МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
федеральное государственное бюджетное образовательное учреждение высшего образования
«Тольяттинский государственный университет»

Кафедра «Прикладная математика и информатика»
(наименование)

01.04.02 Прикладная математика и информатика
(код и наименование направления подготовки)

Математическое моделирование
(направленность (профиль))

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА (МАГИСТЕРСКАЯ ДИССЕРТАЦИЯ)

на тему: «Исследование алгоритмов минимизации в обратной задаче кинематики манипулятора»

Обучающийся В.В. Тарасова
(Инициалы Фамилия) (личная подпись)

Научный руководитель д.ф.-м.н., доцент, С.В. Талалов
(ученая степень (при наличии), ученое звание (при наличии), Инициалы Фамилия)

Тольятти 2024

Содержание

Введение 3
1. Литературный обзор 6
1.1. Методы решения прямой задачи кинематики манипулятора 6
1.2. Методы решения ОЗК МП 9
1.3. Методы многомерной минимизации 12
1.4. Математические пакеты инверсной кинематики 13
2. Метод решения ОЗК с помощью условной минимизации 15
2.1. Методы решения ОЗК с помощью модуля Robotics Toolbox for Python 15
2.2. Критерии условной минимизации 22
1.1 Градиенты функций минимизации 24
2.3. Градиент координат узлов 25
3. Разработка необходимых программ 26
3.1. Программа расчета координат и соответствующих матриц 26
3.2. Программа расчета градиента координат узлов 28
3.3. Программа расчета функций минимизации 29
3.4. Решение задач в среде MATLAB 29
3.5. Численные эксперименты 30
4. Численное решение ОЗК МП методами условной минимизации с использованием градиентов 39
4.1. Сокращение количества вычислений с использованием градиентов 39
4.2. Численные эксперименты с применением градиентов 40
4.3. Улучшение обусловленности ОЗК МП с помощью абсолютных углов 52
Заключение 66
Список используемой литературы и используемых источников 67

Введение

Манипулятор (МП) – это управляемое механическое устройство для выполнения работ в пространстве. Классическим примером МП является грузоподъемный механизм, выполняющий перемещение различного груза в трехмерном пространстве. Важнейшей особенностью МП является возможность выполнения работ в недоступных или опасных для человека местах, таких как: площадки на высоте, подводные глубины, вакуум, радиоактивная или другая агрессивная среда и т.д. Благодаря совершенствованию способов управления МП появилась возможность удаленно выполнять работы как в открытом космосе, так и в медицине со сложными хирургическими операциями.

МП представляет собой разомкнутую цепь, состоящую из последовательности вращающихся или поступательно перемещающихся звеньев с помощью силовых приводов. Если управление приводами осуществляется человеком-оператором, то манипуляции осуществляются аналогично действиям руки человека. Если же приводы управляются автоматически с помощью специальных программ, то такие МП представляют собой разновидность роботов.

Автоматическое управление МП невозможно без знания его кинематики, т.е. аналитического описания геометрии движения МП относительно некоторой заданной абсолютной системы. Существуют прямая и обратная задачи кинематики положения МП. В прямой задаче кинематики (ПЗК) при заданных значениях обобщенных координат и аналитических зависимостей требуется найти положение рабочего органа. В обратной задаче кинематики (ОЗК) при известном положении рабочего органа требуется найти соответствующие обобщенные координаты. Обобщенные координаты – это независимые параметры, однозначно определяющие положение механической системы в пространстве. Для МП с вращательными сочленениями независимыми параметрами являются относительные углы поворота.

Решение ОЗК МП определяется из условия совпадения координат рабочего органа и целевой точки в виде нелинейного уравнения:

$$g(\boldsymbol{c}) = 0, \tag{1}$$

где **c** – вектор обобщенных координат.

Для МП с кинематической избыточностью уравнение (1) обычно имеет множество решений. Из-за неоднозначности решений ОЗК возникает дополнительная задача выбора наилучшего решения с помощью минимизации дополнительной функции $f(\boldsymbol{c})$. В этом случае ОЗК рассматривается как задача условной минимизации вида:

$$\begin{cases} f(\boldsymbol{c}) \to min \\ g(\boldsymbol{c}) = 0 \end{cases} \tag{2}$$

Функция минимизации обычно представляет собой некоторую физическую величину, зависящую от характера работ, выполняемых МП. Для эффективного выполнения работ МП требуется эффективное управление движением его элементов по результатам решения ОЗК.

Целью данной работы является исследование существующих алгоритмов многомерной минимизации для совершенствования решения ОЗК МП.

Для решения поставленной цели требуется выполнение следующих задач:

1) Разработать алгоритм и программу численного решения ПЗК МП.

2) Разработать алгоритм и программу численного решения ОЗК МП с применением стандартных методов условной минимизации с помощью MATLAB или библиотеки SciPy Python.

3) Разработать алгоритм и программу вычисления частных производных координат узлов манипулятора для вычисления градиентов целевой и ограничивающих функций.

4) Проведение численных экспериментов решения ОЗК МП с применением стандартных и альтернативных методов минимизации с помощью градиентов целевой и ограничивающих функций.

Научная новизна

Разработаны алгоритм и программа решения ОЗК МП методами условной минимизации MATLAB и SciPy Python.

Предлагается для манипуляторов со сферическими соединениями большой размерности минимизацию проводить с помощью производных матрицы преобразования.

Для улучшения обусловленности ОЗК для манипуляторов со сферическими соединениями предлагается использовать абсолютные углы поворота.

Показано, что решение обратной задачи манипулятора с абсолютными углами может служить основой для решения манипуляторов со специальной схемой.

1. Литературный обзор

1.1. Методы решения прямой задачи кинематики манипулятора

Расчет координат звеньев манипулятора обычно выполняют с помощью матричных преобразований. При относительном вращении звеньев манипулятора их новые координаты можно получить с помощью известной матрицы поворота с углами Эйлера (см. рисунок 1) путем умножения на вектор с относительными координатами [14].

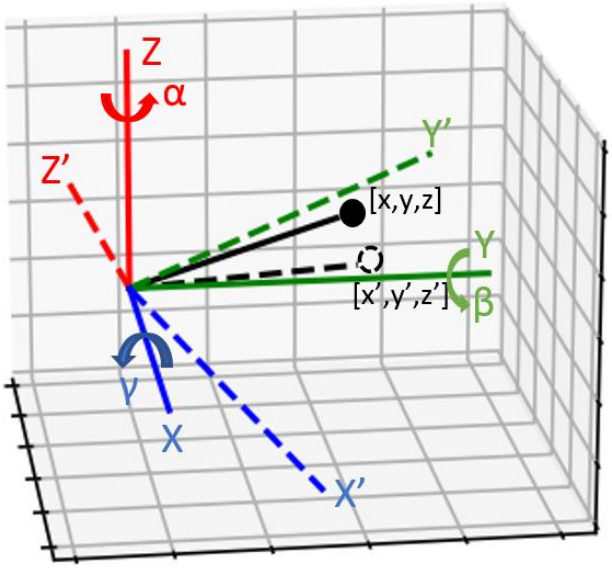

Рисунок 1 - Углы Эйлера для описания вращательного движения

$$\begin{pmatrix} x_i' \\ y_i' \\ z_i' \end{pmatrix} = \mathrm{R}(\propto, \beta, \gamma) \begin{pmatrix} x_i - x_{i-1} \\ y_i - y_{i-1} \\ z_i - z_{i-1} \end{pmatrix} + \begin{pmatrix} x_{i-1} \\ y_{i-1} \\ z_{i-1} \end{pmatrix}, \quad (3)$$

где

$$\mathrm{R}(\propto, \beta, \gamma) = \begin{pmatrix} c_\propto c_\beta & c_\propto s_\beta s_\gamma - s_\propto c_\gamma & c_\propto s_\beta c_\gamma + s_\alpha s_\gamma \\ s_\propto c_\beta & s_\propto s_\beta s_\gamma + c_\propto c_\gamma & s_\propto s_\beta c_\gamma - c_\alpha s_\gamma \\ -s_\beta & c_\beta s_\gamma & c_\beta s_\gamma \end{pmatrix}, \quad (4)$$

$c_\propto = \cos \propto;\ c_\beta = \cos \beta;\ c_\propto = \cos\gamma;\ s_\propto = \sin \propto;\ s_\beta = \sin \beta;\ s_\propto = \sin\gamma.$

Здесь надо отметить, что матрица поворота может иметь и другой вид, в зависимости от порядка вращения [1], [12].

В кинематике промышленных роботов [4], [23] применяют более компактное уравнение с матрицей однородного преобразования размерностью 4x4 и вектора однородных с четвертым единичным компонентом.

Координаты звеньев манипулятора рассчитываются с помощью произведения матриц преобразований A_i. Данные формулы подробно описаны в [14].

$$\begin{pmatrix} x_n \\ y_n \\ z_n \\ 1 \end{pmatrix} = T_n \begin{pmatrix} \Delta x_n \\ \Delta y_n \\ \Delta z_n \\ 1 \end{pmatrix}, \tag{5}$$

где

$$T_n = \left(\prod_{k=1}^{n} A_k \right) = A_1 A_2 \dots A_n$$

$$\mathrm{A}_i = \begin{pmatrix} \mathrm{R}(\propto_i, \beta_i, \gamma_i) & \begin{matrix} \Delta x_i \\ \Delta y_i \\ \Delta z_i \end{matrix} \\ \begin{matrix} 0 & 0 & 0 \end{matrix} & 1 \end{pmatrix}$$

Δx_i, Δy_i, Δz_i – относительные координаты узлов.

Тем не менее, многократное произведение матриц с тремя неизвестными затрудняет получение аналитического решения. В промышленности обычно применяются манипуляторы, работающие по схеме, предложенной еще в 1955 г., Денавитом и Хартенбергом [17], [34]. В данном методе используются перемещения с двумя поворотами и двумя параллельными переносами в соответствии со схемой на рисунке 2.

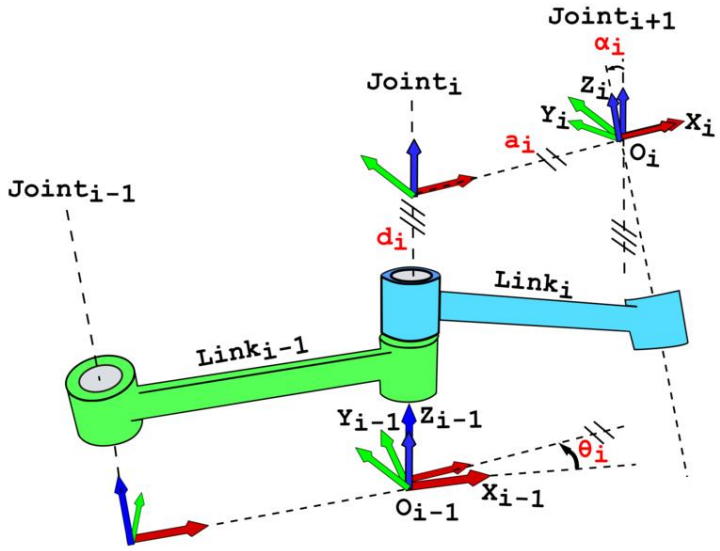

Рисунок 2 - Параметры МП Денавита — Хартенберга

Матрица преобразований систем координат Денавита — Хартенберга:

$$A_i = \begin{bmatrix} \cos\theta_i & -\cos\alpha_i \sin\theta_i & \sin\alpha_i \sin\theta_i & a_i \cos\theta_i \\ \sin\theta_i & \cos\alpha_i \cos\theta_i & -\sin\alpha \cos\theta & a_i \sin\theta_i \\ 0 & \sin\alpha_i & \cos\alpha_i & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad (6)$$

где $\propto_i, a_i, d_i$ −константы,

ϑ_i − присоединенная переменная, если рассматриваемое сочленение − вращательное;

$\vartheta_i, \propto_i, a_i$ − константы, d_i − присоединенная переменная, если рассматриваемое сочленение − поступательное.

Абсолютные координаты x_n, y_n, z_n n-го сочленения определяются из результирующей матрицы T_n, равной произведению матриц A_i:

$$T_n = (\prod_{k=1}^{n} A_k) = A_1 A_2 \dots A_n = \begin{pmatrix} R(\propto_n, \beta_n, \gamma_n) & \begin{matrix} x_n \\ y_n \\ z_n \end{matrix} \\ \begin{matrix} 0 & 0 & 0 \end{matrix} & 1 \end{pmatrix} \quad (7)$$

В статье [20] приводится решение расширенной прямой задачи кинематики специального шестизвенного манипулятора. В работе [5] представлена математическая модель и приводится решение прямой задачи кинематики для манипулятора с управляемым изгибом.

Классические подходы к прямой и обратной задачам кинематики, метод построения математической модели в среде MATLAB представлено в [2].

1.2. Методы решения ОЗК МП

Данная задача является некорректно поставленной [29], для решения которой минимизируется некоторая невязка [28]. Существуют аналитический и численный метод решения ОЗК. Преимуществом аналитического решения является то, что оно позволяет качественно оценивать результат и получать закономерности в поведении манипулятора. В работе [46] для решения ОЗК применяются алгебраические методы. В работе [42] приведено решение ОЗК МП PUMA геометрическим методом.

Аналитический метод имеет свои недостатки. Наиболее простым методом получения аналитического решения является метод обратных преобразований [25], [45] с последовательным умножением результирующей матрицы преобразования прямой задачи

$$T_n = A_1 A_{2\ldots} A_{n-1} A_n$$

на матрицу, обратную матрице преобразования от i-й системы координат в (i-1)-ю систему координат $A_i^{-1}(q_i)$ [14]:

$$A_n^{-1} A_{n-1}^{-1} \ldots A_2^{-1} A_1^{-1} T_n = E.$$

Но наличие выражений с обратными тригонометрическими функциями не всегда позволяет получать точные и однозначные величины в явном виде. Кроме того, аналитическое решение может быть получено только для конкретных случаев с определенной структурой и с ограниченной степенью

свободы [3]. В статье [18] рассмотрены аналитическое и численное решения ОЗК для робота-манипулятора DELTA.

Численные методы применяют, когда отсутствует возможность получения решения в аналитическом виде. Одной из первых работ с решением численным итерационным методом нелинейной ОЗК является работа А. Ф. Верещагина и JI. Н. Минаева [6]. В работе [19] с помощью MATLAB решена ОЗК манипуляционного робота шарнирного типа методом Ньютона-Рафсона. В статьях [9] и [27] для решения ОЗК предлагается применение искусственных нейронных сетей. В статье [33] предлагается также применение метода Монте Карло. В статье [15] предлагается решение ОЗК с иерархическим подходом.

В статье [32] предложен бикватернионный подход для решения ОЗК и кинематики движения твердого тела. В работе [7] предложено решение ОЗК для МП с кинематическими парами пятого класса.

Задачу поиска необходимых обобщенных координат для достижения заданной цели представляется естественным рассматривать как задачу минимизации расстояния между рабочим органом целевой точкой:

$$g(\boldsymbol{c}) \to 0 \text{ или } g(\boldsymbol{c}) = 0.$$

С другой стороны, факт о возможном существовании нескольких таких решений предполагает поиск наилучшего варианта с минимальной критериальной функцией:

$$f(\boldsymbol{c}) = min \text{ или } f(\boldsymbol{c}) \to min.$$

Таким образом, численное решение ОЗК МП сводится к численному решению задачи условной минимизации вида (2). Соответственно, решение ОЗК в этом случае может зависеть от выбранного алгоритма решения задачи условной минимизации.

Для непрерывно дифференцируемых функций предпочтительнее применение методов минимизации с помощью производных минимизируемых функций. В статье [10] для приближенного решения ОЗК многозвенного МП применен метод штрафных функций. Приведена схема решения задачи минимизации расширенного критерия, основанная на градиентном методе. Метод обобщенного приведенного градиента применен также в статье [28] для поиска оптимальных углов Эйлера для двигателей трехзвенного антропоморфного манипулятора. Данные углов приведены в матричный вид представлением Денавита-Хартенберга. Получение оптимальных значений по критерию минимизации энергопотребления достигается за счет решения задачи нелинейной оптимизации.

Как известно, главным недостатком градиентных методов заключается в том, что точность решения зависит от принятой начальной области поиска для функций с локальными минимумами. Представляется, что для оценки эффективности данного подхода для систем с кинематической избыточностью требуются дополнительные исследования.

Имеются публикации решения ОЗК с применением стохастических методов минимизации. Так, в статье [39] представлен метод, основанный на формулировке многовариантных моделей распределения Гаусса (MGDMs), которые точно определяют ограничения мягких соединений кинематического каркаса. В статье [30] для кинематического управления роботом-манипулятором «Intelbot» с пятой степенью подвижности использована способность к обучению адаптивной нейронечеткой системы вывода.

Как известно, применение стохастических методов, основанных на концепции вероятности не гарантирует получение глобального минимума многоэкстремальных функций.

В работе [37] представлена разработка роботизированного манипулятора с зрительной функцией с применением пропорциональной производно-псевдоинверсной якобианской (PD-PIJ) обратной кинематики и прямой кинематики Денавита-Хартенберга. Отмечено, что эксперименты по

моделированию и реализации проводились с использованием функции плотности вероятности и распределения вероятностей Вейбулла. Результат показывает, что манипулятор навигационной системы имел хорошую производительность. Реальное выполнение задачи сортировки цветов на манипуляторе показывает, что вероятность успешности составляет 94,46% для евклидовой погрешности расстояния менее 1,2 см.

Представляет интерес применения специальных приложений, таких как MATLAB Robotics Toolbox и библиотек SciPy и roboticstoolbox-python для исследования существующих методов многомерной минимизации в решении ОЗК МП. Так, в статье [11] с помощью программного пакета MATLAB Robotics Toolbox приведен алгоритм создания модели МП с кинематической схемой Денавита-Хартенберга и решения ПЗ ОЗК с помощью собственных функций программного пакета.

