проводятся без симуляции физики сцены, т.е. в режиме остановленной симуляции.



Положение при 140°



Положение при 180°

рисунок 8. Соответствие положений шлагбаума и введенных параметров Revolute_join

5. Контрольные вопросы

- Для чего используется joint в робототехнике?
- Какие параметры определяют движение шарнира?
- Зачем ограничивать диапазон углов?
- Что такое Position и чем отличается от Target Position?

6. Отчёт

- Титульный лист
- Цель работы
- Краткая теория
- Скриншоты сцены (3 состояния)
- Таблица параметров Joint
- Ответы на контрольные вопросы
- Выводы

Лабораторная работа №2

1. Цель:

- Рассчитать момент силы.
- Определить минимальный Torque, при котором шлагбаум поднимается.
- Сравнить расчёт и фактическое поведение в симуляции.

2. Теоретическая часть

Подъёмный механизм шлагбаума моделируется вращательным приводом.

Основные формулы:

1. Вес груза:

$$F = m \cdot g$$

2. Момент силы относительно шарнира:

$$M = F \cdot L$$

где

m — масса,

g — ускорение свободного падения (9.81),

L — плечо рычага.

3. Крутящий момент привода (Torque) в joint:

- определяет, сможет ли привод поднять нагрузку
- задаётся в панели свойств шарнира
- ограничивает максимальную вращающую силу

3. Практическое задание

Для каждого из значений плотности, согласно выданному варианту (см.таблицу вариантов, 4 столбец), выполнить нижеперечисленные шаги:

- а. Установите настройки значений соединения поворотного типа (Revolute joint)
 - i. Pos. min = 90 градусов
 - ii. Pos. rang = 110 градусов

iii. Position [deg] = 180 градусов, соответствует положению (“закрыт”)

- b. Изменить массу объекта Load Mass используя значение плотности
- c. Рассчитать момент сил, который оказывает груз
- d. Изменить значение крутящего момента поворотного узла основываясь на рассчитанном значении момента сил
- e. Запустить симуляцию и проверить корректную, достаточно ли текущего крутящего момента для перевода шлагбаума из положения 180 градусов (“закрыт”) в положение 90 градусов (“открыт”)
- f. При необходимости, увеличить/уменьшить (с шагом 1.0, 0.5, 0.1 [Н*м]) значение крутящего момента поворотного узла для определения минимально допустимого значения для перевода в положение 90 градусов (“открыт”)
- g. Найти величину относительной разности между вычисленным и требуемым фактически крутящим моментом (Torque)

4. Ход выполнения работы:

Увеличение массы груза и проверка работы поворотного узла (первый запуск симуляции)

Изменить массу объекта LoadMass, возможно двумя способами:

- 1) путем прямого ввода значения массы в килограммах
- 2) путем ввода плотности объекта в размерности кг/м³

Второй способ предпочтительнее, т.к. он рассчитывает исходя из геометрических размеров объекта не только массу, но и его инерциальные характеристики, которые критичны для корректной симуляции физики взаимодействия.

Для настройки параметров твердые тела, необходимо открыть окно “Scene Object Properties” путем двойного нажатия ЛКМ на соответствующую пиктограмму



в окне иерархии сцены. В открывшемся окне выполнить следующие шаги (см. рис. 9):

- 1) Нажать на кнопку “Show dynamic properties dialog”
- 2) В появившемся окне “Rigid Body Dynamic Properties” нажать на кнопку “Compute mass and inertial properties ...”
- 3) В появившемся окне “Body density” ввести значение плотности материала и нажать кнопку подтверждения “ОК”
- 4) Проверить полученный результат

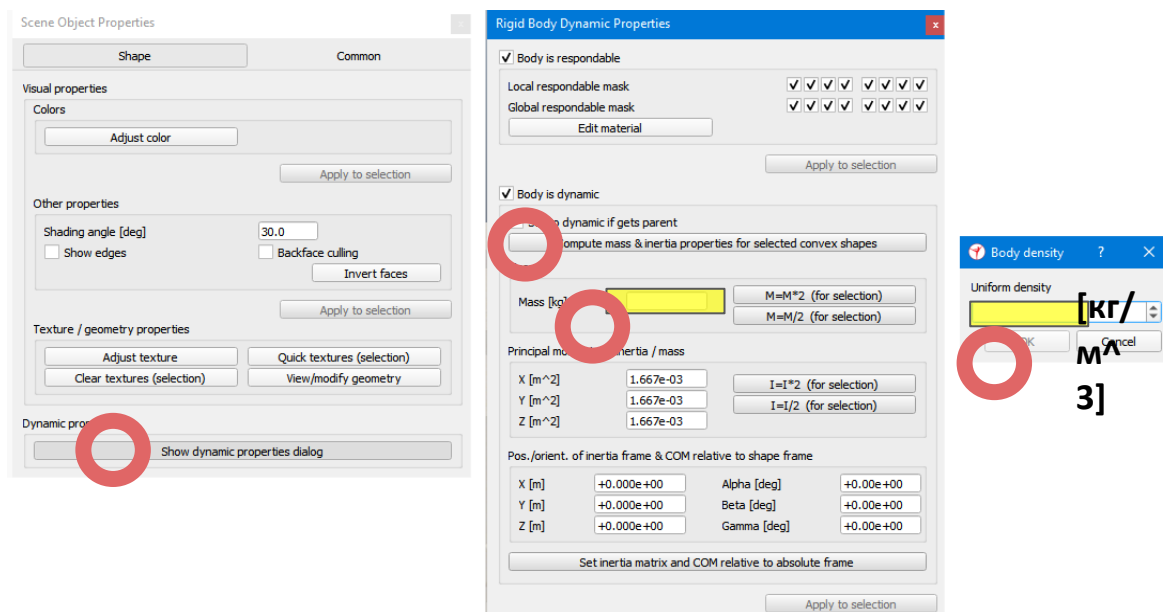


рисунок 9. Последовательность шагов для расчета массы объекта из плотности

Управление режимами стимуляции осуществляется кнопками на горизонтальной панели инструментов и имеют вид представленный на рис. 10.



рисунок 10. Кнопки запуска, паузы, остановки режима симуляции

Важно помнить:

все изменения сцены во время запущенной симуляции (перемещения объектов, добавление и их удаление и пр.), после завершения симуляции будут отменены и не сохранятся.

Запустите симуляцию сцены и попробуйте задать крайние положения шлагбаума путем ввода соответствующих целевых значений угла в градусах в поле “Position [deg]” окна “Scene Object Properties” объекта “Revolute_joint” (см. рис. 6). При запущенной симуляции поля ввода “Pos. min” и “Pos. range” будут заблокированы.

Остановите симуляцию и сделайте заключение, о том, соответствовало ли положение шлагбаума введенным целевым значениям углов.

Момент силы груза и момент инерции поворотного узла

Рассмотрим модель шлагбаума как рычаг с некоторым грузом LoadMass, в ней можно вычислить величину момента силы M , которую оказывает объект LoadMass в точке В, закрепленный на плече длиной L , относительно точки А.

При этом момент силы вычисляется как $M = F * L = m * g * L$.

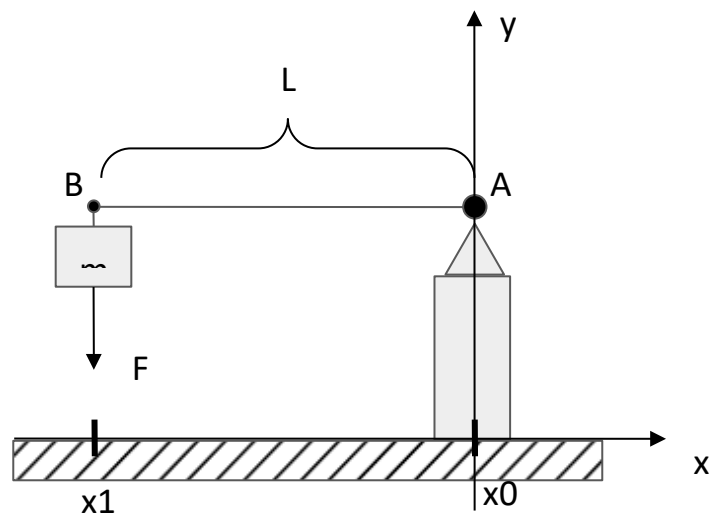


рисунок 11. Схема соединения составных частей

Для вычислений потребуются точные значения параметров рассматриваемых объектов. Для получения величины L воспользуемся окном просмотра абсолютных координат точек A (Revolute_joint) и B (ForceSensor) при этом положение объекта Barrier должно быть строго параллельно оси x (поверхности “земли”).

