Федеральное государственное автономное образовательное учреждение
высшего образования «Московский физико-технический институт
(национальный исследовательский университет)»

На правах рукописи

Тормагов Тимофей Алексеевич

Задачи полуопределенного программирования в спутниковой навигации и планировании путей колесных роботов

Специальность 2.3.1 —
«Системный анализ, управление и обработка информации, статистика»

Диссертация на соискание ученой степени
кандидата физико-математических наук

Научный руководитель:
доктор физ.-мат. наук, профессор
Рапопорт Лев Борисович

Москва — 2026

Оглавление

Стр.

Стр.

Стр.

Введение

Актуальность темы исследования. В данной диссертационной работе рассматриваются подходы к решению реальных практических задач, возникающих в спутниковой навигации и в планировании путей колесных роботов, основанные на переходе к задачам полуопределенного программирования.

В стандартной постановке задачи полуопределенного программирования состоят в минимизации (или максимизации) линейной функции на ограничениях, заданных линейными матричными неравенствами. Полуопределенное программирование является одним из классов задач выпуклой оптимизации, включающим в себя классы задач линейного программирования и конического программирования второго порядка. Свойство выпуклости задач полуопределенного программирования позволяет использовать для их численного решения вычислительно эффективные алгоритмы. Задачи полуопределенного программирования возникают в различных областях техники, например, в автоматическом управлении и обработке сигналов. Однако они редко формулируются напрямую в стандартной постановке. Часто задачи, возникающие на практике, свойством выпуклости не обладают. Это относится в первую очередь к задачам дискретного программирования, а также к некоторым задачам оптимизации с вещественными переменными. Численные методы решения таких задач отличаются высокой трудоемкостью, подчас экспоненциальной. В приложениях широкое распространение получил подход, называемый выпуклой релаксацией. Этот подход состоит в таком ослаблении ограничений исходной задачи, что допустимая область задачи с ослабленными ограничениями (релаксированной задачи) оказывается выпуклой. Вместо исходной задачи рассматривается релаксированная задача, для решения которой могут использоваться вычислительно эффективные методы. Часто они имеют полиномиальную трудоемкость: число операций полиномиально зависит от размерности задачи. При этом для релаксированной задачи минимизации (максимизации) мы получим значение целевой функции равное либо меньшее (большее), чем у исходной. Практическую значимость имеет оценка разницы между этими значениями.

В настоящее время активно развиваются существующие глобальные спутниковые навигационные системы и появляются новые. При совместном использовании сигналов нескольких глобальных навигационных спутниковых

систем (ГНСС), таких, как ГЛОНАСС, GPS, Бэйдоу и Galileo, для навигационного приемника число видимых спутников может достигать нескольких десятков. При этом сложность решения задачи точного позиционирования экспоненциально зависит от числа обрабатываемых сигналов. Актуальной является задача выбора ограниченного числа видимых спутников, геометрическая конфигурация которых обеспечивает наилучшую точность позиционирования. До последнего времени эта задача не имела метода решения с гарантированной оценкой точности, несмотря на огромное количество опубликованных эвристических алгоритмов. Для определения относительной ориентации твердого тела методом G.Wahba [1] с помощью спутниковых навигационных измерений многоантенного приемника необходим выбор конфигурации базовых линий – набора векторов, попарно соединяющих антенны ГНСС. Одним из критериев выбора может быть число обусловленности матрицы базовых линий, используемой при определении относительной ориентации твердого тела указанным способом. Задачи выбора базовых линий и выбора подмножества видимых спутников являются комбинаторными и могут быть решены точно с использованием перебора. Однако размер множества перебора для ряда случаев не позволяет решать указанные задачи в режиме реального времени на навигационных приемниках. В связи с этим на практике строится приближенное решение. В данной диссертационной работе получен алгоритм выбора спутников с двусторонней оценкой гарантированной точности приближенного решения.

Область применения разработок данного исследования по планированию путей – точное земледелие. В точном земледелии предполагается использование точных навигационных измерений, геоинформационных систем, мониторинга урожайности в целях повышения производительности. Точное земледелие позволяет эффективно использовать посевные площади и сокращать время выполнения работ. Одним из современных направлений развития точного земледелия являются автономные колесные сельскохозяйственные роботы. Эти машины могут проводить посадку растений, мониторинг их состояния, опрыскивание от вредителей, внесение удобрений и сбор урожая. Для работы автономных сельскохозяйственных машин требуются измерения позиции (как правило, с сантиметровой точностью) и относительной ориентации. Достижение требуемой точности позиционирования возможно, например, при использовании ГНСС в фазово-дифференциальном режиме позиционирования в реальном времени (Real Time Kinematic, RTK) [2] с наземной базовой станцией для

передачи поправок. Для обеспечения работы автономных роботов на сельскохозяйственном поле требуется с использованием собранных данных строить траектории и расписания их движения. При решении задач планировании путей должны учитываться такие факторы, как рельеф поля, наличие препятствий, ограничение на нормальную кривизну траектории движения сельскохозяйственных машин. Предложенный в диссертационной работе метод позволяет с учетом этих факторов строить пути, покрывающие поле, с помощью решения задач конического программирования второго порядка.

Степень научной разработанности темы. Практическое применение задач полуопределенного программирования и линейных матричных неравенств рассматривается в ряде работ, в частности по системам автоматического управления [3; 4], синтезу цифровых фильтров [5; 6] и спутниковой навигации [7; 8]. Для задачи выбора спутников ранее предложены методы построения приближенного решения на основе набора правил выбора по углу возвышения и азимуту (совместные работы F. Meng, B. Zhu и S. Wang [9; 10]), анализа вклада отдельных спутников в точность позиционирования (квазиоптимальный алгоритм Ch. Park и J. How [11], рекурсивный алгоритм M. Liu, M.-A. Fortin и Jr. R. Landry [12], алгоритм G. Li с соавторами [13]), использования нейронных сетей (работа J. Wei с соавторами [14]). Разработаны методы определения относительной ориентации твердых тел в пространстве с помощью спутниковой навигации при заранее определенных базовых линиях на основе решения задачи G. Wahba [1] с применением SVD-разложения (изложен в работе C. E. Cohen [15]), полуопределенного программирования (предложен Л. Б. Рапопортом [16]). Планирование путей, полностью покрывающих ограниченный участок поверхности, в литературе рассматривается отдельно для плоских [17—19] и для трехмерных рельефов (серии работ J. Jin и L. Tang, I. A. Hameed и других авторов [20—24]). Для оптимизации покрытия предложены различные виды целевых функций [20], учитывающих число разворотов, время разворота между рядами, водную эрозию почвы, кривизну получаемых путей. Для учета препятствий при построении путей разработаны методы, основанные на декомпозиции рабочей области [25] (трапециевидная декомпозиция [26], декомпозиция бустрофедон [27], декомпозиция Морзе [28]), поиске на графе (алгоритмы A* [29] и D* [30]), деформации путей с применением штрафных функций (искусственных потенциальных полей, работы J. Chuang [31], Л. Б. Рапопорта и Р. Ф. Гилимьянова [32]). Также А. В. Пестере-

вым и Р. Ф. Гилимьяновым рассмотрены особенности применения B-сплайнов для планирования путей колесных роботов [33].

Объектом исследования являются оптимизационные задачи, возникающие в спутниковой навигации и ее применении к планированию путей колесных роботов.

Предмет исследования — формализация и способы решения оптимизационных задач, возникающих при применении методов спутниковой навигации к управлению колесными роботами.

Целью исследования является построение вычислительно эффективных методов решения следующих задач:

- выбора оптимального множества навигационных сигналов, используемых в позиционировании;
- выбора базовых линий, используемых при определении относительной ориентации твердого тела по измерениям от нескольких навигационных антенн;
- планирования путей с ограниченной нормальной кривизной, покрывающих заданный ландшафт с препятствиями.

Соответствие области исследования научной специальности. Работа соответствует следующим направлениям исследований, указанным в паспорте специальности 2.3.1. Системный анализ, управление и обработка информации, статистика (физико-математические науки).

п. 2. Формализация и постановка задач системного анализа, оптимизации, управления, принятия решений, обработки информации и искусственного интеллекта.

п. 4. Разработка методов и алгоритмов решения задач системного анализа, оптимизации, управления, принятия решений, обработки информации и искусственного интеллекта.

Положения, выносимые на защиту.

1. Формализация релаксированной задачи выбора подмножества видимых спутников ограниченного размера по критерию параметра геометрического снижения точности по местоположению и времени в виде полуопределенного программирования (соответствует п.2 паспорта специальности).

2. Метод и алгоритм выбора спутников, используемых при вычислении позиции, с двусторонней оценкой точности (соответствует п.4 паспорта специальности).
3. Метод и алгоритм оценки оптимального выбора базовых линий для определения относительной ориентации твердого тела в пространстве на основе спутниковой навигации (соответствует п.4 паспорта специальности).
4. Формализация построения покрытия заданного трехмерного ландшафта путями в виде задачи конического программирования второго порядка (соответствует п.2 паспорта специальности).
5. Формализация деформации путей для обхода препятствий колесным роботом с ограничением на кривизну реализуемой траектории в виде задачи конического программирования второго порядка (соответствует п.2 паспорта специальности).
6. Метод построения покрытия трехмерного ландшафта путями колесных роботов с механизмом руления поворотом передних колес (соответствует п.4 паспорта специальности).

Научная новизна исследования заключается в том, что рассматриваемые практические задачи формализованы в виде задач полуопределенного программирования, что позволило построить вычислительно эффективные алгоритмы их решения. Для выбора множества навигационных спутников новизна также заключается в том, что построен алгоритм с двусторонней оценкой точности. Для построения путей условия на нормальную кривизну траектории сформулированы в виде конусных ограничений, что позволяет строить покрытие заданного ландшафта решением задач выпуклой оптимизации.

Достоверность полученных в исследовании результатов подтверждается использованием строгого математического аппарата, а также результатами вычислительных экспериментов.

Теоретическая значимость работы заключается в математической формализации нескольких задач навигации и точного земледелия с использованием полуопределенного программирования, формализации ограничений на нормальную кривизну траектории в виде конусных ограничений, построении двусторонних оценок точности решения задач выбора спутников и конфигурации базовых линий.

Практическая значимость работы состоит в разработке алгоритмов решения в режиме реального времени ряда задач, возникающих в спутниковой навигации и планировании путей колесных роботов. При этом вычислительная эффективность предложенных алгоритмов достигается за счет использования полуопределенного программирования.

Апробация работы. Результаты данной диссертационной работы докладывались автором на научном семинаре «Автоматическое управление» (Москва, ИПУ РАН, в 2021 году и в 2025 году), на научном семинаре международной общественной организации «Академия навигации и управления движением» и журнала «Гироскопия и навигация» (Санкт-Петербург, в 2022 году), на Международной конференции Optimization and Applications (Черногория, Петровац, OPTIMA-2019 и OPTIMA-2021), на Международной научной конференции «Устойчивость и колебания нелинейных систем управления (конференция Пятницкого)» (Москва, ИПУ РАН, STAB-18, STAB-20 и STAB-22), на 13-ом Всероссийском совещании по проблемам управления (Москва, ИПУ РАН, ВСПУ-2019), на Всероссийской научной конференции МФТИ (Долгопрудный, МФТИ, в 2018 году, в 2020 году и в 2021 году).

Публикации. Результаты диссертационного исследования опубликованы в 17 научных работах. В их число входят три статьи [34—36] в рецензируемых научных изданиях категории К1 Перечня ВАК по специальности 2.3.1 (физ.-мат.) и в приравниваемых к ним научных изданиях, индексируемых международными наукометрическими базами данных. Шесть докладов в сборниках материалов научных конференций [37—42] индексируются системой Scopus. Остальные работы [43—50] опубликованы в иных сборниках материалов научных конференций.

Личный вклад автора. Методы, алгоритмы и формализации задач, полученные в данном исследовании, разработаны автором лично. Автором доказаны сформулированные в диссертации математические утверждения, реализованы основанные на них алгоритмы и выполнялась экспериментальная проверка результатов с помощью компьютерного моделирования.

Объем и структура работы. Диссертация состоит из введения, четырех глав и заключения. Полный объем диссертации составляет 123 страницы, включая 39 рисунков и две таблицы. Список литературы содержит 119 наименований.

Диссертация организована следующим образом. В главе 1 приведены в общем виде формулировки задач полуопределенного программирования и частично-целочисленного полуопределенного программирования, обсуждается вопрос о выпуклой релаксации комбинаторных задач. Эти сведения являются необходимыми для решения ряда задач спутниковой навигации и планирования путей, обсуждаемых в главах 2–4, с помощью полуопределенного программирования. Глава 2 посвящена задаче поиска оптимального рабочего созвездия навигационных спутников для позиционирования, а глава 3 — оптимального выбора базовых линий при определении относительной ориентации твердого тела в пространстве методами спутниковой навигации. В главе 4 обсуждается задача построения путей ограниченной нормальной кривизны для колесных роботов, покрывающих заданную поверхность.

Глава 1. Полуопределенное программирование

1.1 Формулировка задачи полуопределенного программирования

Будем обозначать как $\mathrm{trace}(A)$ след (сумму диагональных элементов) некоторой квадратной матрицы A, пространство вещественных симметричных матриц размера $n \times n$ — как $\mathbb{S}^n$, пространство n-мерных векторов с действительными компонентами — как R^n. Для единичной матрицы размерности $n \times n$ будем использовать обозначение I_n.

Приведем определения неотрицательно определенной и положительно определенной матриц в соответствии с [51; 52].

Определение 1.1. *Матрица $A \in \mathbb{S}^n$ называется неотрицательно определенной, если для любого вектора $x \in R^n$ выполняется $x^T Ax \geqslant 0$.*

Определение 1.2. *Матрица $A \in \mathbb{S}^n$ называется положительно определенной, если для любого ненулевого вектора $x \in R^n$ выполняется $x^T Ax > 0$.*

Чтобы показать, что матрица A положительно определена, будем использовать обозначение $A \succ 0$; неотрицательно определена — $A \succeq 0$. Для собственных значений неотрицательно определенной матрицы можно привести следующую теорему.

Теорема 1.1. [51; 53] *Матрица $A \in \mathbb{S}^n$ неотрицательно определена тогда и только тогда, когда все ее собственные числа неотрицательны.*

Для двух матриц M и N одинакового размера $a \times b$ их скалярное произведение $\langle M, N \rangle$ определяется в соответствии с [54] выражением

$$\langle M, N \rangle = \mathrm{trace}(M^T N) = \sum_{i=1}^{a} \sum_{j=1}^{b} m_{ij} n_{ij}, \tag{1.1}$$

где m_{ij} и n_{ij} — элементы матриц M и N. Причем из (1.1) следует, что

$$\langle M, N \rangle = \mathrm{trace}(M^T N) = \mathrm{trace}(N^T M) = \langle N, M \rangle, \tag{1.2}$$

$$\langle M^T, N^T \rangle = \langle M, N \rangle = \mathrm{trace}(M^T N) = \mathrm{trace}(N M^T) = \langle N, M \rangle. \tag{1.3}$$

Из равенства (1.3), в частности, следует, что сомножители под знаком $\mathrm{trace}(\cdot)$ можно менять местами. Скалярное произведение квадратной матрицы M с самою собой соответствует квадрату нормы Фробениуса для M:

$$\|M\|_F^2 = \langle M, M \rangle. \tag{1.4}$$

Линейными матричными неравенствами (англ. *linear matrix inequality, LMI*) обычно называют условия вида

$$F(x) = F_0 + x_1 F_1 + \cdots + x_n F_n \succeq 0 \tag{1.5}$$

где матрицы F_0, F_1, $\ldots$, F_n являются симметричными, $x = (x_1, \ldots, x_n)^T$, $x \in R^n$ — переменная.