1.3. Методы многомерной минимизации

Обзор методов многомерной минимизации приведен в [13]. Несмотря на большое количество существующих методов, вопрос о выборе метода минимизации для решения ОЗК остается актуальным. Так как расчет кинематики МП основан на многократном перемножении матриц с нелинейными элементами, с произведениями тригонометрических функций, то соответственно, и минимизируемые функции являются нелинейными и многоэкстремальными.

В настоящее время существует множество специальных алгоритмов и пакетов прикладных программ, таких как MATLAB, LINGO, MINITAB, GAMS [53], для поиска глобального минимума многоэкстремальных многомерных функций. Так, для поиска глобального минимума с автоматической генерацией множества пробных точек в приложении MATLAB существует объект GlobalSearch.

Вероятностная генерация пробных начальных приближений способствовала появлению различных эвристических методов с имитацией природных или физических процессов явлений: отжиг [38], рой пчел [40],

поведение муравьев [35], генетический отбор [26] и т.д. Приводятся алгоритмы с метаэвристическими [36], гибридными и метаоптимизационными методами [16].

Поиск глобального минимума непрерывно дифференцируемых функций предпочтительнее выполнять с помощью их градиентов. Как известно, градиентный метод минимизации можно рассматривать как задачу Коши с системой дифференциальных уравнений. Минимизация с системой дифференциальных уравнений предложена в [31]. Метод основан на концепции диффузионного сглаживания поверхности целевой функции [48] и сведения многомерной минимизации на отрезке [52]. Минимизация выполнялась в MATLAB.

Однако, для решения ОЗК МП применение глобального безусловного метода минимизации недостаточно. Неоднозначность решений обратной задачи синтеза с множеством вариантов предполагает наличие множества локальных минимумов с одинаковыми значениями минимизируемой функции в виде квадрата расстояния до целевой точки. Выбор необходимого решения должен выполняться по некоторому критерию.

Для решения ОЗК МП требуется решение задачи условной минимизации. Наиболее распространенный подход решения задач условной минимизации основан на сведении данной задачи к задаче безусловной минимизации с помощью штрафных слагаемых функций. Примером является популярный метод внутренней точки [8].

В MATLAB существует пакет функций Optimization Toolbox [44]. Список функций для решения оптимизационных задач с нелинейными целевыми и ограничивающими функциями с помощью ScyPi Python приведен [43].

1.4. Математические пакеты инверсной кинематики

Пакет расширения SimMechanics в Matlab позволяет исследовать кинематику манипулятора, определять кинематические и динамические параметры движения выходного звена [21], [22].

Алгоритм проектирования инверсной кинематики с помощью MATLAB и Simulink изложен в [49].

Свободно распространяемый модуль для проектирования манипуляторов в python со схемой Денавита-Хартенберга можно загрузить на сайте https://pypi.org/project/roboticstoolbox-python/ [47], [50].

Согласно документации с помощью наборов инструментов можно легко создать свой собственный в манипулятор по схеме Денавита-Хартенберга, импортировать файл URDF или использовать более 30 поставляемых моделей для известных современных роботов от Franka-Emika, Kinova, Universal Robotics, Rethink, а также классические роботы, такие как Puma 560 и Stanford arm.

Панель инструментов содержит быстрые реализации кинематических операций. Прямая кинематика и якобиан манипулятора могут быть вычислены менее чем за 1 микросекунду, в то время как численная обратная кинематика может быть решена всего за 4 микросекунды.

Таким образом, применение данных инструментов может быть полезным для решения ОЗК МП методами минимизации с помощью производных.

2. Метод решения ОЗК с помощью условной минимизации

2.1. Методы решения ОЗК с помощью модуля Robotics Toolbox for Python

Данный модуль является преемником набора инструментов Robotics Toolbox для MATLAB. Как уже отмечалось выше, с помощью данного модуля можно легко сформировать МП со по схемой Денавита-Хартенберга и решать для него прямую и обратную задачи. Для решения ОЗК имеется несколько методов, каждый из которых имеет свое название [41]. Для численного решения используются алгоритмы глобальной минимизации:

ikine_LM – алгоритм Левенберга-Марквадта с возможностью глобального поиска;

ikine_LMS – алгоритм Левенберга-Марквадта с поправкой Сугихары;

ikine_min – библиотека SciPy.minimize, функцию стоимости пользователя, жесткость.

Все методы принимают необязательные аргументы, специфичные для метода, и возвращают именованный кортеж.

Для исследования взята модель МП с 7-ю степенями свободы с названием Panda.

```
import roboticstoolbox as rtb
panda = rtb.models.DH.Panda()
print(panda)
DHRobot: Panda (by Franka Emika), 7 joints (RRRRRRR),
dynamics, geometry, modified DH parameters
```

a_{j-1}	α_{j-1}	θ_j	d_j	q^-	q^+
0.0	0.0°	q1	0.333	-166.0°	166.0°
0.0	-90.0°	q2	0.0	-101.0°	101.0°
0.0	90.0°	q3	0.316	-166.0°	166.0°
0.0825	90.0°	q4	0.0	-176.0°	-4.0°
-0.0825	-90.0°	q5	0.384	-166.0°	166.0°
0.0	90.0°	q6	0.0	-1.0°	215.0°
0.088	90.0°	q7	0.107	-166.0°	166.0°

```
┌──────┬──────────────────────────────────────┐
│ tool │  t = 0, 0, 0.1; rpy/xyz = -45°, 0°, 0°  │
└──────┴──────────────────────────────────────┘

┌──────┬──────┬──────────┬──────┬──────────┬──────┬──────────┬──────┐
│ name │ q0   │ q1       │ q2   │ q3       │ q4   │ q5       │ q6   │
├──────┼──────┼──────────┼──────┼──────────┼──────┼──────────┼──────┤
│   qr │   0° │ -17.2°   │   0° │ -126°    │   0° │   115°   │   45° │
│   qz │   0° │   0°     │   0° │   0°     │   0° │   0°     │   0° │
└──────┴──────┴──────────┴──────┴──────────┴──────┴──────────┴──────┘
```

Начальная конфигурация с вариантом qr изображена на рисунке 3 с помощью команды:

```
puma.plot(puma.qr)
```

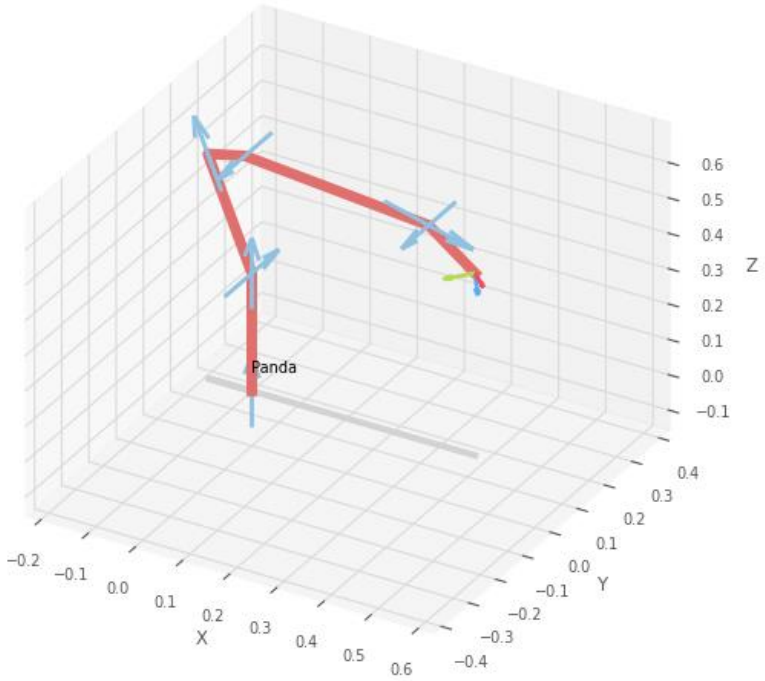

Рисунок 3 - Кинематическая схема МП Panda

Целью ОЗК является достижение рабочего органа заданной точки с координатами (0.5, -0.25, 0.5). На рисунках 4 и 5 представлены решения ОЗК, выполненные методом ikine_LMS и ikine_min соответственно:

```
from spatialmath import SE3
Tep = SE3(0.5, -0.25, 0.5) #целевая точка

sol = puma.ikine_LMS(Tep) # решение ОЗК
puma.plot(sol.q)
print(sol)

IKsolution(q=array([-0.2324326   ,      0.92029401,   -
0.39199561, -1.38324611,  0.39360074,
       -0.91523015,     -1.63051658]),     success=True,
reason=None, iterations=11, residual=3.395351038626567e-
11)
```

```
sol = puma.ikine_min(Tep) # решение ОЗК
puma.plot(sol.q)
print(sol)

IKsolution(q=array([-5.46147802e+00, -1.61708527e+05, -
7.86684066e+00, -1.61699848e+05,
       -7.64675960e+00, -1.61702359e+05,
2.02265141e+00]), success=True, reason='Optimization
terminated successfully', iterations=49,
residual=2.927297203181904e-15)
```

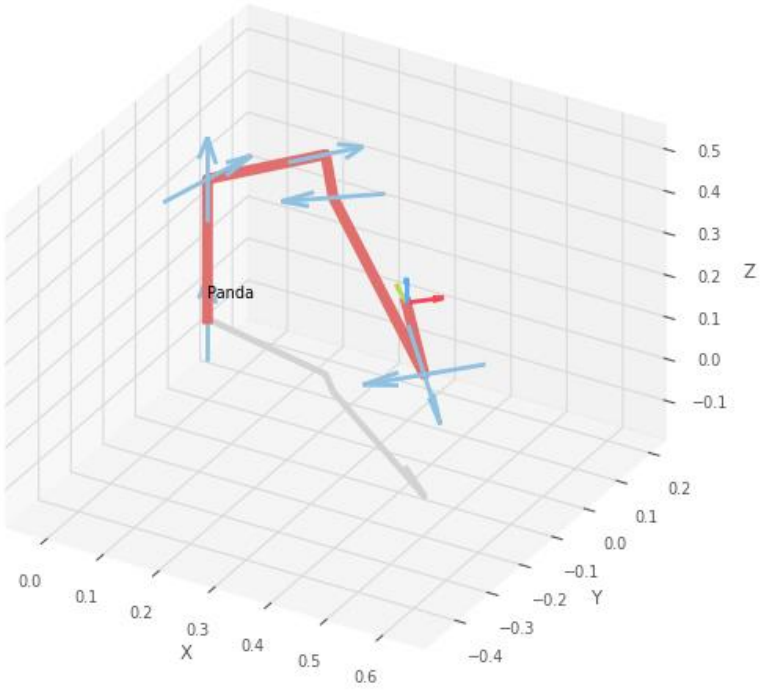

Рисунок 4 - Решение ОЗК МП Panda методом ikine_LMS

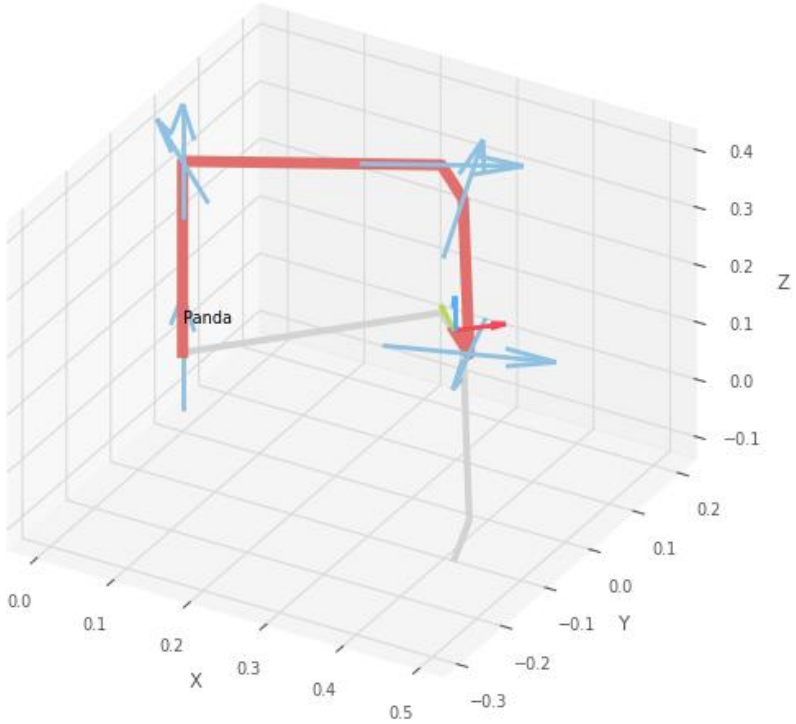

Рисунок 5 - Решение ОЗК МП Panda методом ikine_min

Как видно из рисунков, получаются разные результаты при решении разными методами.

Доступ к данным результата осуществляется с помощью кортежа T:

T = puma.fkine_all(q) ,

где q – вектор относительных углов поворота.

Значения векторов с координатами узлов

Начальное положение

0 [0. 0. 0. 1.]

1 [0. 0. 0.333 1.]

2 [0. 0. 0.333 1.]

3 [-9.33843853e-02 8.64213005e-19 6.34886331e-01 1.00000000e+00]

4 [-1.45691250e-02 -6.28656980e-19 6.59266748e-01 1.00000000e+00]

5 [3.75481498e-01 -4.08272911e-17 6.13193311e-01 1.00000000e+00]

6 [3.75481498e-01 -4.08272911e-17 6.13193311e-01 1.00000000e+00]

7 [4.73724040e-01 -5.63815267e-17 5.15513206e-01 1.00000000e+00]

ОЗК методом ikine_LMS

0 [0. 0. 0. 1.]

1 [0. 0. 0.333 1.]

2 [0. 0. 0.333 1.]

3 [0.24470419 -0.05792412 0.52436524 1.   ]

4 [0.28237388 -0.09922967 0.46369321 1.   ]

5 [0.56582088 -0.30839681 0.29000051 1.   ]

6 [0.56582088 -0.30839681 0.29000051 1.   ]

7 [0.49999261 -0.24999633 0.39700062 1.   ]

ОЗК методом ikine_min

0 [0. 0. 0. 1.]

1 [0. 0. 0.333 1.]

2 [0. 0. 0.333 1.]

3 [0.20890846 0.22465779 0.25722312 1.]

4 [0.26949223 0.16866854 0.25825301 1.]

5 [0.47118946 -0.16684987 0.29000005 1.]

6 [0.47118946 -0.16684987 0.29000005 1.]

7 [0.50000003 -0.25000004 0.39700005 1.]

Для наглядности полученные результаты решения ОЗК представлены на одном рисунке 6. Текст программы приведен ниже.

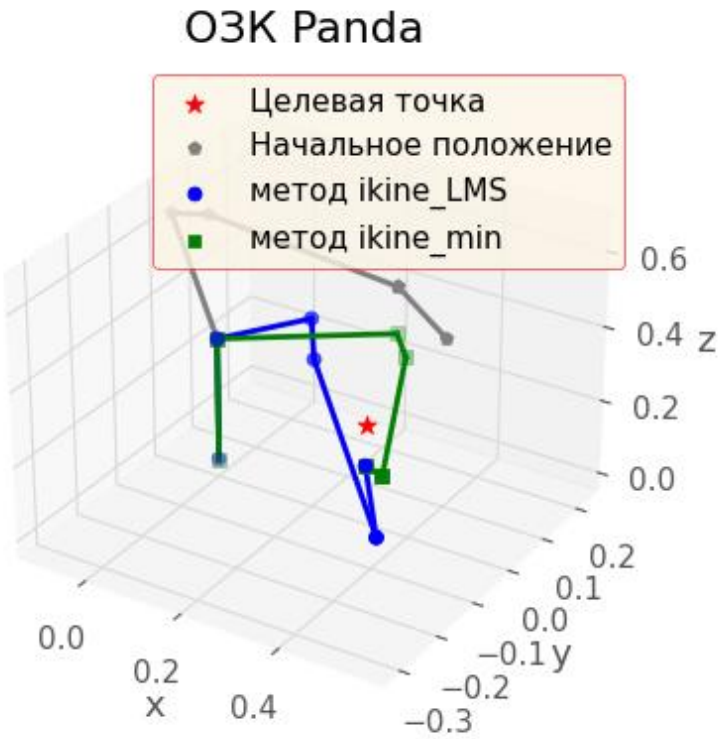

Рисунок 6 - Варианты расположения звеньев МП Panda

Программа решения ОЗК МП Panda

```
import roboticstoolbox as rtb
import numpy as np
from matplotlib import pyplot as plt
puma = rtb.models.DH.Panda()
print(puma)
q0=puma.qr # начальная конфигурация
#puma.plot(q0)
T0 = puma.fkine_all(q0)
from spatialmath import SE3
Tep = SE3(0.5, -0.25, 0.5)# целевая точка
sol = puma.ikine_LMS(Tep) # решение ОЗК
Tsol = puma.fkine_all(sol.q) # кортеж с результатами
#puma.plot(sol.q)
print(sol)

n=len(Tsol)
p = np.full((4, n), 0.0)

sol2 = puma.ikine_min(Tep) # решение ОЗК
Tsol2 = puma.fkine_all(sol2.q)
#puma.plot(sol2.q)
print(sol2)

global ax
def pltInit():
#Подготовка графической фигуры
 global ax
 plt.rcParams["figure.figsize"] = [5, 4]
 plt.rcParams["figure.facecolor"] = "white"
 plt.rcParams["figure.autolayout"] = True
 plt.rc('font', size= 11)
 fig = plt.figure()
 ax = fig.add_subplot(projection="3d",facecolor = "white")
 ax.set_xlabel('x')
 ax.set_ylabel('y')
 ax.set_zlabel('z')
 ax.set_title('ОЗК Panda')

def initVectors(T):
 n=len(T)
```

```
 for i in range(n):
   A=np.array(T[i])
   p[:,i]=A[:,3]

def drawpoints(colorp,m,labelp):
 ax.scatter(p[0,:], p[1,:], p[2,:], c=colorp, s=25, marker=m, label = labelp)
 for i in range(n-1):
   ax.plot3D([p[0,i], p[0,i+1]], [p[1,i], p[1,i+1]], [p[2,i], p[2,i+1]], colorp,
linewidth=2)


pltInit()
A=np.array(Tep)
pn=A[:,3]
ax.scatter(pn[0],pn[1],pn[2], c='red',s=50, marker='*',
          label = 'Целевая точка')
initVectors(T0)
drawpoints('gray','p','Начальное положение')
initVectors(Tsol)
drawpoints('blue','o','метод ikine_LMS')
initVectors(Tsol2)
drawpoints('green','s','метод ikine_min')
print("residual = ", np.linalg.norm(Tep - puma.fkine(sol.q)))


ax.legend(
   ncol = 1,
   facecolor = 'oldlace',
   edgecolor = 'r',
       )
plt.show()
```

Таким образом, для получения необходимого решения ОЗК МП с кинематической избыточностью может потребоваться выполнение поиска наилучшего варианта с дополнительными ограничениями.