Для работы с координатами, необходимо выполнить следующие шаги (см. рис. 12):

- 1) Выбрать интересующий объект в окне иерархии сцены
- 2) Открыть окно “Object/Item translation/Position” нажатием на



соответствующую кнопку горизонтальной панели инструментов

- 3) В появившемся окне выбрать вкладку “Position”
- 4) Установить переключатель “Relative to” в положение “World”, что соответствует режиму абсолютных (мировых) координат
- 5) Получить значение X-coord. [m]

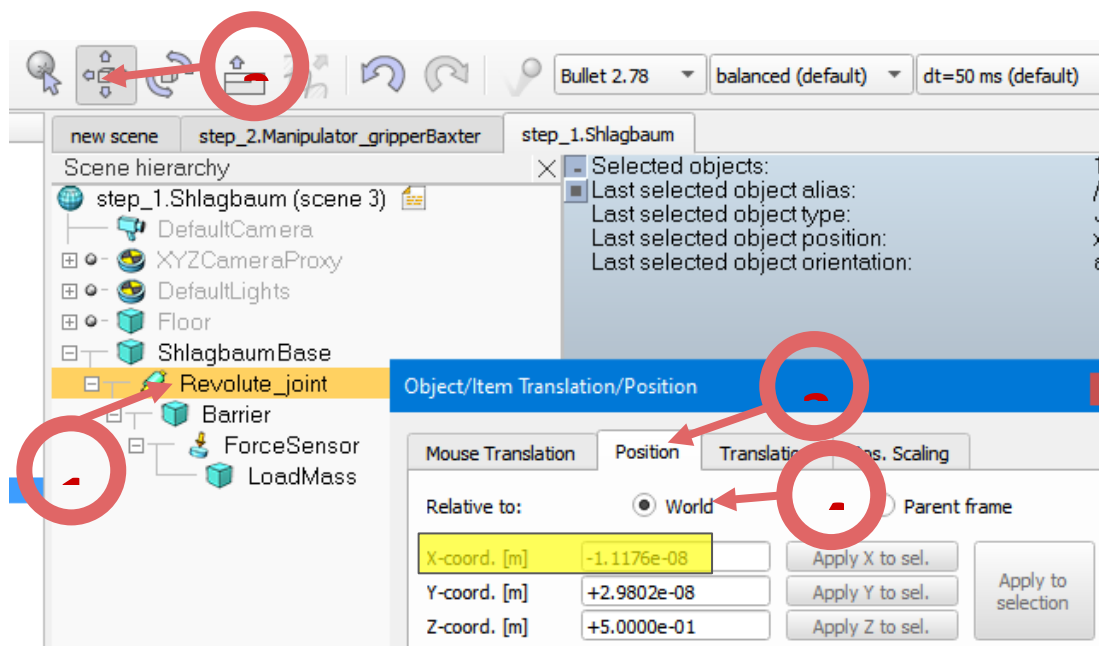


рисунок 12.1 Шаги для получения координат объекта

Получив значения координат точек А и В, вычислим величину момента силы М.
Пример:

$$x(A) = -1.1176 \cdot 10^{-8} \approx 0.000 \text{ [м]}$$

$$x(B) = -5 \cdot 10^{-1} = -0.5 \text{ [м]}$$

$$L = x(A) - x(B) = 0.000 - (-0.5) = 0.5 \text{ [м]}$$

$$F = m \cdot g$$

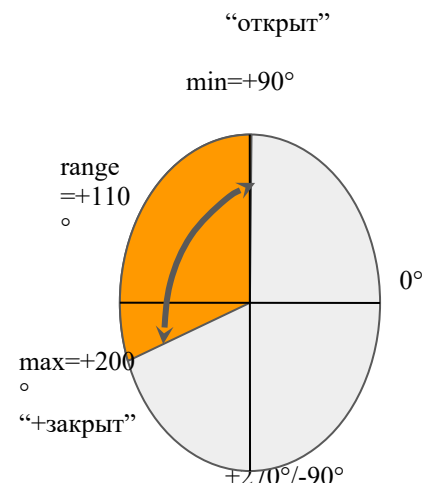
$$g = 9.81 \text{ [м/с}^2\text{]}$$

$$M = F \cdot L = m \cdot g \cdot L = 1 \text{ [кг]} \cdot 9.81 \text{ [м/с}^2\text{]} \cdot 0.5 \text{ [м]} = 4.9 \text{ [Н*м]}$$

Установите настройки значений соединения поворотного типа (Revolute_joint), такими, чтобы стрела шлагбаума могла перемещаться:

- из вертикального положения (“открыт”, 90 градусов)
- в положение на несколько градусов больше, чем горизонтальное (“закрыт”, 180+”запас 10-20” градусов)

Таким образом значение параметров Revolute_joint должны быть равны:






- Pos. min = 90 градусов
- Pos. rang = 90 + 20 = 110 градусов
- Position [deg] = 180 градусов

рисунок 13.1. Сектор для положений “открыт”/”закрыт”

Для того, чтобы соединения поворотного типа (Revolute_joint) могло совершать вращение, оно должно обладать крутящим моментом не меньшим, чем оказываемая нагрузка - момент силы М.

Изменить параметр крутящего момента вращательного соединения (Revolute_joint) на вычисленный.

Для настройки параметров объекта, необходимо выполнить следующие шаги (см. рис. 13):

- 1) Выполнить двойное нажатие ЛКМ на пиктограмму    в окне иерархии сцены
- 2) В появившемся окне “Scene Object Properties” нажать кнопку “Show dynamic properties dialog”
- 3) В появившемся окне “Joint dynamic properties” ввести значение крутящего момента в поле “Max. torque [N*m]” и нажать клавишу Enter

