Содержание

1 Введение **3**

2 Обзорно-постановочный раздел работы **6**

2.1 Основные понятия и определения 6

2.2 Обзор литературы 7

2.3 Постановка задачи 9

3 Метод **12**

3.1 Определение цели для планирования МП 12

3.1.1 Футпринт мобильного манипулятора 13

3.1.2 Область достижимости манипулятора 14

3.1.3 Выбор оптимальной точки 16

3.2 Переносимость метода 16

3.3 Определение позы захвата по облаку точек 17

4 Вычислительные эксперименты **19**

4.1 Симуляционные среды 19

4.2 Сравниваемые методы 20

4.3 Результаты экспериментов 22

5 Заключение **26**

Аннотация

Рассматривается задача сбора и перемещения объектов с помощью мобильного манипулятора (мобильной робототехнической платформы, оснащенной робототехническим манипулятором). Для ее решения требуется определить целевую конфигурацию мобильного манипулятора для захвата целевого объекта с учетом положения этого объекта и препятствий. Целевая конфигурация включает положение мобильной платформы и конфигурацию звеньев манипулятора. Предложен и реализован алгоритм определения целевой конфигурации мобильного манипулятора для тесной среды, когда высокая плотность препятствий требует учитывать при планировании захвата геометрическую форму колесной базы и рабочего пространства манипулятора. В рамках алгоритма выполняется серия сверток карты препятствий ядрами, сформированными с учетом формы базы и рабочего пространства манипулятора, в результате чего определяется множество допустимых положений платформы, из которых манипулятор может захватить объект. В ходе экспериментов в симуляционной среде Isaac Sim предложенный подход показал свою эффективность для тесных сред.

1 Введение

Мобильный манипулятор (ММ) — это роботизированная система, сочетающая мобильную платформу с манипулятором (роботизированной рукой). Такая конструкция позволяет роботу не только перемещаться в пространстве, но и выполнять сложные задачи по взаимодействию с объектами, что делает мобильные манипуляторы очень гибкими и функциональными.

Основные компоненты мобильного манипулятора включают:

- Мобильная платформа (МП): Оснащена колесами, гусеницами или ногами, что позволяет роботу перемещаться по различным поверхностям. Мобильные платформы часто снабжаются датчиками для навигации и избегания препятствий.
- Манипулятор: Роботизированная рука, обычно с несколькими степенями свободы, что позволяет выполнять разнообразные движения и манипуляции. Манипуляторы оснащены захватами или специализированными инструментами для взаимодействия с объектами.

Мобильные манипуляторы, благодаря своей гибкости и многофункциональности, значительно расширяют возможности автоматизации и могут работать в условиях, где традиционные стационарные роботы или человеческий труд были бы менее эффективны. Разработка ММ для автоматизации захвата объектов является одной из ключевых областей исследований в робототехнике. Такие системы находят широкое применение в различных сферах, включая промышленное производство, логистику, сельское хозяйство и медицинские технологии. Одной из главных задач при создании ММ является определение их целевой конфигурации, то есть позиций и ориентаций, необходимых для успешного выполнения захватов объектов в различных условиях.

Целевая конфигурация ММ определяется множеством факторов, включая геометрию и расположение объектов, траекторию движения манипулятора, а также ограничения окружающей среды. Для эффективного решения этой задачи необходимо учитывать как статические, так и динамические аспекты взаимодействия манипулятора с объектами и пространством. Важным элементом является разработка алгоритмов планирования движений, которые позволяют манипулятору адаптироваться к изменяющимся условиям и обеспечивать точное и безопасное выполнение захватов.

Целью данной работы является разработка теоретических и практических основ для создания мобильных манипуляторов, способных эффективно решать задачи захвата объектов в различных условиях. Для достижения этой цели в работе рассматриваются следующие ключевые задачи:

- **Исследование методов захвата объектов:** Изучение различных подходов к захвату объектов манипуляторами, включая анализ существующих техник и их адаптация к условиям реальной среды. Это включает разработку стратегий захвата для объектов с различной формой, размером и ориентацией, а также оптимизацию параметров захвата для обеспечения надежности и точности.

- **Выбор целевых конфигураций для планировщика мобильной платформы с учетом препятствий:** Разработка алгоритмов, позволяющих манипулятору выбирать оптимальные цели и планировать траекторию движения с учетом наличия статических и динамических препятствий. Этот процесс включает оценку окружающей среды, предсказание возможных изменений и адаптацию маршрута для безопасного и эффективного достижения цели.

- **Моделирование и симуляция:** Создание виртуальных моделей мобильных манипуляторов и их тестирование в различных сценариях. Моделирование позволяет проводить эксперименты, оценивать эффективность разработанных алгоритмов и оптимизировать конструкцию манипулятора до его реальной реализации.

- **Экспериментальная валидация:** Проведение экспериментов на прототипах мобильных манипуляторов в реальных условиях. Это позволяет проверить работоспособность предложенных решений, выявить возможные недостатки и внести необходимые коррективы.

Результаты этого исследования могут значительно улучшить сферу роботизации и автоматизации, обеспечивая более эффективное функционирование мобильных манипуляторов в человекоориентированных средах за счет совместного целеполагания платформы с учетом возможностей манипулятора.

В контексте мобильной робототехники существует разнообразие подвижных платформ, которые могут служить основой для ММ. Среди них можно выделить беспилотные летательные аппараты (БПЛА)[4, 5] и шагоходы[7], которые обладают

уникальными характеристиками и применяются в различных областях, от исследований до промышленности. Однако данное исследование сфокусированно на широко распространенном варианте использования колесной платформы в качестве мобильной базы для ММ.

Мобильный манипулятор, оборудованный колесной платформой, представляет собой комплексную робототехническую систему с высокой степенью гибкости и маневренности. Обычно такие системы имеют девять или более степеней свободы, где на три степени свободы приходится мобильная платформа, а оставшиеся шесть степеней свободы обеспечиваются манипулятором. Такая конфигурация делает мобильный манипулятор кинематически избыточной системой, что требует совместного планирования и управления для эффективного выполнения задач.

Совместное целеполагание для мобильного манипулятора с колесной платформой включает учет различных ограничений как базы, так и манипулятора. Это важно для обеспечения координированных и оптимальных движений в пространстве, учитывая особенности работы колесной платформы и возможности манипулятора. Такой подход позволяет максимально эффективно использовать возможности мобильного манипулятора в различных задачах, обеспечивая точность и скорость выполнения операций в разнообразных условиях.

Не полная наблюдаемость среды создает трудности для выполнения задач ММ, а ограниченная маневренность колесных платформ замедляет выполнение задач. Когда ММ получает задачу захвата объекта для оценки позы схвата требуется обладать полной информацией о форме и положении объекта в пространстве. Ее можно получить только совершив манёвр и подъехав к цели по приблизительным координатам, и положение объекта может оказаться таким, что потребует дополнительного маневра. Поэтому важно по неполным данным определить некоторую цель для планирования, из которой робот сможет захватить объект с высоким шансом на успех.