2.2. Критерии условной минимизации

Предлагается ОЗК рассматривать как задачу условной минимизации в виде уравнения (2). На рисунке 7 в качестве примера схематично изображены два варианта критериев для оценки эффективности работы МП: перемещение

узлов от начального положения в конечное и отклонение центра тяжести узлов МП от вертикального положения равновесия.

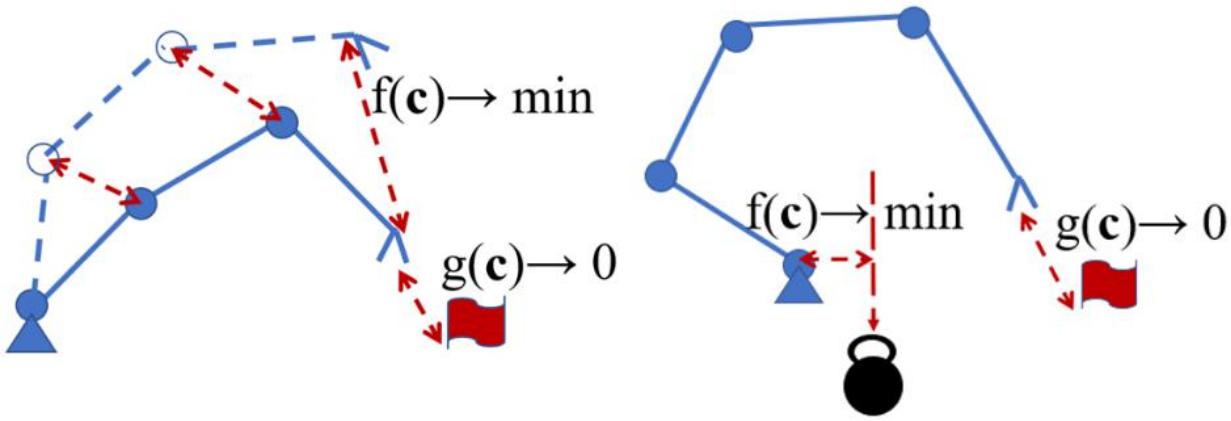

Рисунок 7 - Варианты критериев минимизации

Соответственно, для первого случая целевая функция запишется:

$$f_1(\boldsymbol{c}) = \sum_{i=2}^{n+1}[(x_i - x_i')^2 + (y_i - y_i')^2 + (z_i - z_i')^2], \quad (8)$$

где x_i, y_i, z_i и x_i', y_i', z_i' – конечные и начальные координаты узлов.

Для второго случая:

$$f_2(\boldsymbol{c}) = \frac{1}{n}\sum_{i=2}^{n}({x_i}^2 + {y_i}^2), \quad (9)$$

Ограничивающая функция:

$$g(\boldsymbol{c}) = (x_n - x_c)^2 + (y_n - y_c)^2 + (z_n - z_c)^2, \quad (10)$$

где x_c, y_c, z_c – координаты целевой точки.

Так как в нашем случае функции минимизации вычисляются по конкретным аналитическим формулам, то представляется целесообразным

выполнение минимизации с помощью их градиентов. Согласно выражениям (8) и (9), целевые функции $f_1(\boldsymbol{c})$и $f_2(\boldsymbol{c})$, в отличие от g($\boldsymbol{c}$), зависят от координат каждого узла. Соответственно, для вычисления их градиентов требуется вычисление частных производных координат каждого узла.

1.1 Градиенты функций минимизации

Градиент целевой функции $f_1(\boldsymbol{c})$

$$J_1 = \left[\frac{\partial f_1}{\partial c_1}, \frac{\partial f_1}{\partial c_2}, \dots, \frac{\partial f_1}{\partial c_{n_c}}\right],$$

где $n_c = 3n -$ количество переменных для МП с шаровыми соединениями.

$$\frac{\partial f_1}{\partial c_m} = 2\sum_{i=2}^{n}(\mathbf{r}_i - \mathbf{r}_i{}')\frac{\partial \mathbf{r}_i}{\partial c_m}, \qquad m = 1,2, \dots n_c,$$

где $\mathbf{r}_i$, $\mathbf{r}_i{}'$ – векторы конечных и начальных координат узлов.

Градиент функции $f_2(\mathbf{c})$

$$J_2 = \left[\frac{\partial f_2}{\partial c_1}, \frac{\partial f_2}{\partial c_2}, \dots, \frac{\partial f_2}{\partial c_{n_c}}\right],$$

где

$$\frac{\partial f_2}{\partial c_m} = \frac{2}{n}\left(\sum_{i=2}^{n}\boldsymbol{r}_i\right)\sum_{i=2}^{n}\frac{\partial \boldsymbol{r}_i}{\partial c_m}, \qquad m = 1,2, \dots n_c$$

Градиент ограничивающей функции $g(\mathbf{c})$

$$J_g = \left[\frac{\partial g}{\partial c_1}, \frac{\partial g}{\partial c_2}, \dots, \frac{\partial g}{\partial c_{n_c}}\right],$$

где

$$\frac{\partial g}{\partial c_m} = 2(\boldsymbol{r}_n - \boldsymbol{r}_c)\frac{\partial \boldsymbol{r}_n}{\partial c_m}.$$

2.3. Градиент координат узлов

Компоненты градиента координат i-го узла можно получить дифференцированием обеих частей уравнения (5):

$$\begin{pmatrix} \frac{\partial \boldsymbol{r}_i}{\partial c_m} \\ 1 \end{pmatrix} = \left(\prod_{j=1}^{k-1} \mathrm{T}_j \right) \frac{\partial \mathrm{T}_k}{\partial c_m} \left(\prod_{j=k+1}^{i} \mathrm{T}_j \right) \begin{pmatrix} \Delta \boldsymbol{r}_i \\ 1 \end{pmatrix},$$

где $k = div(m-1,3)+1$, −целая часть от деления на 3 и увеличенное на 1,

$$\frac{\partial T_k}{\partial c_i} = \begin{cases} \partial T_k / \partial \propto_k \text{ при } j = 1 \\ \partial T_k / \partial \beta_k \text{ при } j = 2 \\ \partial T / \partial \gamma_k \text{ при } j = 3 \end{cases},$$

$$j = i - 3(k-1),$$

$$\frac{\partial T}{\partial \propto_k} = \begin{pmatrix} -s_\propto c_\beta & -c_\propto c_\gamma - s_\propto s_\beta s_\gamma & c_\propto s_\gamma - s_\alpha s_\beta c_\gamma & 0 \\ c_\propto c_\beta & c_\propto s_\beta s_\gamma - s_\propto c_\gamma & c_\propto s_\beta c_\gamma + s_\alpha s_\gamma & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix},$$

$$\frac{\partial T_k}{\partial \beta_k} = \begin{pmatrix} c_\propto s_\beta & c_\propto c_\beta s_\gamma & c_\propto s_\beta c_\gamma & 0 \\ -s_\propto s_\beta & s_\propto c_\beta s_\gamma & s_\propto c_\beta c_\gamma & 0 \\ -c_\beta & -s_\beta s_\gamma & -s_\beta s_\gamma & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix},$$

$$\frac{\partial T_k}{\partial \gamma_k} = \begin{pmatrix} 0 & c_\propto s_\beta s_\gamma + s_\propto s_\gamma & s_\propto c_\gamma - c_\gamma s_\beta s_\gamma & 0 \\ 0 & s_\propto s_\beta s_\gamma - c_\propto s_\gamma & -c_\propto c_\propto - s_\propto s_\beta s_\gamma & 0 \\ 0 & c_\beta c_\gamma & -c_\beta c_\beta & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}.$$

Так как координаты их градиентов узлов присутствуют во всех функциях минимизации, то их вычисление целесообразно выполнять с помощью отдельных программных функций.

3. Разработка необходимых программ

3.1. Программа расчета координат и соответствующих матриц

Функция [f]=CalcR(c) вычисляет координаты узлов МП и вспомогательные матрицы для расчета градиента функций минимизации.

Возвращает квадрат расстояния рабочего органа до целевой точки.

Входные переменные:

c – вектор относительных углов поворота звеньев МП

```
function [f]=CalcR(c)
global dr;
global r;
global rn;
%промежуточные матрицы
global T;
global DT;

n=length(dr(1,:));
al=zeros(3,n);
for i=0:n-1
   al(1,i+1)=c(i*3+1);
   al(2,i+1)=c(i*3+2);
   al(3,i+1)=c(i*3+3);
end;
p=zeros(4,1);
dp=zeros(4,1);
r=zeros(3,n+1);
A=zeros(3,3);
T=zeros(4,4,n);
DT=zeros(4,4,3,n);
P=zeros(4,4);
for k=1:4
   P(k,k)=1;
end;

r(:,1)=0.;
dp(4)=1;
for i=1:n
   A = eul2rotm([al(1,i) al(2,i) al(3,i)]);
   for k=1:3
      for m=1:3
```

```
            T(k,m,i)=A(k,m);
        end;
        dp(k)=dr(k,i);
    end;
    T(4,4,i)=1;
    if i>1
        for k=1:3
            T(k,4,i)=dr(k,i-1);
        end;
    end;
    DT(:,:,:,i)=DEuM(al(1,i), al(2,i), al(3,i));
    P(:,:)=P(:,:)*T(:,:,i);
    p=P(:,:)*dp;
    for k=1:3
        r(k,i+1)=p(k);
    end;
end;
f=(r(1,n+1)-rn(1))^2+ (r(2,n+1)-rn(2))^2+ (r(3,n+1)-rn(3))^2;
end
%---------- функция вычисления производных матриц преобразования
function [D]=dA(a,b,g)
sa=sin(a); ca=cos(a);
sb=sin(b); cb=cos(b);
sg=sin(g); cg=cos(g);
D=zeros(4,4,3);
    D(:,:,1)=[ -sa*cb  -ca*cg-sa*sb*sg  ca*sg-sa*sb*cg 0;
      ca*cb    ca*sb*sg-sa*cg   ca*sb*cg+sa*sg 0;
      0  0  0  0;
      0  0  0  0];

  D(:,:,2)=[ -ca*sb  ca*cb*sg  ca*cb*cg 0;
    -sa*sb  sa*cb*sg  sa*cb*cg 0;
    -cb         -sb*sg       -sb*cg 0;
      0  0  0  0];

    D(:,:,3)=[0  ca*sb*cg+sa*sg   sa*cg-ca*sb*sg 0;
    0  sa*sb*cg-ca*sg  -ca*cg-sa*sb*sg 0;
    0   cb*cg  -cb*sg 0;
      0  0  0  0];
 end
```

3.2. Программа расчета градиента координат узлов

Функция [d]=Drc(nr) вычисляет и возвращает градиент координат узлов.

Входные переменные:

nr –номер узла МП

```
function d=Drc(nr)
global dr;
global DT;
global T;
nc=length(T(1,1,:))*3;
d=zeros(3,nc);
n=nr-1;
p=zeros(4,nc);
dp=zeros(4,1);
for k=1:3
   dp(k)=dr(k,n);
end;
dp(4)=1;

for i=1:n
   P1=zeros(4,4);
   P2=zeros(4,4);
   for k=1:4
      P1(k,k)=1;
      P2(k,k)=1;
   end;
   for k=1:(i-1)
      P1(:,:)=P1(:,:)*T(:,:,k);
   end;
   for k=(i+1):n
      P2(:,:)=P2(:,:)*T(:,:,k);
   end;
   for j=1:3
      ic=(i-1)*3+j;
      p(:,ic)=P1(:,:)*DT(:,:,j,i)*P2(:,:)*dp(:);
      for k=1:3
         d(k,ic)=p(k,ic);
      end;
   end;
end;
end
```

3.3. Программа расчета функций минимизации

Функция [f]=CalcW(c) возвращает сумму квадратов перемещений узлов.

```
function [f]=CalcW(c)
%функция для расчета критериальной функции перемещения узлов
global r;
global r0;
n=length(r(1,:));
CalcR(c);
f=0;
for i=2:n
f=f+norm( [r(1,i)-r0(1,i) r(2,i)-r0(2,i) r(3,i)-r0(3,i)] );
end;
end
```

Функция [f]=CalcEq(c) – возвращает квадрат расстояния центра тяжести от начала координат.

```
function [f]=CalcEq(c)
%функция для расчета критериальной функции отклонения центра тяжести
global r;
n=length(r(1,:));
CalcR(c);
xc=sum(r(1,:))/n;
yc=sum(r(2,:))/n;
f=(xc*xc+yc*yc);
end
```

Функция [clt, f] = CalcCon(c) – возвращает условия минимизации:

f – разность квадрата расстояния рабочего органа до целевой точки;

clt – пустое множество.

```
function [clt, f] = CalcCon(c)
%функция ограничения
f=CalcR(c);
clt=[];
end
```

3.4. Решение задач в среде MATLAB

Разработана программа расчета прямой задачи

```
function [f]=CalcR(c)
```

c – вектор углов поворота в шарнирах

Применяются методы без использования производных:

метод безусловной минимизации

```
c = fminsearch(@CalcR, c0)
```

и метод условной минимизации

```
[c, fval] = fmincon(@CalcEq,c0,[],[],[],[],[],[], @CalcCon)
```

Решение задачи условной минимизации методом внутренней точки с помощью команды в MATLAB

```
options = optimset('Algorithm','interior-point');
[c, fval] = fmincon(@CalcEq, c0, [], [], [], [], [], [], @CalcCon,options);
```

3.5. Численные эксперименты

Вычисления выполнялись для МП с шаровыми шарнирами, обладающих кинематической избыточностью. Исходное положение МП расположено в горизонтальной нулевой плоскости. Предварительно ОЗК решалась с минимизацией функции g($\boldsymbol{c}$) и методом условной минимизации вида (2).

Множество возможных решений было рассчитано методом многомерной минимизации с функцией для MATLAB, приведенной в [41]. Разные решения также могут быть получены при бычной безусловной минимизации любым методом с разными начальными приближениями. Для условной минимизации в MATLAB использовался метод внутренней точки.

Множество минимумов с малыми значениями целевой функции $f=g(\boldsymbol{c})$ при решении ОЗК без ограничений для МП количеством звеньев 6 представлено на рисунке 8.

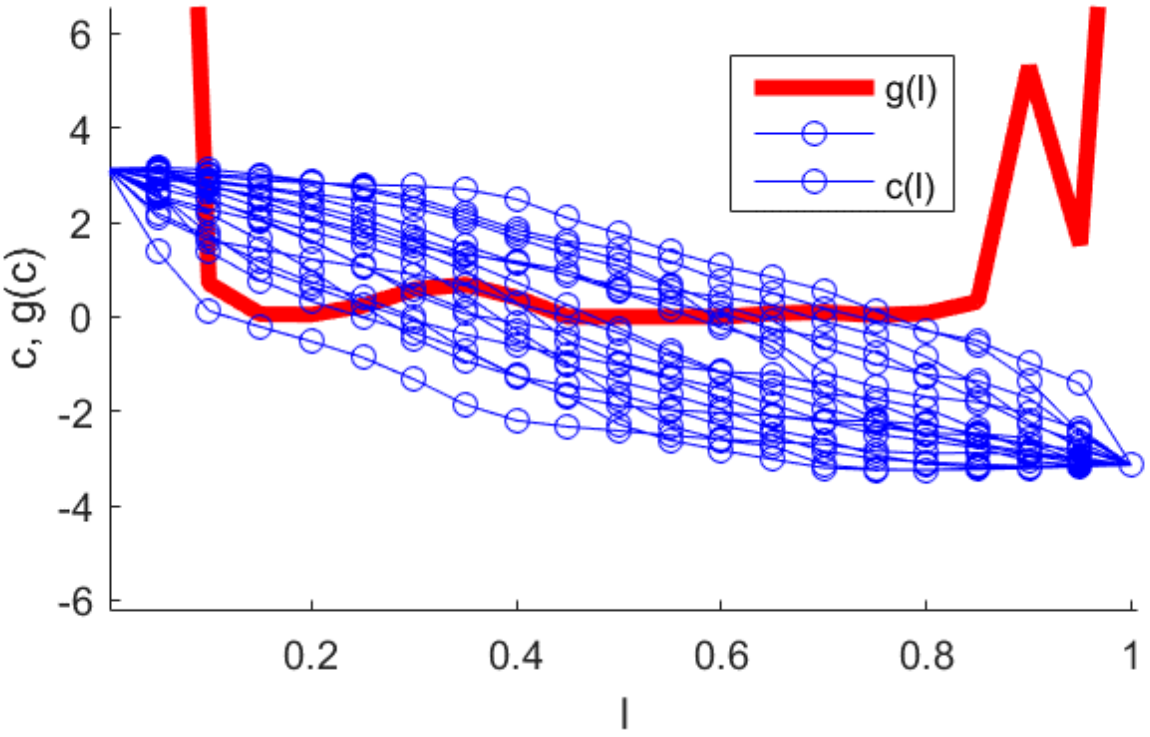

Рисунок 8 - Множество минимумов с малыми значениями функции $f=g(\boldsymbol{c})$

На рисунках 9 и 10 представлены разные решения ОЗК, которые были получены при безусловной минимизации и совпадающие решения, которые были получены при условной минимизации $f_1(\boldsymbol{c})$ и $f_2(\boldsymbol{c})$ с разными начальными значениями переменных.