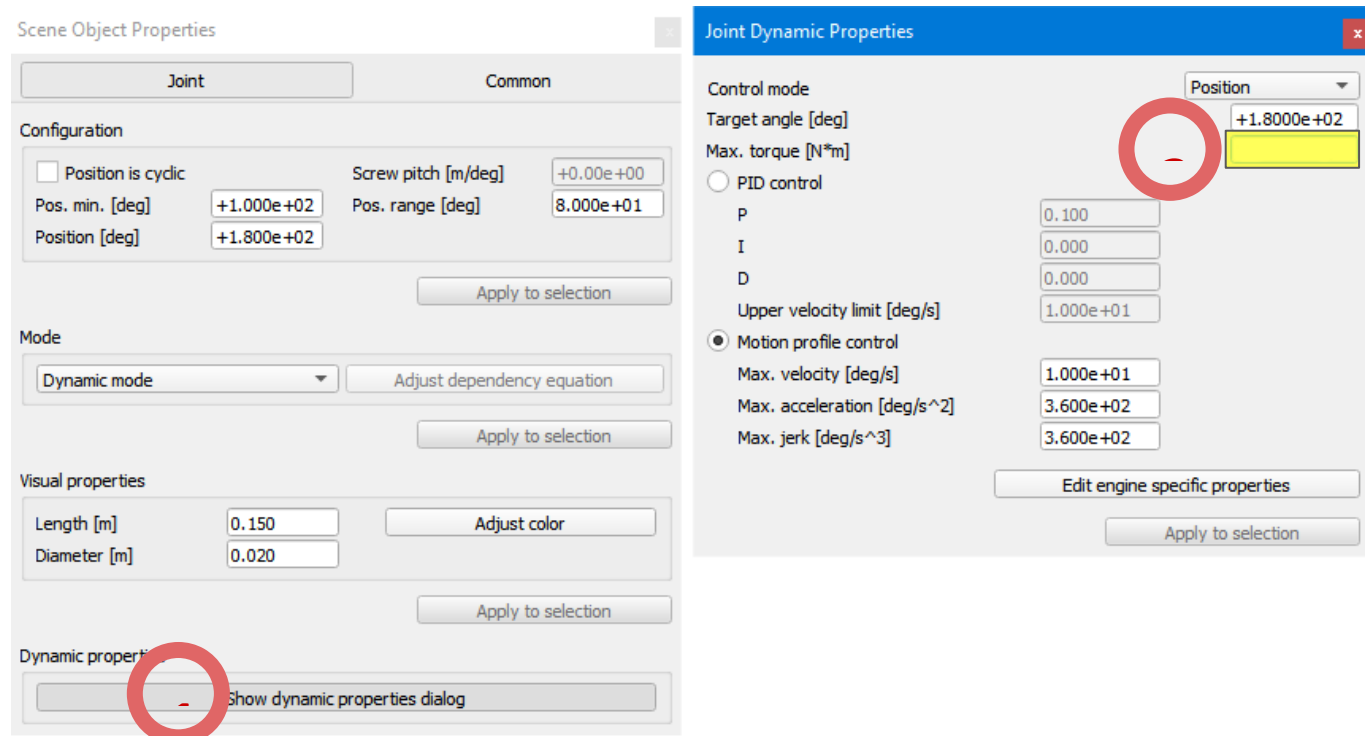


рисунок 13.2. Шаги для изменения значения крутящего момента

Повторно запустите симуляцию сцены и попробуйте задать крайние положения шлагбаума путем ввода соответствующих целевых значений угла в градусах в поле “Position [deg]” окна “Scene Object Properties” объекта “Revolute_joint” (см. рис. 6).

При необходимости увеличивайте величину крутящего момента “Max. torque [N*m]” поворотного узла (Revolute_joint) (с шагом 1.0, 0.5, 0.1 [Н*м]) до величины достаточной для уверенного, пусть даже и медленного, подъема шлагбаума.

Остановите симуляцию и сделайте заключение, о том, соответствовало ли положение шлагбаума введенным целевым значениям углов.

Зафиксируйте в отчете найденную требуемую фактически величину крутящего момента, при котором достигнута уверенная работа модели.

Определите величину относительной разности между вычисленным и требуемым фактически крутящими моментами (Torque)

Сделайте предположения о причинах разности этих значений.

Пример:

Вычисленное = 4.9 [Н*м]

Фактическое = 5.3 [Н*м]

Разность абсолютная = $5.3 - 4.9 = 0.4$ [Н*м]

Разность относительная $0.4/4.9 * 100 = 8,2\%$

5. Контрольные вопросы

- Что такое момент силы?
- Почему фактический момент может отличаться от расчётного?
- Какие физические свойства моделирует CoppeliaSim?
- Как влияет изменение массы на работу шарнира?

6. Отчёт

- Титульный лист
- Цель работы
- Краткая теория
- Таблица: масса \rightarrow M(теор) \rightarrow Torque(факт)
- График или таблица погрешностей

- Скриншоты тестов
- Ответы на контрольные вопросы
- Выводы

Лабораторная работа № 3

1. Цель:

- Освоить работу с Python child script в CoppeliaSim.
- Изучить базовые функции API: управление объектами, изменение положения, цвета, вывод информации.
- Научиться выполнять программное управление объектами в симуляции.

2. Теоретическая часть

В CoppeliaSim встроены два типа скриптов:

- **Non-threaded script** — выполняется синхронно, подходит для простых одноразовых действий.
- **Threaded (Python / Lua) script** — выполняется в отдельном потоке, позволяет использовать циклы, задержки, управление движением объектов.

Python child script использует модуль:

```
sim = require('sim')
```

Основные команды API:

Команда Назначение

`sim.getObject('/obj')` Получить объект в сцене

`sim.setObjectPosition(obj, parent, pos)` Установить XYZ координаты

`sim.setObjectOrientation(obj, parent, ori)` Установить ориентацию

`sim.wait(seconds)` Задержка внутри симуляции

`sim.setJointTargetPosition(joint, angle)` Повернуть шарнир (рад)

`sim.addLog(level, message)` Вывод в консоль

Python script размещается на объекте или в самом корне сцены тремя способами:

- Add → Add script → Python script
- Add → Associated child script
- Через дерево объектов (Scene hierarchy)

3. Практическое задание




1. Добавить Python child script к объекту по инструкции из ФОС.
2. Запустить пример из методички (вывод времени каждые 1 сек).
3. Реализовать программу, выполняющую:
 - а) перемещение объекта в три произвольные точки;


- b) задержку между позициями 1–2 сек;
 - c) изменение цвета объекта (любой).
- 4. Реализовать логирование в консоль:
 - a) вывод текущего времени,
 - b) вывод текущих координат


5. Ход выполнения работы:



Написание простого скрипта для управления поворотным узлом

Среда CoppeliaSim имеет возможность исполнять управляющие команды на языке Python3, но требует установленного интерпретатора, библиотек и определенных настроек. Подробности по установке см. в файле “АРиМС-2023.02 CoppeliaSim. Установка”.

Сцена CoppeliaSim может содержать встроенные скрипты на языке LUA и Python. Встроенные скрипты бывают двух типов  child и  customization. За конкретным объектом сцены может быть закреплены только по одному скрипту каждого типа. Далее будут использованы  child скрипты для управления моделями сцены.

Добавьте новый  child скрипт python к любому объекту сцены:

1.1. Создайте новый объект ПКМ на модели в окне иерархии сцены → Add → Dummy. Объекты  Dummy являются “материальной точкой” и часто применяется как вспомогательный.

1.2. К объекту  Dummy добавьте  child скрипт для управления в многопоточном режиме. Для этого выберите пункты: ПКМ на модели в окне иерархии сцены → Add → Associated child script → Threaded → Python.

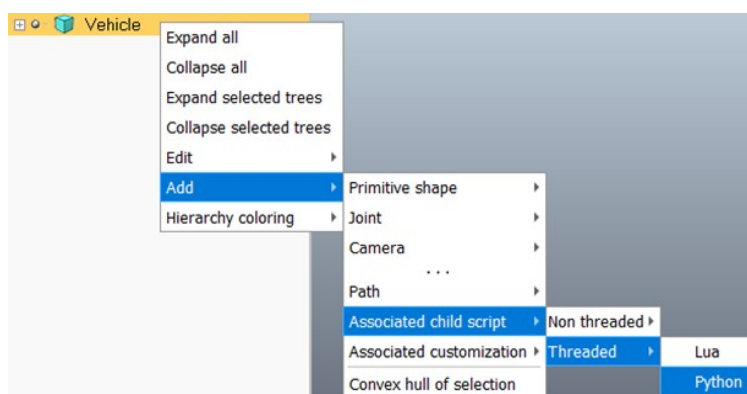


Рис. 14. Добавление скрипта Threaded → Python

1.3. Откройте окно редактирования созданного скрипта. Для этого выполните двойное нажатие ЛКМ на значке в окне иерархии сцены (рис. 15).

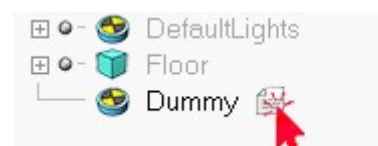


Рис. 15. Открытие окна редактора

1.4. Ознакомьтесь с приведенным ниже примерами кода на языке Python. Данный код рассчитан для работы во внутреннем редакторе CoppeliaSim и имеет ряд особенностей написания, по сравнению с классическим языком Python.

Добавьте к объекту потоковый Python child script.

Вставьте приведённый ниже пример и выполните его в режиме симуляции:

```
def sysCall_thread():  
  
    sim = require('sim')  
  
    for i in range(10):  
  
        sim.addLog(sim.verbosity_scriptinfos, sim.getSystemTime())  
  
        sim.wait(1)
```

После выполнения примера измените позицию объекта с помощью команды `sim.setObjectPosition`.

Повторите пример несколько раз, изменяя значения координат.

По аналогии выполните задание с изменением цвета, координат, ориентации.