Далее будем обозначать как $\max\limits_x f(x)$ максимальное значение функции $f(x)$, как $\min\limits_x f(x)$ минимальное значение функции $f(x)$, где x – переменная. В задачах поиска максимального или минимального значения функции будем подразумевать, что оптимальное значение переменной также требуется найти.

Приведем в соответствии с [54—57], следующие две постановки задачи полуопределенного программирования (англ. *semidefinite programming, SDP*).

Задача 1.1. *Найти*

$$\min_X \langle C, X \rangle \tag{1.6}$$

при ограничениях

$$\langle A_i, X \rangle = b_i,\ i = 1, \ldots, n, \tag{1.7}$$

$$X \succeq 0, \tag{1.8}$$

где $C \in \mathbb{S}^m$, $A_i \in \mathbb{S}^m$, $b_i \in R$, $i = 1, \ldots, n$, $X \in \mathbb{S}^m$.

Задача 1.2 (двойственная к задаче 1.1). *Найти*

$$\max_u b^T u \tag{1.9}$$

при ограничениях

$$C - \sum_{i=1}^{n} u_i A_i \succeq 0, \tag{1.10}$$

где $C \in \mathbb{S}^m$, $A_i \in \mathbb{S}^m$, $i = 1, \ldots, n$, $b \in R^n$, $u \in R^n$.

В постановке 1.1 требуется найти неотрицательно определенную матрицу X, удовлетворяющую ограничениям (1.7), для которой значение $\mathrm{trace}(C^T X)$ минимально. В постановке 1.2 нужно максимизировать линейную функцию $b^T u$

при ограничениях, заданных линейными матричными неравенствами (1.10). Задача 1.2 является двойственной к 1.1, доказательство приведено, например, в работе [54]. Очевидно, что заменой переменных можно перейти от задачи максимизации 1.2 к следующей задаче минимизации, также относящейся к полуопределенному программированию.

Задача 1.3 (SDP). *Найти*

$$\min_x c^T x \tag{1.11}$$

при ограничениях

$$F_0 + x_1 F_1 + \cdots + x_n F_n \succeq 0, \tag{1.12}$$

где $F_i \in \mathbb{S}^m$, $i = 0, \ldots, n$, $c \in R^n$, $x \in R^n$.

Отметим, что совместные задачи полуопределенного программирования являются выпуклыми: критерий оптимизации в них — линейная функция, а ограничения задают выпуклое множество. В несовместных задачах множество допустимых значений переменных, задаваемое ограничениями задачи, является пустым.

1.2 Подклассы задач полуопределенного программирования

1.2.1 Линейное программирование

Задачи линейного программирования (англ. *linear programming, LP*) можно представить в следующем виде [55; 57; 58].

Задача 1.4 (LP). *Найти*

$$\min_x c^T x \tag{1.13}$$

при ограничениях

$$Ax = b, \tag{1.14}$$

$$Gx \geqslant h, \tag{1.15}$$

где $G \in R^{m \times n}$, $A \in R^{l \times n}$, $c \in R^n$, $b \in R^l$, $h \in R^m$, $x \in R^n$, *неравенство* (1.15) *покомпонентное.*

Ограничение вида равенство (1.14) можно представить в виде двух неравенств вида (1.15). В тоже время, можно получить представление задачи 1.4 в виде 1.3 (полуопределенного программирования). Для этого нужно в задаче 1.3 взять F_0 диагональной с элементами вектора $-h$, а каждую матрицу F_i, где $i = 1, \ldots, n$, задать в виде диагональной матрицы с элементами из i-й строки матрицы G. Таким образом, задачи линейного программирования входят в класс задач полуопределенного программирования.

1.2.2 Коническое программирование второго порядка

Здесь и далее обозначение $\|\cdot\|$ будем использовать для евклидовой нормы вектора: для $u \in R^n$ выполняется $\|u\| = \sqrt{u^T u}$. Приведем общую формулировку задач конического программирования второго порядка (англ. *second-order cone programming, SOCP*) в соответствии с работой [59].

Задача 1.5 (SOCP). *Найти*

$$\min_x f^T x \tag{1.16}$$

при ограничениях

$$\|A_i x + b_i\| \leqslant c_i^T x + d_i, i = 1, \ldots, N, \tag{1.17}$$

где $x \in R^n$, $f \in R^n$, $A_i \in R^{(n_i-1)\times n}$, $b_i \in R^{n_i-1}$, $c_i \in R^n$, $d_i \in R$, $i = 1, \ldots, N$, N *— число ограничений,* n_i *— размерность* i*-ого ограничения.*

При $n_i = 1$, $i = 1, \ldots, N$, задача 1.5 относится к линейному программированию. Поэтому класс задач конического программирования включает в себя задачи линейного программирования. Конус второго порядка размерности k можно определить следующим образом.

$$\left\{ \begin{pmatrix} u \\ t \end{pmatrix} \middle| \, u \in R^{k-1}, \, t \in R, \, \|u\| \leqslant t \right\}. \tag{1.18}$$

Заметим, что для произвольного вектора $u \in R^n$ условие $\|u\| \leqslant t$ эквивалентно линейному матричному неравенству

$$\begin{pmatrix} tI_n & u \\ u & t \end{pmatrix} \succeq 0. \tag{1.19}$$

Это означает, что ограничения (1.17), задающие конуса второго порядка, могут быть представлены в виде линейных матричных неравенств. Тогда задачи конического программирования второго порядка являются частными случаями задач полуопределенного программирования.

1.3 Частично-целочисленное полуопределенное программирование

Приведем формулировку задачи частично-целочисленного полуопределенного программирования (англ. *mixed-integer semidefinite programming, MISDP*) [60].

Задача 1.6 (MISDP). *Найти*

$$\max_{u} b^T u \tag{1.20}$$

при ограничениях

$$C - \sum_{i=1}^{n} u_i A_i \succeq 0, \tag{1.21}$$

$$u_i \in \mathbb{Z}\ \forall i \in M, \tag{1.22}$$

где $C \in \mathbb{S}^m$, $A_i \in \mathbb{S}^m$, $i = 1, \ldots, n$, $b \in R^n$, $u \in R^n$, $M \subseteq \{1, \ldots, n\}$.

Формировка задачи 1.6 отличается от 1.2 только тем, что область допустимых значений для переменных, индексы которых входят в множество M, ограничена условием (1.22) целыми числами. Частыми случаями являются задачи целочисленного линейного программирования (англ. *integer linear programming*), смешанного линейного программирования (англ. *mixed-integer linear programming, MILP*), частично-целочисленного конического программирования второго порядка (англ. *mixed-integer second order cone programing, MISOCP*). Для всех этих задач область допустимых значений в общем случае является невыпуклым множеством.

Решение задачи 1.6 может предполагать перебор всевозможных значений целочисленных переменных с последующим решением задач полуопредленного программирования относительно вещественных переменных. Однако таким способом его не всегда можно получить за приемлемое время. Сократить время решения, как правило, позволяют приведенные в работе [60] алгоритмы на основе выпуклой релаксации, метода ветвей и границ.

1.4 Выпуклая релаксация оптимизационных задач

Релаксацией называется подход к решению задачи, заключающийся в ослаблении ее ограничений (расширении области допустимых значений переменных) для получения нижних или верхних оценок ее решения. Целью релаксации может быть сведение невыпуклой задачи к выпуклой, допускающей эффективное решение. После решения релаксированной задачи можно сделать следующие выводы.

1. Если релаксированная задача несовместна, то и исходная задача несовместна. В таком случае область допустимых значений обеих задач — пустое множество. Алгоритмы решения выпуклых задач часто позволяют сделать вывод о их несовместности.
2. Если оптимальное решение релаксированной задачи удовлетворяет ограничениям исходной, то оно является оптимальным для обеих задач.
3. Если решение релаксированной задачи минимизации (максимизации) существует, то оно является нижней (верхней) оценкой оптимального значения критерия оптимизации для решения исходной задачи.

Приведем один из классических примеров применения полуопределенной релаксации к комбинаторной оптимизации — решение задачи о максимальном разрезе графа (англ. *MAX CUT*) [51; 54; 58; 61]. Предлагается рассмотреть полный взвешенный неориентированный граф $G(V,\, E)$ с множествами вершин $V = \{1,\, \ldots,\, n\}$ и ребер между ними E. Вес ребра $(i,\, j) \in E$ задается элементом a_{ij} симметричной матрицы A. Пусть V_c — подмножество множества вершин V. Под разрезом графа относительно V_c понимается множество ребер из E, таких, что один из их концов лежит в V_c, а второй — нет. Требуется найти такой разрез графа $G(V,\, E)$, что сумма весов ребер разреза будет максимальной. Максимальный разрез может быть найдет как решение задачи комбинаторной оптимизации поиска

$$\max_{x \in \{-1,1\}^n} \sum_{i,\, j,\, i<j} a_{ij} \frac{1 - x_i x_j}{2}. \tag{1.23}$$

Решение интерпретируется следующим образом: если вершина i принадлежит множеству V_c, то $x_i = 1$, иначе $x_i = -1$. Тогда максимальный разрез — множество всех ребер (i, j), для которых $x_i x_j = -1$. При этом целевая функция

(1.23) построена так, что ее значение для оптимального решения равно сумме весов ребер максимального разреза. Также ее можно преобразовать следующим образом [54]

$$\sum_{i,\,j,\,i<j} a_{ij}\frac{1-x_ix_j}{2} = \sum_{i,\,j} a_{ij}\frac{x_ix_i - x_ix_j}{4} = x^TCx =$$

$$= \operatorname{trace}(x^TCx) = \operatorname{trace}(Cxx^T) = \langle C, xx^T\rangle, \tag{1.24}$$

где $C = 0{,}25\,(\operatorname{diag}(A\bar{e}) - A))$, $\bar{e}$ — вектор из единиц размера n. При этом матрица xx^T неотрицательно определена, ее ранг равен единице: $\operatorname{rank}(xx^T) = 1$. Рассматривается следующая эквивалентная задача оптимизации.

Задача 1.7 (о максимальном разрезе графа). *Найти*

$$\max_X \langle C, X\rangle \tag{1.25}$$

при ограничениях

$$X_{ii} = 1,\ i = 1,\ \ldots,\ n,\ X \succeq 0, \tag{1.26}$$

$$\operatorname{rank}(X) = 1. \tag{1.27}$$

Задача (1.25) с ограничением (1.26) относится к полуопределенному программированию и является полуопределенной релаксацией задачи (1.25) с ограничениями (1.26) и (1.27). При этом существует оценка точности: теорема Геманса-Виллиамсона [61] утверждает, что значение критерия оптимизации для решения задачи (1.23) лежит в диапазоне от γf до f, где f — значение критерия для решения релаксированной задачи (1.25) с ограничением (1.26), $\gamma \approx 0{,}87856$.

1.5 Методы решения задач полуопределенного программирования

Для того, чтобы задачи полуопределенного программирования можно было применять на практике, как правило, требуются численные методы их решения. Приведем примеры таких методов. Часть из них предназначены для задач более общих классов задач, включающих в себя полуопределенное программирование.

Метод эллипсоидов, который предложили Д.Б. Юдин и А.С. Немировский [62], Н.З. Шор [63], основан на использовании отсекающей гиперплоскости и генерации последовательности эллипсоидов уменьшающегося объема. Приведем

алгоритм метода эллипсоидов в соответствии с [3; 58], который предназначен для минимизации выпуклой непрерывной функции $f(x)$ на выпуклом компактном множестве Q:

$$\min_{x \in Q} f(x). \tag{1.28}$$

Алгоритм является итеративным. Эллипсоидом на k-ой итерации алгоритма является выпуклое множество вида

$$\mathcal{E}_k = \left\{ x \in R^n \,\middle|\, (x - c_k)^T H_k^{-1} (x - c_k) \leqslant 1 \right\}, \tag{1.29}$$

где $c_k \in R^n$ — центр эллипсоида, $H_k \in \mathbb{S}^n$. Полуоси эллипсоида соответствуют собственным векторам H_k, их длины — собственным значениям H_k. На нулевой итерации выбирается точка c_0 и эллипсоид $\mathcal{E}_0$, гарантированно содержащий оптимальное решение. Таким является евклидов шар с центром c_0, содержащий все множество Q: $H_0 = R^2 I_n$, R — радиус шара. На каждой итерации (с номером $k+1$) генерируется эллипсоид $\mathcal{E}_{k+1}$ с центром в точке

$$c_{k+1} = c_k - \frac{1}{n+1} \frac{H_k g_k}{\sqrt{g_k^T H_k g_k}} \tag{1.30}$$

и матрицей

$$H_{k+1} = \frac{n^2}{n^2 - 1} \left(H_k - \frac{2}{n+1} \frac{H_k g_k g_k^T H_k}{g_k^T H_k g_k} \right), \tag{1.31}$$

где в случае $c_k \notin Q$ вектор g_k — отсекающая гиперплоскость, такая, что оптимальное решение лежит в полупространстве $\left\{ x \,\middle|\, (x - c_k)^T g_k \leqslant 0 \right\}$, а в случае $c_k \in Q$ вектор g_k— один из субградиентов целевой функции в c_k. На каждой итерации получается эллипсоид $\mathcal{E}_{k+1}$ меньшего, чем $\mathcal{E}_k$ объема, который включает в себя половину $\mathcal{E}_k$, отделенную секущей плоскостью и включающей оптимальное решение. Можно показать (см. [58]), что объем эллипсоида $\mathcal{E}_k$ не превышает $\exp(-0{,}5k/n)$ от объема $\mathcal{E}_0$, что подтверждает сходимость метода.

На практике, как указано в [3], более эффективными, чем метод эллипсоидов, оказываются методы внутренней точки. Они нацелены на решение задач конического программирования, которые в общем виде можно записать как [58]

$$\min_{x \in K \times R^p} \langle c, x \rangle \tag{1.32}$$

при ограничении $Ax = b$, где A — матрица размера $m \times n$, $b \in R^m$, $K \subset R^{n-p}$ — замкнутый выпуклый конус, который не содержит прямых и внутренность которого непустое множество. Методы внутренней точки генерируют последовательность итераций во внутренности выпуклого конуса. В основе метода лежит использование самосогласованных функций, введенных Ю.Е. Нестеровым и А.С. Немировским, и самосогласованных барьеров для ограничений задачи, см. [52]. Как правило, методы внутренней точки сочетают шаги по методу Ньютона и следование центральному пути. Подробное описание метода внутренней точки можно найти в книге [64], в применении к задачам полуопределенного программирования — в работе [65].