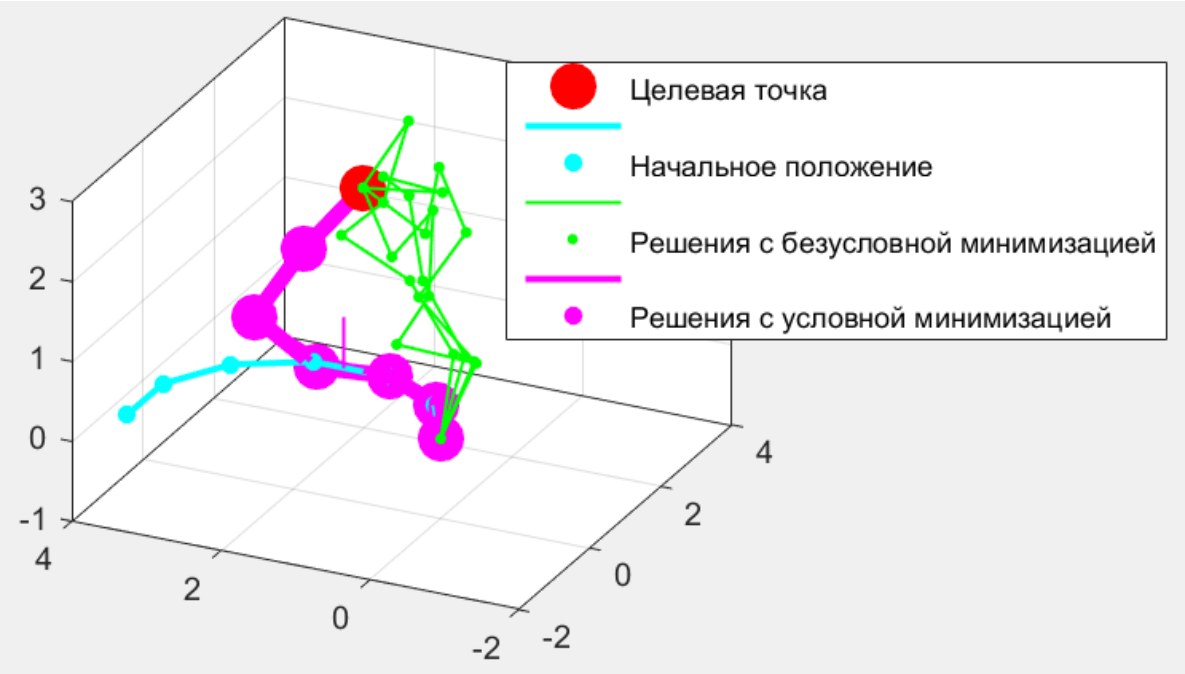

Рисунок 9 - Разные решения ОЗК с безусловной минимизацией функции g($\boldsymbol{c}$) и совпадающие решения при условной минимизации $f_1(\boldsymbol{c})$ с разными начальными значениями переменных

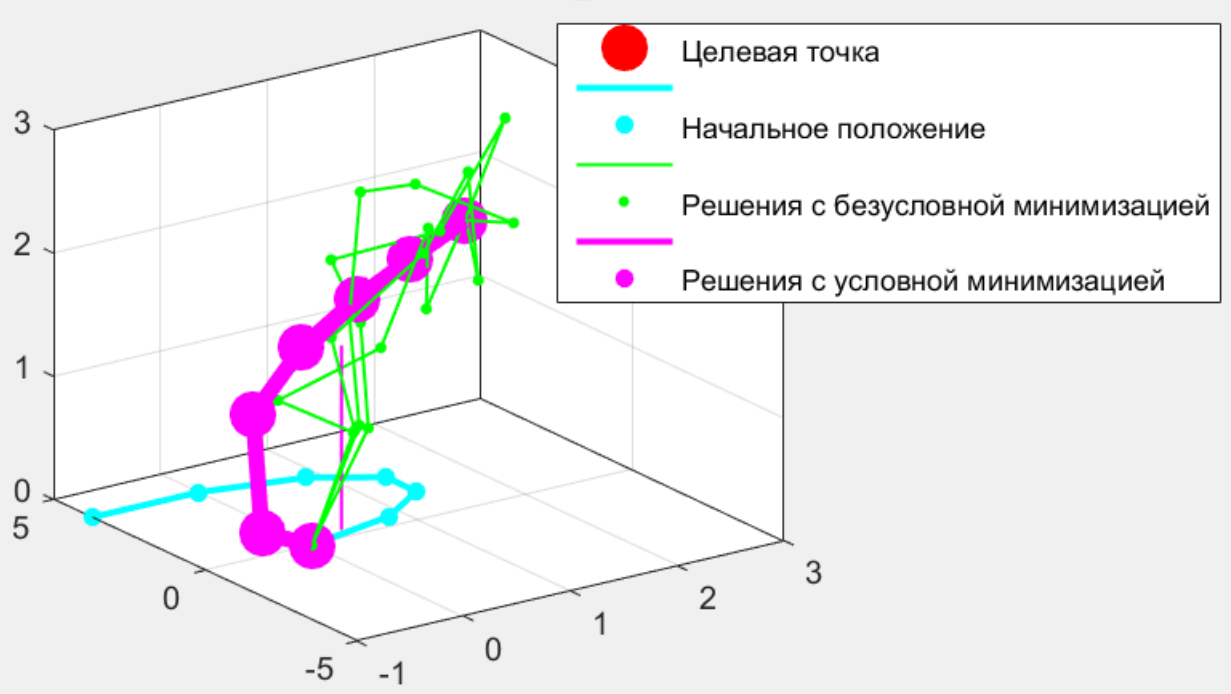

Рисунок 10 - Разные решения ОЗК с безусловной минимизацией функции $g(\boldsymbol{c})$ и совпадающие решения при условной минимизации $f_2(\boldsymbol{c})$ с разными начальными значениями переменных

Здесь представляется интересным отметить тот факт, что на рисунке 9 для минимизации перемещений узлов первые 4 узла остаются неподвижными, а на рисунке 10 для минимального отклонения центра тяжести узлы располагаются в одной вертикальной плоскости.

Ниже приведены численные результаты, полученные при решении ОЗК МП.

Численные результаты при минимизации $f_1(\boldsymbol{c})$

```
Начальное положение
ct = (углы поворота)
  Columns 1 through 12
    0.5236         0         0    0.5236         0         0    0.5236         0         0    0.5236         0
0
  Columns 13 through 18
    0.5236         0         0    0.5236         0         0
r =
         0    0.8660    1.3660    1.3660    0.8660    0.0000   -1.0000
         0    0.5000    1.3660    2.3660    3.2321    3.7321    3.7321
         0         0         0         0         0         0         0
dr =
    0.8660    0.5000    0.0000   -0.5000   -0.8660   -1.0000
    0.5000    0.8660    1.0000    0.8660    0.5000    0.0000
         0         0         0         0         0         0
fl = (значения целевой функции на отрезке)
  Columns 1 through 12
   16.0000   28.1703    0.7067    0.0575    0.0487    0.2330    0.5803    0.6832    0.4110
0.0000    0.0008    0.0000
  Columns 13 through 21
    0.0003    0.0747    0.0988    0.0354    0.0752    0.3213    5.3098    1.5283   16.0000


Варианты решения ОЗК с безусловной минимизацией

nump =
l =
    10
ans =
   2.2851e-06
ct =
  Columns 1 through 12
   -1.6586   -0.8168    0.8786   -2.3272   -0.5079    2.0909   -0.0015   -1.0142    1.5066
-0.3423   -1.6799    0.9746
  Columns 13 through 18
```

```
   1.6025   -1.4597    1.3267    0.8900   -0.9755    0.2491
r =
         0    0.2905   -0.1744    0.7995    1.2942    1.4548    1.9990
         0   -0.3391    0.5080    0.5408    0.7235    1.3517    1.9991
         0    0.8948    1.1524    1.3773    2.2269    1.4656    1.9993
Solver stopped prematurely.
fmincon stopped because it exceeded the function evaluation limit,
options.MaxFunEvals = 3000 (the default value).
```

Решение ОЗК условной минимизацией

```
fval =

   21.6642
g =
  -9.3623e-09
ct =
  Columns 1 through 12
   -2.1724   -2.8911    0.4588   -3.8043   -1.9464    2.3884   -1.0329    0.2701   -0.3129
1.0942   -3.1956    1.2675
  Columns 13 through 18
    0.3827   -2.4443   -1.1788    2.8357   -3.0089    0.2491
r =
         0    0.8755    1.3838    1.4104    1.0526    1.4236    1.9932
         0    0.4832    1.3444    2.3402    3.0113    2.5246    2.0047
         0    0.0001    0.0049   -0.0825    0.5669    1.3578    1.9944
fval =
   21.6642
```

…….

Решение ОЗК с безусловной минимизацией

```
nump =
     4
l =
    13
ans =
   2.8090e-04
ct =
```

```
  Columns 1 through 12
   -2.0334   -1.7150    0.4244   -2.5959   -1.5199    1.0826   -1.2005   -1.9643    0.2346
-1.1650   -2.8290    0.0669
  Columns 13 through 18
    0.7624   -2.3030   -0.1208   -0.2351   -2.6137   -1.1718
r =
         0    0.5542    0.6624    0.6453    1.6407    2.2354    2.0107
         0    0.0903    0.7304    1.6434    1.5563    1.5010    2.0100
         0    0.8275    1.5881    1.9958    2.0370    2.8391    2.0081
Solver stopped prematurely.
fmincon stopped because it exceeded the function evaluation limit,
options.MaxFunEvals = 3000 (the default value).
ans =


Решение ОЗК условной минимизацией
fval =
   21.6744
g =
  -5.6645e-06
ct =
  Columns 1 through 12
   -3.0362   -2.8144    2.4928   -4.1763   -0.4538    2.9220   -0.9062   -5.9317    0.0468
-1.1576   -2.0812    0.7104
  Columns 13 through 18
    3.2411   -3.1929   -0.9236    2.3621   -3.1895   -1.1718
r =
         0    0.8702    1.3360    1.4556    1.0509    1.4221    1.9933
         0    0.4927    1.3766    2.3609    2.9962    2.5211    2.0045
         0   -0.0078    0.0314   -0.0988    0.5589    1.3568    1.9946
Solver stopped prematurely.
fmincon stopped because it exceeded the function evaluation limit,
options.MaxFunEvals = 3000 (the default value).
fminc =
   21.6642
>>
```

Численные результаты при минимизации $f_2(\boldsymbol{c})$

```
Начальное положение
ct =
  Columns 1 through 12
```

```
   0.5236         0         0    0.5236         0         0    0.5236         0         0    0.5236         0
0
  Columns 13 through 18
   0.5236         0         0    0.5236         0         0
r =
         0    0.8660    1.3660    1.3660    0.8660    0.0000   -1.0000
         0    0.5000    1.3660    2.3660    3.2321    3.7321    3.7321
         0         0         0         0         0         0         0
dr =
    0.8660    0.5000    0.0000   -0.5000   -0.8660   -1.0000
    0.5000    0.8660    1.0000    0.8660    0.5000    0.0000
         0         0         0         0         0         0

fl = (значения целевой функции при безусловной минимизации)
  Columns 1 through 12
   16.0000   28.1703    0.7067    0.0575    0.0487    0.2330    0.5803    0.6832    0.4110
0.0000    0.0008    0.0000
  Columns 13 through 21
    0.0003    0.0747    0.0988    0.0354    0.0752    0.3213    5.3098    1.5283   16.0000



Solver stopped prematurely.

fmincon stopped because it exceeded the function evaluation limit,
options.MaxFunEvals = 3000 (the default value).

Решение ОЗК с безусловной минимизацией
nump =
     1
l =
```

```
   10
ans =
  2.2851e-06
ct =
 Columns 1 through 12
  -1.6586  -0.8168   0.8786  -2.3272  -0.5079   2.0909  -0.0015  -1.0142   1.5066
-0.3423  -1.6799   0.9746
 Columns 13 through 18
   1.6025  -1.4597   1.3267   0.8900  -0.9755   0.2491
r =
       0   0.2905  -0.1744   0.7995   1.2942   1.4548   1.9990
       0  -0.3391   0.5080   0.5408   0.7235   1.3517   1.9991
       0   0.8948   1.1524   1.3773   2.2269   1.4656   1.9993
Решение ОЗК условной минимизацией
fval =
   0.2201
g =
 -6.8664e-05
ct =
 Columns 1 through 12
  -2.0832   0.1377   2.0783  -1.8245  -2.5830   1.8330  -1.2869  -0.4998   1.7513
0.5111  -2.9383  -0.3103
 Columns 13 through 18
   1.6504  -3.2824   0.3860  -0.5357   0.0086   0.2491
r =
       0  -0.6618  -0.7702  -0.1228   0.5859   1.2915   1.9959
       0  -0.6808  -0.7776  -0.0936   0.5902   1.2916   1.9962
       0   0.3139   1.3033   1.6395   1.8133   1.9136   1.9995
fval =
   0.2201
……
```

Решение ОЗК с безусловной минимизацией

```
nump =
    4
l =
   13
ans =
  2.8090e-04
ct =
 Columns 1 through 12
```

```
  -2.0334  -1.7150   0.4244  -2.5959  -1.5199   1.0826  -1.2005  -1.9643   0.2346
-1.1650  -2.8290   0.0669
  Columns 13 through 18
   0.7624  -2.3030  -0.1208  -0.2351  -2.6137  -1.1718
r =
       0   0.5542   0.6624   0.6453   1.6407   2.2354   2.0107
       0   0.0903   0.7304   1.6434   1.5563   1.5010   2.0100
       0   0.8275   1.5881   1.9958   2.0370   2.8391   2.0081
Solver stopped prematurely.
fmincon stopped because it exceeded the function evaluation limit,
options.MaxFunEvals = 3000 (the default value).

Решение ОЗК условной минимизацией
fval =
   0.2226
g =
  -9.7076e-05
ct =
  Columns 1 through 12
  -2.8907   -0.2309    0.2203   -2.7073   -0.9717    1.6862   -1.8884   -1.9607   -
0.0208  -0.8135  -1.0558   0.7376
  Columns 13 through 18
   1.6196  -3.2467   0.1790  -0.5704  -0.0128  -1.1718
r =
       0  -0.6713  -0.7740  -0.1083   0.5913   1.2939   1.9988
       0  -0.6757  -0.7722  -0.0993   0.5948   1.2943   1.9988
       0   0.3045   1.2945   1.6168   1.7867   1.9173   1.9998
Solver stopped prematurely.

fmincon stopped because it exceeded the function evaluation limit,
options.MaxFunEvals = 3000 (the default value).
fminc =
   0.2201
>>
```

4. Численное решение ОЗК МП методами условной минимизации с использованием градиентов

4.1. Сокращение количества вычислений с использованием градиентов

В программных функциях минимизации MATLAB и SciPy предусмотрен автоматический расчет производных методом конечных разностей, при отсутствии ввода собственных функций для расчета градиентов. Тем не менее, полагается, что при задании собственных функций расчета градиентов можно сократить количество необходимых вычислений и ускорить решение ОЗК МП.

Расчет градиентов методом конечных разностей

Частные производные функции $g(\boldsymbol{c})$ методом конечных разностей приближенно вычисляются по формуле:

$$\frac{\partial g(\boldsymbol{c})}{\partial c_m} \approx \frac{g(c_1, \dots, c_m + \varepsilon, \dots, c_{3n}) - g(\boldsymbol{c})}{\varepsilon},$$

где ε – малое число.

Количество необходимых вычислений в этом случае будет пропорционально произведению количеству переменных, умноженного на количество матричных преобразований:

$$C_1 \sim 3n \cdot n$$

Так как в манипуляторе при повороте i-го звена меняются координаты только последующих узлов с номерами больше i, то нет необходимости пересчитывать предыдущие узлы. Количество необходимых вычислений в этом случае будет пропорционально количеству преобразований, равного сумме убывающей арифметической прогрессии:

$$C_2 \sim 3 \cdot (n + n-1 + n-2 \ldots + 1) = 3n \cdot \frac{n+1}{2}$$

С учетом того, что

$$\lim_{n\to\infty} \frac{C_1}{C_2} \sim \lim_{n\to\infty} \frac{3n \cdot n}{3n \cdot \frac{n+1}{2}} = 2$$

получается, что для МП большой размерности при использовании собственных функций для расчета градиентов время решения оптимизационной задачи уменьшается практически в два раза, даже при вычислении производных этим же методом конечных разностей. Полагается, что дополнительное сокращение времени решения можно получить с применением более точных выражений для частных производных.

Расчет с помощью производных матриц преобразования

Расчет градиентов функции $g(\mathbf{c})$ на основании пункта 2.3.

$$\begin{pmatrix} \frac{\partial \boldsymbol{r}_n}{\partial c_m} \\ 1 \end{pmatrix} = Z_{1,k} \frac{\partial T_k}{\partial c_m} Z_{2,k} \begin{pmatrix} \Delta \boldsymbol{r}_n \\ 1 \end{pmatrix}, \qquad m = 1, \ldots, 3n$$

$$Z_{1,k} = \prod_{j=1}^{k-1} T_j \; ; \; Z_{2,k} = \prod_{j=k+1}^{n} T_j$$

Из формулы видно, что количество матричных преобразований можно дополнительно уменьшить, если предварительно сохранить повторяющиеся значения матриц $Z_{1,k}$ и $Z_{2,k}$ при решении прямой задачи.