6. Контрольные вопросы

- Чем отличается threaded script от non-threaded?
- Как получить объект в сцене по его имени?
- Для чего используется `sim.wait()`?
- Что будет, если попытаться изменить положение объекта, когда он закреплён?

7. Требования к отчёту

- Титульный лист
- Цель работы
- Теоретическое краткое описание Python API
- Код программ
- Скриншоты выполнения
- Ответы на контрольные вопросы
- Выводы

Лабораторная работа № 4

1. Цель

- Освоить управление вращательным шарниром шлагбаума через Python child script.

- Научиться программно изменять угловое положение joint.
- Реализовать циклический алгоритм работы механизма.

2. Теоретическая часть

Вращательный шарнир (**Revolute Joint**) управляется через API:

```
sim.setJointTargetPosition(joint, angle)
```

где angle — угол в **радианах**.

Для организации задержки используется:

```
sim.wait(seconds)
```

Для бесконечного цикла:

```
while True:
    # действия
```

3. Практическое задание

Реализовать программу:

1. Получить объект Revolute_joint функцией sim.getObject.
2. Установить целевой угол **вверх** (например math.radians(90)).
3. Остановиться на 5 секунд.
4. Установить целевой угол **вниз** (math.radians(0)).
5. Остановиться на 5 секунд.
6. Организовать бесконечный цикл.

Дополнительно (по желанию):

- выводить в консоль текущее положение joint,
- менять цвет флажка:
 - зелёный — открыт;
 - красный — закрыт.

Ход выполнения работы

Название основной функции sysCall_thread() используется для работы в многопоточном режиме. В этом режиме необходимо обязательно назначить

режим переключения потоков, функцией `sim.setThreadAutomaticSwitch(True)` мы установим автоматический режим.

Замените содержимое окна редактирования скрипта на следующие примеры программ. Обратите внимание на структуру и особенности выполнения.

Пример 1. Обязательные структурные элементы

Напишите следующий код, запустите симуляцию и осмотрите консоль вывода

```
#python
def sysCall_thread():
    sim.setThreadAutomaticSwitch(True)
    while True:
        pass
```

Обязательный заголовок для python в CoppeliaSim

Имя основной функции для многопоточного режима

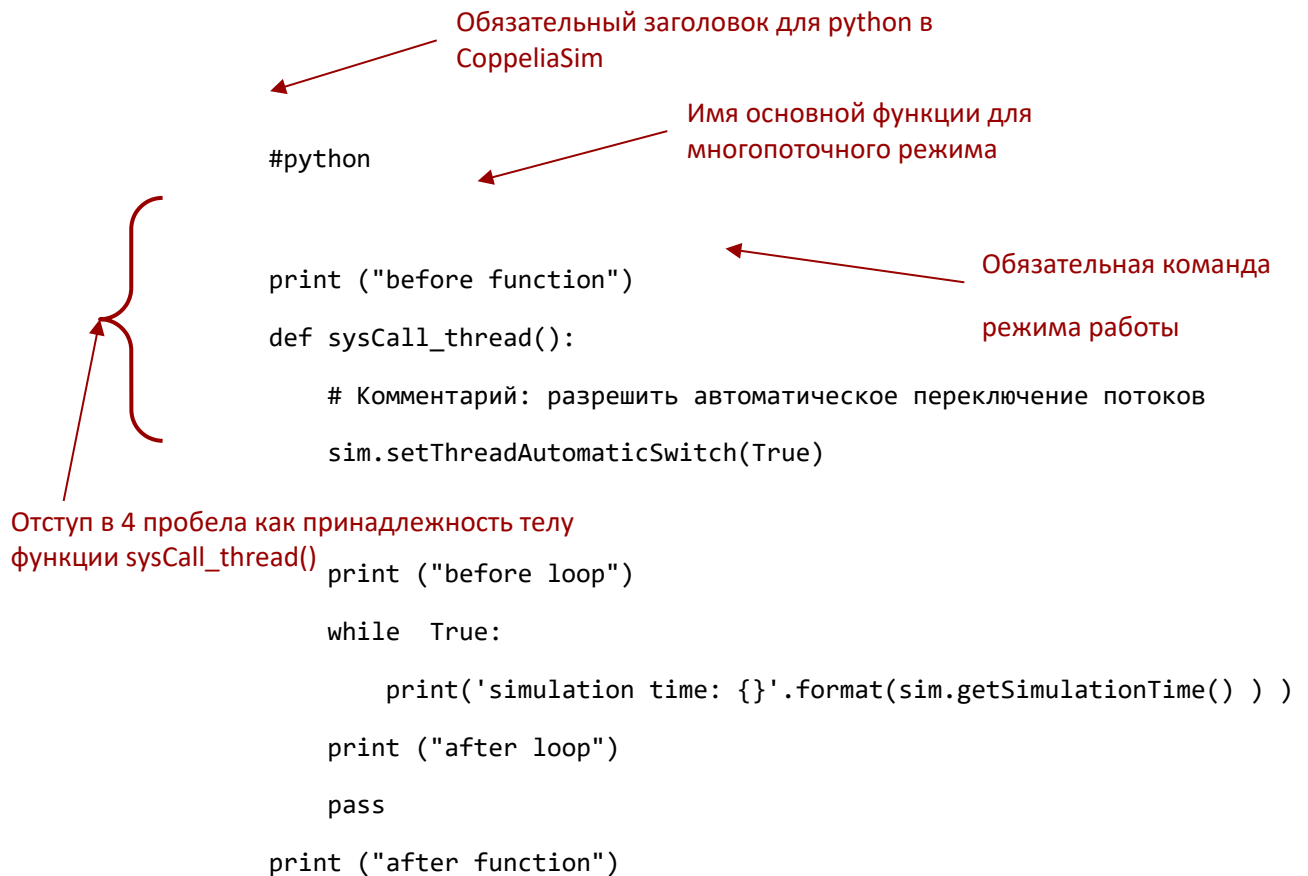
Обязательная команда режима работы

Отступ в 4 пробела как принадлежность телу функции sysCall_thread()

Отступ дополнительно в 4 пробела как принадлежность телу блока while

Пример 2. Демонстрация области выполнения

Напишите следующий код, запустите симуляцию и осмотрите консоль вывода. Определите те команды `print` которые не были выполнены.



```
#python

print ("before function")

def sysCall_thread():
    # Комментарий: разрешить автоматическое переключение потоков
    sim.setThreadAutomaticSwitch(True)

    print ("before loop")
    while True:
        print('simulation time: {}'.format(sim.getSimulationTime() ) )
        print ("after loop")
        pass
    print ("after function")
```

Обязательный заголовок для python в CoppeliaSim

Имя основной функции для многопоточного режима

Обязательная команда режима работы

Отступ в 4 пробела как принадлежность телу функции sysCall_thread()

Пример 3. Перемещение в заданные координаты.

Напишите следующий код, запустите симуляцию и осмотрите консоль вывода.

Обратите внимание на вывод времени срабатывания скрипта, которое выводится в консоль.

```
#python

def sysCall_thread():

    sim.setThreadAutomaticSwitch(True)

    thisObjID = sim.getObject('.')
    print(thisObjID )

    while True:
```

```

coord1 = sim.getObjectPosition(thisObjID , -1)

coord1[0] = coord1[0]+0.001

sim.setObjectPosition(thisObjID ,-1,coord1)

print(coord1)

pass

```

4.