Метод множителей переменного направления (англ. *alternating direction method of multipliers, ADMM*) [66] позволяет разделять вычисления по переменным. Задача оптимизации представляется в виде

$$\min_{x \in R^n,\, z \in R^m} \; f(x) + g(z) \tag{1.33}$$

при ограничении

$$Ax + Bz = c, \tag{1.34}$$

где $f(x)$ и $g(z)$ — выпуклые функции, $A \in R^{p \times n}$, $B \in R^{p \times m}$, $c \in R^p$. Для этой задачи строится модифицированная функция Лагранжа

$$L_{\rho}(x, z, y) = f(x) + f(z) + y^T(Ax + Bz - c) + \frac{\rho}{2}\|Ax + Bz - c\|^2, \tag{1.35}$$

где ρ — неотрицательный параметр. На каждой k-й итерации ADMM вычисляется приближение для следующей итерации $(x_{k+1}, z_{k+1}, y_{k+1})$ по следующему алгоритму.

1. Определяется x_{k+1} как решение задачи

$$\min_{x \in R^n} \; L_{\rho}\left(x,\, z_k,\, y_k\right). \tag{1.36}$$

2. Определяется z_{k+1} как решение задачи

$$\min_{z \in R^m} \; L_{\rho}\left(x_{k+1},\, z,\, y_k\right). \tag{1.37}$$

3. Вычисляется y_{k+1} по формуле

$$y_{k+1} = y_k + \rho\left(Ax_{k+1} + Bz_{k+1} - c\right). \tag{1.38}$$

Для решения прямой и двойственной задач 1.1 и 1.2 полуопределенного программирования также были предложены прямой и двойственный симплекс-методы, описание которых можно найти в [54]. Однако, в отличие от линейного программирования, допустимое множество задач полуопределенного программирования может иметь бесконечное число крайних точек, что вызывает сложности со сходимостью симплекс-методов. В работе [67] предложен рандомизированный подход к решению задач полуопределенного программирования для поиска оптимального значения критерия, использующий отсекающую гиперплоскость.

Компьютерные программы для решения задач полуопределенной оптимизации доступны в виде оптимизационных пакетов программ и библиотек. В оптимизационном пакете SDPA (от англ. *SemiDefinite Programming Algorithm*) [68—71] доступно решение задач полуопредленного программирования с помощью прямо-двойственного метода внутренний точки, поддерживаются версии для программирования на языках C++ и Python, в среде Matlab. Для Matlab также доступны пакеты SDPT3 [72; 73] и SeDuMi [74], реализующие разные варианты прямо-двойственных методов внутренней точки. В пакете SCS (англ. *Splitting Conic Solver*) [75; 76] доступно решение задач конического программирования. Существуют реализации библиотек SCS для C/C++, Python, Julia, R, Matlab и Ruby. Алгоритм в SCS использует, в частности, метод ADMM и позволяет решать задачи большой размерности, с достаточно большим количеством переменных и ограничений. Для задач конического программирования второго порядка разработан также встраиваемый пакет ECOS [77], использующий метод внутренней точки со схемой предиктор-корректор и логарифмическими барьерными функциями для конусов. Его особенность заключается в независимости от внешних библиотек (например, он не требует LAPACK и BLAS). Код реализован на C, но также имеются интерфейсы для его использования на Python, Julia, R и Matlab. ECOS показывает хорошие результаты по времени работы для задач небольшой размерности, в работе [77] рассмотрен тест времени работы ECOS для задачи оптимизации инвестиционного портфеля, в которой он показал время, на порядок меньшее, чем у ряда распространенных алгоритмов решения задач полуопределенного и конического программирования.

Большинство оптимизационных пакетов программ требуют представление задачи в строго определенном виде. Иногда это заметно усложняет написание кода компьютерных программ, использующих решение задач полуопределен-

ного программирования. При проведении экспериментов или компьютерного моделирования часто удобно использовать вспомогательные библиотеки, позволяющие представить оптимизационную задачу и ее ограничения с помощью заданного набора конструкций. Примерами таких библиотек являются CVX (для Matlab) и CVXPY (для Python) [78] и YALMIP [79]. CVX и CVXPY используют подход DCP (англ. *Disciplined Convex Programming* — управляемое выпуклое программирование) [80] к формированию задач в некоторой заданной форме. В DCP программа формируется по определенным правилам, позволяющим, например, проверить достаточные условия ее выпуклости.

1.6 Выводы к главе 1

Приведены постановки оптимизационных задач следующих классов.

1. Полуопределенного программирования (SDP).
2. Линейного программирования (LP).
3. Конического программирования второго порядка (SOCP).
4. Частично-целочисленного полуопределенного программирования (MISDP).

При этом выполнено соотношение вложенности между этими классами задач

$$\text{LP} \subset \text{SOCP} \subset \text{SDP} \subset \text{MISDP}. \tag{1.39}$$

MISDP расширяют класс задач полуопределенного программирования на случай целочисленных переменных. Для любой задачи SOCP существует эквивалентная задача полуопределенного программирования. Ограничения задач линейного программирования представимы в виде условия принадлежности конусу размерности 1. Обсужден вопрос о выпуклой релаксации, приведен классический пример полуопределенной релаксации — решение задачи о максимальном разрезе графа, имеющее гарантированную оценку точности решения. Приведены сведения о численных методах решения задач полуопределенного программирования и их реализациях в виде компьютерных программ.

Сведения, содержащиеся в главе 1, являются необходимыми для описания решения задач спутниковой навигации и планирования путей с помощью полуопределенного программирования и полуопределенной релаксации.

Глава 2. Задача выбора навигационных спутников

2.1 Постановка задачи

Глобальные навигационные спутниковые системы (ГНСС, англ. Global Navigation Satellite Systems — GNSS) позволяют производить позиционирование приемников спутниковых навигационных сигналов. Примерами ГНСС являются GPS (США), ГЛОНАСС (Россия) и Бэйдоу (Китай) и Galileo (Европейское космическое агентство). В настоящее время эти системы активно развиваются. Навигационные приемники могут принимать сигналы сразу нескольких систем, при этом общее число одновременно принимаемых сигналов может достигать нескольких десятков. Использование большого числа спутников, как правило, позволяет повысить точность определения координат. При этом использование избыточного числа спутников приводит к значительному увеличению времени решения навигационных уравнений приемником сигнала. По этой причине возникает задача выбора подмножества навигационных спутников ограниченного размера.

Навигационные спутники могут передавать сигналы в разных полосах частот, находящихся, как правило, внутри L-диапазона радиоволн (1–2 ГГц). Например, в системе GPS для разных целей выделяют несущие частоты (и диапазоны) L1, L2, L5; в ГЛОНАСС — L1, L2, L3. В данной работе рассматривается случай, когда приемник получает сигналы L1, которые в настоящее время используются в большей части приемников гражданской навигации.

Для координат спутников и антенны приемника будем использовать глобальную геоцентрическую систему координат ECEF (от англ. Earth-fixed coordinate system). ECEF представляет собой правую декартову систему координат (X, Y, Z) с началом координат в центре Земли. Ось Z направлена к географическому северному полюсу, ось X – к точке пересечения нулевого меридиана и экватора, ось Y направлена так, чтобы оси X, Y, Z в данном порядке составляли правую тройку векторов.

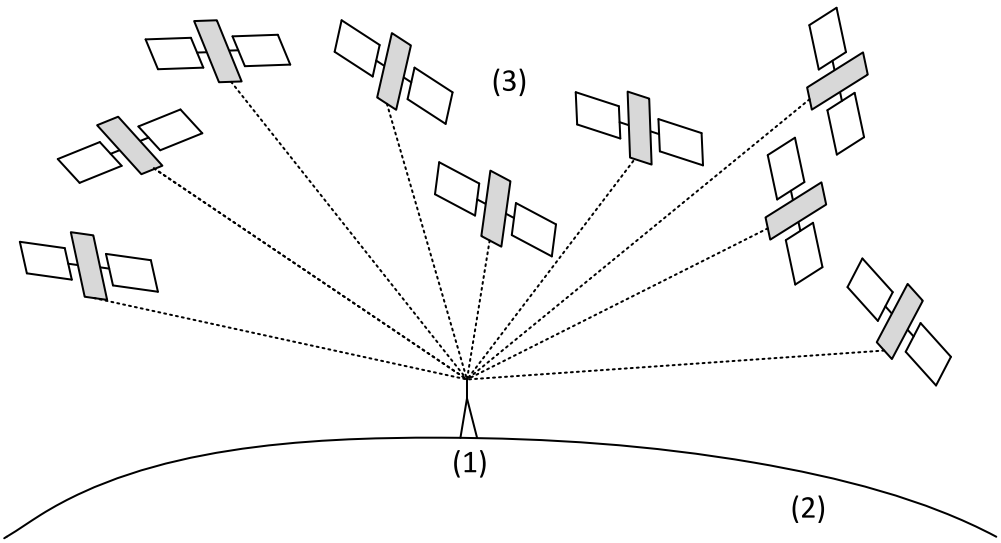


Рисунок 2.1 — Прием сигналов антенной навигационного приемника (1), находящегося на поверхности Земли (2) от группы спутников (3).

Предположим, что навигационный приемник использует псевдодальномерный метод позиционирования и за некоторый интервал измерения получает сигналы от n спутников (рисунок 2.1), относящихся к J ГНСС. Для систем будем использовать индексы $j = 1, \ldots, J$, а для отдельных спутников — индексы $s = 1, \ldots, n$. Обозначим $j(s)$ — индекс (от 1 до J) спутниковой системы, к которой принадлежит спутник с индексом s. Определяется псевдодальность p_s каждого навигационного спутника s (см. [2]):

$$p_s = c(t_{\text{прием.}} - t_{\text{передат.}}), \tag{2.1}$$

где c — скорость света, $t_{\text{прием.}}$ — показание часов приемника при получении сигнала и $t_{\text{передат.}}$ — показание часов передатчика (спутника) в момент отправки сигнала. При этом не учитывается несинхронность внутренних часов спутника и приемника.

Для используемого набора спутников справедлива система из уравнений вида

$$p_s = ||\mathbf{r} - \mathbf{r}_s|| + c\Delta t_{j(s)} + \varepsilon_s,\ s = 1, \ldots, n, \tag{2.2}$$

где $\mathbf{r}$ — координата антенны приемника, $\mathbf{r}_s$ — координата спутника s, $||\mathbf{r} - \mathbf{r}_s||$ — евклидова норма разности этих векторов, $\Delta t_{j(s)}$ — смещение часов для навигационной системы, в которую входит спутник s, c — скорость света в вакууме, ε_s — ошибка измерения псевдодальности спутника s. Поскольку система навигационных уравнений (2.2) нелинейная, то, как правило, она решается (в смысле

минимизации взвешенной суммы квадратов ошибок измерений псевдодальностей) итеративно, например, методом Гаусса-Ньютона [81]. Решение системы уравнений (2.2) относительно координаты приемника и смещений часов навигационных систем имеет смысл только в случае $n \geqslant J + 3$.

Линеаризуем систему уравнений (2.2). Пусть $\hat{\mathbf{r}}$ – оценка координаты антенны приемника, сделанная на предыдущей итерации (на первой итерации можно выбрать нулевое решение). Тогда направляющий вектор $\mathbf{h}_s$ спутника s вычисляется по формуле

$$\mathbf{h}_s = \frac{\hat{\mathbf{r}} - \mathbf{r}_s}{||\hat{\mathbf{r}} - \mathbf{r}_s||}. \tag{2.3}$$

Координаты вектора $\mathbf{h}_s$ являются направляющими косинусами спутника. Обозначим $\Delta p_s = p_s - ||\hat{\mathbf{r}} - \mathbf{r_s}||$, $\mathbf{e}_j$ — вектор размерности J, для которого координата j равна единице, а остальные — нулю. Получаем систему линейных уравнений

$$\begin{pmatrix} \Delta p_1 \\ \Delta p_2 \\ \dots \\ \Delta p_n \end{pmatrix} = \begin{pmatrix} \mathbf{h}_1^T & \mathbf{e}_{j(1)}^T \\ \mathbf{h}_2^T & \mathbf{e}_{j(2)}^T \\ \dots & \dots \\ \mathbf{h}_n^T & \mathbf{e}_{j(n)}^T \end{pmatrix} \times \begin{pmatrix} \mathbf{r} - \hat{\mathbf{r}} \\ c\Delta t_1 \\ \dots \\ c\Delta t_J \end{pmatrix} + \begin{pmatrix} \boldsymbol{\nu}_1 \\ \boldsymbol{\nu}_2 \\ \dots \\ \boldsymbol{\nu}_n \end{pmatrix}, \tag{2.4}$$

где компоненты $\boldsymbol{\nu}_1$, $\boldsymbol{\nu}_2$, …, $\boldsymbol{\nu}_n$ отвечают нормально распределенной ошибке с нулевым средним. Обозначим

$$\Delta\mathbf{x} = \Big(\mathbf{r} - \hat{\mathbf{r}},\ \ c\Delta t_1,\ \ \dots,\ \ c\Delta t_J\Big)^T, \tag{2.5}$$

$$\Delta\mathbf{p} = \Big(\Delta p_1,\ \ \Delta p_2,\ \ \dots,\ \ \Delta p_n\Big)^T, \tag{2.6}$$

$$\mathbf{v} = \Big(\boldsymbol{\nu}_1,\ \ \boldsymbol{\nu}_2,\ \ \dots,\ \ \boldsymbol{\nu}_n\Big)^T, \tag{2.7}$$

$$H = \begin{pmatrix} \mathbf{h_1}^T & \mathbf{e}_{j(1)}^T \\ \mathbf{h_2}^T & \mathbf{e}_{j(2)}^T \\ \dots & \dots \\ \mathbf{h_n}^T & \mathbf{e}_{j(n)}^T \end{pmatrix}. \tag{2.8}$$

Тогда систему уравнений (2.4) можно записать в виде

$$\Delta\mathbf{p} = H\Delta\mathbf{x} + \mathbf{v}. \tag{2.9}$$

Из-за ошибок измерений системы (2.4) и (2.9) могут не иметь решений. Поэтому на практике находят и используют в дальнейших расчетах значение $\Delta\mathbf{x}$, при котором квадрат евклидовой нормы вектора ошибки $\mathbf{v}$ минимален:

$$\Delta\widetilde{\mathbf{x}} = (H^T H)^{-1} H^T \Delta\mathbf{p}. \tag{2.10}$$

Точность позиционирования зависит от геометрического расположения спутников. Кроме того, отличие часов для разных ГНСС требует решение навигационных уравнений для J дополнительных временных переменных. Для того, чтобы оценивать влияние этих факторов, для множества используемых в данный момент спутников определяют параметр GDOP (от англ. Geometric dilution of precision), который вычисляется по формуле

$$\mathrm{GDOP} = \sqrt{\mathrm{trace}((H^T H)^{-1})}. \tag{2.11}$$

GDOP отражает для выбранного множества спутников суммарное снижение точности позиционирования антенны приемника по местоположению и времени. При отсутствии информации об ошибках измерений, обусловленных другими причинами (ошибки эфемерид, многолучевость), из двух множеств видимых спутников, возможно пересекающихся, для позиционирования следует выбрать то, GDOP которого меньше.

Использование сигналов от большого числа ГНСС спутников, как правило, позволяет минимизировать GDOP. Однако, как отмечено выше, сложность решения задачи позиционирования экспоненциально зависит от числа учитываемых спутников.