4.2. Численные эксперименты с применением градиентов

Решение ОЗК с разными способами вычисления градиентов

Для решения ОЗК МП применялись методы условной минимизации SLSQP и trust-constr библиотеки SciPy. Искомыми переменными являлись углы относительного поворота для достижения целевой точки с минимальным

перемещением элементов МП с начальным положением в горизонтальной плоскости. Минимизируемая функция определялась по уравнению

$$f(\boldsymbol{c}) = \sum_{j=1}^{n} \left(\boldsymbol{r}_j - \boldsymbol{r}_{0,j}\right)^2,$$

а ее производная по формуле

$$\frac{\partial f(\boldsymbol{c})}{\partial c_m} = 2 \sum_{j=k}^{n} \left(\boldsymbol{r}_j - \boldsymbol{r}_{0,j}\right) \frac{\partial \boldsymbol{r}_j}{\partial c_m},$$

где k – номер звена с переменной c_m; $\boldsymbol{r}_{0,j}$ – векторы координат начального положения МП.

Ниже приведен фрагмент функции в Python для проведения минимизации разными методами с помощью интерфейса вида (рисунок 11).

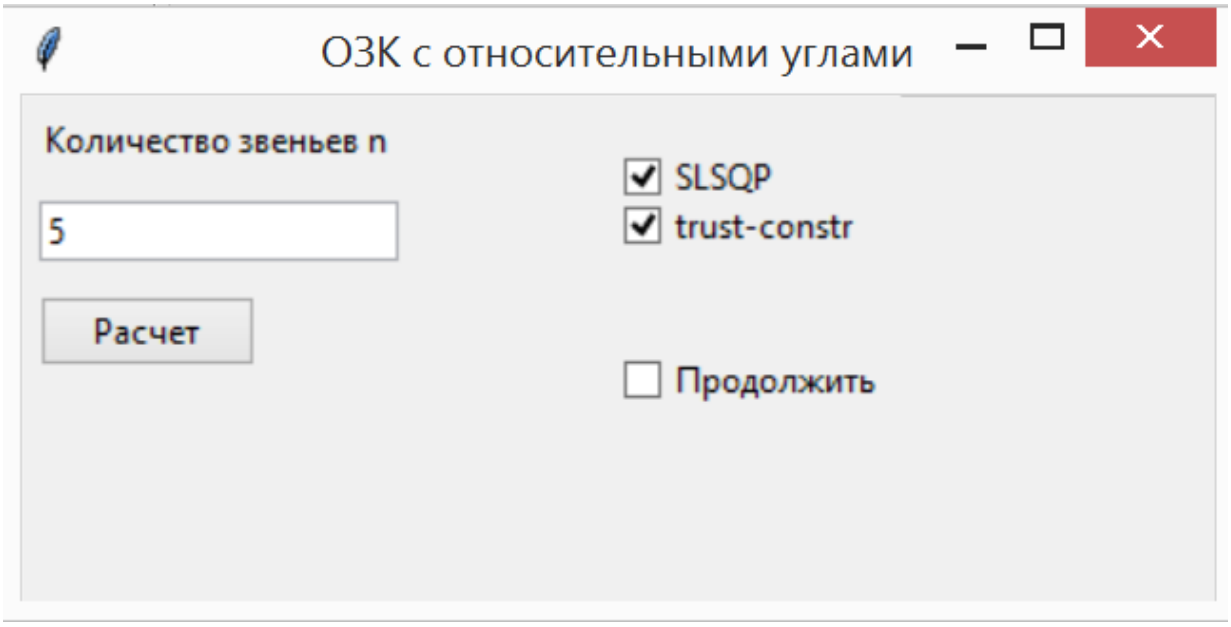

Рисунок 11 – Интерфейс для проведения минимизации разными методами

```
def MpMinim(smethod):
 #минимизация перемещений f=CalcWC(c)
 #при  условии совпадения рабочего органа с целевой точкой
g(c)=CalcRC(c)=0
 con1 = {'type':'eq','fun':mp.CalcRC,'jac':mp.CalcRDgCnew}
#  con1 = {'type':'eq','fun':mp.CalcRC}
 cons = (con1)
#  opt = {'disp':True,'maxiter':500}
 opt = {'maxiter':500}
 t1=time.time()
 result = optimize.minimize(
   mp.CalcWC,  mp.c0,
         method=smethod,
         jac=mp.CalcWDfCnew,
         constraints=cons,options = opt, tol = 1e-5)
 dt=time.time()-t1
```

mp.CalcWC, mp.CalcRC, mp.CalcWDfCnew, mp.CalcRDgCnew – функции для расчета значений $f(\mathbf{c})$, $g(\mathbf{c})$ и их градиентов, находящиеся в отдельном модуле.

Тексты данных функций аналогичны текстам, составленных в MATLAB.

mp.CalcRC(c) – возвращает $g(\mathbf{c})$

```
# функция расчета ПЗ
def CalcRC(c):
 CalcRCib(0,c)
 #квадрат расстояния до целевой точки
 return (r[0,n]-rn[0])**2+ (r[1,n]-rn[1])**2+ (r[2,n]-rn[2])**2
```

где CalcRCib – функция последовательного расчета координат узлов с номерами, превышающими заданный:

```
# функция расчета узлов с определенными номерами
def CalcRCib(ib,c):
 A = np.full((3,3), 0.0)
 p = np.full(4, 0.0)
 dp = np.full(4, 0.0)
 dp[3]=1.

 for i in range(ib,n):
   A[:,:] = Eu(c[i*3], c[i*3+1], c[i*3+2])
```

```
  for k in range(3):
    dp[k]=dr[k,i]
    for m in range(3):
      TT[k, m,i]=A[k, m]
  TT[3,3,i]=1
  if i>0:
    for k in range(3):
      TT[k,3,i]=dr[k,i-1]

  PP[:,:,i+1] = np.matmul(PP[:,:,i], TT[:,:,i])
  p[:] = np.matmul(PP[:,:,i+1], dp[:])
  for k in range(3):
    r[k,i+1]=p[k]
```

mp.CalcWC(c) – возвращает *f*(**c**)

```
def CalcWC(c):
  CalcRC(c)
  f=0
  for i in range(n+1):
     f=f  + (r[0,i]-r0[0,i])**2+ (r[1,i]-r0[1,i])**2+ (r[2,i]-r0[2,i])**2
  return f
```

mp.CalcWDfCnew(c) – возвращает градиенты *f*(**c**)

```
def CalcWDfCnew(c):
  DA=np.full((3,3,3), 0.)
  dra=np.full((3,3), 0.)
  d=np.full((4,3), 0.)
  dp=np.full((4), 0.)
  Pr=np.full((4,4,3), 0.)
  ds=np.full((3), 0.)
  dp[3]=1
  df=[]

  CalcRC(c)

  for i in range(n):
    DA=DEuM(c[i*3],c[i*3+1],c[i*3+2])
    Pr[:,:,:]=0
    for k in range(3):
      for m in range(3):
```

```
      Pr[k,m,:]=DA[k,m,:]
  for k in range(3):Pr[:,:,k]= np.matmul( PP[:,:,i], Pr[:,:,k])
  ds[:]=0

  for j in range(i,n):
    if j>i:
      for k in range(3): Pr[:,:,k]= np.matmul( Pr[:,:,k], TT[:,:,j] )
    for m in range(3): dp[m]=dr[m,j]
    for k in range(3): d[:,k]= np.matmul(Pr[:,:,k],dp)*2

    for k in range(3):
      s=0
      for m in range(3):s+=(r[m,j+1]-r0[m,j+1])*d[m,k]
      ds[k]+=s

  for k in range(3): df.append(ds[k])
 return df
```

mp. CalcRDgCnew (c) – возвращает градиенты $g(\mathbf{c})$

```
def CalcRDgCnew(c):
 DA=np.full((3,3,3), 0.)
 dra=np.full((3,3), 0.)
 d=np.full((4,3), 0.)
 dp=np.full((4), 0.)
 Pr=np.full((4,4,3), 0.)
 rr=np.full((4,n+1), 0.)
 dp[3]=1
 df=[]
 CalcRC(c)
 for i in range(3):rr[i,n]=dr[i,n-1]
 rr[3,n]=1
 for i in range(n-1,0,-1):
  rr[:,i]= np.matmul(TT[:,:,i],rr[:,i+1])

 for i in range(n):
  DA=DEuM(c[i*3],c[i*3+1],c[i*3+2])
  Pr[:,:,:]=0
  for k in range(3):
    for m in range(3): Pr[k,m,:]=DA[k,m,:]
  for k in range(3):Pr[:,:,k]= np.matmul( PP[:,:,i], Pr[:,:,k])
  for k in range(3):d[:,k]= np.matmul( Pr[:,:,k], rr[:,i+1] )*2
  for k in range(3):
    s=0
```

```
    for m in range(3): s+= (r[m,n]-rn[m])*d[m,k]
    df.append(s)
return df
```

На рисунках 12-21 приведены результаты решения для МП с разной размерностью в графическом виде.

При n=5 без указания градиентов: время расчета методом SLSQP – 3.7 с; время расчета методом trust-constr – 6.0 с (рисунок 12).

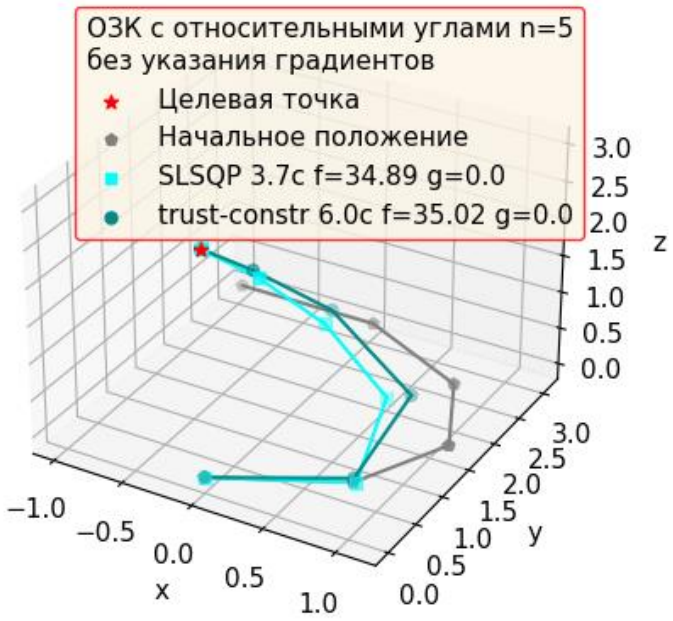

Рисунок 12 - Решение ОЗК МП при n=5 без указания градиентов

При n=5 с указанием градиентов: время расчета методом SLSQP – 2.3 с; время расчета методом trust-constr – 1.7 с (рисунок 13).

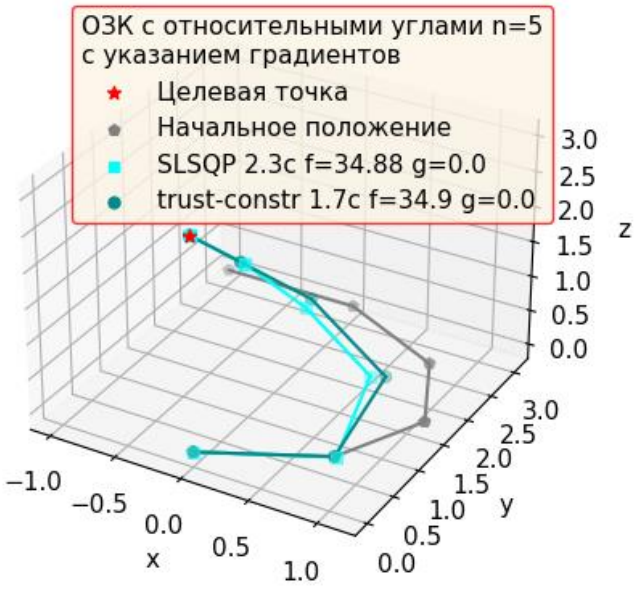

Рисунок 13 - Решение ОЗК МП при n=5 с указанием градиентов

При n=10 без указания градиентов: время расчета методом SLSQP – 19 с; время расчета методом trust-constr – 17.4 с (рисунок 14).

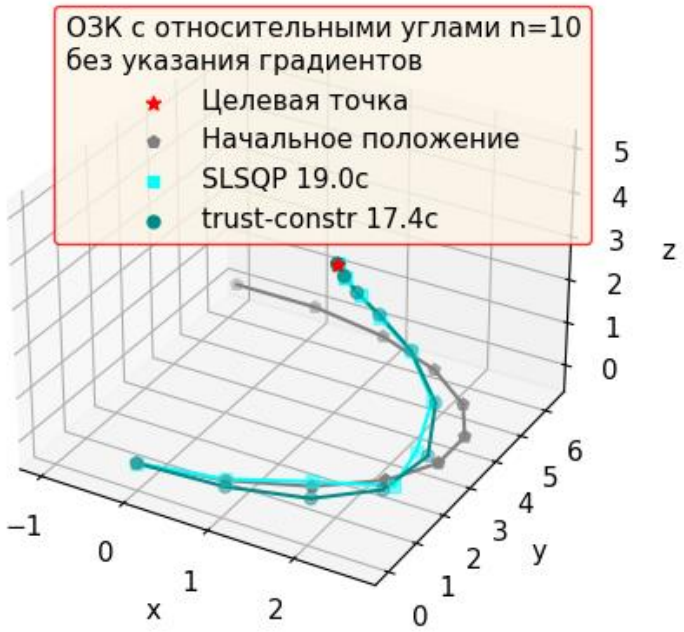

Рисунок 14 - Решение ОЗК МП при n=10 без указания градиентов

При n=10 с указанием градиентов: время расчета методом SLSQP – 5.2 с; время расчета методом trust-constr – 4.1 с (рисунок 15).

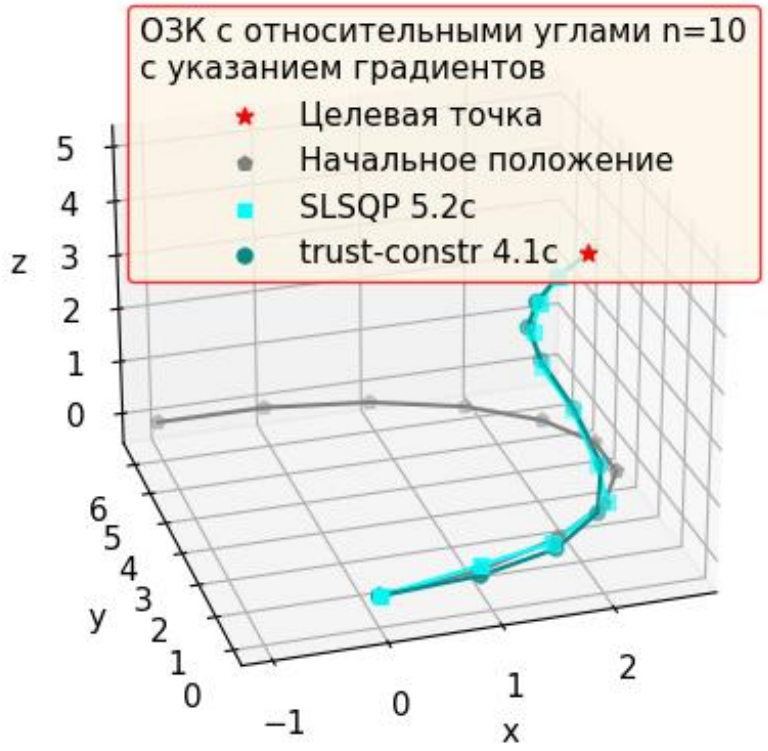

Рисунок 15 - Решение ОЗК МП при n=10 с указанием градиентов

При n=20 без указания градиентов: время расчета методом SLSQP – 119.5 с; время расчета методом trust-constr – 54.4 с (рисунок 16).

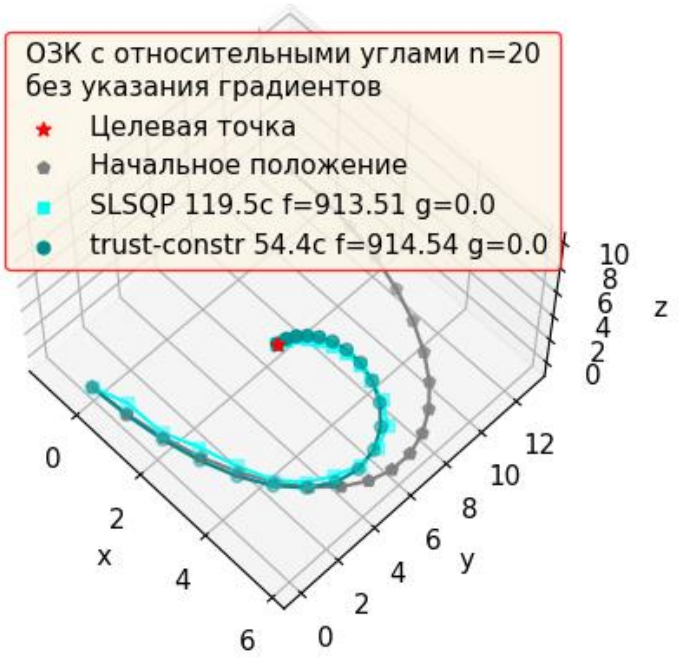

Рисунок 16 - Решение ОЗК МП при n=20 без указания градиентов

При n=20 с указанием градиентов: время расчета методом SLSQP – 49.0 с; время расчета методом trust-constr – 16.4 с (рисунок 17).