Таблица 1. Основные команды CoppeliaSim на языке Python3

Описание команды		Код команды	
Пауза выполнения скрипта		sim.wait(second)	
Работа с параметрами объектов			
Параметр		Задать значение	Получить значение
Получить идентификатор объекта сцены		—	objID = sim.getObject('.')
Цвет объекта		sim.setObjectColor(objID , 0, sim.colorcomponent_ambient_diffuse,[0,1,0])	sim.getObjectColor(objID , 0, sim.colorcomponent_ambient_diffuse)
Координаты положения (x,y,z)		sim.setObjectPosition(objID,-1, [x,y,z])	[x,y,z]= sim.getObjectPosition(objID , -1)
Ориентация (углы поворота)		sim.setObjectOrientation(objID,-1, [a,b,g])	[a,b,g]=sim.getObjectOrientation(objID , -1)
Работа с сообщениями (signals)			
Целочисленные сообщения		sim.setInt32Signal ("SIGNAL_CHANNEL_NAME", intVal)	recived = sim.getInt32Signal ("SIGNAL_CHANNEL_NAME")
Объекты поворотные узлы (joint)			
Крутящий момент [Н*м]	Целевой / Макс-ый	sim.setJointTargetForce(objID , ...)	sim.getJointTargetForce(objID)
	Мгновенный	—	sim.getJointForce(objID)
Угол /	Мгновенный	sim.setJointPosition(objID , ...)	sim.getJointPosition(objID)

положение [радиан, либо метр]			
	Целевой	sim.setJointTargetPosition(objID , ...)	sim.getJointTargetPosition(objID)

5. * <https://coppeliarobotics.com/helpFiles/en/apiFunctions.htm#jointObject>

6.

Таблица 2. Режимы управления подвижным узлом и соответствующие управляющие функции						
		Dynamic Control modes*				
Параметр	Код команды	Free (no control)	Force / torque	Velocity	Position	
Максимальный/целевой Крутящий момент [Н*м]	sim.setJointTargetForce(objHandle r, ...)	—	+	+	+	
Мгновенный угол/положение [радиан, либо метр]	sim.setJointPosition(objHandler, ...)	—	—	—	—	
Целевой угол/положение [радиан, либо метр]	sim.setJointTargetPosition(objHan dler, ...)	—	—	—	+	
Целевая скорость [радиан/сек, либо м/сек]	sim.setJointTargetVelocity(objHan dler, ...)	—	—	+	—	
*При различных режимах работы необходимо использовать соответствующие функции, которые изменяют параметры оказывающие влияние на выбранный режим. Использование не соответствующей функции не окажет эффекта.						

5. Контрольные вопросы

- Какое значение угла нужно передавать — градусы или радианы?
- Почему важно использовать sim.wait() в поточном скрипте?
- Как получить объект joint по пути /joint?
- Что может произойти, если Torque недостаточен?

6. Требования к отчёту

- Цель работы

- Краткое описание теории
- Код программы
- Скриншоты работы
- Пояснение логики работы
- Вывод

Лабораторная работа № 5

Цель:

- Научиться управлять мобильным роботом в CoppeliaSim.
- Освоить управление скоростями колёс и линейной/угловой скоростью.

Теория:

Мобильный дифференциальный робот имеет две скорости колёс:

$$v = \frac{R(\omega_L + \omega_R)}{2}$$

$$\omega = \frac{R(\omega_R - \omega_L)}{L}$$

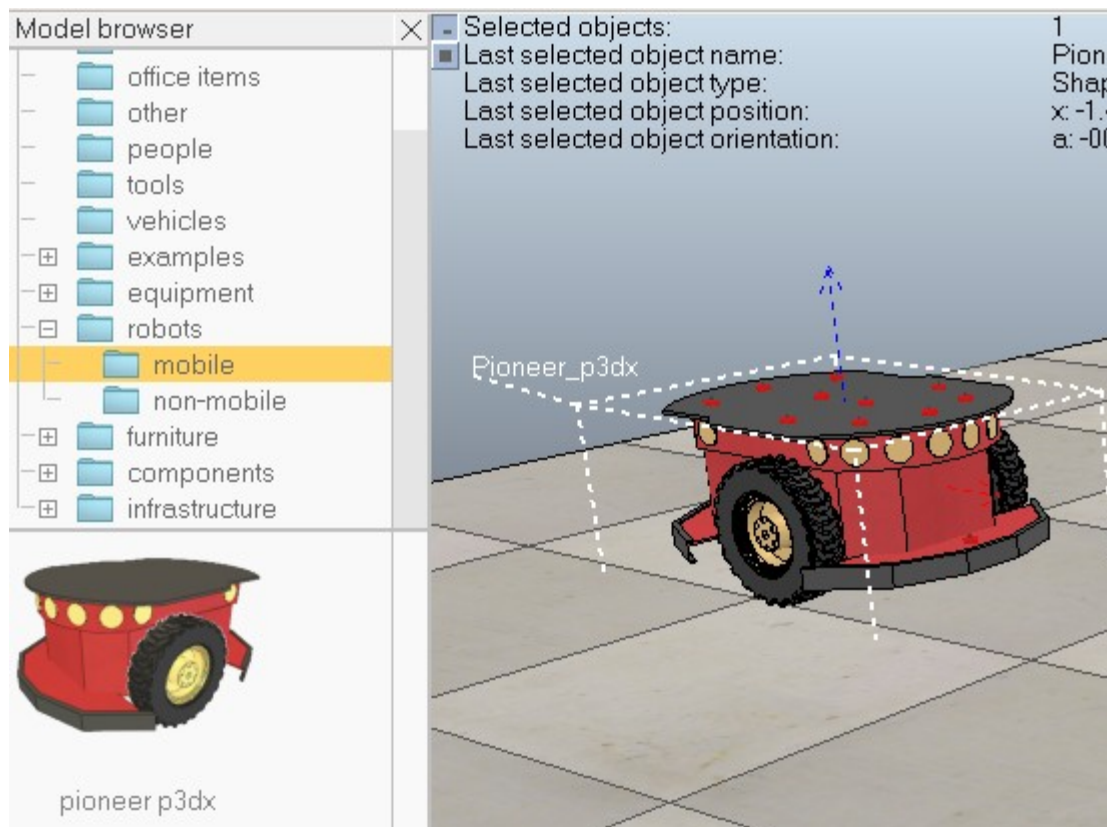
где

R — радиус, L — база,
 ω_L , ω_R — скорости колёс.

Выполнение хода работы:

Построим график траектории движения мобильного робота *pioneer 3dx*. Для этого на панели меню выберем *Add --> Graph*. Доступ к его основным свойствам и их настройка осуществляются в диалоговом окне графика.

Добавим на сцену модель робота *pioneer 3dx*. Для этого в обзорщике моделей в дереве *robots* выберем *mobile* и перетащим робота *pioneer 3dx* на сцену.



Если сейчас нажать на кнопку *Play* (Старт сцены), то робот поедет в прямом направлении, логика его работы содержится в прикрепленном скрипте.

Отредактируем скрипт таким образом, чтобы вместо прямолинейного движения робот осуществлял круговое. Для этого в функции `sysCall_actuation()` скорость правого колеса увеличим в два раза.

```
vRight=v0*2
```

Убедимся в круговой траектории движения мобильного робота запустив симуляцию.

Далее в скрипте содержащего логику работы мобильного робота в функции `sysCall_init()` добавим следующий код:

```
graph=sim.getObjectHandle('Graph')
objectHandle=sim.getObjectHandle('Pioneer_p3dx')
objectPosX=sim.addGraphStream(graph,'x','m',1)
objectPosY=sim.addGraphStream(graph,'y','m',1)
sim.addGraphCurve(graph,'object pos x/y',2,{objectPosX,objectPosY},{0,0},'m by m')
```

Где `'Graph'` и `'Pioneer_p3dx'` это имена объектов из иерархии сцены.