Эта работа посвящена построению алгоритма, решающего две задачи:

- Выделение цели для планирования МП по неполным данным о местоположении объекта и его форме
- Определение позы захвата манипулятора по облаку точек

Метод построен на основе сверки футпринта ММ и области достижимости манипулятора с картой занятости получаемой с LiDAR. Футпринт ММ используется в качестве ядра свертки, что позволяет соотнести каждый пиксель карты с количеством

коллизий платформы с препятствиями. Область область достижимости манипулятора таким же образом позволяет оценить достижимость объекта манипулятором.

Метод был протестирован в симуляторе Nvidia Isaac Sim[1], в сравнении с другими бейзлайнами показывает хорошую эффективность. Для тестирования были разработаны различные среды, которые обычно вызывают проблемы на практике.

Работа состоит из трех глав, каждая из которых раскрывает ключевые аспекты исследования. В обзорно-постановочном разделе рассматриваются современные подходы и результаты исследований в области определения целевой конфигурации мобильного манипулятора для захвата объектов, что помогает обосновать выбор темы и сформулировать цели исследования. В главе о методе подробно описана выбранная методология, включая алгоритмы и методы, используемые для решения поставленных задач. Здесь также объясняется процедура проведения вычислительных экспериментов, направленных на тестирование и верификацию предложенных решений. Завершающая глава посвящена подведению итогов поставленных экспериментов.

2 Обзорно-постановочный раздел работы

2.1 Основные понятия и определения

ММ - Мобильнный манипулятор

МП - Мобильная платформа

$\otimes$ обозначает 2D-свертку с 3D-ядром. Пусть $\mathbf{A}$ — $m \times n$ 2D-матрица, а $\mathbf{B}$ — $a \times b \times k$ 3D-ядро. $\mathbf{C} = \mathbf{A} \otimes \mathbf{B}$ — это $(m - a) \times (n - b) \times k$ 3D-тензор. Каждый $(m - a) \times (n - b)$ слой $\mathbf{C}$ является результатом свертки $\mathbf{A}$ с соответствующим слоем $\mathbf{B}$.

$\odot$ обозначают поэлементное умножение тензоров. $\mathbf{C} = \mathbf{A} \odot \mathbf{B}$ означает, что каждый элемент $\mathbf{C}$ равен произведению соответствующих элементов $\mathbf{A}$ и $\mathbf{B}$.

$\overline{\mathbf{A}}$ обозначают логический обратный каждому элементу в $\mathbf{A}$. Элементы со значением 0 заменяются на 1, остальные элементы заменяются на 0.

[1] `https://developer.nvidia.com/isaac/sim`

2.2 Обзор литературы

Обычно решение проблемы захвата объектов рассматривается для статичных манипуляторов. Примером может быть ROS пайплайн [3]. CAD-модель объекта, который необходимо захватить, оптимизируется к облаку точек. Затем компонент планирования вычисляет траекторию для манипулятора, позволяющую захватить объект. Соответственно в случае ММ планирование придется разделить на 2 независимых этапа. Кроме того, этот подход исходит из предположения о закрытом мире: для каждого объекта, который необходимо захватить, существует точная CAD-модель. В общем случае для объектов без заранее известной CAD-модели он не подходит. Также возникнут проблемы с классификацией, если слой детекции имеет открытый словарь. Есть работы напрямую использующие облака точек без их классификации, что позволяет обходиться без 3Д моделей объектов [13, 18]. Обе эти работы базируются на фильтрации сгенерированных кандидатов точек схвата с помощью CNN. Для мобильного робота, где вычислительные ресурсы довольно ограничены, это может оказаться вычислительно трудным. Помимо детекции захвата на роботе в реальном времени должны работать подсистемы локализации, сегментации и трекинга. Авторы [18] учли времязатратность прогона сначала сегментации, а затем детекции захвата, и объединили их в одну сеть. Но эти работы реализуют только манипуляторную часть для схвата объекта. Для ММ требуется выдачу целей для платформы поставить в зависимость от возможностей манипулятора. Наиболее близка к такому коллаборативному выполнению задач манипулятора и платформы работа [17]. Статья представляет реализацию замкнутого цикла динамического отслеживания и захвата объекта с предварительным знанием целевой позы. В реальных случаях целевая поза заранее неизвестна, а оценить позу робот сможет только доехав до объекта. Перепарковка робота после оценки позы займет некоторое время, это крайне не эффективно. В статье [16] ставится оптимизационная задача совместного движения рабочего органа ММ по ключевым точкам. Приоритет отдается минимизации перемещений платформы, но разница между конфигурациями манипулятора для передвижения между точками также включена в функцию потерь. Авторы демонстрируют этот подход на искусственном примере, без практического приложения. Решение может быть полезным, когда ММ требуется произвести серию действий в близко расположенных точках пространства. Проблема поставлена так, что конфигурации рабочего органа заранее известны и точны, т.е. отсутствует элемент неопре-

деленности среды. В реальных условиях ММ нужно решать задачи, когда целевая конфигурация не известна до выполнения части плана: ММ находится в одной комнате, а целевой объект в другой.

В области роботизированных манипуляций исследования преимущественно сосредоточены либо на статичных манипуляторах с фиксированной базой [30, 29, 28, 27, 19, 20] либо ММ действуют в достаточно простых средах [22, 19, 21, 23]. Однако, существует нарастающий интерес к более сложным сценариям применения, где роботы должны оперировать в динамически меняющихся средах и взаимодействовать с непредсказуемыми объектами и людьми [26, 24, 25]. Эти задачи требуют от манипуляторов не только высокой точности и повторяемости движений, но и способности к адаптации и обучению в реальном времени.

Современные подходы, такие как использование глубокого обучения и усиленного обучения, позволяют разрабатывать модели, способные к более сложным формам взаимодействия и обучению на основе опыта [14, 15, 8]. Например, алгоритмы, основанные на нейронных сетях, могут обучаться на больших объемах данных, имитируя человеческие действия и улучшая свои навыки через пробу и ошибки. Это открывает новые горизонты для роботизированных систем, делая их более автономными и универсальными.

Несмотря на достигнутые успехи, многие вызовы остаются нерешенными. Одной из основных проблем является необходимость значительных вычислительных ресурсов для обучения и работы таких систем в реальном времени [9, 6]. Более того, существует необходимость разработки более надежных методов для обеспечения безопасности и устойчивости роботизированных манипуляторов в условиях неопределенности [12, 11].

Тем не менее прогресс в этой области идет быстрыми темпами, и будущее роботизированных манипуляторов, способных эффективно функционировать в сложных и динамичных средах, выглядит весьма перспективным.

Одной из ключевых проблем в разработке и применении алгоритмов для мобильных манипуляторов является их переносимость между различными конструктивными моделями. Мобильные манипуляторы могут существенно различаться по своему устройству, включая тип мобильной базы (колесная, гусеничная, шагоход или летательный аппарат), количество степеней свободы, тип манипулятора и используемые сенсоры. Эти различия создают значительные трудности при попытке переноса алгоритмов и методов, разработанных для одной модели, на другую.

Основные аспекты проблемы переносимости

- **Различия в кинематике и динамике:** Разные типы мобильных баз имеют различные кинематические и динамические характеристики. Например, колесные платформы обладают высокой маневренностью на ровных поверхностях, но испытывают трудности на пересеченной местности, в то время как шагоходы могут преодолевать сложные препятствия, но обладают меньшей скоростью передвижения. Манипуляторы с разным количеством степеней свободы требуют разных подходов к планированию движения и управления, что усложняет переносимость методов.