Рисунок 17 - Решение ОЗК МП при n=20 с указанием градиентов

При n=30 без указания градиентов: время расчета методом SLSQP – 249.1 с; время расчета методом trust-constr – 123.5 с (рисунок 18).

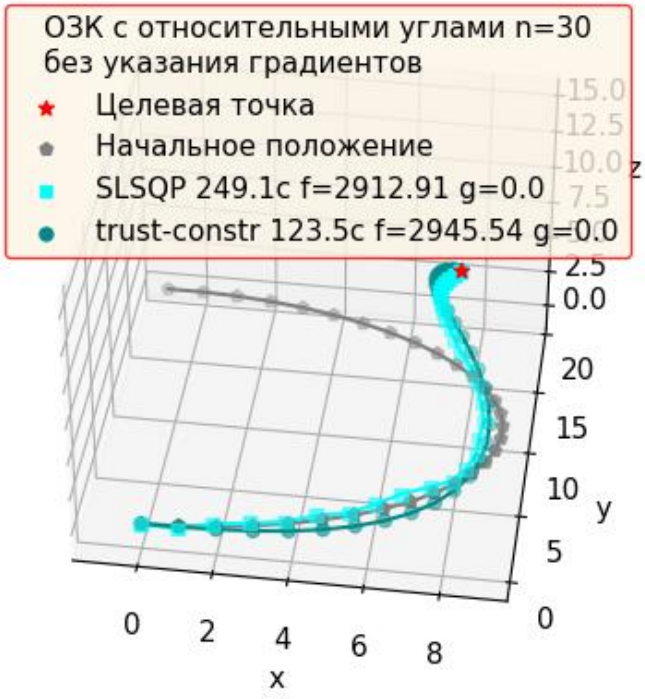

Рисунок 18 - Решение ОЗК МП при n=30 без указания градиентов

При n=30 с указанием градиентов: время расчета методом SLSQP – 81.8 с; время расчета методом trust-constr – 37.0 с (рисунок 19).

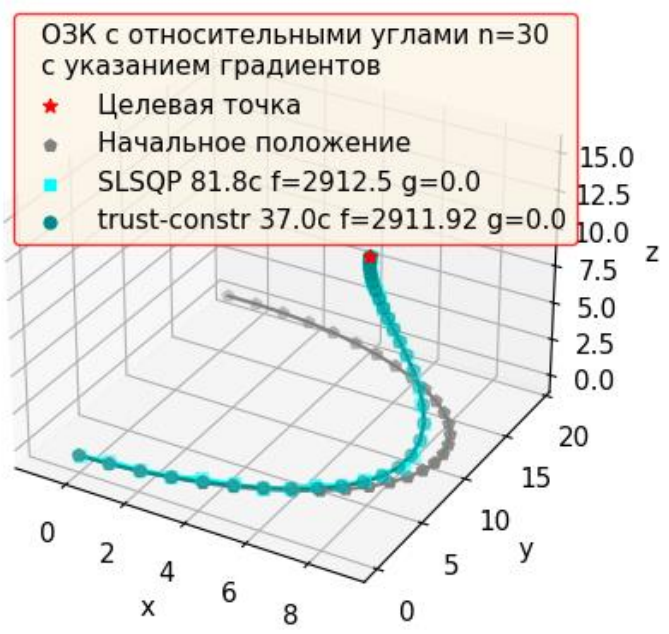

Рисунок 19 - Решение ОЗК МП при n=30 с указанием градиентов

При n=40 без указания градиентов: время расчета методом SLSQP – 600.4 с; время расчета методом trust-constr – 256.4 с (рисунок 20).

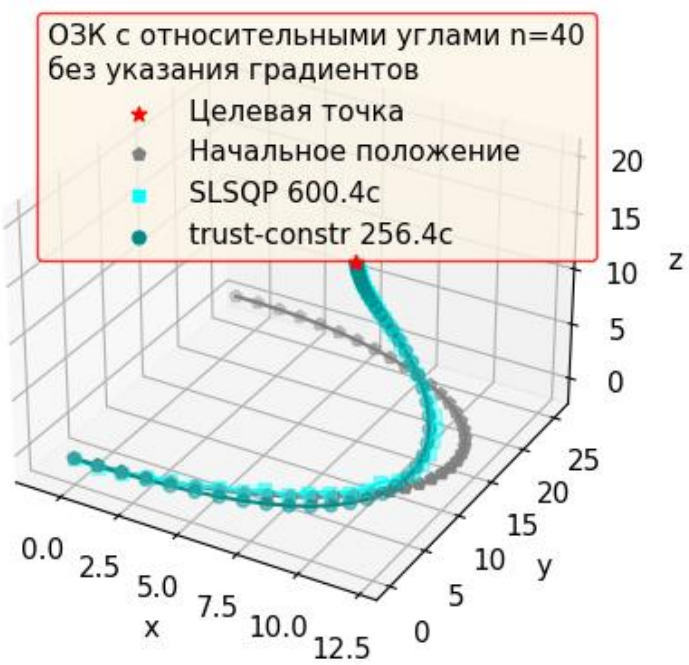

Рисунок 20 - Решение ОЗК МП при n=40 без указания градиентов

При n=40 с указанием градиентов: время расчета методом SLSQP – 163.5 с; время расчета методом trust-constr – 44.6 с (рисунок 21).

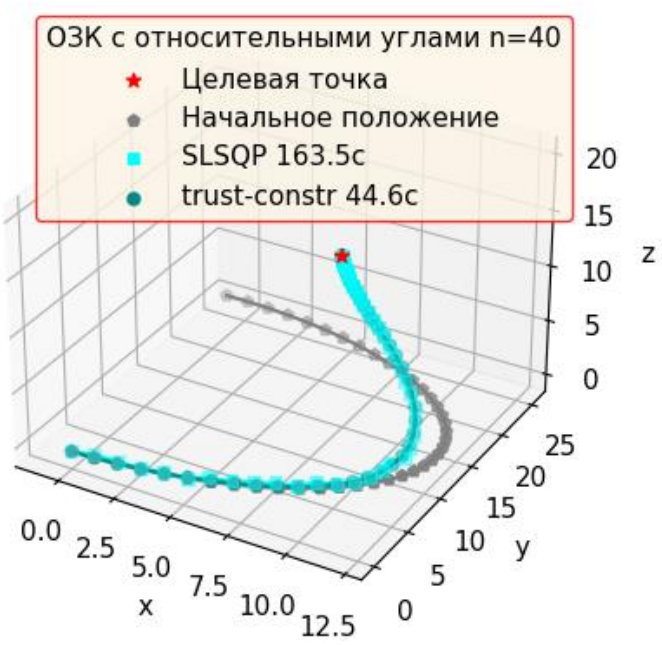

Рисунок 21 - Решение ОЗК МП при n=40 с указанием градиентов

Таблица 1 - Сравнение времени расчета с указанием и без указания градиентов

	Метод SLSQP		Метод trust-constr	
	время, с	время, с	время, с	время, с
n	без градиентов	с градиентами	без градиентов	с градиентами
5	3.7	2.3	6	1.7
10	19	5.2	17.4	4.1
20	119.5	49	54.4	16.4
30	249.1	81.8	123.5	37
40	600.4	163.5	256.4	44.6

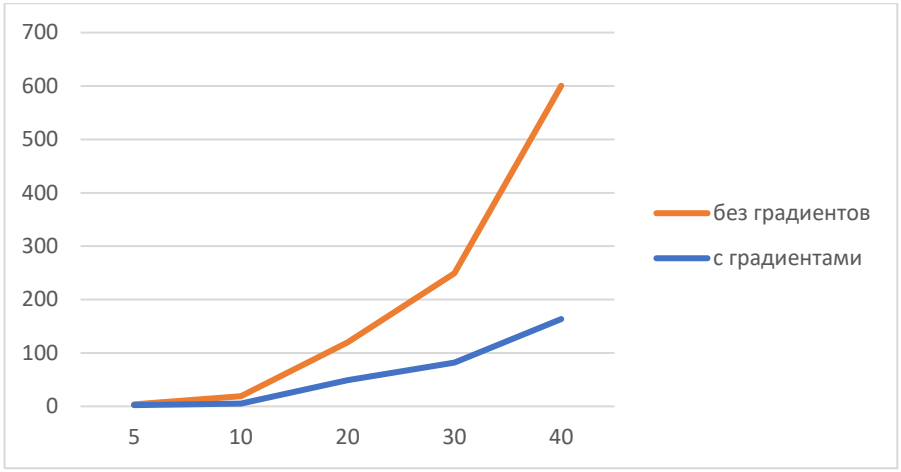

Рисунок 22 - Графики зависимости времени расчета от размерности МП методом SLSQP

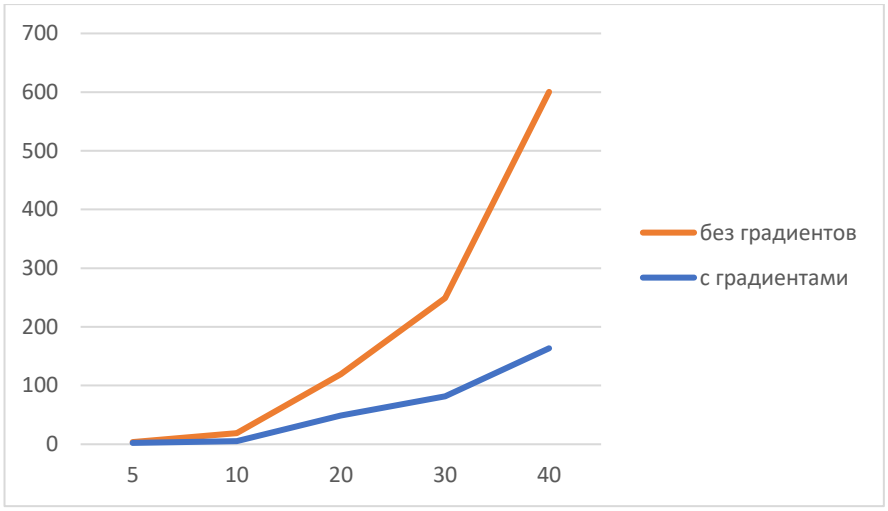

Рисунок 23 - Графики зависимости времени расчета от размерности МП методом trust-constr

Время решения ОЗК с использованием собственных функций для расчета градиентов функций $f(\boldsymbol{c})$ и $g(\boldsymbol{c})$ уменьшалось в 3 – 4 раза.

4.3. Улучшение обусловленности ОЗК МП с помощью абсолютных углов

Многочисленные произведения матриц с тригонометрическими функциями увеличивают как время, так и погрешности вычислений. В результате решение может оказаться неустойчивым. В качестве примера на

рисунках 24 и 25 приведены результаты решения ОЗК с минимальным отклонением центра тяжести МП разными методами условной минимизации в Python и MATLAB.

Рисунок 24 - Решение ОЗК с минимальным отклонением центра тяжести МП методами условной минимизации SLSQP и trust-constr библиотеки SciPy

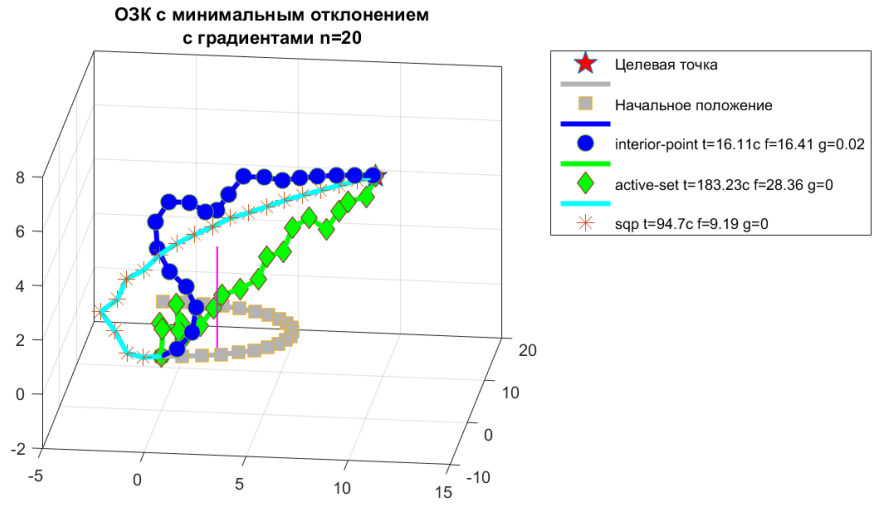

Рисунок 25 - Решение ОЗК с минимальным отклонением центра тяжести МП методами условной минимизации interior-point, active-set и sqp в среде MATLAB

Наилучшие решения получаются методами SLSQP и sqp, а остальные решения сильно отличаются друг от друга.

Для МП со сферическими соединениями с тремя вращательными степенями свободы в качестве независимых переменных можно использовать абсолютные углы вращения. В этом случае расчет функций и их градиентов значительно упрощается с помощью выражений:

$$\boldsymbol{r}'_k = A(\propto_k, \beta_k, \gamma_k)(\boldsymbol{r}_k - \boldsymbol{r}_{k-1}) + \boldsymbol{r}_{k-1}$$

$$\frac{\partial \boldsymbol{r}_n}{\partial c_m} = \frac{\partial A(\propto_k, \beta_k, \gamma_k)}{\partial c_m}(\boldsymbol{r}_k - \boldsymbol{r}_{k-1}),$$

где k – номер звена с переменной c_m.

Для сравнения на рисунке 26 приведено повторное решение ОЗК МП, приведенного ранее на рисунке 24. Получены лучшие результаты обоими методами с меньшими вычислительными затратами. Как показано на рисунке 26, время перерасчета абсолютных углов в относительные пренебрежимо мало.

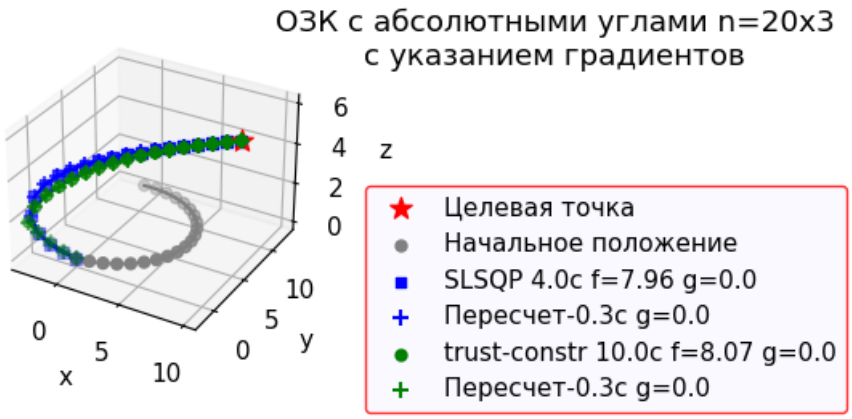

Рисунок 26 - Решение ОЗК с минимальным отклонением центра тяжести МП методами условной минимизации SLSQP и trust-constr библиотеки SciPy с абсолютными углами

Следует отметить, что вопрос об эффективности применения абсолютных углов также рассматривается исследователями в области

биомеханики. Так в статье [51] сравниваются абсолютные и относительные углы для изучения биомеханических ограничений походки (Рисунок 27).

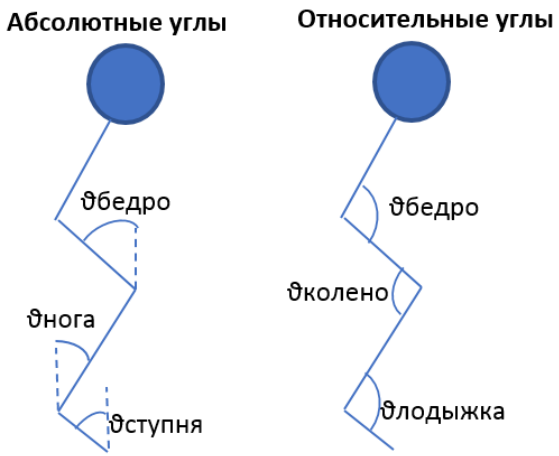

Рисунок 27 - Абсолютные и относительные углы для изучения биомеханических ограничений походки

Для сравнительной оценки по входному параметру на рисунках 28 и 29 изображены отклонения координат рабочего органа, полученные при изменении угла поворота c[0]. Видно, что отклонение рабочего органа с относительными углами на порядок выше отклонения рабочего органа с абсолютными углами.

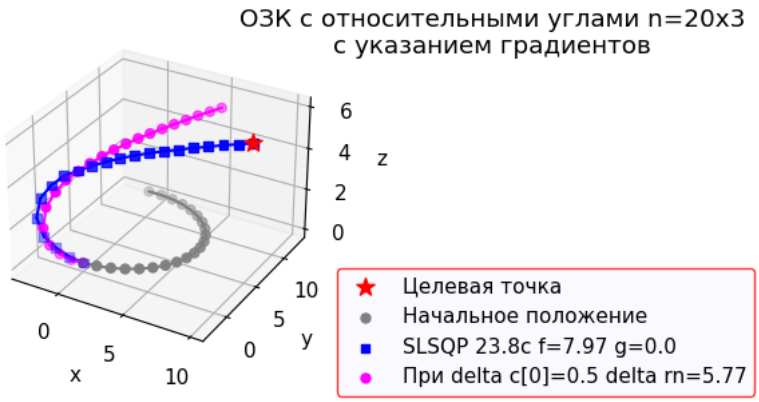

Рисунок 28 - Чувствительность по параметру c[0]

Рисунок 29 - Чувствительность по параметру c[0]

Меньшие значения градиентов МП с абсолютными углами также могут свидетельствовать о лучшей обусловленности ОЗК с абсолютными углами (см. рисунки 30, 31).