В теле скрипта добавим функцию `sysCall_sensing()`.

```
function sysCall_sensing()
```

```
end
```

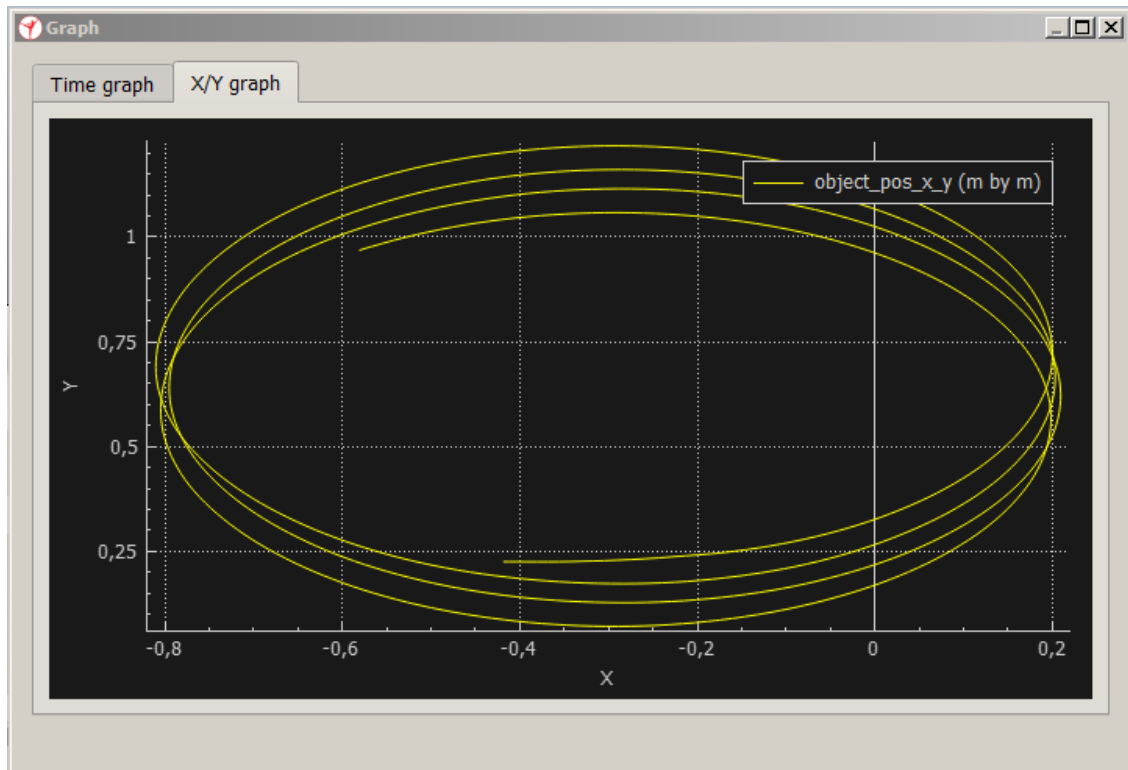
В функции `sysCall_sensing()` добавим следующий код:

```
pos=sim.getObjectPosition(objectHandle,-1)
sim.setGraphStreamValue(graph,objectPosX,pos[1])
sim.setGraphStreamValue(graph,objectPosY,pos[2])
```

Очистим график после окончания процесса симулирования. Для этого в функции `sysCall_cleanup()` добавим следующую строку:

```
sim.resetGraph(graph)
```

И запустим процесс симулирования.



Точность позиционирования обусловлена влиянием физики на нашего мобильного робота.

Одним из самых интересных графиков – пространственная кривая движения механизма.

Приведем скрипт к первоначальному виду, оставив лишь увеличенную в два раза скорость правого колеса.

Далее в скрипте содержащую логику работы мобильного робота в функции `sysCall_init()` добавим следующий код:

```
graph=sim.getObjectHandle('Graph')
base=sim.getObjectHandle('Pioneer_p3dx')
x=sim.addGraphStream(graph, 'x', 'm')
```

Строить будем зависимость x от времени t .

Добавим функцию `sysCall_sensing()`.

```
function sysCall_sensing()
```

```
end
```

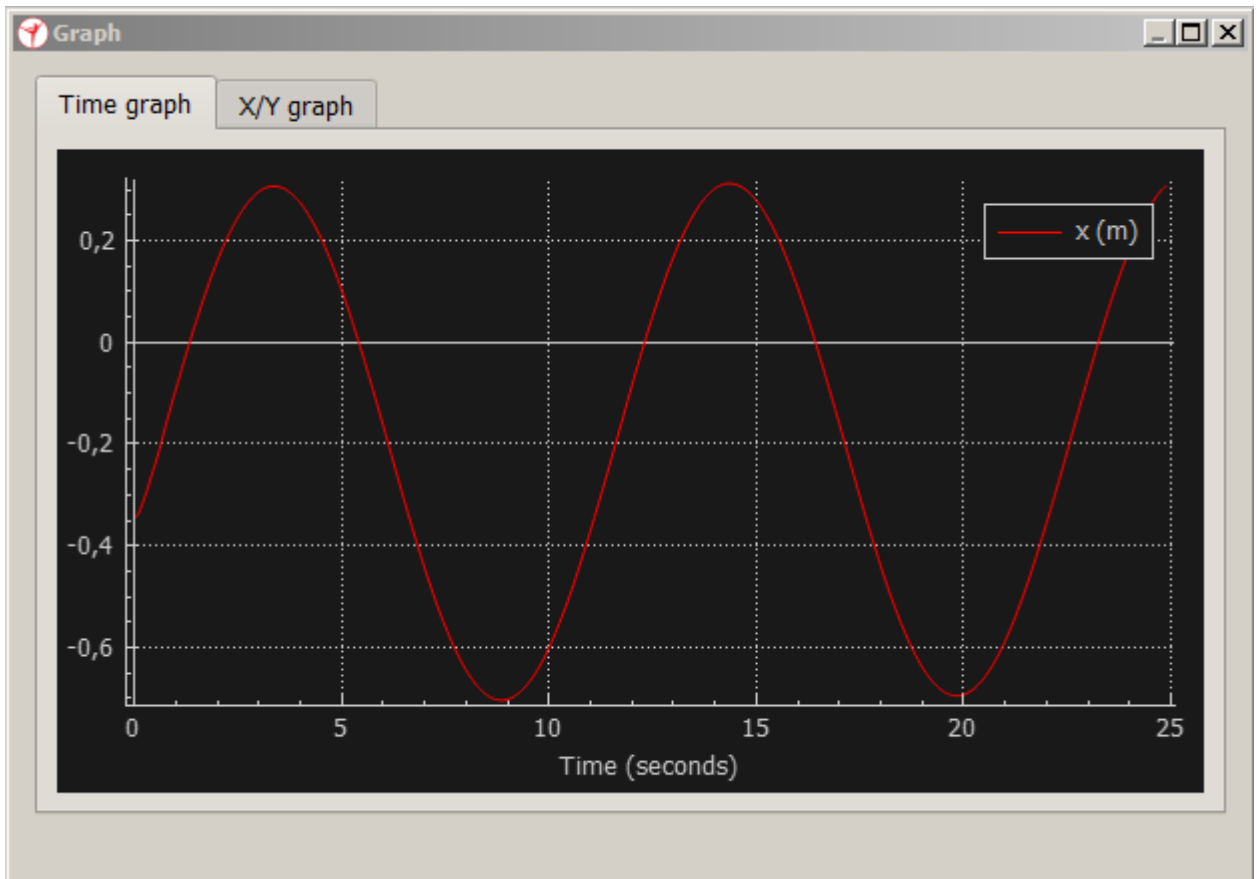
В функции `sysCall_sensing()` добавим следующий код:

```
p=sim.getObjectPosition(base,-1)
sim.setGraphStreamValue(graph, x, p[1])
```

Очистим график после окончания процесса симулирования. Для этого в функцию очистки `sysCall_cleanup()` добавим следующую строку:

```
sim.resetGraph(graph)
```

И запустим процесс симулирования.



Добавим вторую зависимость y от времени t .