- **Различия в сенсорных системах:** Различные модели мобильных манипуляторов могут быть оснащены различными наборами сенсоров (LiDAR, камеры, ультразвуковые датчики и т.д.). Это требует адаптации алгоритмов для обработки и интерпретации данных с этих сенсоров. Например, алгоритмы, использующие данные LiDAR для построения карт препятствий, могут не работать корректно на манипуляторах, оснащенных только камерами.

- **Геометрические и конструктивные особенности:** Форма и размеры мобильной базы и манипулятора могут значительно влиять на алгоритмы планирования движения и захвата объектов. Методы, учитывающие геометрию одной модели, могут быть неприменимы к другой модели с иной конструкцией. Например, методы, разработанные для манипуляторов с длинными манипуляционными звеньями, могут не подходить для компактных манипуляторов с ограниченным рабочим пространством.

Из-за этих особенностей многие методы могут оказаться несравнимыми и не проверяемыми, а отсутствие открытых симуляционных сред лишь усугубляет проблему переносимости. Для преодоления проблемы переносимости требуется адаптивный алгоритм, который можно легко подстроить под любую форму робота и любые возможности манипулятора.

2.3 Постановка задачи

Эта работа сосредоточена на выделении целей для планировщиков платформы и манипулятора при решении задач схвата объектов в помещениях. Объекты могут находиться на открытых поверхностях (пол, стул, стол и т.д.), заранее известно, что



Рис. 1: Мобильный манипулятор husky.

существует такая конфигурация мобильной платформы и манипулятора, которая позволяет захватить объект.

В качестве модельной робототехнической системы рассматривается мобильный манипулятор, который состоит из 4-х колесной платформы Clearpath Husky[2] с дифференциальным приводом и шестистепенного манипулятора UR5[3], как на 1. На рабочем органе манипулятора установлен двупалый захват Robotiq[4]. На манипуляторе закреплено несколько камер глубины (Intel Realsense[5] и Zed Camera[6]) и Velodyne LiDAR[7]. Камера глубины Intel Realsense установлена на рабочем органе манипулятора, именно эта камера используется для точного позиционирования объектов и извлечения облаков точек.

В реальных задачах после поступления команды на хватание целевого объекта, робот не может определить заранее ориентацию и точную позу, требуемую для успешного захвата. Данные для этого появляются уже после того, как робот совершил манёвр и приблизился к объекту. Ограничением является область достижимости точек манипулятором, которая меняется в зависимости от требуемой ориентации. Самым худшим случаем является ориентация захвата перпендикулярная к поверхности, на которой лежит объект. Такая поза сильно уменьшает область достижимости манипулятора. Поэтому метод должен полагаться на худший случай ориентации захвата.

---

[2] https://clearpathrobotics.com/husky-unmanned-ground-vehicle-robot

[3] https://www.universal-robots.com/products/ur5-robot

[4] https://robotiq.com/products/2f85-140-adaptive-robot-gripper

[5] https://www.intelrealsense.com/depth-camera-d435

[6] https://www.stereolabs.com/docs

[7] https://velodynelidar.com

Для платформы требуется определить точку с данными свойствами:

- В точку существует путь.
- Из точки возможно построить траекторию для манипулятора в координаты объекта с худшей ориентацией.

Для манипулятора требуется по облаку точек целевого объекта определить конфигурацию манипулятора, позволяющую захватить объект.

Конфигурация мобильного манипулятора $\mathbf{S}$ включает в себя положение мобильной базы $\mathbf{X}_b$ (декартовы координаты и ориентация) и конфигурацию звеньев манипулятора $\mathbf{\Theta}$ (углы поворота каждого звена). Известна функция прямой кинематики мобильного манипулятора, позволяющая рассчитать положение и ориентацию гриппера $\mathbf{X}_g = \mathbf{X}_b + FK(\mathbf{\Theta})$. Отметим, что нам также известен алгоритм расчета обратной кинематики, позволяющий рассчитать конфигурацию звеньев для заданной пары положения платформы и положения гриппера $\hat{\mathbf{\Theta}} = IK(\mathbf{X}_g - \mathbf{X}_b)$.

Входными данными для задачи являются: начальная конфигурация ММ $\mathbf{S}(0)$, множество точек целевого объекта $\mathcal{T}$, множество препятствий $\mathcal{O}$. С учетом того, что движение мобильной базы и манипулятора планируется и выполняется по отдельности, задача планирования захвата разбивается на ряд подзадач:

1. Определить целевое положение гриппера $\mathbf{X}_g(T)$ для захвата объекта;
2. Определить целевое положение платформы $\mathbf{X}_b(T)$, которое не будет содержать коллизий с препятствиями и из которого будет достижимо $\mathbf{X}_g(T)$;
3. Определить план движения платформы из $\mathbf{X}_b(0)$ в $\mathbf{X}_b(T)$;
4. Определить план движения манипулятора из конфигурации $\mathbf{\Theta}(0)$ в конфигурацию $\mathbf{\Theta}(T)$, такую, что $\mathbf{X}_g(T) = \mathbf{X}_b(T) + FK(\mathbf{\Theta}(T))$

Подзадача 1 решается разработанным подходом на основе анализа облаков точек. Для решения подзадачи 4 применяются простые правила - эксперименты показали, что их достаточно. Интерес представляют прежде всего новые подходы к подзадачам 2 и 3. Описание алгоритма в виде псевдокода приведено в пункте 2.1.

Алгоритм 2.1 Планирование захвата для мобильного манипулятора

Вход: Начальная конфигурация $\mathbf{S}(0)$, множество точек целевого объекта $\mathcal{T}$, множество препятствий $\mathcal{O}$

Выход: Траектории движения платформы и манипулятора

Function PlanGripperTarget($\mathcal{T}$)
  Определить целевое положение гриппера $\mathbf{X}_g(T)$ для захвата объекта
  return $\mathbf{X}_g(T)$
$\mathbf{X}_g(T) \leftarrow$ PlanGripperTarget($\mathcal{T}$)
Function PlanBaseTarget($\mathbf{X}_g(T)$, $\mathcal{O}$)
  Определить целевое положение платформы $\mathbf{X}_b(T)$, которое:
    (a) Не будет содержать коллизий с препятствиями из $\mathcal{O}$
    (b) Позволит достичь $\mathbf{X}_g(T)$ манипулятором
  return $\mathbf{X}_b(T)$
$\mathbf{X}_b(T) \leftarrow$ PlanBaseTarget($\mathbf{X}_g(T)$, $\mathcal{O}$)
Function PlanBaseTrajectory($\mathbf{X}_b(0)$, $\mathbf{X}_b(T)$)
  Определить траекторию движения платформы из $\mathbf{X}_b(0)$ в $\mathbf{X}_b(T)$
  return Траектория движения платформы
ТраекторияПлатформы $\leftarrow$ PlanBaseTrajectory($\mathbf{X}_b(0)$, $\mathbf{X}_b(T)$)
Function PlanManipulatorTrajectory($\mathbf{\Theta}(0)$, $\mathbf{X}_g(T)$, $\mathbf{X}_b(T)$)
  Определить конфигурацию манипулятора $\mathbf{\Theta}(T)$ такую, что:
    $\mathbf{X}_g(T) = \mathbf{X}_b(T) + FK(\mathbf{\Theta}(T))$
  Определить траекторию движения манипулятора из $\mathbf{\Theta}(0)$ в $\mathbf{\Theta}(T)$
  return Траектория движения манипулятора
ТраекторияМанипулятора $\leftarrow$ PlanManipulatorTrajectory($\mathbf{\Theta}(0)$, $\mathbf{X}_g(T)$, $\mathbf{X}_b(T)$)
return ТраекторияПлатформы, ТраекторияМанипулятора

3 Метод

3.1 Определение цели для планирования МП

У мобильных платформ есть надежный источник точных данных - LiDAR. С его помощью строится достаточно точная карта занятости, идеей алгоритма является использование данных именно с этого сенсора. Алгоритм описывается для двумерной карты занятости, но без модификаций возможно использование 2.5Д карты.