Рассмотрим задачу выбора не более m спутников из n возможных (где $m \leqslant n$). Введем n-мерный вектор x, такой, что

$$x = \Big(x_1,\ \ x_2,\ \ \ldots,\ \ x_n\Big)^T, \tag{2.12}$$

где компонента x_s принимает значение 1, если спутник s используется для решения задачи позиционирования, и 0 — если не используется. Обозначим

$$\mathrm{diag}(x) = \begin{pmatrix} x_1 & 0 & \cdots & 0 \\ 0 & x_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \ldots & x_n \end{pmatrix}. \tag{2.13}$$

Множество выбранных спутников ГНСС также называется рабочим созвездием. GDOP рабочего созвездия может быть вычислен по формуле

$$\mathrm{GDOP} = \sqrt{\mathrm{trace}((H^T \mathrm{diag}(x) H)^{-1})}. \tag{2.14}$$

Чтобы решение системы навигационных уравнений соответствовало позиции приемника, необходимо выбрать не менее $p = J + 3$ спутников. Учитывая, что квадратный корень является монотонной функцией своего аргумента, получаем следующую оптимизационную задачу.

Задача 2.1 (выбора спутников). *Найти*

$$\min_x \mathrm{trace}((H^T \mathrm{diag}(x) H)^{-1}), \tag{2.15}$$

при ограничениях

$$x_s \in \{0,1\},\ s = 1,\ 2, \ldots, n, \tag{2.16}$$

$$p \leqslant \sum_{s=1}^{n} x_s \leqslant m, \tag{2.17}$$

$$\sum_{s:\ j(s)=j} x_s \geqslant 1,\ j = 1,\ \ldots, J. \tag{2.18}$$

Формулировка задачи 2.1 предполагает, что ее нужно решать отдельно для спутников каждой навигационной системы и для всех комбинаций выбора ГНСС. Причем, при решении для нескольких ГНСС нужно потребовать выбор хотя бы одного спутника каждой из систем (условия (2.18)). В противном случае матрица $H^T \mathrm{diag}(x) H$ вырождается и ее нельзя обратить. Соблюдение условий (2.18) позволяет предотвратить появление нулевых столбцов матрицы (2.8) для каждого возможного выбора спутников.

2.2 Оценка вычислительной сложности

Задача 2.1 относится к комбинаторной оптимизации. Число возможных значений переменных, удовлетворяющих ее ограничениям (2.16) и (2.17), конечно и равно $\sum_{k=p}^{m} C_n^k$, где C_n^k – число сочетаний, определяемое формулой $C_n^k = \frac{n!}{k!(n-k)!}$. Точное решение задачи 2.1 может быть получено, например, полным перебором.

Однако нет необходимости при переборе рассматривать все возможные варианты. Как доказано в работе [82], в случае, когда используется только одна ГНСС система, при добавлении новых спутников GDOP всегда уменьшается. Тогда для одной ГНСС для оптимального решения задачи 2.1 выбирается ровно m спутников и можно ограничиться рассмотрением только C_n^m возможных комбинаций.

Для случая нескольких ГНСС множество перебора можно сократить, если использовать следующие утверждения об изменении GDOP параметра при добавлении спутников, доказанные в работе [83]. Если спутники некоторой ГНСС уже содержатся в наборе, что добавление еще одного спутника из этой системы уменьшает GDOP. Если к набору добавляется один спутник из ГНСС, которая еще не использовалась для позиционирования, то GDOP увеличивается; но при добавлении двух таких спутников об увеличении или об уменьшении GDOP определенно сказать нельзя.

В качестве примера рассмотрим случай позиционирования с использованием двух ГНСС систем. Пусть число видимых спутников каждой из систем n_1 и n_2 соответственно, $n_1 + n_2 = n$, $n > m$. Если $4 \leqslant n_1 \leqslant m$, то при выборе спутников только из первой ГНСС достаточно рассмотреть только один вариант, в котором выбираются все спутники, поскольку при исключении любых из них GDOP будет увеличиваться. Если $n_1 < 4$, позиционирование с использованием спутников только первой ГНСС невозможно. Если $n_1 > m$, то требуется перебор $C_{n_1}^m$ вариантов. Аналогичные выводы можно сделать о выборе спутников только из второй ГНСС.

Каждый элемент множества перебора нужно оценить, определив GDOP (или его квадрат) для соответствующего выбора спутников. Если вычислять GDOP по формуле (2.14), то для этого требуется обращение матрицы размерности $p \times p$. Значительное число обращений матриц требует вычислительных ресурсов. Помимо этого существуют аналитические формулы вычисления GDOP, не требующие нахождения обратной матрицы. Например, в работе [84] для случая одной ГНСС ($p = 4$) предложена формула

$$GDOP = \sqrt{(0.5h_1^3 - 1.5h_1h_2 + h_3)/(3h_4)}, \tag{2.19}$$

где $M = H^T H$, $h_1 = \mathrm{trace}(M)$, $h_2 = \mathrm{trace}(M^2)$, $h_3 = \mathrm{trace}(M^3)$, $h_4 = \det(M)$. Использование формулы (2.19) позволяет незначительно сократить число операций с плавающей точкой при расчете GDOP. В работе [85] рассмотрены

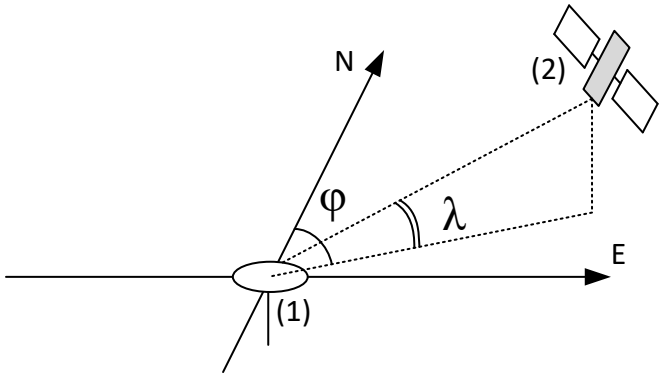


Рисунок 2.2 — Азимут φ и угол возвышения λ спутника (2) относительно навигационной антенны (1), плоскость локального горизонта с направлениями N (север) и E (восток).

алгоритмы эффективного вычисления GDOP при использовании нескольких ГНСС. В [14] предлагается использовать обобщенно – регрессионную нейронную сеть (GRNN, [86]) для вычисления оценки GDOP.

2.3 Существующие методы решения

Большое количество операций делает невозможным решение задачи выбора спутников с помощью полного перебора в режиме реального времени. В связи с этим на практике используются приближенные алгоритмы, применение которых в общем случае не гарантирует оптимальность выбора по критерию (2.1). Приведем примеры таких алгоритмов.

Местоположение спутника ГНСС относительно антенны приемника можно характеризовать лучом, выпущенным из точки наблюдения на спутник. Этот луч задается либо углом φ с направлением на север (азимутом) и углом возвышения λ (рисунок 2.2) над плоскостью локального горизонта, либо направляющими косинусами вектора, направленного от навигационного приемника на спутник. Как видно из формул (2.3), (2.8) и (2.11), GDOP зависит только от направлений на спутники и того, к каким ГНСС они относятся. Таким образом, входными данными для алгоритма выбора спутников могут быть как их направляющие косинусы, так и азимуты с углами возвышения.

В работе [87] приведен метод выбора $m = 4$ спутников одной ГНСС (GPS), основанный на вычислении объема V тетраэдра, вершины которого получаются

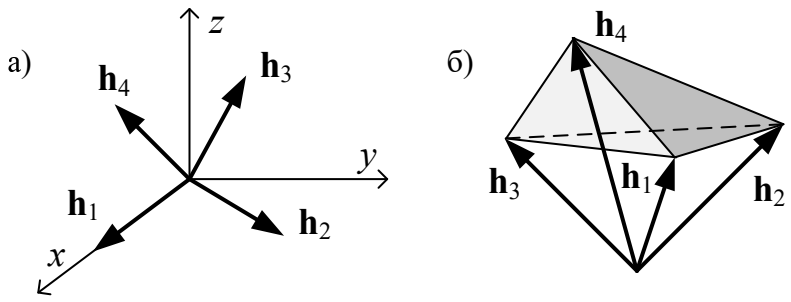


Рисунок 2.3 — Определение тетраэдра (а) и системы координат (б) для алгоритма на основе объема многогранника.

с помощью направляющих векторов спутников. Матрица H для случая четырех спутников будет квадратной с размером 4×4. Показано, что V прямо пропорционален $\det H$. Также указано на взаимосвязь между значениями V и GDOP. Доказано, что минимальное значение GDOP для 4 спутников достигается, когда углы между их направляющими векторами составляют 109,47°. Предложен следующий алгоритм выбора спутников.

1. В качестве первого выбирается спутник с максимальным углом возвышения λ. Направляющий вектор этого спутника $\mathbf{h}_1$.
2. Второй спутник выбирается так, чтобы угол между векторами $\mathbf{h}_1$ и $\mathbf{h}_2$ был близок по значению к 109,47°.
3. Третий спутник выбирается так, чтобы принимала максимальное значение функция

$$f(l_3, m_3, n_3) = \frac{1 - l_3}{6} \left(\sqrt{\frac{2(1 - l_2)(1 + l_3)}{(1 - l_2 l_3 - m_2 m_3)^{-1}}} + |m_2 n_3| \right), \qquad (2.20)$$

   где $(l_2, m_2, 0)$, (l_3, m_3, n_3) – координаты направляющих векторов $\mathbf{h}_2$ и $\mathbf{h}_3$ второго и третьего спутников в правой декартовой системе координат $(x,\ y,\ z)$ (рисунок 2.3а), начало которой совпадает с антенной приемника, ось x сонаправлена $\mathbf{h}_1$, а вектор $\mathbf{h}_2$ лежит в плоскости $(x,\ y)$.
4. Четвертый спутник (с направляющим вектором $\mathbf{h}_4$) выбирается так, чтобы максимизировать объем тетраэдра (рисунок 2.3б), вершинами которого служат концы векторов $\mathbf{h}_1$, $\mathbf{h}_2$, $\mathbf{h}_3$, $\mathbf{h}_4$.

Асимптотическую сложность алгоритма можно оценить как $O(n)$, перебор на каждом шаге идет только по оставшимся спутникам, а не по их комбинациям. Однако данный алгоритм предназначен для выбора только четырех спутников. Как отмечено в обзоре [88], четырех спутников на практике может быть недостаточно для обеспечения надежности и точности позиционирования. Метод,

основанный на вычислении объема тетраэдра для двух ГНСС (GPS и Beidou), предложен в работе [89].

В работе [11] рассмотрен квазиоптимальный алгоритм, использующий упрощенную целевую функцию. Пусть $\theta_{i,j}$ – угол между направляющими векторами $\mathbf{h}_i$ и $\mathbf{h}_j$ спутников i и j, $i = 1, \ldots, n$, $j = 1, \ldots, n$. Для каждого спутника i определяется вес по формуле $J_i = \sum_{j=1}^{n} \cos(2\theta_{i,j})$. Из набора исключается спутник с наибольшим весом. Затем веса спутников определяются заново. Для этого достаточно вычесть из каждого J_i слагаемое, относящееся к исключенному спутнику. Процедура повторяется до тех пор, пока не останется ровно m спутников. Основная идея заключается в том, что если $\mathbf{h}_i$ и $\mathbf{h}_j$ перпендикулярны, то $\cos(2\theta_{i,j})$ принимает минимальное значение, равное -1. Если же они сонаправлены и, вероятно, не вносят существенного вклада в точность позиционирования, то значение $\cos(2\theta_{i,j})$ максимальное, равное 1. В данном алгоритме значительно сокращено число операций по сравнению с полным перебором, он может быть использован в системе реального времени. Однако в 24-часовом тесте, приведенном в работе [9], среднеквадратичная ошибка GDOP составила $\Delta_{rms} = 1{,}0099$ для $m = 6$. Также при тестировании в отдельных случаях выявлена ошибка GDOP на 2,5 единицы.

В работе [12] введен рекурсивный квазиоптимальный метод выбора спутников, основанный на эвристическом предположении о том, что оптимальное множество из m спутников целиком содержится в оптимальном из $m + 1$ спутника. Выбор спутников предлагается проводить по следующему итеративному алгоритму, где M — текущее множество спутников, k — его размер.

1. Определяются начальные условия: M — множество всех видимых спутников, $k = n$.
2. Из подмножеств множества M с размером $k - 1$ выбирается то, GDOP которого минимален.
3. M заменяется на выбранное подмножество, k уменьшается на единицу.
4. Если $k > m$, то осуществляется возврат на п. 2.

Для данного алгоритма размер множества перебора существенно меньше, чем у полного перебора: на каждой итерации проводится перебор только k рабочих созвездий, $k \leqslant n$, а число итераций $n-m$. При этом также не требуется перемножать матрицы геометрии для каждого рабочего созвездия. Пусть $Q_k = H_k^T H_k$, где H_k состоит из строк матрицы H, соответствующих спутникам выбранного рабочего созвездия из k элементов. Если исключить из набора спутник s, то

приходим к матрице

$$Q_{k-1} = Q_k - w^T w, \tag{2.21}$$

где w — строка спутника s в матрице H_k:

$$w = \left(\mathbf{h_s}^T \quad \mathbf{e^T_{j(s)}} \right). \tag{2.22}$$

Матрицы Q_k и Q_{k-1} имеют размерность $p \times p$, где $p = J+3$. На каждой итерации требуется сравнивать значения квадрата GDOP, равного $\mathrm{trace}(Q_{k-1}^{-1})$.

Алгоритмы выбора рабочего созвездия, основанные на исключении спутников, могут быть построены и другим способом. В работе [13] рассматривается исключение избыточных спутников, не вносящих заметного вклада в GDOP. Обозначая $G_k = Q_k^{-1}$ и используя формулу Шермана—Моррисона (см. [90]), из (2.21) получаем

$$G_{k-1} = (H_k^T H_k - w^T w)^{-1} = G_k + G_k w^T (1 - w G_k w^T)^{-1} w G_k. \tag{2.23}$$

Тогда изменение квадрата GDOP рабочего созвездия при исключении спутника s составляет $\mathrm{trace}(G_k w^T w G_k / \alpha)$, где $\alpha = 1 - w G_k w^T$. При этом α является скалярной величиной. В приближенном алгоритме предполагается исключение спутников, для которых величина $\mathrm{trace}(G_k w^T w G_k / \alpha)$ не превышает некоторого заранее заданного значения, например 0,15. Размер рабочего созвездия с течением времени для данного алгоритма может изменяться. На практике, GDOP созвездий всех видимых спутников и рабочего отличаются незначительно, а значит, как правило, исключаются избыточные измерения.

Приближенный алгоритм определения рабочего созвездия также может представлять собой набор применяемых последовательно правил выбора спутников по информации об их угле возвышения λ и азимуте φ. Для двух ГНСС (Beidou и GPS) в работе [9] предложен такой алгоритм для $m = 6$. В нем по углу возвышения λ видимые спутники классифицируются на высокие ($65^\circ \div 90^\circ$), средние ($30^\circ \div 65^\circ$) и низкие ($5^\circ \div 30^\circ$). Спутники, находящиеся близко к горизонту, для которых $\lambda < 5^\circ$, не рассматриваются. Первым выбирается спутник, наиболее близкий к зениту (с максимальным углом возвышения λ), вторым — наиболее близкий к горизонту (с минимальным углом возвышения λ, таким, что $\lambda \geqslant 5^\circ$). Третий спутник выбирается в зависимости от того, сколько низких спутников доступны для позиционирования.