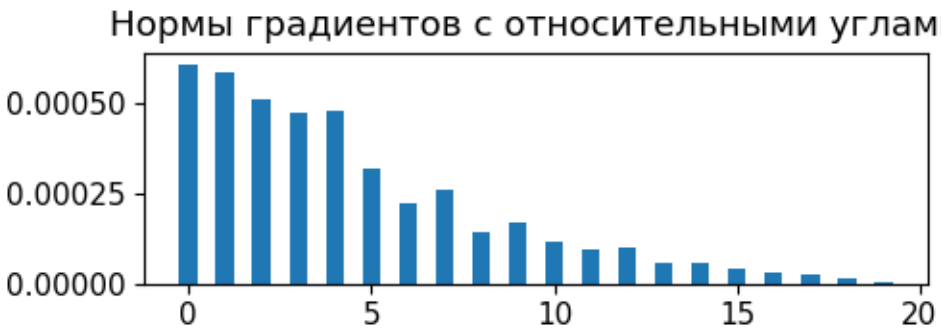

Рисунок 30 - Нормы градиентов

Рисунок 31 - Нормы градиентов

Таким образом, МП со сферическими соединениями, с улучшенной обусловленностью задачи может служить основой для решения ОЗК со специальной схемой. Как показано на рисунке 32, по полученным координатам каждого узла можно найти с минимальными вычислительными затратами соответствующие углы поворота МП со специальной схемой Денавита – Хартенберга.

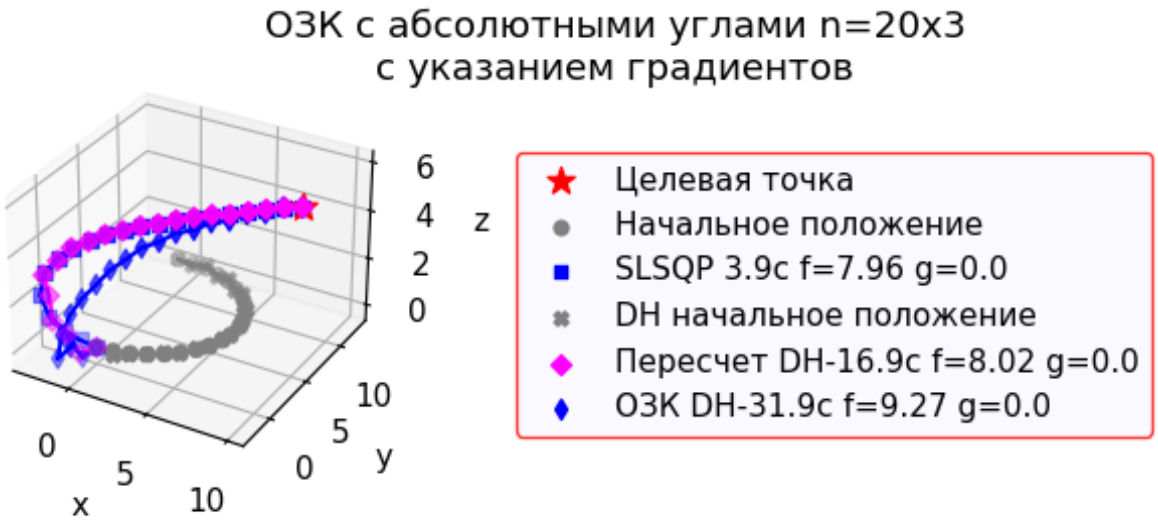

Рисунок 32 - Расчет МП со схемой Денавита-Хартенеберга

На рисунке 33 показаны результаты предварительного решения ОЗК с помощью МП со сферическими соединениями. В третьей строке легенды указано время решения методом SLSQP 3.9 с. По полученным координатам узлов искомые относительные углы МП со схемой Денавита – Хартенберга вычислены методом безусловной минимизации отклонений с временем расчета 16.9 с (в 5-й строке легенды). Наконец, для сравнения в последней строке легенды отмечены результаты независимого решения ОЗК МП со схемой Денавит-Хартенберга с большим временем расчета 31.9 и с худшим результатом.

Аналогично, МП с абсолютными углами был использован для решения ОЗК МП, имитирующего ходьбу робота (см. рисунки 33-37). Целью задачи являлась минимизация перемещения узлов при условии размещения центра тяжести в опорной целевой точке.

Так как в данном случае МП представляет собой разветвленную систему звеньев, то была запрограммирована специальная последовательность расчета узлов. Ниже представлен фрагмент функции для решения ПЗ с выбранным порядком вычисления координат звеньев.

```
def CalcR(c):
   A = np.full((3,3), 0.0)
   for i in range(n):
      al[0,i]=c[i*3]
      al[1,i]=c[i*3+1]
      al[2,i]=c[i*3+2]

   def calcP(i,i0):
      A = Eu(al[0,i], al[1,i], al[2,i])
      r[:, i+1] = np.matmul(A, dr[:,i]) +r[:, i0]
   for i in range(4):  calcP(i,i)
   calcP(4,2)
   calcP(5,5)
   calcP(6,5)
   calcP(7,7)
   calcP(8,5)
   calcP(9,9)
```

В данном случае решались две задачи при последовательных шагах правой и левой ногой с соответствующими условиями минимизации:

$$\begin{cases} f(\boldsymbol{c}) \to min \\ g(\boldsymbol{c}) = 0 \end{cases},$$

где

$$f(\boldsymbol{c}) = \sum_{k=0}^{10} \left(\boldsymbol{r}_j - \boldsymbol{r}_{0,j}\right)^2,$$

$$g(\boldsymbol{c}) = (\boldsymbol{r}_{nn} - \boldsymbol{r}^*)^2 + x^2 + y^2,$$

$$x = \sum_{k=0}^{10} \frac{x_k - x_c}{11}, \qquad y = \sum_{k=0}^{10} \frac{y_k - y_c}{11}$$

$\boldsymbol{r}_{nn}$ – расчетный вектор узла, стремящегося к целевой точке $\boldsymbol{r}_c$,

x_k, y_k – расчетные координаты k-го узла,

x_c, y_c – координаты точки, где должен быть центр тяжести.

```
tolc=1e-2
#Расчет оптимизации
#условие минимизации
conm = {'type':'eq','fun':CalcEq2,'jac':CalcRDgA}
#conm = {'type':'eq','fun':CalcEq2}
#Минимизация суммы перемещений CalcW(c) -> min
#1-е условие-нога в целевой точке; CalcR(c)=0
#2-е условие - центр тяжести в опорной точке CalcEq(c)=0
numc=4 # номер опорного узла
c[:]=0
t1=time.time()


result = optimize.minimize(CalcW, c,
                   method='SLSQP',
#                    method='trust-constr',
                   jac=CalcWDfA,
                   constraints=conm,  tol=tolc, options={'maxiter':200})
dt1=time.time()-t1
c[:]=result.x[:]
f=CalcR(c)
```

```
r0[:,:]=r[:,:]
numc=0 # номер опорного  узла
r[:,numc]=[2*ls, 0, 2*hs] #первый узел на новой ступеньке
rc[:]=r[:,numc] #координаты для центра тяжести
#Расчет оптимизации повторно
t1=time.time()
result = optimize.minimize(CalcW, c,
#                    method='trust-constr',
                     method='SLSQP',
                     jac=CalcWDfA,
                     constraints=conm, tol=tolc, options={'maxiter':200})
dt2=time.time()-t1
c[:]=result.x[:]
f=CalcR(c)
```

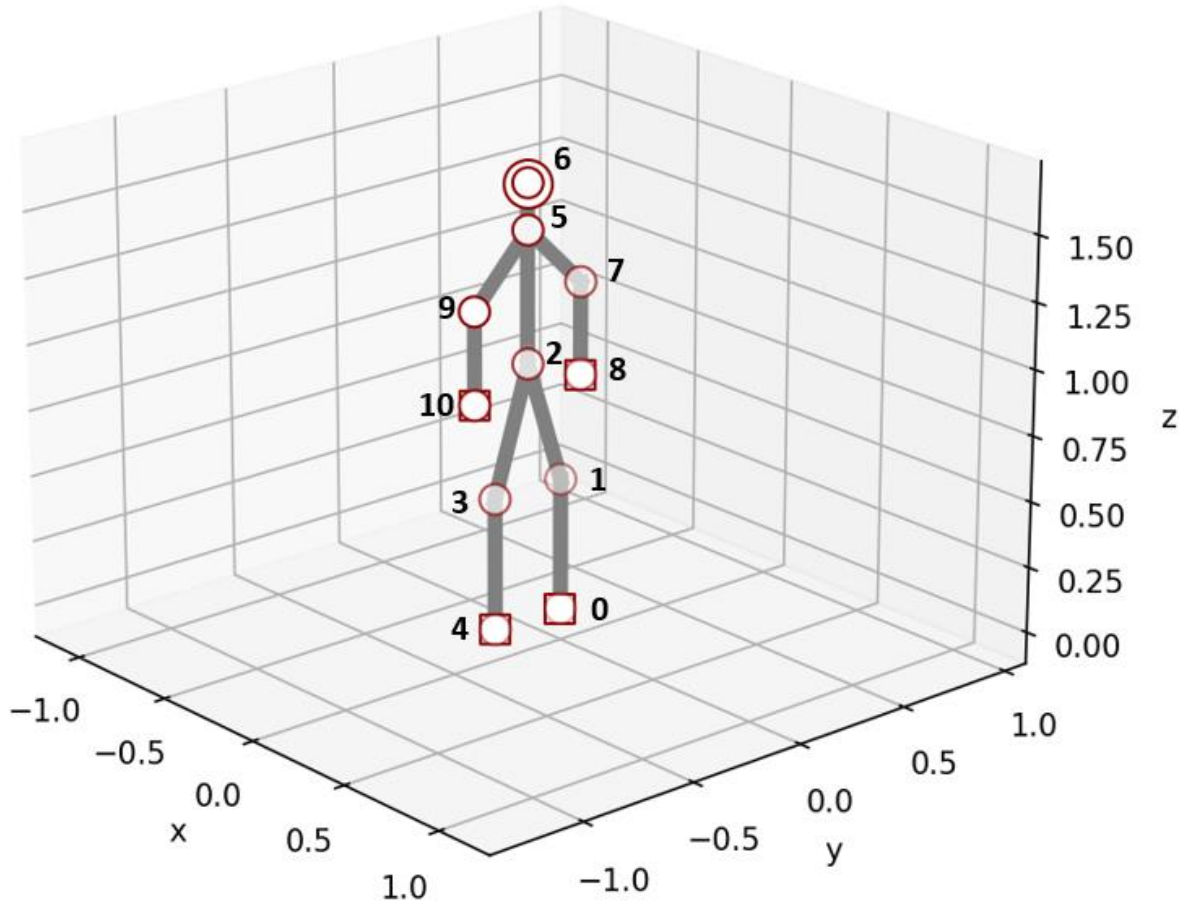

Рисунок 33 - Схема МП

Данные МП в начальном положении

Длина звеньев
0 0.5
1 0.5
2 0.5
3 0.5
4 0.5
5 0.17500000000000004
6 0.35
7 0.35
8 0.35
9 0.35
Абсолютные координаты
0 [0. 0. 0.]
1 [0. 0. 0.5]
2 [0.   -0.1545085   0.97552826]
3 [0.   -0.30901699 0.5   ]
4 [0.   -0.30901699 0.   ]
5 [0.   -0.1545085   1.47552826]
6 [0.   -0.1545085   1.65052826]
7 [0.   0.09297888 1.22804088]
8 [0.   0.09297888 0.87804088]
9 [0.   -0.40199587 1.22804088]

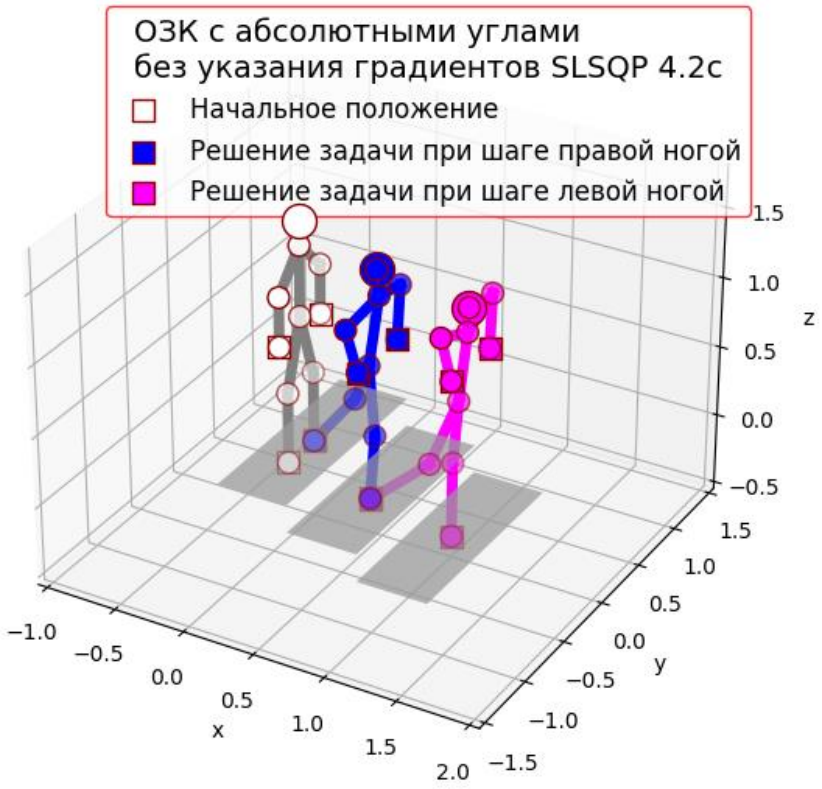

Рисунок 34 - Решение ОЗК МП с имитацией ходьбы робота по ступенькам без указания градиентов: время расчета 4.2 с

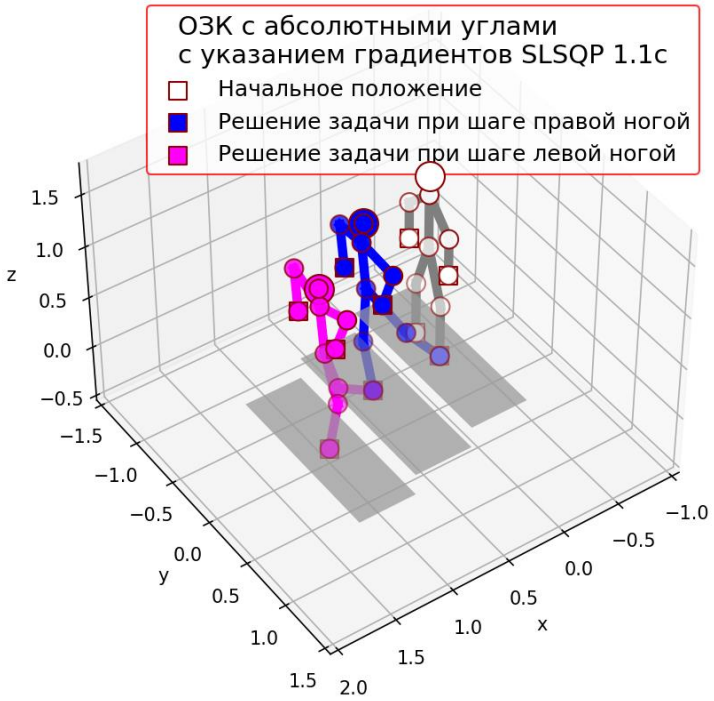

Рисунок 35 - Решение ОЗК МП с имитацией ходьбы робота по ступенькам без указания градиентов: время расчета 1.1 с

Результаты решения ОЗК с имитацией ходьбы робота по ступенькам

Абсолютные координаты при шаге правой ногой

0 [0. 0. 0.]
1 [0.3046128 -0.00419706 0.39647626]
2 [0.52350899 -0.20453639 0.79890465]
3 [0.55827719 -0.19351115 0.30023681]
4 [0.49947597 -0.15042759 -0.19442087]
5 [0.579797   -0.17697508 1.29496113]
6 [0.5616457 -0.17524324 1.46900863]
7 [0.53748018 0.1573994 1.20060548]
8 [0.56768325 0.06602189 0.86409705]
9 [0.53896234 -0.51284944 1.20540138]

время dt1(c)= 0.5470280647277832
минимум fmin1= 3.048473026734824
CalcR= 3.158412039854874e-05
CalcEq2= 0.0006092341264346175
Абсолютные координаты при шаге левой ногой
0 [1.   0. -0.4]
1 [0.96503853 0.06556814 0.09444769]
2 [1.03085141e+00 5.54859239e-04 5.85815105e-01]
3 [0.87922694 -0.05372887 0.11246186]
4 [0.50080463 -0.14986338 -0.19987564]
5 [1.07810489 0.01687944 1.08330944]
6 [1.08435283 0.01703236 1.25819781]
7 [1.06876523 0.347641   1.19736892]
8 [1.10498724 0.25040891 0.86310273]
9 [1.07481095 -0.31176306 1.20365583]
время dt2(c)= 0.5937738418579102
минимум fmin2= 4.173361180643127
CalcR= 6.815579651099117e-07
CalcEq2= 0.0001176033610944018 8