Метод состоит из трех этапов:

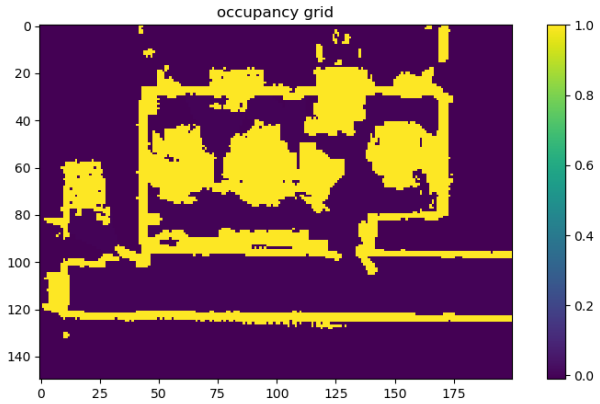

Рис. 2: Двухмерная карта занятости. Каждая клетка матрицы заполнена значениями: 0 - свободное место, 1 - препятствие.

1. Свертка карты занятости с футпринтом ММ
2. Свертка матрица положения цели с областью достижимости манипулятора
3. Выбор оптимальной точки

Далее будет подобно описан каждый из этапов алгоритма. Этапы 1 и 2 протекают независимо друг от друга. Этап 3 объединяет результаты их выполнения.

3.1.1 Футпринт мобильного манипулятора

Определение 3.1. *Футпринтом называется область, которую занимает робот. Представляется в виде черно-белого изображения, где занятые ячейки помечаются значением* 1*, а не занятые* 0*. Центр изображения соответствует центру робота. Пример футпринта представлен на рисунке 3.*

Для учета различных ориентаций ММ футпринта поворачивается на дискретные углы, таким образом получается трехмерное ядро свертки. Пусть $\mathbf{M}_{OG}$ — двумерная сетка занятости (рис. 2), а $\mathbf{G}_{fp}$ — трёхмерное ядро свертки, представляющее изображения клеток занимаемых роботом (футпринт) для различных ориентации (центр $\mathbf{G}_{fp}$ соответствует центру системы координат основания робота, 3). Здесь и n — количество ячеек занятости, которое должно содержать максимальный размер футпринта робота; m — разрешение для дискретизации возможных ориентаций робота. Тензор безопасных поз $\mathbf{T}_{sp}$ определяется как результат 2D-свертки (рис. 4) $\mathbf{M}_{OG}$ и $\mathbf{G}_{fp}$:

$$\mathbf{T}_{sp} = \mathbf{M}_{OG} \otimes \mathbf{G}_{fp}.$$

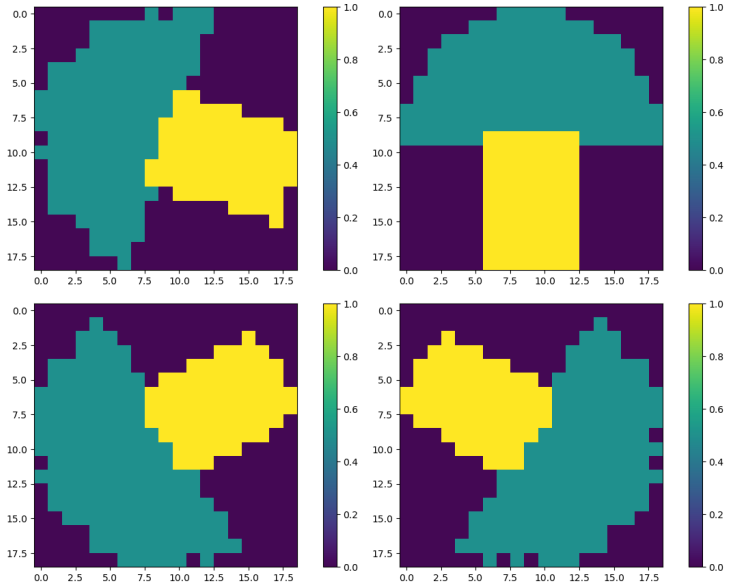

Рис. 3: Футпринт робота с различными углами поворота и разрешением равным разрешению карты занятости. Желтым цветом обозначена платформа, бирюзовым - область достижимости манипулятора.

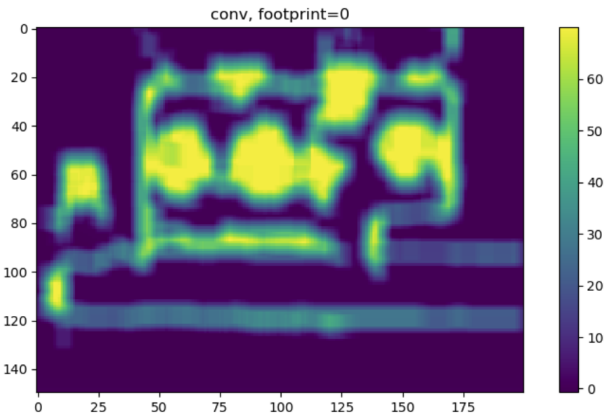

Рис. 4: Визуализация одного канала результата свертки футпринта и карты занятости. Каждая клетка содержит значение равное количеству пикселей с препятствиями, которые задевает робот в данных координатах.

Таким образом в каждой ячейке тензора $\mathbf{T}_{sp}$ будет содержаться количество точек, с которыми соприкасается футпринт в данной позиции и ориентации. Это позволяет выбрать из этого множества подходящие точки.

3.1.2 Область достижимости манипулятора

Определение 3.2. *Областью достижимости манипулятора называется область, в которую манипулятор может привести рабочий орган в любых ориентациях. Пред-*

ставляется в виде черно-белого изображения, где достижимые ячейки помечаются значением 1*, а не достижимые* 0*. Центр изображения соответствует центру робота. Пример области достижимости представлен на рисунке 3.*

Область достижимости меняется в зависимости от высоты, которую необходимо достичь манипулятору, поэтому выбирается двумерный срез этой области, исходя из высоты целевого объекта относительно базы манипулятора. Точки, которые манипулятор способен достичь можно представить в виде сферы или полусферы, если необходимо исключить опасные для бортового оборудования конфигурации. Для рисования сферы на изображении размером $n \times m$ с центром в точке (x, y) и радиусом R, можно использовать алгоритм, который проверяет каждую точку изображения и определяет, попадает ли эта точка в сферу (окружность на 2D-плоскости). Алгоритм можно описать следующим образом:

1. Инициализируем пустое изображение размером $n \times m$.
2. Для каждой точки (i, j) на изображении:
   - Вычисляем расстояние от этой точки до центра сферы.
   - Если расстояние меньше или равно радиусу R, закрашиваем эту точку (например, устанавливаем значение пикселя в 1).
3. Сохраняем или отображаем изображение.