1. Если низких спутников меньше двух, то выбирается средний спутник с наименьшим λ, азимут которого отличается от азимута второго спутника более чем на 60°.
2. Если низких спутников два, то выбирается оставшийся при условии, что его азимут отличается от азимута второго спутника более чем на 60°.
3. Если низких спутников больше двух, то выбирается низкий спутник, для которого разница азимута со вторым будет максимальной.

При этом, если не удается выбрать третий спутник по заданному набору правил, то выбирается тот, для которого разница азимута со вторым наибольшая. Четвертый спутник выбирается так, чтобы GDOP рабочего созвездия из четырех спутников был минимален. Точно также выбираются последовательно пятый и шестой. Средняя абсолютная ошибка (MAE) GDOP в 24-часовом тесте составила 0,3026, среднеквадратичная (MSE) 0,1119. При этом для теста решение почти всегда не совпадает с оптимальным. Модификация данного алгоритма для трех ГНСС (GPS, Galileo и Beidou) предложена в работе [10].

В работе [91] предложен алгоритм, использующий данные об оптимальном рабочем созвездии предыдущих навигационных эпох. Он построен на предположении, что переходе от одной эпохи к другой оптимальное подмножество может отличаться лишь на небольшое число спутников.

2.4 Предлагаемый метод решения

2.4.1 Релаксация бинарных условий

В задаче 2.1 выбора m спутников к невыпуклой области допустимых значений приводят ограничения (2.16), которые требуют бинарных значений переменных $x_s \in \{0;\ 1\}$, $s = 1, 2, \ldots, n$ для решения задачи. Рассмотрим линейную релаксацию (ослабление ограничений), заключающуюся в замене условий (2.16) на $x_s \in [0;\ 1]$, $s = 1, 2, \ldots, n$ и приводящую к следующей задаче оптимизации.

Задача 2.2 (релаксированная задача выбора спутников). *Найти*

$$\min_x \operatorname{trace}((H^T \operatorname{diag}(x) H)^{-1}) \tag{2.24}$$

при ограничениях (2.18),

$$x_s \in [0,1],\ s = 1,\ 2, \ldots, n, \tag{2.25}$$

$$p \leqslant \sum_{s=1}^{n} x_s \leqslant m. \tag{2.26}$$

Лемма 2.1. *Задача оптимизации 2.2 является выпуклой.*

Доказательство. Пусть $M \subset R^n$ — множество допустимых значений x для задачи 2.2, которое не является пустым. Каждое ограничение вида (2.25), как и условие (2.26), описывает выпуклое множество. Тогда M — выпуклое множество, поскольку оно образовано пересечением выпуклых множеств. Рассмотрим критерий оптимизации (2.24). Покажем, что эпиграф (надграфик) функции

$$\varphi(x) = \operatorname{trace}((H^T \operatorname{diag}(x) H)^{-1}), \tag{2.27}$$

определяемый как

$$\operatorname{epi} \varphi = \{x, t | x \in M, \varphi(x) \leqslant t\}, \tag{2.28}$$

является выпуклым множеством. Запишем эпиграф в эквивалентном виде

$$\operatorname{epi} \varphi = \{x, t | x \in M, \sum_{j=1}^{p} e_j^T (H^T \operatorname{diag}(x) H)^{-1} e_j \leqslant t\}, \tag{2.29}$$

где $e_j \in R^p$ — единичный вектор, j-я компонента которого равна единице, а остальные — нулю. Для каждого слагаемого суммы введем дополнительную переменную q_j, $j = 1, \ldots, p$. Определим $q = (q_1,\ q_2,\ \ldots,\ q_p)^T$. Используя лемму Шура для нестрогих линейных матричных неравенств ([55; 92]), можно записать (2.29) в виде

$$\operatorname{epi} \varphi = \{x, t, q | x \in M, \sum_{j=1}^{p} q_j \leqslant t, \begin{bmatrix} H^T \operatorname{diag}(x) H & e_j \\ e_j^T & q_j \end{bmatrix} \succeq 0,\ j = 1, ..., p\}. \tag{2.30}$$

Получаем, что epi φ описывается пересечением выпуклых множеств. Поэтому он является выпуклым множеством. Тогда задача 2.2 является выпуклой. $\square$

Лемма 2.2. *Значение критерия оптимизации для оптимального решения задачи 2.2 является нижней оценкой квадрата GDOP оптимального рабочего созвездия, определяемого решением задачи 2.1.*

Доказательство. Область допустимых значений переменных задачи 2.1 целиком входит в область допустимых значений задачи 2.2. Критерии оптимизации совпадают. Это означает, что значение критерия для оптимального решения 2.2 не может быть больше квадрата GDOP оптимального рабочего созвездия. □

Леммы 2.1 и 2.2 позволяют построить нижнюю оценку GDOP оптимального рабочего созвездия с помощью решения задачи выпуклого программирования 2.2. Для того, чтобы использовать вычислительно эффективные методы решения для задачи выпуклого программирования, остается свести ее к стандартной постановке задачи некоторого класса. Рассмотрим сведение к двум задачам полуопределенного программирования и к одной — конического программирования второго порядка.

2.4.2 Сведение к задаче полуопределенного программирования

Введем дополнительную переменную — симметричную неотрицательно определенную матрицу $P \succeq 0$ с размерностью $p \times p$. Обозначим как I_p единичную матрицу размерности $p \times p$. Рассмотрим следующую задачу оптимизации.

Задача 2.3. *Найти*

$$\min_{x,P} \operatorname{trace} P \tag{2.31}$$

при ограничениях (2.18), (2.25), (2.26) *и*

$$\begin{bmatrix} H^T \operatorname{diag}(x) H & I_p \\ I_p & P \end{bmatrix} \succeq 0. \tag{2.32}$$

Критерий оптимизации (2.31) задачи 2.3 представляет собой линейную функцию от элементов матрицы P (сумму диагональных элементов). Ограничения (2.25) и (2.26) являются линейными неравенствами, а (2.32) — линейное матричное неравенство. Это означает, что задача 2.3 относится к классу полуопределенного программирования. Рассмотрим, как соотносятся оптимальные решения задач 2.2 и 2.3.

Лемма 2.3. *Для оптимальноˆго решения* x^*, P^* *задачи 2.3 выполняется равенство*

$$P^* = \left(H^T \mathrm{diag}(x^*) H\right)^{-1}. \tag{2.33}$$

Доказательство. Из ограничений задачи 2.3 следует неотрицательная определенность матрицы $H^T \mathrm{diag}(x^*) H$. В силу специфики выбора H при ограничениях (2.25) и (2.26) матрица $H^T \mathrm{diag}(x) H$ также будет невырожденной. По лемме Шура для нестрогих линейных матричных неравенств условие (2.32) выполняется тогда и только тогда, когда

$$P - \left(H^T \mathrm{diag}(x) H\right)^{-1} \succeq 0. \tag{2.34}$$

Поскольку след квадратной матрицы равен сумме ее собственных значений, которые для неотрицательно определенной матрицы неотрицательны, то из (2.34) следует, что

$$\mathrm{trace}\left(P - \left(H^T \mathrm{diag}(x) H\right)^{-1}\right) \geqslant 0, \tag{2.35}$$

что эквивалентно

$$\mathrm{trace}\, P \geqslant \mathrm{trace}\left(H^T \mathrm{diag}(x) H\right)^{-1}. \tag{2.36}$$

Рассмотрим оптимальное решение x^*, P^* задачи 2.3. Из (2.34) следует, что существует неотрицательно определенная матрица Z^*, такая, что

$$P^* = \left(H^T \mathrm{diag}(x^*) H\right)^{-1} + Z^*. \tag{2.37}$$

Пусть λ_1, ..., λ_p — собственные числа Z^*. Из $Z^* \succeq 0$, следует, что $\lambda_i \geqslant 0$ $\forall i = 1, ..., p$. Используя спектральное разложение симметричной матрицы, запишем Z^* в виде

$$Z^* = Q \mathrm{diag}(\lambda_1, \ldots, \lambda_p) Q^T, \tag{2.38}$$

где Q — некоторая ортогональная матрица. Для оптимального решения задачи

$$\mathrm{trace}\, P^* = \mathrm{trace}\left(H^T \mathrm{diag}(x^*) H\right)^{-1}. \tag{2.39}$$

Тогда $\mathrm{trace}\, Z^* = \sum_{i=1}^{p} \lambda_i = 0$. Используя (2.38), получаем, что Z^* — нулевая матрица. Тогда для оптимального решения x^*, P^* задачи 2.3 выполняется равенство (2.33) □

Основное значение леммы 2.3 состоит в том, оптимальные решения задач 2.3 и 2.2 по переменной $x \in R^n$ совпадают. Вместо матрицы P можно ввести набор скалярных переменных q_j, $j = 1, \ldots, p$ и рассмотреть следующую задачу.

Задача 2.4. *Найти*

$$\min_{x,q_1,\ldots,q_p} \sum_{j=1}^{p} q_j \tag{2.40}$$

при ограничениях (2.18), (2.25), (2.26) *и*

$$\begin{bmatrix} H^T \mathrm{diag}(x) H & e_j \\ e_i^T & q_j \end{bmatrix} \succeq 0,\ j = 1, ..., p, \tag{2.41}$$

где $e_j \in R^p$ *— единичный вектор,* j*-ая компонента которого равна единице, а остальные равны нулю.*

Критерий оптимизации (2.40) представляет собой линейную функцию, ограничения (2.18), (2.25) и (2.26) — линейные неравенства, (2.41) — линейные матричные неравенства. Задача 2.4, также, как и 2.3, относится к классу полуопределенного программирования.

Лемма 2.4. *Оптимальные решения задач 2.2, 2.3 и 2.4 по переменной* x *совпадают.*

Доказательство. Совпадение оптимальных решений задач 2.2 и 2.3 по x следует из леммы 2.3. Заметим, что

$$\mathrm{trace}(H^T \mathrm{diag}(x) H)^{-1} = \sum_{i=1}^{p} e_i^T (H^T \mathrm{diag}(x) H)^{-1} e_i. \tag{2.42}$$

Матрица $H^T \mathrm{diag}(x) H$ положительно определена в силу специфики задачи. Тогда по лемме Шура условие (2.41) эквивалентно

$$e_i^T (H^T \mathrm{diag}(x) H)^{-1} e_i \leqslant q_i,\ j = 1, \ldots, p. \tag{2.43}$$

Тогда критерий оптимизации (2.40) эквивалентен (2.24), причем области допустимых значений для x в задачах 2.2 и 2.4 совпадают. Тогда по переменной x совпадают оптимальные решения задач 2.2, 2.3 и 2.4. □

Совпадение оптимальных решений задач 2.2, 2.3 и 2.4 означает, что можно решить релаксированную задачу 2.2 путем решения задачи полуопределенного программирования 2.3 либо 2.4. При этом результаты будут совпадать по переменной x. Выбор той или иной задачи влияет только на вычислительную сложность решения: они отличаются размерностью переменных и ограничений.

2.4.3 Сведение к задаче конического программирования

Рассмотрим следующую задачу конического программирования второго порядка относительно переменных x_s, w_j, t_{js}, $j = 1, \ldots, p, s = 1, \ldots, n$.

Задача 2.5. *Найти*

$$\min_{x_s,\, w_j,\, t_{js},\, j=1,\ldots,p,\, s=1,\ldots,n} \sum_{j=1}^{p} \sum_{s=1}^{n} t_{js} \tag{2.44}$$

при ограничениях (2.18), (2.25), (2.26) *и*

$$H^T w_j = e_j,\ w_j = (w_{j1}, \ldots, w_{jn})^T,\ j = 1, \ldots, p, \tag{2.45}$$

$$\left\| \begin{pmatrix} 2w_{js} \\ x_s - t_{js} \end{pmatrix} \right\| \leqslant x_s + t_{js},\ j = 1, \ldots, p,\ s = 1, \ldots, n, \tag{2.46}$$

где $e_j \in R^p$ *— единичный вектор,* j*-я компонента которого равна единице, а остальные равны нулю; норма вектора в* (2.46) *евклидова.*

Покажем, что оптимальное решение $(x_1^*, \ldots, x_n^*, w_1^*, \ldots, w_p^*, t_{11}^*, \ldots, t_{pn}^*)$ задачи 2.5 может быть использовано для решения релаксированной задачи 2.2.

Сначала предположим, что

$$0 < x_s < 1, s = 1, \ldots, n. \tag{2.47}$$

В силу специфики задачи из ее ограничений следует невырожденность матрицы $H^T \mathrm{diag}(x) H$. Обозначим $X = \mathrm{diag}(x)$,

$$v_j = (H^T X H)^{-1} e_j,\ v_j \in R^p,\ j = 1, \ldots, p. \tag{2.48}$$

Используя равенство (2.42), задачу 2.2 можно сформулировать в виде поиска

$$\min_{x,\, v_1, \ldots, v_p} \sum_{j=1}^{p} v_j^T H^T X H v_j \tag{2.49}$$

при ограничениях (2.25), (2.26) и

$$v_j = (H^T X H)^{-1} e_j,\ j = 1, \ldots, p. \tag{2.50}$$

Матрица X невырождена в силу предположения (2.47). Обозначая

$$w_j = X H v_j,\ j = 1, \ldots, p, \tag{2.51}$$

получаем эквивалентную формулировку задачи 2.2 в виде поиска

$$\min_{x,\, w_j,\, j=1,\ldots,p} \sum_{j=1}^{p} w_j^T X^{-1} w_j \tag{2.52}$$

при ограничениях (2.25), (2.26), (2.51) и

$$H^T w_j = e_j,\ j = 1, \ldots, p. \tag{2.53}$$

Матрица X обратима в силу условия (2.47). Существование v_j, $j = 1, \ldots, p$, удовлетворяющих ограничениям задачи, гарантируется, поэтому условие (2.51) может быть опущено. Чтобы показать это, зафиксируем $x_s = x_s^*$, $s = 1, \ldots, n$, где x_s^* — часть оптимального решения задачи (2.52) с ограничениями (2.25), (2.26), (2.50) и (2.51), $X^* = \mathrm{diag}(x_1^*, x_2^*, \ldots, x_n^*)$. Тогда решение задачи поиска

$$\min_{w_j,\, j=1,\ldots,p} \sum_{j=1}^{p} w_j^T (X^*)^{-1} w_j \tag{2.54}$$

при ограничениях

$$H^T w_j = e_j,\ j = 1, \ldots, p, \tag{2.55}$$

полностью определяется решениям p задач поиска

$$\min_{w_j} w_j^T (X^*)^{-1} w_j \tag{2.56}$$

при

$$H^T w_j = e_j, \tag{2.57}$$

сформулированных для каждого $j = 1, \ldots, p$. Условия Каруша—Куна—Таккера (формулировку которых можно найти в [55]) для оптимального решения $w_1^*, \ldots, w_p^*$ задач вида (2.56) с ограничением (2.57) гарантируют существование векторов $\boldsymbol{\nu}_j \in R^p$, таких, что

$$\frac{1}{2}(X^*)^{-1} w_j^* + H\boldsymbol{\nu}_j = 0,\ j = 1, \ldots, p. \tag{2.58}$$

Полагая $v_j = -2\boldsymbol{\nu}_j$, мы приходим в выводу, что условия (2.51) всегда выполняются для оптимального решения задачи (2.49) с ограничениями (2.25), (2.26), и (2.53). Введем дополнительные перемененные t_{js}, $j = 1, \ldots, p$, $s = 1, \ldots, n$, такие, что

$$t_{js} \geqslant \frac{w_{js}^2}{x_s},\ j = 1, \ldots, p,\ s = 1, \ldots, n. \tag{2.59}$$

Как и в работах [55; 59], будем интерпретировать w_{js}^2/x_s при $x_s = 0$ как 0 если $w_{js} = 0$ и как $+\infty$ иначе. Тогда задачу оптимизации (2.52) с ограничениями (2.25), (2.26), и (2.53) можно записать в виде поиска

$$\min_{x_s,\, w_j,\, t_{js},\, j=1,\ldots,p,\, s=1,\ldots,n} \sum_{j=1}^{p} \sum_{s=1}^{n} t_{js} \tag{2.60}$$

при (2.25), (2.26) и (2.59). Поскольку $t_{js} \geqslant 0$, $x_s \geqslant 0$ (что следует из (2.25) и (2.59)), неравенства вида

$$\left\| \begin{pmatrix} 2w_{j,s} \\ x_s - t_{js} \end{pmatrix} \right\| \leqslant x_s + t_{js},\ j = 1, \ldots, p,\ s = 1, \ldots, n \tag{2.61}$$

эквивалентны неравенствам

$$4w_{js}^2 + (x_s - t_{js})^2 \leqslant (x_s + t_{js})^2,\ j = 1, \ldots, p,\ s = 1, \ldots, n \tag{2.62}$$

Тогда условие (2.59) можно заменить на (2.61).