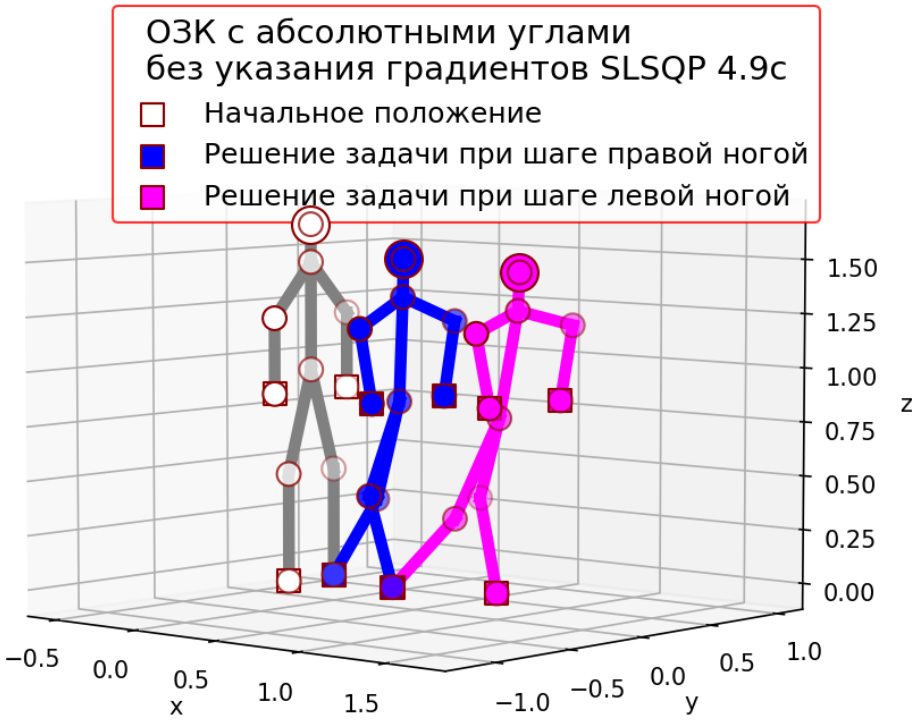

Рисунок 36 - Решение ОЗК МП с имитацией ходьбы робота без указания градиентов

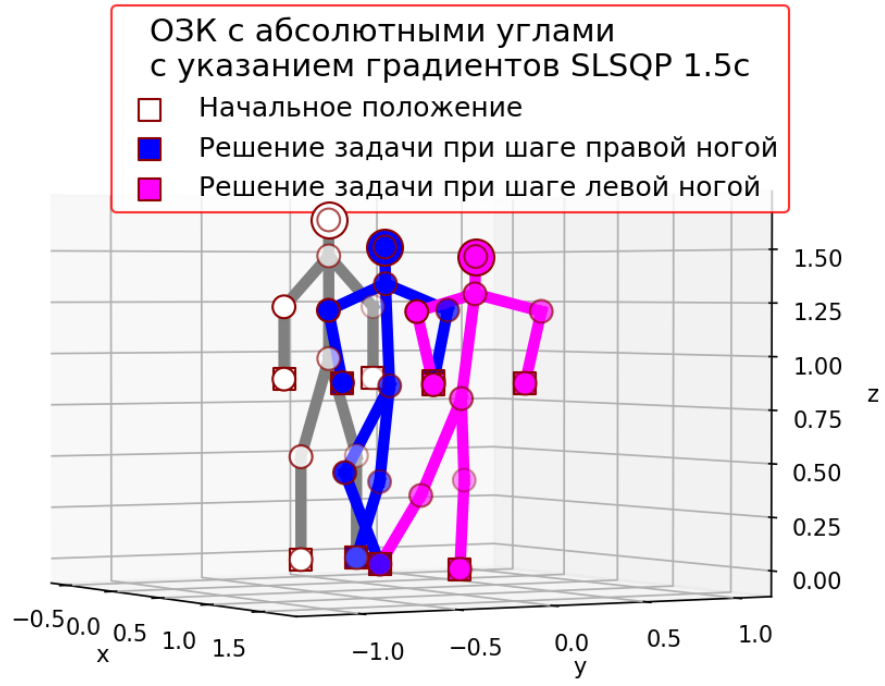

Рисунок 37 - Решение ОЗК МП с имитацией ходьбы робота с указанием градиентов

Результаты решения ОЗК с имитацией ходьбы робота по ровному полу

Абсолютные координаты при шаге правой ногой

0 [0. 0. 0.]
1 [0.31530725 -0.04768466 0.38510714]
2 [0.4896246 -0.09077137 0.85175154]
3 [0.5555567 -0.3731892   0.44445174]
4 [5.00032468e-01 -1.50372301e-01 3.01217234e-04]
5 [0.57347588 -0.16113733 1.33962201]
6 [0.58381136 -0.16987756 1.51409776]
7 [0.60506815 0.1636268 1.21301163]
8 [0.61071937 0.08263765 0.87255778]
9 [0.60231872 -0.48672893 1.21448929]
время dt1(c)= 0.718970775604248
минимум fmin1= 3.160778634000511
CalcR= 2.3039429487593604e-07
CalcEq2= 2.3160622917188377e-05

Абсолютные координаты при шаге левой ногой

0 [1. 0. 0.]
1 [0.75798507 0.16582099 0.40488538]
2 [1.01343241e+00 -3.83814816e-04 8.01272333e-01]
3 [0.86355206 -0.13364368 0.34325739]
4 [5.00288425e-01 -1.49723720e-01 6.50790950e-05]
5 [1.1049084 0.01921254 1.29244249]
6 [1.11784568 0.01549075 1.46692394]
7 [1.14859888 0.35653044 1.20993322]
8 [1.15301815 0.26467376 0.87223095]
9 [1.14671948 -0.31850982 1.21061684]
время dt2(c)= 0.7657992839813232
минимум fmin2= 3.6401456945025608
CalcR= 1.637543599143588e-07
CalcEq2= 1.5085791977558804e-05

Заключение

В работе рассмотрен подход решения ОЗК МП с помощью задачи условной минимизации некоторой целевой функции. Условием минимизации является совпадение расчетных координат рабочего органа с координатами целевой точки.

Для исследования эффективности данного подхода разработаны алгоритмы и программы численного решения ПЗК и ОЗК с применением существующих методов минимизации MATLAB и Python с библиотекой SciPy.

В численных экспериментах ОЗК решалась с помощью безусловной минимизации с алгоритмом Нелдера-Мида (симплекс методом) и условной минимизации методом внутренней точки (interior-point), active-set и sqp в MATLAB и методами SLSQP и trust-constr библиотеки SciPy Python.

Для уменьшения вычислительных затрат решения ОЗК предлагается применение градиентов минимизируемой и ограничивающих функций. По результатам численных экспериментов время расчета ОЗК при указании собственных функций для расчета градиентов уменьшается в несколько раз.

Дополнительно снизить вычислительные затраты и увеличить точность решения ОЗК МП со сферическими соединениями можно с помощью абсолютных углов. С помощью специально разработанной программы проиллюстрированы преимущества применения абсолютных углов для МП со сферическими соединениями.

Предварительное решение ОЗК МП со сферическими соединениями может быть использовано для решения МП со специальной схемой.

Список используемой литературы и используемых источников

1. Берёзкин Е. Н. Курс теоретической механики — 2-е изд., пер. — М.: Изд-во МГУ. 1974. — 641 с.

2. Бойко М.В., Дамбаев Б. Д. Математическая модель манипулятора livingstick. Кинематика и позиционирование. Электронное научное издание «Ученые заметки ТОГУ» 2017, Том 8, № 1, С. 153 – 160.

3. Братчиков С.А., Абрамова Е.А., Федосов Ю.В. Решение обратной задачи кинематики манипулятора. Вестник томского государственного университета. 2021, №56.

4. Булгаков А. Г., , Воробьев В. А. Промышленные роботы. Кинематика, динамика, контроль и управление. Москва : СОЛОН-ПРЕСС, 2021. - 488 с. - ISBN 978-5-91359-013-8.

5. Бурлачук А. М. Разработка двухсекционного прототипа манипулятора на гибких связях [Электронный ресурс]: Молодой ученый. — 2020. — № 30 (320). — С. 57-61. — URL: https://moluch.ru/archive/320/72805/ (дата обращения: 25.01.2023).

6. Верещагин А. Ф., Минаев Л. Н. Принципы построения специализированных вычислений для позиционного супервизорного управления манипуляционными роботами // Изв. АН СССР. Техническая кибернетика. 1978. №4. С. 56-65.

7. Данилов А.В., Кропотов А.Н., Трифонов О.В. Общий подход к решению обратной задачи кинематики для манипулятора последовательной структуры с помощью конечного поворота и смещения [Электронный ресурс]: Препринты ИПМ им. М.В.Келдыша. 2018. № 81. 15 с. doi:10.20948/prepr-2018-81. URL: http://library.keldysh.ru/preprint.asp?id=2018-81 (дата обращения: 25.10.2018).

8. Дикин И. И. Итеративное решение задач линейного и квадратичного программирования // Докл. АН СССР. — 1967. — Т. 174, № 4. — С. 747—748.

9. Дыда А.А., Оськин Д.А. Константинова Е.А. Нейросетевое моделирование задачи обратной кинематики для манипуляционногоробота [Электронный ресурс]: Современные наукоемкие технологии. – 2015. – № 12 (часть 2) – С. 254-257. URL: https://www.toptechnologies.ru/ ru/aricle/view?id=35248 (дата обращения 12.09.2015).

10. Дыда А.А., Оськин Д.А. Решение обратной задачи кинематики для манипуляционного робота методом штрафных функций. Фундаментальные исследования. № 11, 2015.

11. Егоров Е.Е. Моделирование работы манипуляционного робота в программном пакете MATLAB robotics toolbox. Политехнический молодежный журнал. 2020. № 01.

12. Журавлёв В. Ф. Основы теоретической механики — 2-е изд. — М.: Физматлит, 2001. С. 23.

13. Захарова Е. М., Минашина И. К. Обзор методов многомерной оптимизации. Информационные процессы. 2014. Т.14. №3. С.256-274.

14. Зенкевич С.Л., Ющенко А.С. Основы управления манипуляционными роботами. – М. Изд-во МГТУ им. Н.Э. Баумана, 2004 – 480 с.

15. Каргинов Л.А. Иерархический подход к решению обратной задачи кинематики // Наука и Образование. МГТУ им. Н.Э. Баумана. Электрон.журн. 2016. № 03. С. 37–63.

16. Карпенко А.П. Современные алгоритмы поисковой оптимизации. Алгоритмы, вдохновленные природой. М.: Изд-во МГТУ им. Н.Э. Баумана, 2017. С. 448.

17. Ковальчук А.К. Применение системы координат Денавита-Хартенберга для исполнительных механизмов роботов с древовидной кинематической структурой. Конструкторское Бюро, №10, 2016.

18. Колтыгин Д.С., Седельников И.А., Петухов Н.В. Аналитический и численный методы решения обратной задачи кинематики для робота DELTA.

А // Вестник Иркутского государственного технического университета. 2017. Т. 21. № 5. С. 87–96. DOI: 10.21285/1814-3520-2017-5-87-96.

19. Кондрашов Д.А. Решение обратной задачи кинематики шестизвенного манипулятора с использованием метода Ньютона-Рафсона. https://doi.org/10.26160/2618-7493-2021-4-9-13.

20. Круглов С.П., Иванченко С.А., Ковыршин С.В. Решение прямой расширенной задачи кинематики для манипулятора инвалидного кресла. ВЕСТНИКПНИПУ. № 41.2022. DOI: 10.15593/2224-9397/2022.1.07.

21. Махов А. А. Моделирование механических систем с помощью пакета расширения SimMechanics [Электронный ресурс]. URL: https://masters.donntu.ru/2013/fimm/pupysheva/library/article02.htm (дата обращения: 20.02.2012).

22. Мирзаев Р. А., Каменюк О.В. Исследование кинематики манипулятора с помощью пакета SimMechanics. Сибирский государственный аэрокосмический университет. 2012.

23. Околов А. Р., Дрозд А. В., Матрунчик Ю. Н., Лицкевич О. В. Математическое обеспечение промышленных роботов. Минск. БНТУ. 2015.

24. Петренко В. И., Тебуева Ф. Б., Антонов В. О., Антонов В. О. Математическая модель поиска оптимальных углов Эйлера для двигателей трехзвенного манипулятора. Информатика, вычислительная техника и управление, №3 март 2018 г.

25. Пол Р. Моделирование, планирование траекторий и управление движением робота-манипулятора: Пер. с англ. М.: Наука, 1976.

26. Саймон Д. Алгоритмы эволюционной оптимизации. — М.: ДМК Пресс, 2020. — 940 с. — ISBN 978-5-97060-812-8.

27. Смирнов П. А., Яковлев Р. Н. Решение прямой и обратной задач кинематики в системе позиционирования звеньев манипулятора. Мехатроника, автоматизация, управление. 2019. Том 20, № 12.

28. Талалов С.В. Сборник: Информационные технологии в моделировании и управлении: подходы, методы, решения. V Всероссийская

научная конференция с международным участием: сборник материалов. Тольятти, 2022. С. 111-115.

29. Тихонов А. Н., Арсенин В. Я. Методы решения некорректных задач. М.: Наука, 1979. 285 с.

30. Ту Раин, В.М. Довгаль, Ян Найнг Co.Thu Rain, V.M. Dovgal, Yan Naing Soe. Моделирование кинематического управления роботом-манипулятором «INTELBOT» на основе адаптивной нейро-нечеткой системы вывода (ANFIS). Научные ведомости. Серия Экономика. Информатика. 2018. Том 45, № 3.

31. Федоров В.В. Алгоритм глобальной многомерной минимизации с помощью уравнений конвективной диффузии [Электронный ресурс]: Программные продукты, системы и алгоритмы. № 4, 2018. URL: http://swsys-web.ru/en/global-multidimensional-minimization-algorithm-using-convective-diffusion-equations.html (дата обращения 20.05.2024).

32. Челноков Ю.Н. Бикватернионное решение кинематической задачи управления движением твердого тела и его приложение к решению обратных задач кинематики роботов-манипуляторов. Изв. РАН. Механика твердого тела 2012 №5 С. 30-42.

33. Courty N., Arnaud E. Inverse Kinematics Using Sequential Monte Carlo Methods. Proc. of the 5th International Conference on Articulated Motion and Deformable Objects (AMDO2008). 2008, 9-11 July, Port d'Andratx, Mallorca,. Spain. Port d'Andratx, 2008, pp. 1-10. DOI: 10.1007/978-3-540-70517-81.

34. Denavit J., Hartenberg R. S. Kinematic notation for Lower-Pair Mechanisms Based on Matrices // J. Appl. Mech. 77. P. 215-221, 1955.

35. Dorigo M., Stutzle T. Ant Colony Optimization. MIT Press, 2004.

36. Du K.-L., Swamy M.N. Search and Optimization by Metaheuristics. 2016, 437 p. DOI: 10.1007/978-3-319-41192-7.

37. Indra Agustian, Novalio Daratha, Ruvita Faurina, Agus Suandi, Sulistyaningsih. Robot Manipulator Control with Inverse Kinematics PD-

Pseudoinverse Jacobian and Forward Kinematics Denavit Hartenberg. Jurnal Elektronika dan Telekomunikasi (JET), Vol. 21, No. 1, August 2021, pp. 8-18.

38. Ingber L. Simulated Annealing: Practice versus theory. Simulated Annealing: Practice versus theory. P. 29–57.

39. Jing Huang, Qi Wang, Marco Fratarcangeli, Ke Yan, and Catherine Pelachaud. Multi-Variate Gaussian-Based Inverse Kinematics. Computer Graphics Forum _c 2017 The Eurographics Association and John Wiley & Sons Ltd. Published by John Wiley & Sons Ltd.

40. Karaboga, Dervis. Artificial bee colony algorithm. Wayback Machine Scholarpedia. 2010. 5(3): 6915.

41. Kinematics [Электронный ресурс] : Robotics-toolbox-python. URL: https://github.com/petercorke/robotics-toolbox-python/wiki/Kinematics (дата обращения: 15.05.2024).

42. Lee C. S., Ziegler M. A. A Geometric Approach in Solving of Inverse Kinematics of PUMA Robots // IEEE Trans. Aero, and Electr. Systems, AES-20. No 6. P. 695-706.

43. Optimization (scipy.optimize) [Электронный ресурс] : SciPy User Guide. URL: https://docs.scipy.org/doc/scipy/tutorial/optimize.html (дата обращения: 15.05.2024).

44. OptimizationToolbox. Nonlinear Programming [Электронный ресурс] : Optimization Toolbox. URL: https://www.mathworks.com/products/optimization.html#nonlinear-programming (дата обращения: 20.05.2024).

45. Paul R. P. Robot Manipulators: Mathematics, Programming and Control. Cambridge, Mas. MIT Press, 1981.

46. Paul R., Shimano B. Kinematic Control Equation for Simple Manipulators// IEEE Proc. of Confer, on Decision and Control. N-Y, 1979. P. 1398-1406.

47. Peter Corke, Jesse Haviland. 2021 IEEE International Conference on Robotics and Automation (ICRA 2021), May 31 - June 4, 2021, Xi'an, China, URL: https://eprints.qut.edu.au/229002/ (дата обращения: 17.05.2024).

48. Piela L., Kostrowicki J., and Scheraga H.A. The multiple-minima problem in the conformational analysis of molecules. Deformation of the potential energy hypersurface by the diffusion equation method. J. of Physical Chemistry. 1989, no. 93, pp. 3339–3346.

49. Robotics System Toolbox. Inverse Kinematics [Электронный ресурс]: Inverse Kinematics. URL: https://www.mathworks.com/discovery/inverse-kinematics.html (дата обращения 20.05.2024).

50. Robotics Toolbox for Python [Электронный ресурс]: roboticstoolbox-python 1.1.1. URL: https://pypi.org/project/roboticstoolbox-python/ (дата обращения 17.05.2024).

51. Silvia Minosse, Martina Favetta. Comparison of the Gait Biomechanical Constraints in Three Different Type of Neuromotor Damages. Frontiers in Human Neuroscience. March 2022, V. 16.

52. Snyman A.J. A new and dynamic method for unconstrained optimization. Applied Mathematical Modelling, 1982, vol. 6, pp. 449–462.

53. Suman Dutta. Optimization in Chemical Engineering. s.l. Cambridge Univ. Press, 2015. DOI: 10.1017/CBO9781316134504.