Для проверки, попадает ли точка (i, j) в сферу, используем уравнение окружности:

$$(i - x)^2 + (j - y)^2 \leqslant R^2$$

Для определения R зададим l как максимальное безопасное расстояние, которое может достичь манипулятор относительно базы, h_m как позицию базы манипулятора относительно мировых координат и h_t как высоту объекта в мировых координатах. Тогда $R = \sqrt{l^2 - (h_t - h_m)^2}$. Такое упрощение допустимо так как в нашем случае ММ перемещается вдоль плоскости.

С рабочей областью манипулятора производятся аналогичные действия, как и с футпринтом мобильной платформы: для учета различных ориентаций ММ область достижимости поворачивается на дискретные углы, таким образом получается трехмерное ядро свертки. Пусть $\mathbf{M}_d$ — матрица положения цели (она имеет те же размеры, что и сетка занятости, ячейки в некоторой области рядом с целевым положением захвата помечаются 1, остальные ячейки имеют нулевое значение) и $\mathbf{G}_{ws}$ —

трёхмерное ядро свертки, представляющее рабочее пространство манипулятора для различных ориентаций (центр $\mathbf{G}_{ws}$ соответствует центру системы координат следа основания робота). Мы определяем тензор достижимости цели $\mathbf{T}_{gr}$ как результат двумерной свертки $\mathbf{M}_d$ и $\mathbf{G}_{ws}$:

$$\mathbf{T}_{gr} = \mathbf{M}_d \otimes \mathbf{G}_{ws}.$$

Таким образом в каждой ячейке тензора $\mathbf{T}_{gr}$ будет содержаться оценка вероятности возможности схватить цель в данной позиции и ориентации. Это позволяет выбрать из этого множества подходящие точки.

3.1.3 Выбор оптимальной точки

В тензорах $\mathbf{T}_{gr}$ и $\mathbf{T}_{fp}$ содержаться некоторые оценки для каждой позиции и ориентации робота. Из этого множества точек требуется выбрать оптимальные. Зададим σ_{sp} и σ_{gr} как пороги для элементов из $\mathbf{T}_{sp}$ и $\mathbf{T}_{bf}$ соответственно. Перебирая все точки расположенные не дальше R от целевого объекта найдем такие, что $\mathbf{T}_{sp}(\theta, x, y) \leq \sigma_{sp}$ и $\mathbf{T}_{gr}(\theta, x, y) \leq \sigma_{gr}$. Это позволит найти множество точек, в которых платформа не имеет коллизий с препятствиями и манипулятор имеет наибольшие шансы достичь объект. Точка с подходящими параметрами существовать будет не всегда, в этом случае никаких действий ММ совершать не будет.

3.2 Переносимость метода

Так как в робототехнике довольно актуальной проблемой является вопрос переносимости методов с одной конструкции роботов на другую имеет смысл рассмотреть переносимость создаваемого метода.

Описанный выше метод создавался с учетом возможности изменения его под различные конструкции роботов. Такое свойство обеспечивается за счет представления футпринта и области достижимости манипулятора в виде изображений. Поэтому для подстройки метода под иную конструкцию достаточно будет их отредактировать. Например, если необходимо адаптировать метод для робота с другой формой платформы или с манипулятором, закрепленным в другом месте, достаточно будет обновить соответствующие изображения.

Таким образом, использование изображений для представления футпринта и области достижимости манипулятора делает метод гибким, наглядным и удобным

для адаптации под различные конструкции роботов, что значительно упрощает и ускоряет процессы настройки и модификации.

3.3 Определение позы захвата по облаку точек

Когда робот завершит свой маневр и приблизится к объекту, он сможет создать точное облако точек, описывающее объект. Положение захвата определяется в системе координат самого объекта. Из этого облака точек необходимо извлечь предсказуемую систему координат. Поэтому в основе метода лежит поиск главной диагонали как довольно предсказуемая система координат, извлекаемая из неполного облака точек. Это относится к анализу главных компонент (PCA) и часто связано с понятием последней главной компоненты. В контексте PCA главные компоненты вычисляются таким образом, что первая главная компонента объясняет наибольшую часть дисперсии данных, вторая главная компонента объясняет наибольшую возможную дисперсию среди оставшихся и так далее. Последняя главная компонента будет направлением, вдоль которого дисперсия минимальна. Для большого числа объектов такого подхода достаточно. Но если требуется расширить число объектов, с которыми такой подход сможет эффективно работать, достаточно добавить поиск вдоль главной диагонали оптимальной точки.

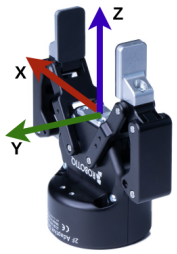

Рис. 5: Визуализация базиса захвата.

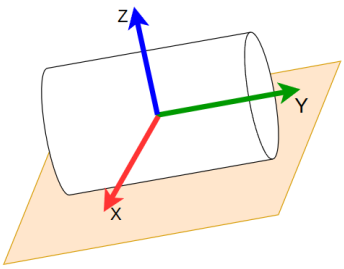

Рис. 6: Схематичное представления базиса объекта, полученного из облака точек с помощью линейной регрессии.

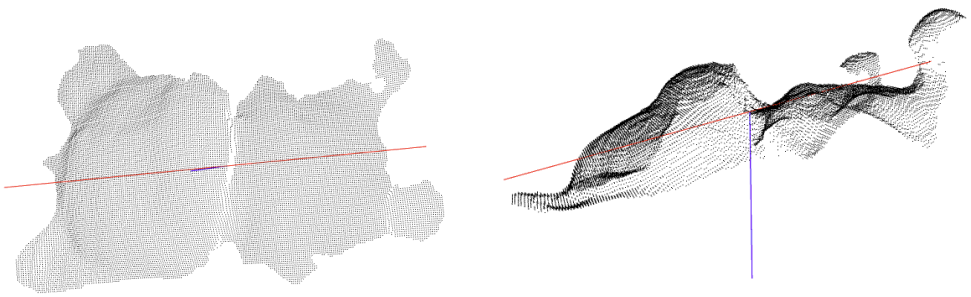

Рис. 7: Визуализация результата работы алгоритма построения базиса объекта. Красная линяя - главная диагональ, синяя - нормаль разделяющей плоскости.