Для этого в функции инициализации `sysCall_init()` добавим следующую строку:

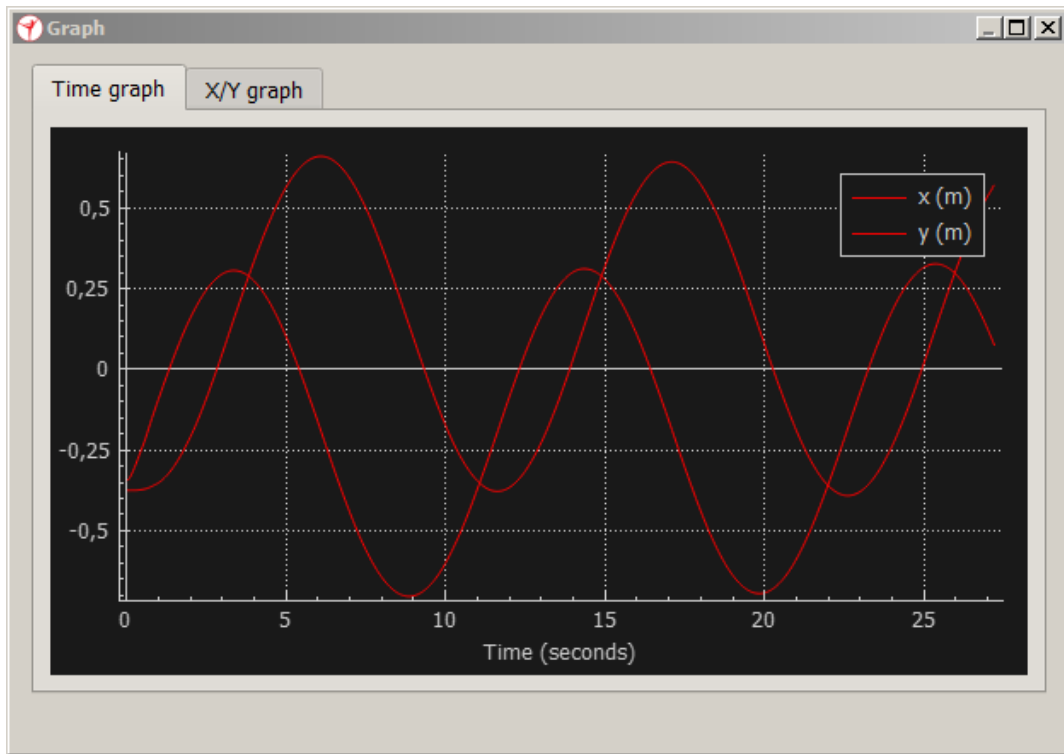
```
y=sim.addGraphStream(graph, 'y', 'm')
```

В функции `sysCall_sensing()` добавим следующую строку:

```
sim.setGraphStreamValue(graph, y, p[2])
```

$p[3]$ - это соответственно координата z .

Запустим процесс симулирования. Получим график изображенный на рисунке.

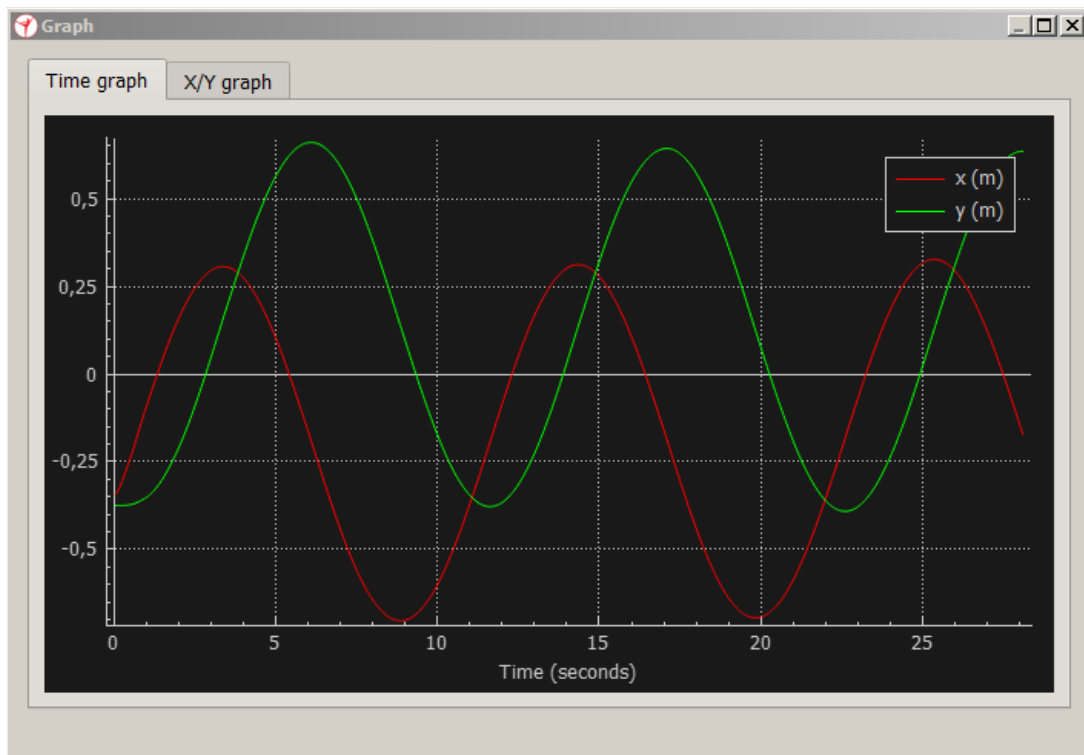


Из-за того что кривые одного цвета из графика сложно понять где какая переменная. Поэтому введем следующие изменения в описании переменных в функции `sysCall_init()`:

```
x=sim.addGraphStream(graph, 'x', 'm', 0, {1,0,0})  
y=sim.addGraphStream(graph, 'y', 'm', 0, {0,1,0})
```

Где $\{1,0,0\}$ и $\{0,1,0\}$ это значение цвета в RGB формате.

Запустим процесс симулирования. Получим график изображенный на рисунке.



Для построения графика зависимости скорости от времени создадим пользовательский интерфейс.

CoppeliaSim предлагает создание настраиваемого пользовательского интерфейса (*custom user interfaces*) с помощью встроенного плагина *Qt*. При создании пользовательского интерфейса используется специальный *XML* синтаксис. Начиная с версии 4.2.0 разработчики полностью исключили возможность создания пользовательского интерфейса с помощью "*OpenGL-based custom UI*".

Пример синтаксиса плагина *Qt* наиболее хорошо отображен в учебной сцене *customUI.ttt* находящейся в корне папки *scenes* в месте установки программы.

В общем удалим и снова добавим на сцену модель робота *pioneer 3dx*.

Нам потребуется написать немного кода.

В функции `sysCall_init()` значение переменной `v0=2` изменим на ноль:

```
v0=0
```

В конце функции `sysCall_init()` добавим следующий код:

```
xml = '<ui title="Speed" closeable="false" resizable="false" activate="false">..'[[  
<hslider minimum="0" maximum="1000" on-change="speedChange_callback" id="1"/>  
</ui>  
]]  
ui=simUI.create(xml)
```

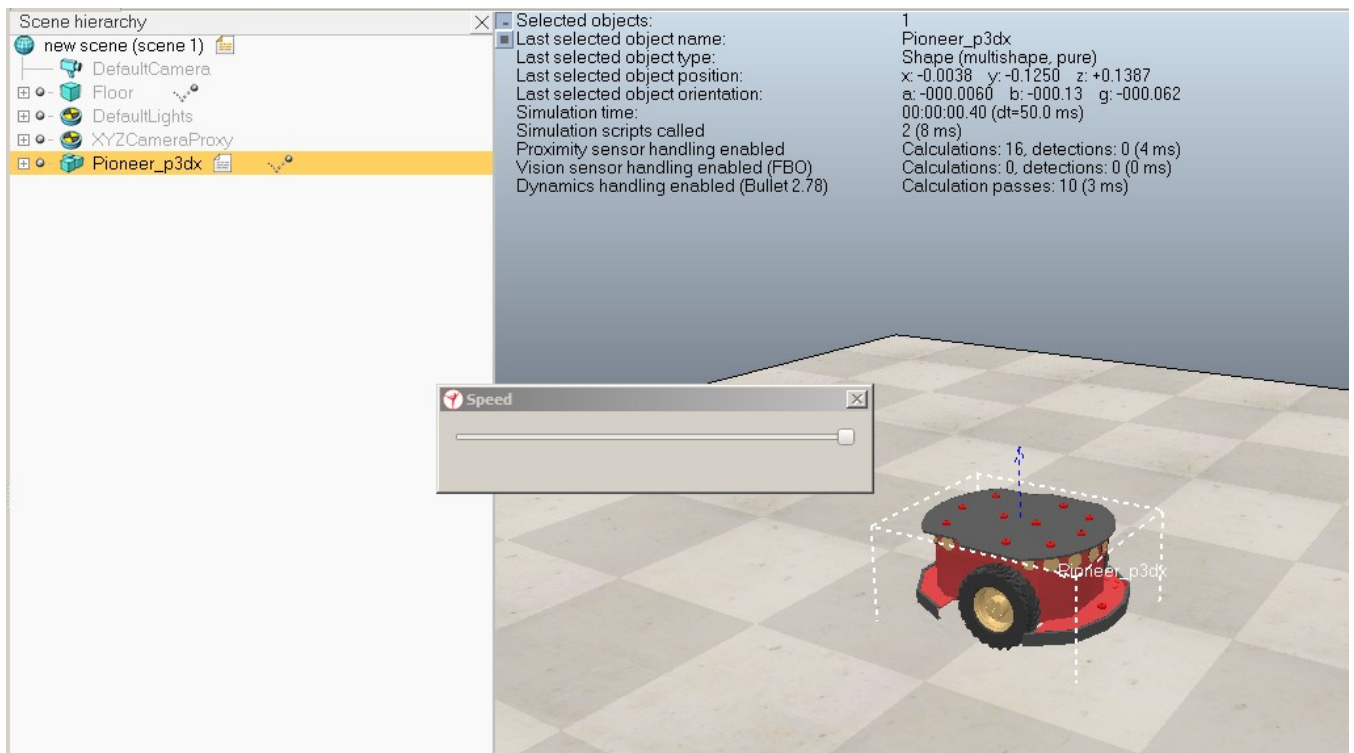
В функции `sysCall_cleanup()` добавим следующую строку:

```
simUI.destroy(ui)
```