При численном решении задач полуопределенного программирования часто используют следующее допущение. Строгие линейные матричные неравенства вида $M(x) \succ 0$, где x — вектор неизвестных переменных, заменяют на нестрогие $M(x) \succeq \varepsilon I$, где I — единичная матрица соответствующей размерности, ε — достаточно малое положительно число. Если задача полуопределенного программирования решается методом внутренней точки с барьерной функцией

$$\Phi(x) = -\ln \det M(x), \tag{2.63}$$

то в случае вырождения матрицы $M(x)$ барьерная функция $\Phi(x)$ принимает неограниченные значения. Минимизация суммы барьерной и целевой функции не дает приблизиться слишком близко к границе допустимой области. Близость решения к границе регулируется выбором малого ε (например, это может быть число из диапазона $10^{-8} \div 10^{-6}$). Тогда строгое неравенство (2.47) при численном решении заменяется на нестрогое:

$$\varepsilon \leqslant x_s \leqslant 1 - \varepsilon, s = 1, \ldots, n. \tag{2.64}$$

2.4.4 Алгоритм выбора спутников

Для задач полуопределенного программирования 2.3, 2.4 и 2.5 можно ограничить область допустимых значений путем введения бинарных ограничений (2.16). В таком случае получаются задачи частично-целочисленного полуопределенного программирования, и их можно использовать для оптимального выбора спутников. Точное решение этих задач можно получить, например, с использованием метода ветвей и границ. Однако на практике время, за которое требуется выбрать спутники, меньше, чем то, которое требуется для точного решения.

Приведем алгоритм для двух ГНСС (например, ГЛОНАСС и GPS), использующий предположение о близости решения исходной задачи комбинаторной оптимизации 2.1 и релаксированной задачи 2.2. Данное предположение не всегда выполняется, однако как уже было отмечено, задачи 2.3, 2.4 и 2.5 могут быть использованы для определения нижней оценки точности приближенного решения.

Алгоритм 2.1 (выбора спутников из двух ГНСС).

1. Решается релаксированная задача для всех спутников при условии, что выбирается хотя бы один спутник каждой ГНСС.
2. Определяются решения релаксированной задачи для спутников только первой и только второй ГНСС. Задача решается только в том случае, когда к выбранной ГНСС относятся хотя бы четыре спутника.
3. Для всех трех случаев выбираются m спутников с наибольшими значениями x_s для решения релаксированной задачи (выбор ближайшего подходящего решения).
4. Из рабочих созвездий, полученных на предыдущем шаге, выбирается то, GDOP которого минимален.

Алгоритм 2.1 обобщается на случай трех и более ГНСС: нужно решать релаксированную задачу для каждой ГНСС и всех комбинаций их использования. Поскольку алгоритм является приближенным, то требуется оценка гарантированной точности.

2.4.5 Двусторонняя оценка точности решения

Рассмотрим более подробно построение оценки точности приближенного решения, получаемого в приведенном выше алгоритме выбора спутников. Обозначим

$$\varphi(x) = \sqrt{\mathrm{trace}((H^T \mathrm{diag}(x) H)^{-1})}. \tag{2.65}$$

Лемма 2.5. *Пусть x^* — решение исходной комбинаторной задачи 2.1, $\widetilde{x}$ — решение релаксированной задачи 2.2, $\widehat{x}$ — результат выбора спутников по приближенному алгоритму 2.1. Тогда справедлива двусторонняя оценка*

$$\varphi(\widetilde{x}) \leqslant \varphi(x^*) \leqslant \varphi(\widehat{x}), \tag{2.66}$$

где $\varphi(x^*)$ равен GDOP оптимального рабочего созвездия, а GDOP выбранного рабочего созвездия — $\varphi(\widehat{x})$.

Доказательство. По лемме 2.2 GDOP оптимального рабочего созвездия не может быть больше, чем значение критерия (2.65) для решения релаксированной задачи:

$$\varphi(\widetilde{x}) \leqslant \varphi(x^*). \tag{2.67}$$

GDOP выбранного рабочего созвездия не может быть меньше GDOP оптимального:

$$\varphi(x^*) \leqslant \varphi(\widehat{x}). \tag{2.68}$$

Из (2.67) и (2.68) получаем двустороннюю оценку (2.66). □

Практическое значение имеет оценка точности приближенного решения в единицах GDOP. Можно привести также следующую лемму.

Лемма 2.6. *Пусть Δ — разница GDOP истинного и приближенного решения задачи выбора спутников*

$$\Delta = \varphi(\widehat{x}) - \varphi(x^*), \tag{2.69}$$

а $\widetilde{\Delta}$ — разница значений функции $\varphi(x)$ для решения релаксированной задачи 2.2 и GDOP приближенного решения

$$\widetilde{\Delta} = \varphi(\widehat{x}) - \varphi(\widetilde{x}). \tag{2.70}$$

Тогда $\widetilde{\Delta}$ – верхняя оценка величины Δ (точности решения задачи), получаемая без вычисления точного решения.

Доказательство. Вычтем из обеих частей двустороннего неравенства (2.66) значение $\varphi(\widehat{x})$. Получаем

$$\varphi(\widetilde{x}) - \varphi(\widehat{x}) \leqslant \varphi(x^*) - \varphi(\widehat{x}) \leqslant 0. \tag{2.71}$$

Тогда

$$0 \leqslant \Delta \leqslant \widetilde{\Delta}. \tag{2.72}$$

□

Заметим, что оценка $\widetilde{\Delta}$ получается в процессе работы алгоритма при решении релаксированной задачи. Если задача решается для нескольких ГНСС, то нужно выбирать максимальную оценку $\widetilde{\Delta}$, получаемую при решении релаксированных задач, описанных в алгоритме выше.

2.5 Результаты экспериментов

Экспериментальное исследование было проведено с использованием измерений, полученных с использованием ГНСС приемника в 12-часовом тесте. За время теста спутники сделали полный оборот вокруг Земли: для GPS период обращения 11 часов 58 минут, ГЛОНАСС — 11 часов 15 минут. Рассматривались только сигналы диапазона L1 систем GPS и ГЛОНАСС. Поскольку требовалось сравнение с точным решением, вычисление которого часто не может проводиться с частотой получения навигационных измерений, в рамках данного тестирования выбор спутников проводился раз в минуту.

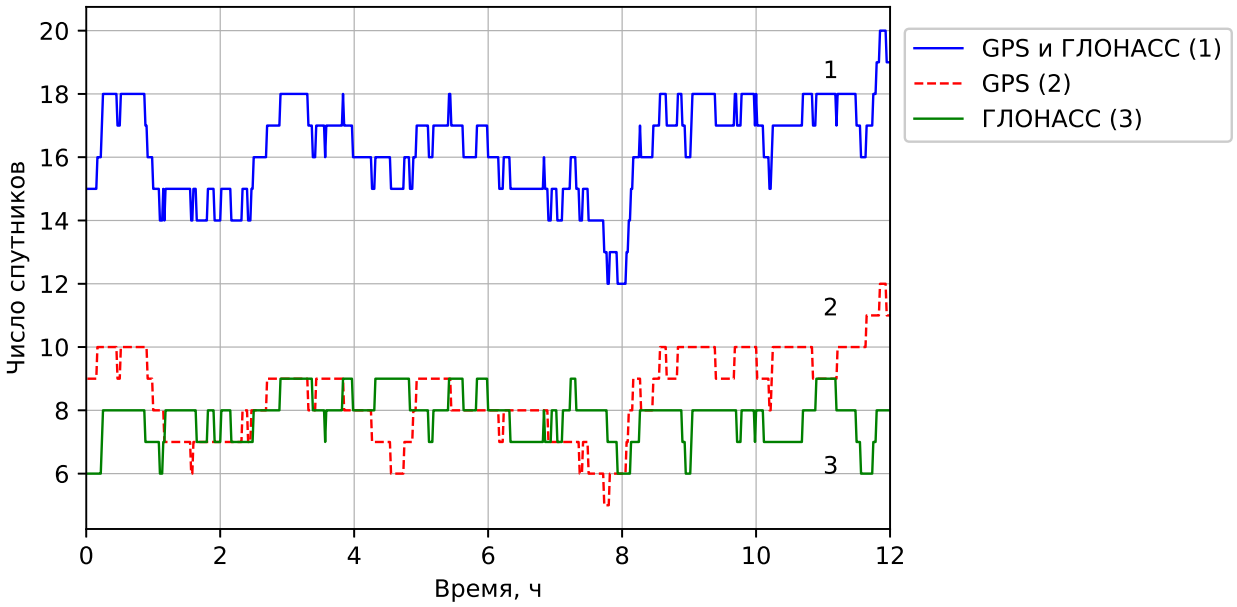


Рисунок 2.4 — Зависимость числа спутников GPS и ГЛОНАСС от времени от начала теста (навигационной эпохи).

На рисунке 2.4 показано, как изменялось число видимых спутников каждой ГНСС и их общее число в течение теста. Время указано от начала тестирования. Как видно из графика, общее число спутников n находилось в диапазоне $12 \leqslant n \leqslant 20$, число спутников GPS — в диапазоне $5 \leqslant n_1 \leqslant 12$, и ГЛОНАСС — в диапазоне $6 \leqslant n_1 \leqslant 9$. Антенна ГНСС приемника была неподвижно закреплена под открытым небом с целью принять как можно большее число навигационных сигналов.

2.5.1 Оценка точности работы алгоритма

Для оценки точности работы алгоритма вычислялись Δ (абсолютная величина разницы GDOP истинного и приближенного решения задачи выбора спутников) и $\widetilde{\Delta}$ (абсолютная величина разницы значения функции $\varphi(x)$ для решения релаксированной задачи и GDOP приближенного решения), определяемые формулами (2.69) и (2.70). Определялись $\Delta_{\max}$ и $\widetilde{\Delta}_{\max}$ — максимальные значения Δ и $\widetilde{\Delta}$ в процессе тестирования, Δ_{rms} и $\widetilde{\Delta}_{\text{rms}}$ — среднеквадратичные значения Δ и $\widetilde{\Delta}$. Также определялся процент навигационных эпох от их общего числа, для которых приближенным алгоритмом получено точное решение. Для

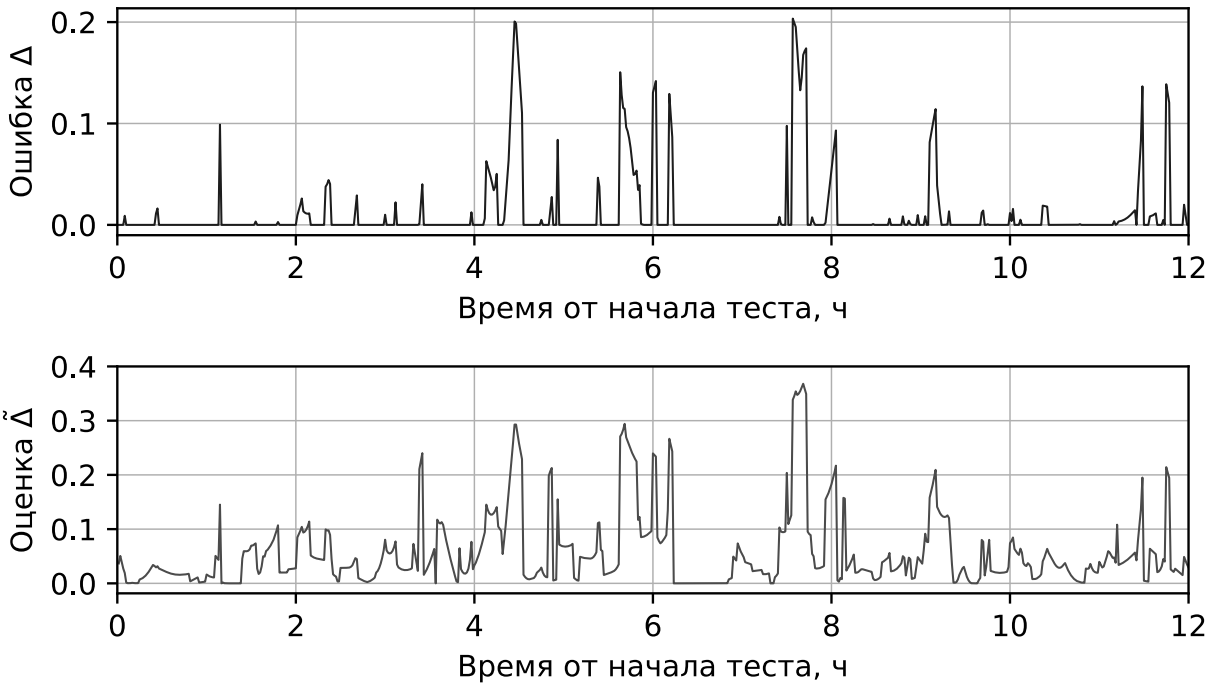


Рисунок 2.5 — Величина ошибки GDOP Δ и ее оценка $\widetilde{\Delta}$ для $m = 6$

оценки точности работы алгоритма выбора спутников использовалась задача конического программирования второго порядка 2.5, поскольку с помощью задач 2.3 и 2.4 получились эквивалентные решения. Они могут отличаться лишь из-за конечной разрядности вычислений и разных доверительных интервалов численных алгоритмов решения задач полуопределенного программирования. Такие случаи при тестировании не выявлены.