Конструкция базиса по облаку точек вдоль главной диагонали, вдоль которой минимальная дисперсия, может быть выполнена с использованием линейной регрессии с L2 регуляризацией[2]. Начнем с подготовки данных, которые собираются в матрицу X, где каждая строка представляет одну точку, а каждый столбец — одну координату. Данные центрируются, вычитая среднее значение из каждого признака, чтобы сместить центр облака точек в начало координат:

$$X_{\text{centered}} = X - \mu,$$

где μ — вектор средних значений по каждому признаку. Линейная регрессия с L2 регуляризацией решает задачу минимизации функции

$$\min_{w} \left(\|Xw - y\|^2 + \lambda \|w\|^2 \right),$$

где y — целевой вектор, w — вектор весов регрессии, а λ — параметр регуляризации.

Главная диагональ определяется направлением вектора регрессии w, который указывает на направление, вдоль которого изменяется целевая переменная. После определения главной диагонали новый базис для данных строится следующим образом: первый вектор нового базиса u_1 направлен вдоль вектора w, а второй и последующие векторы определяются с использованием ортогонализации, чтобы они были перпендикулярны к u_1. Если необходимо построить полный ортонормированный базис, используется процесс Грама-Шмидта для ортогонализации оставшихся векторов и их последующей нормализации. Итоговый базис состоит из ортонормированных векторов, где первый вектор соответствует направлению наибольшей дисперсии вдоль главной диагонали, определенной линейной регрессией с L2 регуляризацией.

Относительно полученного базиса (рис. 6) задана матрица, которая совмещает главную диагональ объекта (y ось) с y осью манипулятора и z ось объекта с z осью захвата. Точка схвата определяется как центр облака точек. Пример работы такого подхода показан на рис. 7.

4 Вычислительные эксперименты

4.1 Симуляционные среды

В качестве экспериментальной среды был выбран Isaac Sim, разработанный компанией NVIDIA. Эта платформа обеспечивает высокоуровневую симуляцию для тестирования и отладки алгоритмов робототехники и искусственного интеллекта в виртуальных средах, приближенных к реальным.

Isaac Sim позволяет моделировать сложные сценарии, включая взаимодействие роботов с окружающей средой, что крайне важно для проверки и оптимизации гиперпараметров алгоритмов планирования и навигации. Благодаря реалистичным физическим моделям и поддержке различных датчиков, таких как лидары, камеры и ИМУ, Isaac Sim предоставляет разработчикам возможность создавать точные и воспроизводимые тесты. Также этот симулятор бы выбран по той причине, что в нем уже имелись некоторые наработки и часть воссозданной реальной среды.

В данном симуляторе было разработано 3 сценария:

1. Два стола стоят перпендикулярно друг другу, объект создается рядом с пересечением (рис. 8).

2. Два стола стоят параллельно друг другу, объект создается на столе в углу ближе к стене (рис. 9).

3. Аналогичен предыдущему, но между столами стоит стул, объект создается на стуле (рис. 10).

На каждом эксперименте начальная позиция робота задается случайным образом в пределах одной комнаты. Этот случайный выбор позиции обеспечивает разнообразие условий для тестирования и позволяет оценить способность робота к захвату объекта из различных начальных положений для алгоритмов, которые зависят от начального положения робота.

После прохода алгоритма робот телепортируется в полученную точку, вместо того чтобы выполнять манёвр к этой точки. Это позволяет существенно сократить время проведения одного эксперимента, потому что Isaac Sim крайне ресурсоемкий симулятор, и даже со всеми возможными оптимизациями проведения эксперимента, один эксперимент занимал $2-5$ минут (в зависимости от того успешный он или нет). Затем запускается алгоритм захвата объекта: робот получает облако точек целевого объекта и планирует траекторию к этому объекту, используя простое перемещение по точкам в декартовой системе координат. Выполнение траектории обеспечивается функцией MoveL из стандартного API Universal Robotics[8]. И в конце подается команда на гриппер для его закрытия.

Успешность эксперимента определяется по наличию события коллизии между захватом манипулятора и целевым объектом. Если за отведенный промежуток времени на эксперимент коллизия не произошла, то такой эксперимент помечается как не успешный. Эти события генерирует сама среда симуляции и их можно принять за успешное выполнение захвата. Если создать более сложные условия, например: робот должен сложиться с захваченным объектом не выронив его, велик шанс, что большая часть экспериментов будет засчитана за провальные из-за особенностей физики самого симулятора. В ходе работ с симулятором было выяснено, что он крайне не надежно работает с нашей моделью захвата. Пальцы захвата могут выскакивать из пазов или не достаточно сильно сжимать объект, что приводит к его выпадению из захвата.

В конце всех экспериментов полученные результаты перепроверялись вручную по изображениям с камеры закрепленной на манипуляторе, так как симулятор не всегда генерировал событие коллизии по неизвестным причинам. Результаты экспериментов будут содержать уже перепроверенные данные.

4.2 Сравниваемые методы

Для сравнения было разработано несколько простых методов, решающих задачу выделения цели для планирования платформы. Некоторые из них были созданы в ходе решения других задач на реальном роботе в качестве временного решения. Одной из ключевых причин, по которым не проводится сравнение описываемого метода

[8] `https://www.universal-robots.com/download/manuals-e-seriesur20ur30/script/script-manual-e-series-sw-511`



Рис. 8: Тестовая среда. Сценарий №1. Область появления объекта обозначена прямоугольником.



Рис. 9: Тестовая среда. Сценарий №2. Область появления объекта обозначена прямоугольником.

с методами из других статей, является отсутствие общих данных и тестовых условий. Это исследование использует уникальные наборы данных и специфические тестовые условия, разработанные для решения конкретных задач, которые не совпадают с теми, что использованы в других работах. Методы из других статей часто тестировались в различных контекстах и на разных данных, что делает прямое сравнение некорректным и потенциально вводящим в заблуждение. Для обеспечения объективности и достоверности результатов, основное внимание было уделено тщательному тестированию и валидации методов в условиях, наиболее близких к реальным сценариям, для которых они предназначены.



Рис. 10: Тестовая среда. Сценарий №3. Область появления объекта обозначена прямоугольником.

1. **Simple**: Проводит вектор из текущей позиции робота до целевого объекта и отступает некоторый радиус от цели, эту точку и возвращает (алг. 4.1).

2. **A***: Карта занятости маскируется вокруг целевого объекта и планировщик строит до него путь. Путь строится классическим алгоритмом A* [1]. Далее узлы из конца пути отбрасываются так, чтобы конечная точка была на некотором радиусе от объекта (алг. 4.2).

3. **A***+: Аналогичен предыдущему, но учитывает занятость соседних узлов при выборе нового (алг. 4.3).

Гиперпараметры алгоритмов представлены в таблице 1. Эти параметры играют ключевую роль в производительности и поведении алгоритмов. Они были выбраны на основе экспериментальных данных и оптимизированы для достижения наилучших результатов на заданном наборе данных. Каждый гиперпараметр в таблице был настроен с учетом специфики задачи и характеристик робота.