В теле скрипта вставим следующую функцию:

```
function speedChange_callback(ui,id,newVal)  
    v0=newVal*2/1000  
end
```

Запустим симулирование. Теперь скоростью робота можно управлять.



Реализуем управление роботом с помощью двух ползунков, чтобы один из них управлял скоростью левого колеса, а другой – правого.

Приведем скрипт к первоначальному виду.

Далее в функции `sysCall_init()` удалим переменную `v0=2` и инициализируем следующие переменные:

```
v1=0
vr=0
```

В конце тела функции `sysCall_init()` вставим следующий код:

```
xml = '<ui title="Speed" closeable="false" resizable="false" activate="false">'.[[
<hslider minimum="0" maximum="1000" on-change="speedChange_callbackL" id="1"/>
<hslider minimum="0" maximum="1000" on-change="speedChange_callbackR" id="2"/>
</ui>
]]
ui=simUI.create(xml)
```

В теле скрипта добавим следующие функции:

```
function speedChange_callbackL(ui,id,newVal)
    v1=newVal*2/1000
end
```

```
function speedChange_callbackR(ui,id,newVal)
    vr=newVal*2/1000
end
```

В функции sysCall_cleanup() добавим следующую строку:

```
simUI.destroy(ui)
```

В функции sysCall_actuation() переменной vLeft и переменной vRight присвоим vl и vr соответственно вместо v0 .

```
vLeft=vl  
vRight=vr
```

Запустим процесс симулирования.

В итоге мобильный двухколесный робот научится поворачивать, и им можно будет управлять.

Построим график зависимости скорости от времени.

В функции sysCall_init() добавим следующий код:

```
graph=sim.getObjectHandle('Graph')  
joint1Vel=sim.addGraphStream(graph,'joint 1 velocity','deg/s',0,{1,0,0})  
joint2Vel=sim.addGraphStream(graph,'joint 2 velocity','deg/s',0,{0,1,0})
```

Добавим функцию sysCall_sensing().

```
function sysCall_sensing()
```

```
end
```

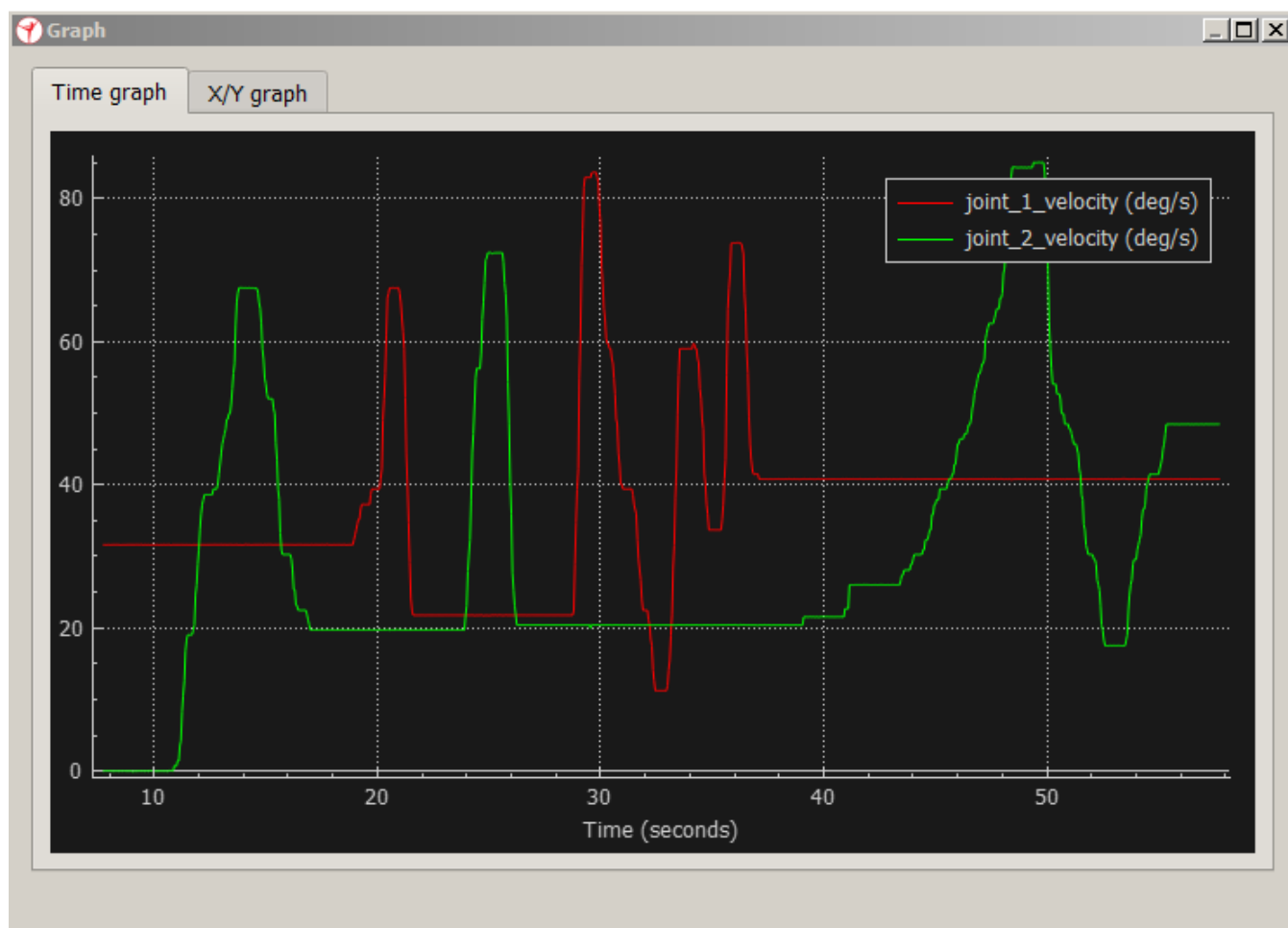
В функции sysCall_sensing() добавим следующий код:

```
sim.setGraphStreamValue(graph,joint1Vel,180*sim.getJointVelocity(motorLeft)/math.pi)  
sim.setGraphStreamValue(graph,joint2Vel,180*sim.getJointVelocity(motorRight)/math.pi)
```

Очистим график после окончания процесса симулирования. Для этого в функцию sysCall_cleanup() добавим следующую строчку.

```
sim.resetGraph(graph)
```

Запустим процесс симулирования, поерзаем ползунками. Получим график изображенный на рисунке.



Таким образом, результаты имитационного моделирования робототехнических систем в программном комплексе *CoppeliaSim* могут быть с легкостью сведены в виде графиков.

5. Контрольные вопросы

1. Что такое дифференциальное управление?
2. Что будет, если вращать колёса в противоположные стороны?

6. Отчёт

- Теория
- Код
- Скриншоты
- Выводы