Таблица 1 — Результаты тестирования точности работы алгоритма выбора спутников.

m	Процент точных решений	$\Delta_{\max}$	Δ_{rms}	$\widetilde{\Delta}_{\max}$	$\widetilde{\Delta}_{\text{rms}}$
6	77,8	0,20	0,0364	0,37	0,0888
7	75,3	0,21	0,0355	0,29	0,0703
8	78,9	0,10	0,0117	0,15	0,0358
9	81,5	0,16	0,0150	0,19	0,0271
10	80,7	0,07	0,0070	0,09	0,0184

Результаты тестирования для разных m (ограничений на максимальный размер рабочего созвездия) представлены в таблице 1. Отметим, что предложенный приближенный алгоритм достаточно часто возвращал точное решение, причем для неточных решений ошибка $\Delta_{\max}$ не превышала 0,21, что свидетельствует о хорошей точности приближенного алгоритма. Кроме того,

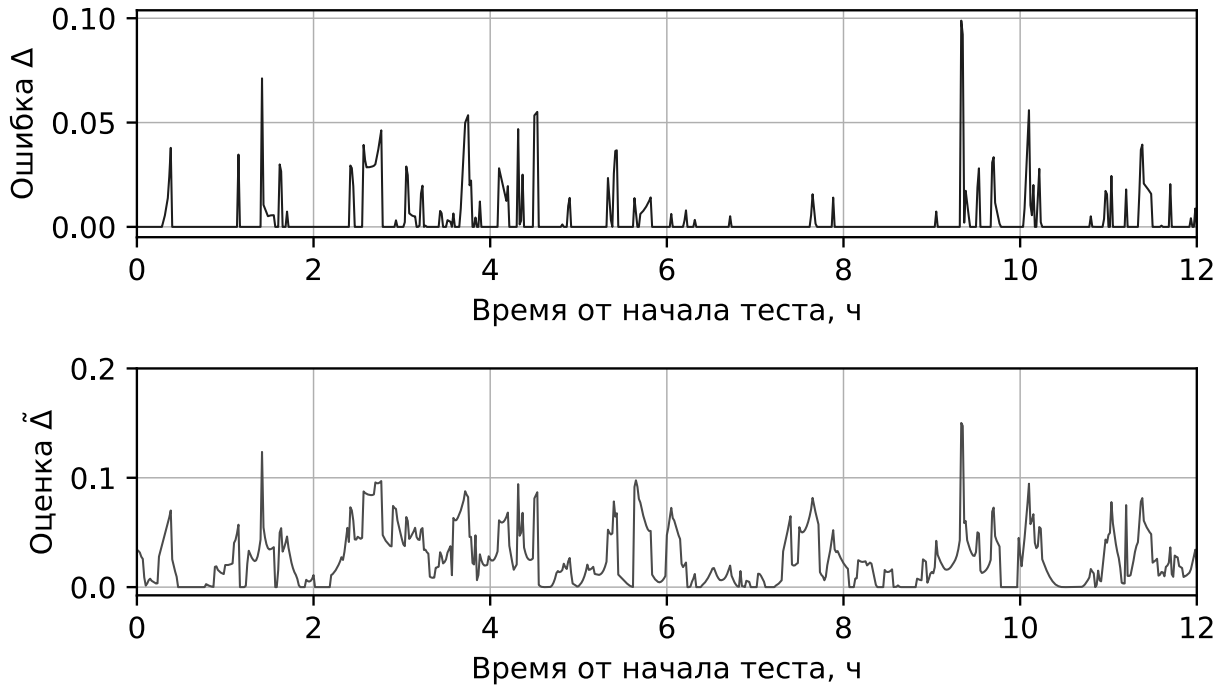


Рисунок 2.6 — Величина ошибки GDOP Δ и ее оценка $\widetilde{\Delta}$ для $m = 8$

среднеквадратичная ошибка Δ_{rms} получилась заметно меньше, чем для ряда существующих методов в аналогичных тестах других приближенных алгоритмов (например, [11], [9]; меньше, но сравнима по величине с полученной в работе [10]). Величина верхней оценки точности $\widetilde{\Delta}$ для всех навигационных эпох в тесте позволяет утверждать без вычисления точного решения путем перебора о том, что полученное в результате работы алгоритма решение достаточно близко к оптимальному.

На рисунках 2.5, 2.6 и 2.7 представлены графики для $m = 6$, 8 и 10, отражающие изменение величины ошибки Δ и ее оценки $\widetilde{\Delta}$ при переходе от одной навигационной эпохи к другой. Видно, что даже при $\Delta = 0$ оценка точности $\widetilde{\Delta}$ может отличаться от нуля. Однако, на практике это отличие оказалось незначительным.

2.5.2 Анализ результатов для отдельной эпохи

Рассмотрим результаты выбора рабочего созвездия для одной навигационной эпохи, для которой число видимых спутников было $n = 19$, из них $n_1 = 11$ относятся к GPS, $n_2 = 8$ — к ГЛОНАСС. Результаты такого тестирования

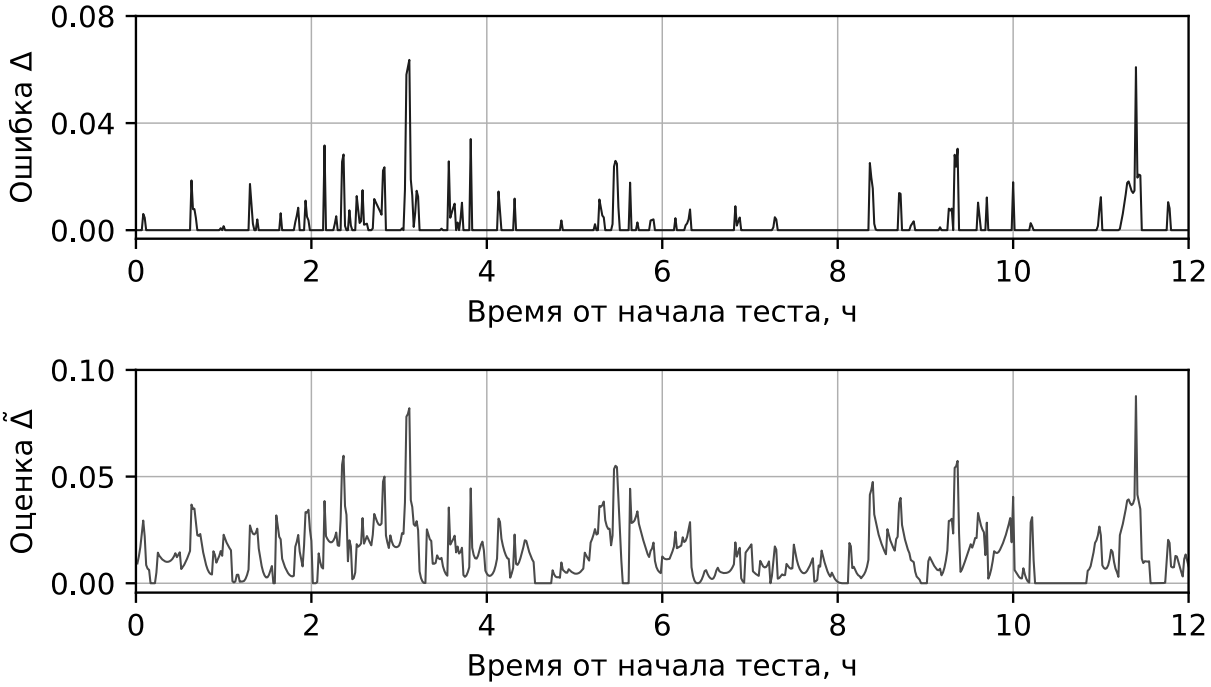


Рисунок 2.7 — Величина ошибки GDOP Δ и ее оценка $\widetilde{\Delta}$ для $m = 10$

представимы в виде графиков созвездий в полярных координатах. Угловая координатная ось на них соответствует азимутам спутников φ, $\varphi \in [0^\circ,\ 360^\circ]$, а радиальная ось — углам $90^\circ - \lambda$, где λ — угол возвышения спутника. Зениту соответствует начало координат, а горизонту — внешняя окружность графика. Закрашенным окружностям и треугольникам соответствуют выбранные спутники GPS и ГЛОНАСС соответственно, не закрашенным — не выбранные.

Для различных m вычислялось приближенное решение по предложенному алгоритму, а также наилучшее (оптимальное) и наихудшее решения по алгоритму полного перебора. Наихудшее решение соответствует ситуации, когда проводится случайный выбор созвездия размера m и выбирается рабочее созвездие с наибольшим GDOP.

В соответствии с классификацией, приведенной в работе [93], созвездие с GDOP параметром 1 можно считать идеальным, $2 \div 4$ — отличным, $4 \div 6$ — хорошим, $6 \div 8$ — удовлетворительным, $8 \div 20$ – нежелательным, а со значениями больше 20 — неудовлетворительным.

Рисунок 2.8 — Выбранное созвездие спутников для $m = 6$ в сравнении с наилучшим и наихудшим выбором

Рисунок 2.9 — Выбранное созвездие спутников для $m = 7$ в сравнении с наилучшим и наихудшим выбором

Рисунок 2.10 — Выбранное созвездие спутников для $m = 8$ в сравнении с наилучшим и наихудшим выбором

На рисунках 2.8, 2.9 и 2.10 представлены результаты выбора созвездий для $m = 6, 7, 8$. Для $m = 6$ и $m = 7$ точное и приближенное решения совпадают, а для $m = 8$ отличаются на небольшую величину. Наихудшие решения для этих случаев приводит к неудовлетворительному GDOP согласно приведенной выше классификации, а приближенное и точное решения — к идеальному GDOP. Это означает, что выбор спутников нельзя проводить случайным образом, поскольку это может привести к существенной потере точности позиционирования.

Рисунок 2.11 — Выбранное созвездие спутников для $m = 11$ в сравнении с наилучшим и наихудшим выбором

На рисунках 2.11 и 2.12 представлены результаты выбора созвездий для $m = 11$ и $m = 12$. Здесь для наихудшего решения GDOP уже классифицируется как приемлемый для позиционирования. Приближенное и точное решения не совпадают, но отличие GDOP составляет не более, чем 0,003.

Рисунок 2.12 — Выбранное созвездие спутников для $m = 12$ в сравнении с наилучшим и наихудшим выбором

В случаях $m = 6$ и $m = 7$ должны были быть выбраны спутники только одной ГНСС, а для $m = 8$, $m = 11$, $m = 12$ — двух. Это подтверждает необходимость решения трех релаксированных задач: только для ГЛОНАСС, только для GPS и для всех видимых спутников. Заметим, что GDOP оптимального рабочего созвездия, состоящего из не более чем $m + 1$ спутника не может быть больше, чем GDOP созвездия из не более чем m спутников. Причем добавление ровно одного спутника из ГНСС, ранее не использованной для позиционирования, согласно выводам работы [83], всегда увеличивает GDOP.

Кроме того, видно, что при переходе от некоторого m к $m-1$ оптимальное рабочее созвездие может отличаться более, чем на один спутник. Соответственно, рекурсивные алгоритмы, основанные на исключении ровно одного спутника при переходе от созвездия с ограничением на размер m к $m - 1$, не позволяют получить оптимальное решение для данной навигационной эпохи.

2.5.3 Оценка времени работы программы

Поскольку позиционирование в навигационном приемнике обычно проводится в режиме реального времени, то накладываются требования на скорость работы алгоритма выбора спутников. Оценка времени работы программы проводилась в описанном выше 12-часовом тесте. Для вычислений использовался центральный процессор CPU 2.60GHz Intel Core i5-3230M. Для исчерпывающего

поиска совершался полный перебор по возможным созвездиям с вычислением GDOP каждого из них. Приближенный алгоритм был реализован с использованием библиотек SCS (для задач полуопределенного программирования 2.3, 2.4) и ECOS (для задачи конического программирования второго порядка 2.5). Как было отмечено ранее, количество переменных и ограничений в оптимизационных задачах 2.3, 2.4 и 2.5 различно, а значит, может отличаться и время их решения.

На рисунках 2.13, 2.14, 2.15 для $m = 6, 8, 10$ переставлены графики зависимости времени принятия решения о выборе рабочего созвездия от навигационной эпохи (времени от начала теста). Под SDP I подразумевается задача 2.3, SDP II — 2.4 и SOCP — 2.5. Все алгоритмы, основанные на решении релаксированных выпуклых задач, вернули результат не позже, чем через 0,6 секунды. При использовании задачи конического программирования второго порядка время решения в течение всего теста не превышало 0,012 секунды. Время работы программы исчерпывающего поиска в ряде случаев превышало 10 секунд, что для большинства применений является неприемлемым.