4.3 Результаты экспериментов

С каждым алгоритмом было произведено по 100 экспериментов (таб. 2), это количество можно считать достаточным для оценки подходов. Извлечение карты занятости из симулятора довольно ресурсоемкая операция, что, в совокупности с длительностью одного эксперимента, замедляло бы проведение тестов. Исходя из этого разрешение карты занятости было выбрано равным 10см. на пиксель. Такое

Алг.	**Параметр**	**Знач.**	**Описание**
Simple	R	0.8	Радиус отступа от целевого объекта
A*	R	0.7	От построенного пути узлы отбрасываются до тех пор, пока расстояние до объекта не будет больше или равно R.
	R_t	2	Препятствия вокруг цели в пределах этого радиуса помечаются как свободные клетки для существования пути в точку.
A*+	R	0.7	От построенного пути узлы отбрасываются до тех пор, пока расстояние до объекта не будет больше или равно R.
	R_t	2	Препятствия вокруг цели в пределах этого радиуса помечаются как свободные клетки для существования пути в точку.
	R_{aware}	2	Область проверки соседей на наличие препятствий
FpConv	R	1.3	Радиус поиска оптимальной точки
	I	2000	Количество итераций семплирования
	R_t	1	Радиус раздутия объекта
	σ_{sp}	0	Количество допустимых коллизий футпринта робота с препятствиями
	σ_{gr}	0.55	Порог для достижимости в относительных величинах

Таблица 1: Описание гиперпараметров алгоритмов.

Алгоритм 4.1 Алгоритм **Simple**

Вход: P_r – текущая позиция робота (x_r, y_r), P_t – позиция целевого объекта (x_t, y_t), R – радиус отступа от целевого объекта

Выход: Точка P_{target}, отступающая радиус R от целевого объекта

1: $\vec{d} \leftarrow (x_t - x_r, y_t - y_r)$ — Вычисляем вектор направления от робота к цели
2: $d \leftarrow \sqrt{(x_t - x_r)^2 + (y_t - y_r)^2}$ — Вычисляем длину вектора $\vec{d}$
3: **если** $d \leqslant R$ **то**
4: **raise error** — Если робот уже находится в радиусе R от цели, возвращаем ошибку
5: $\vec{d_{unit}} \leftarrow \left(\frac{x_t - x_r}{d}, \frac{y_t - y_r}{d}\right)$ — Нормализуем вектор $\vec{d}$
6: $P_{target} \leftarrow (x_t - R \cdot \vec{d_{unitx}}, y_t - R \cdot \vec{d_{unity}})$ — Вычисляем точку, отступающую на радиус R от цели
7: **return** P_{target}

Алгоритм 4.2 Алгоритм **A***

Вход: P_r – текущая позиция робота (x_r, y_r), P_t – позиция целевого объекта (x_t, y_t), R – радиус отступа от целевого объекта, *occupancy_map* – карта занятости

Выход: Путь P до точки, отстоящей на радиус R от целевого объекта

1: $masked_map \leftarrow mask_around_target(occupancy_map, P_t, R)$ — Маскируем карту занятости вокруг цели на радиус R
2: $path \leftarrow plan_path(P_r, P_t, masked_map)$ — Планируем путь от позиции робота до цели на маскированной карте
3: **для** $i = \text{length}(path)$ to 1
4: $P_{current} \leftarrow path[i]$
5: $d \leftarrow \sqrt{(P_{current}.x - P_t.x)^2 + (P_{current}.y - P_t.y)^2}$
6: **если** $d > R$ **то**
7: **continue**
8: **иначе**
9: **break**
10: $P \leftarrow path[1 : i]$ — Обрезаем путь до точки, отстоящей на радиус R от цели
11: **return** P

же разрешение используется в нашем реальном роботе. В этом же разрешении задается футпринт робота, и его объективно недостаточно для полного описания формы.

Алгоритм 4.3 Алгоритм **A+**

Вход: P_r – текущая позиция робота (x_r, y_r), P_t – позиция целевого объекта (x_t, y_t), R – радиус отступа от целевого объекта, *occupancy_map* – карта занятости

Выход: Путь P до точки, отстоящей на радиус R от целевого объекта

$closedSet \leftarrow \{\}$ — Множество посещенных узлов
$openSet \leftarrow \{P_r\}$ — Множество кандидатов на посещение, начинаем с текущей позиции робота
$cameFrom \leftarrow \{\}$ — Словарь для восстановления пути
пока *openSet* не пусто
  $current \leftarrow$ узел из *openSet* с наименьшей стоимостью f — Выбираем узел с наименьшей стоимостью
  если *current* близок к P_t на расстоянии не более R **то**
    return $reconstruct_path(cameFrom, current)$ — Обрезаем путь и возвращаем его
  Удаляем *current* из *openSet*
  Добавляем *current* в *closedSet*
  для каждого соседа *neighbor* узла *current*
    если *neighbor* находится в *closedSet* **то**
      continue
    $tentative_gScore \leftarrow gScore[current] + dist(current, neighbor)$ — Расстояние от начального узла до *neighbor* через *current*
    если *neighbor* не в *openSet* или $tentative_gScore < gScore[neighbor]$ **то**
      $cameFrom[neighbor] \leftarrow current$
      $gScore[neighbor] \leftarrow tentative_gScore$
      $fScore[neighbor] \leftarrow gScore[neighbor] + heuristic(neighbor, P_t)$
      если *neighbor* не в *openSet* **то**
        Добавляем *neighbor* в *openSet*
return failure — Не удалось найти путь

Большинство ошибок нашего метода связано с невозможностью найти точку, удовлетворяющую заданным параметрам поиска. Дополнительно симулятор не позволяет исключить модель робота из генератора карты занятости, и его приходится маскировать на карте. Это также может негативно сказываться на работе алгоритмов, на реальном роботе такого не происходит. Помимо этого Isaac-sim не поддерживает

	Сценарий №1	Сценарий №2	Сценарий №3
Alg	SR, %	SR, %	SR, %
Simple	8	36	10
A*	55	47	76
A*+	58	55	71
FpConv	**84**	**68**	**96**

Таблица 2: Успешность алгоритмов на 100 экспериментах.

сборку из двух роботов (манипулятор и платформа) одного целого, есть возможность их присоединить друг к другу через виртуальное фиксированное сочленение. При телепортации платформы это вызывает много проблем с дерганьем, т.к. манипулятор перемещается на следующем кадре после платформы и толкает ее. Это частично удалось исправить и уменьшить толчки, но полностью они не пропали. Это также портит некоторые эксперименты.

Эксперименты, проведенные в симуляционной, показали, что предложенный подход эффективно справляется с задачами в условиях тесных пространств с высокой плотностью препятствий. Эти результаты демонстрируют, что алгоритм способен надежно определять целевую конфигурацию мобильного манипулятора, учитывая геометрическую форму колесной базы и рабочего пространства манипулятора. Таким образом, можно считать, что эксперименты предоставляют надежную проверку алгоритма.

5 Заключение

Разработан и реализован алгоритм для определения целевой конфигурации мобильного манипулятора в условиях тесной среды, где высокая плотность препятствий требует учета геометрической формы колесной базы и рабочего пространства манипулятора при планировании захвата. В рамках алгоритма выполняется серия сверток карты препятствий с использованием ядер, сформированных на основе формы базы и рабочего пространства манипулятора, что позволяет определить множество допустимых положений платформы для захвата объекта манипулятором. Эксперименты в симуляционной среде Isaac Sim подтвердили эффективность предложенного подхода для тесных сред. Полученный метод **FpConv** показывает хороший процент успеха

в симуляторе, лучший результат порядка 96% успешных захватов. Метод **FpConv** обладает таким свойством, как переносимость, что позволяет использовать его для различных конструкций роботизированных платформ.

Список литературы

1. *Hart P.*, *Nilsson N.*, *Raphael B.* A Formal Basis for the Heuristic Determination of Minimum Cost Paths // IEEE Transactions on Systems Science and Cybernetics. — 1968. — Т. 4, № 2. — С. 100—107. — DOI: 10.1109/tssc.1968.300136. — URL: https://doi.org/10.1109/tssc.1968.300136.

2. *Hinton G. E.* A Practical Guide to Training Restricted Boltzmann Machines // Neural Networks: Tricks of the Trade: Second Edition / под ред. G. Montavon, G. B. Orr, K.-R. Müller. — Berlin, Heidelberg : Springer Berlin Heidelberg, 2012. — С. 599—619. — ISBN 978-3-642-35289-8. — DOI: 10.1007/978-3-642-35289-8_32. — URL: https://doi.org/10.1007/978-3-642-35289-8_32.

3. Mobile Manipulation in Unstructured Environments: Perception, Planning, and Execution / S. Chitta, E. G. Jones, M. Ciocarlie, K. Hsiao // IEEE Robotics & Automation Magazine. — 2012. — Т. 19, № 2. — С. 58—71. — DOI: 10.1109/MRA.2012.2191995.

4. *Papachristos C.*, *Alexis K.*, *Tzes A.* Efficient force exertion for aerial robotic manipulation: Exploiting the thrust-vectoring authority of a tri-tiltrotor uav // 2014 IEEE international conference on robotics and automation (ICRA). — IEEE. 2014. — С. 4500—4505.

5. *Papachristos C.*, *Alexis K.*, *Tzes A.* Technical activities execution with a tiltrotor uas employing explicit model predictive control // IFAC Proceedings Volumes. — 2014. — Т. 47, № 3. — С. 11036—11042.

6. Human-level control through deep reinforcement learning / V. Mnih [и др.] // Nature. — 2015.

7. Mobile manipulation and mobility as manipulation—design and algorithms of RoboSimian / P. Hebert, M. Bajracharya, J. Ma, N. Hudson, A. Aydemir, J. Reid, C. Bergh, J. Borders, M. Frost, M. Hagman [и др.] // Journal of Field Robotics. — 2015. — Т. 32, № 2. — С. 255—274.

8. End-to-End Training of Deep Visuomotor Policies / S. Levine [и др.] // Journal of Machine Learning Research. — 2016.

9. Mastering the game of Go with deep neural networks and tree search / D. Silver [и др.] // Nature. — 2016.

10. 4-DoF tracking for robot fine manipulation tasks / M. Siam, A. Singh, C. Perez, M. Jagersand // 2017 14th Conference on Computer and Robot Vision (CRV). — IEEE. 2017. — С. 329—336.

11. CAD2RL: Real Single-Image Flight Without a Single Real Image / F. Sadeghi [и др.] // Proceedings of Robotics: Science and Systems. — 2017.

12. Domain Randomization for Transferring Deep Neural Networks from Simulation to the Real World / J. Tobin [и др.] // Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems. — 2017.

13. Grasp Pose Detection in Point Clouds / A. ten Pas, M. Gualtieri, K. Saenko, R. Platt. — 2017. — arXiv: 1706.09911 [cs.RO].

14. Model-Agnostic Meta-Learning for Fast Adaptation of Deep Networks / C. Finn [и др.] // Proceedings of the International Conference on Machine Learning. — 2017.

15. QT-Opt: Scalable Deep Reinforcement Learning for Vision-Based Robotic Manipulation / D. Kalashnikov [и др.] // arXiv preprint arXiv:1806.10293. — 2018.

16. *Vafadar S.*, *Olabi A.*, *Panahi M. S.* Optimal motion planning of mobile manipulators with minimum number of platform movements // 2018 IEEE International Conference on Industrial Technology (ICIT). — 2018. — С. 262—267. — DOI: 10.1109/ICIT.2018.8352187.

17. *Arora P.*, *Papachristos C.* Mobile Manipulator Robot Visual Servoing and Guidance for Dynamic Target Grasping // Advances in Visual Computing / под ред. G. Bebis, Z. Yin, E. Kim, J. Bender, K. Subr, B. C. Kwon, J. Zhao, D. Kalkofen, G. Baciu. — Cham : Springer International Publishing, 2020. — С. 223—235. — ISBN 978-3-030-64559-5.

18. Grasp Pose Detection with Affordance-based Task Constraint Learning in Single-view Point Clouds / K. Qian, X. Jing, Y. Duan, B. Zhou, F. Fang, J. Xia, X. Ma // Journal of Intelligent & Robotic Systems. — 2020. — Окт. — Т. 100, № 1. — С. 145—163. — ISSN 1573-0409. — DOI: 10.1007/s10846-020-01202-3. — URL: https://doi.org/10.1007/s10846-020-01202-3.

19. ManiSkill: Generalizable Manipulation Skill Benchmark with Large-Scale Demonstrations / T. Mu, Z. Ling, F. Xiang, D. Yang, X. Li, S. Tao, Z. Huang, Z. Jia, H. Su. — 2021.

20. Meta-World: A Benchmark and Evaluation for Multi-Task and Meta Reinforcement Learning / T. Yu, D. Quillen, Z. He, R. Julian, A. Narayan, H. Shively, A. Bellathur, K. Hausman, C. Finn, S. Levine. — 2021.

21. The Surprising Effectiveness of Representation Learning for Visual Imitation / J. Pari, N. M. Shafiullah, S. P. Arunachalam, L. Pinto. — 2021.

22. *Bahl S.*, *Gupta A.*, *Pathak D.* Human-to-Robot Imitation in the Wild. — 2022.

23. DoorGym: A Scalable Door Opening Environment And Baseline Agent / Y. Urakami, A. Hodgkinson, C. Carlin, R. Leu, L. Rigazio, P. Abbeel. — 2022.

24. Learning Robot Manipulation Skills from Multiple Human Demonstrations / Y. Yang [и др.] // IEEE Transactions on Robotics. — 2022.

25. A Comprehensive Survey on Reinforcement Learning for Robotic Manipulation / Z. Xu [и др.] // IEEE Transactions on Neural Networks and Learning Systems. — 2023.

26. A Survey of Learning-Based Control in Robot Manipulation / A. Zeng [и др.] // Annual Review of Control, Robotics, and Autonomous Systems. — 2023.

27. Large Language Models as General Pattern Machines / S. Mirchandani, F. Xia, P. Florence, B. Ichter, D. Driess, M. G. Arenas, K. Rao, D. Sadigh, A. Zeng. — 2023.

28. ManiSkill2: A Unified Benchmark for Generalizable Manipulation Skills / J. Gu, F. Xiang, X. Li, Z. Ling, X. Liu, T. Mu, Y. Tang, S. Tao, X. Wei, Y. Yao, X. Yuan, P. Xie, Z. Huang, R. Chen, H. Su. — 2023.

29. RT-1: Robotics Transformer for Real-World Control at Scale / A. Brohan [и др.]. — 2023.

30. RT-2: Vision-Language-Action Models Transfer Web Knowledge to Robotic Control / A. Brohan [и др.]. — 2023.