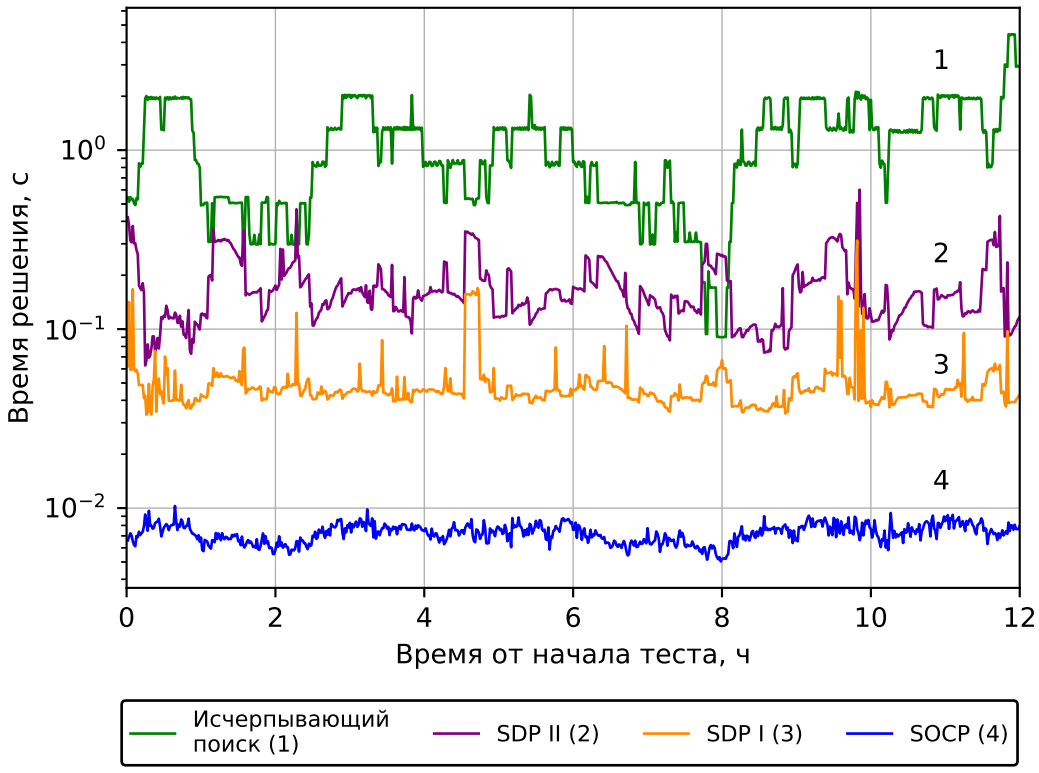


Рисунок 2.13 — Время решения задачи для различных методов для $m = 6$

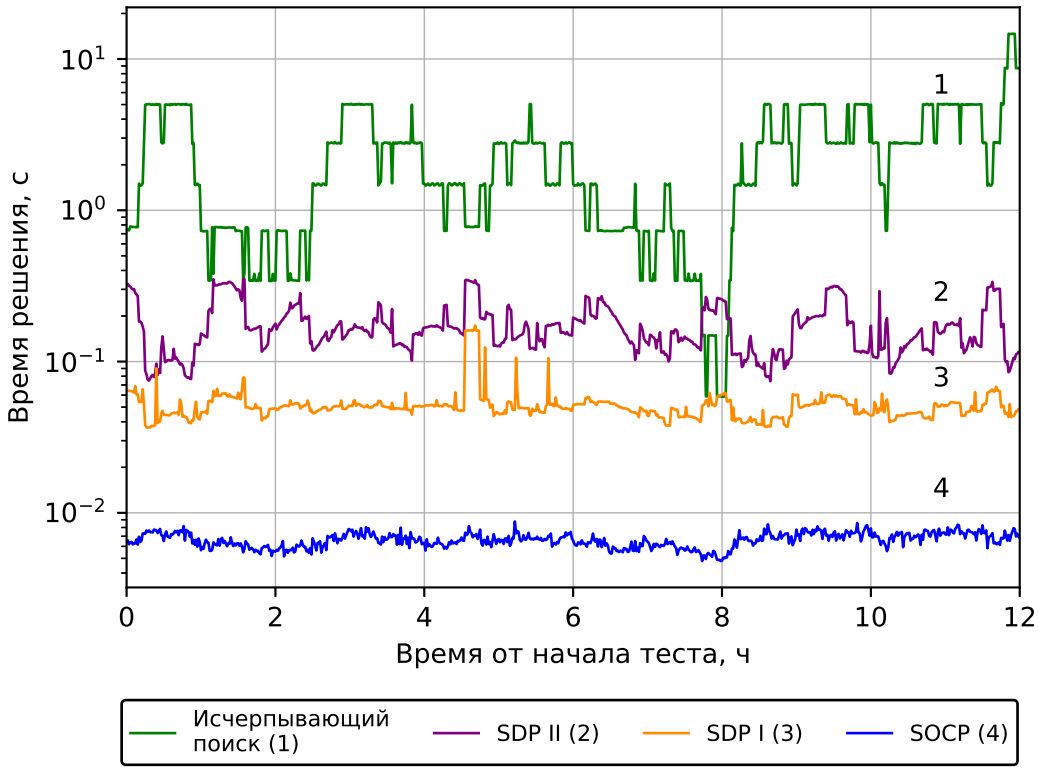


Рисунок 2.14 — Время решения задачи для различных методов для $m = 8$

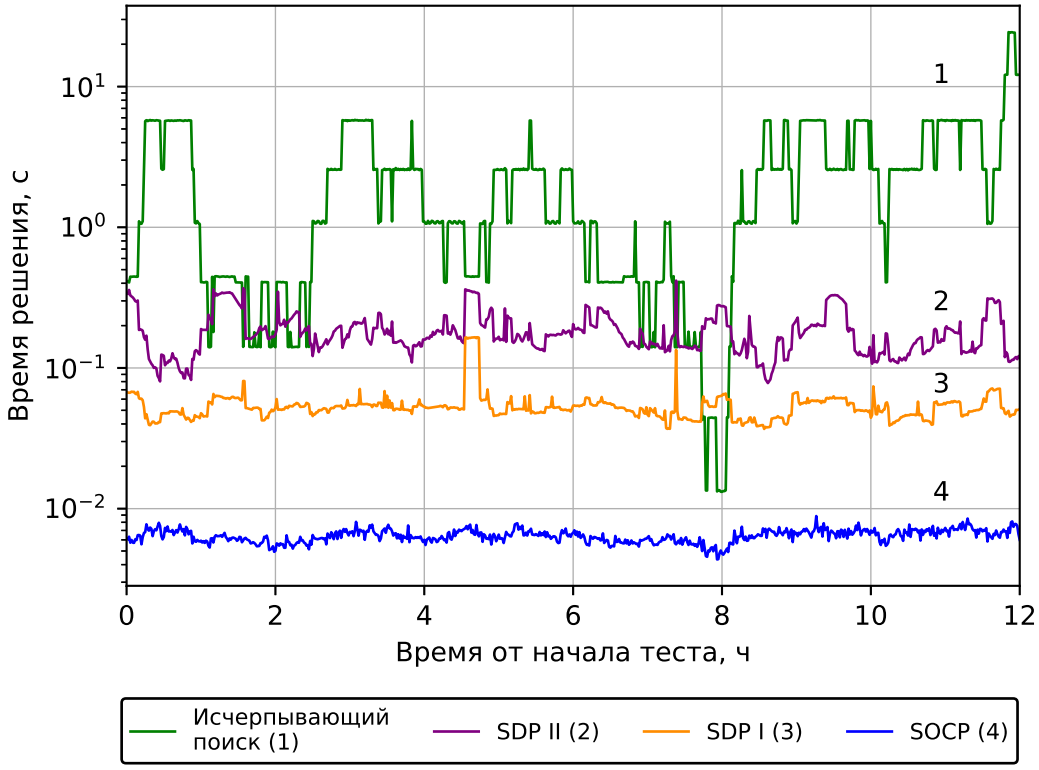


Рисунок 2.15 — Время решения задачи для различных методов для $m = 10$

Необходимо отметить, что в навигационном приемнике, работающем в режиме реального времени, может быть целесообразно вынесение в отдельный

процесс определения рабочего созвездия. Когда позиционирование приемника проводится несколько раз в секунду, изменение геометрической конфигурации спутников при переходе от одной навигационной эпохи к другой может быть незначительным. В связи с этим выбор спутников можно вести лишь с необходимой для этого частотой.

2.6 Выводы к главе 2

Рассмотрена задача выбора навигационным приемником рабочего созвездия ограниченного размера по критерию GDOP. Данная задача относится к комбинаторной оптимизации. В силу большого размера множества перебора в ряде случаев ее не удается решать точно в системах реального времени.

Приведен краткий обзор существующих методов приближенного решения. Некоторые методы применимы только для определенных размеров рабочего созвездия m или только для одной ГНСС системы. При этом большинство методов не позволяют получить приемлемую двустороннюю оценку точности решения.

В данной работе на основе выпуклой релаксации предложен метод построения двусторонней оценки точности приближенного решения. Получены задачи полуопределенного программирования и конического программирования второго порядка, к которым сводится решение релаксированной задачи; сформулированы и доказаны леммы об их оптимальных решениях. Предложен алгоритм построения приближенного решения. Применимость предложенного метода проверена в 12-часовом тесте. Тестирование подтвердило высокую точность и вычислительную эффективность предложенного метода. Результаты исследования опубликованы в работах [34; 37; 39; 43; 44].

Глава 3. Задача выбора базовых линий

3.1 Постановка задачи

Предположим, что на некотором твердом теле (например, на корпусе колесного робота, пример которого показан на рисунке 3.1) закреплено n навигационных антенн A_1, ..., A_n, подключенных к одному навигационному приемнику. Каждая антенна принимает сигналы от спутников ГНСС. Требуется с помощью спутниковых навигационных измерений определить относительную ориентацию твердого тела в пространстве.

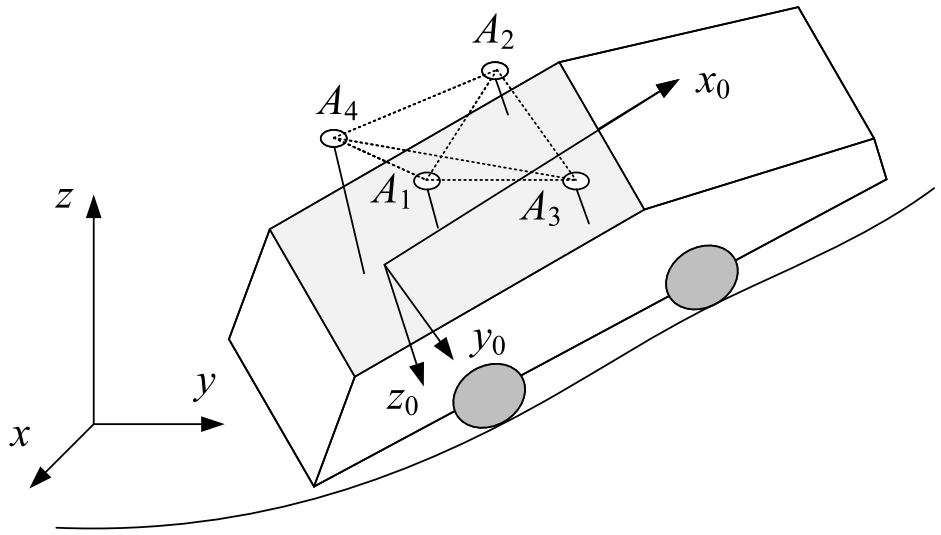


Рисунок 3.1 — Определение относительной ориентации с помощью спутниковой навигации.

Векторы, попарно соединяющие навигационные антенны, называют базовыми линиями. На рисунке 3.1 они показаны пунктиром. Если имеется n антенн, то без учета направлений можно выбрать $m = n(n-1)/2$ базовых линий. Минимально возможное количество базовых линий, при котором используются все антенны, равно $n - 1$. Навигационный приемник может вычислять координаты векторов базовых линий в системе координат локального горизонта (x, y, z). Сравнение с теми же векторами, определенными в связанной с телом системе координат (x_0, y_0, z_0), позволяет вычислить матрицу ориентации данного твердого тела относительно локального горизонта.

Предположим, что выбрано l базовых линий (например, $l = n-1$). Матрицей базовых линий будем называть такую матрицу размера $3 \times l$, что ее каждый i-ый столбец содержит координаты вектора, соединяющего антенны, относящиеся к i-ой базовой линии в выбранной системе координат. Пусть $X \in R^{3\times l}$ — матрица базовых линий в координатах локального горизонта (x, y, z), $X_0 \in R^{3\times l}$ — матрица тех же базовых линий в системе координат (x_0, y_0, z_0), связанной с телом. Обе системы координат правые декартовы. В задаче определения относительной ориентации требуется найти такую ортогональную матрицу $Q \in R^{3\times 3}$, с помощью которой можно было бы осуществить переход от X_0 к X. Матрицу Q можно представить следующим образом

$$Q = \begin{pmatrix} \cos(\psi)\cos(\theta) & \begin{pmatrix}\cos(\psi)\sin(\varphi)\sin(\theta)- \\ -\cos(\varphi)\sin(\psi)\end{pmatrix} & \begin{pmatrix}\sin(\varphi)\sin(\psi)+ \\ +\cos(\varphi)\cos(\psi)\sin(\theta)\end{pmatrix} \\ \cos(\theta)\sin(\psi) & \begin{pmatrix}\cos(\varphi)\cos(\psi)+ \\ +\sin(\varphi)\sin(\psi)\sin(\theta)\end{pmatrix} & \begin{pmatrix}\cos(\varphi)\sin(\psi)\sin(\theta)- \\ -\cos(\psi)\sin(\varphi)\end{pmatrix} \\ -\sin(\theta) & \cos(\theta)\sin(\varphi) & \cos(\varphi)\cos(\theta) \end{pmatrix}, \tag{3.1}$$

где ψ, θ, φ — углы Эйлера, соответствующие азимуту, тангажу и крену.

Обозначим как I_3 единичную матрицу размера 3×3. Один из методов, позволяющий найти матрицу Q с учетом наличия погрешностей измерений, основан на решении задачи Wahba [1; 15]:

$$\min_{Q} \ \|QX_0 - X\|_F^2 \tag{3.2}$$

при ограничении

$$QQ^T = I_3, \ \ Q \in R^{3\times 3}. \tag{3.3}$$

Используя соотношение (1.4) для квадрата нормы Фробениуса, целевую функцию (3.2) можно записать в виде

$$\|QX_0 - X\|_F^2 = \mathrm{trace}(X_0^T Q^T Q X_0) - 2\mathrm{trace}(X_0^T Q^T X) + \mathrm{trace}(X^T X) \tag{3.4}$$

Учитывая ограничение (3.4) и переставляя местами сомножители под знаком $\mathrm{trace}(\cdot)$ в соответствии с (1.3), получаем эквивалентную задачу максимизации

$$\max_{Q} \ \mathrm{trace}(X X_0^T Q^T) \tag{3.5}$$

при ограничении (3.3). Матрица XX_0^T имеет размер 3×3. Используя SVD-разложение, ее можно представить в виде $XX_0^T = U\Sigma V$, где матрицы U и

V ортогональны, а Σ — диагональная матрица из сингулярных чисел XX_0^T. Преобразуем целевую функцию (3.5):

$$\text{trace}(XX_0^TQ^T) = \text{trace}(U\Sigma VQ^T) = \text{trace}(\Sigma VQ^TU). \tag{3.6}$$

Получаем задачу

$$\max_Q \ \text{trace}(\Sigma VQ^TU) \tag{3.7}$$

при ограничении (3.3). Максимум достигается при $VQ^TU = I_3$, откуда $Q = UV$. Тогда для решения задачи Wahba достаточно найти SVD-разложение матрицы XX_0^T. Поскольку матрица базовых линий в координатах (x, y, z) в каждый момент времени может быть различной, то X может изменяться с течением времени. Тогда для каждого измерения требуется вычислять данное SVD-разложение. В работе [38] предложен способ решения задачи Wahba без SVD-разложения, основанный на полуопределенной релаксации условия (3.3). Точность определения ориентации зависит от обусловленности матрицы XX_0^T, которая тем лучше, чем лучше обусловленность матрицы $X_0X_0^T$. Рассмотрим задачу выбора $n-1$ базовой линии, обеспечивающей наилучшую обусловленность матрицы $X_0X_0^T$.

Для того, чтобы охарактеризовать обусловленность некоторой матрицы $M \in R^{p\times q}$, $p \geqslant q$, для нее определяется параметр — число обусловленности. Число обусловленности равно отношению максимального сингулярного числа матрицы к минимальному. Функцию, определяющую число обусловленности для матрицы, будем обозначать cond(). Согласно работе [3], этот параметр может быть определен формулой

$$\text{cond}(M) = \left(\frac{\lambda_{\max}(M^TM)}{\lambda_{\min}(M^TM)}\right)^{1/2}, \tag{3.8}$$

если матрица M полного ранга, и $\text{cond}(M) = \infty$ иначе, где $\lambda_{\min}$ и $\lambda_{\max}$ — минимальное и максимальное собственные числа матрицы соответственно.

Базовые линии будем учитывать только с одним направлением, поскольку противоположно направленные базовые линии вносят одинаковый вклад в вычисления. Пусть Y – матрица, строками которой являются координаты всевозможных базовых линий. Общее число базовых линий составляет $m = n\,(n-1)\,/2$, а матрица Y имеет размер $3 \times m$. Введем бинарные переменные

$s_k,\ k = 1,\ \ldots,\ m$, для которых значение 1 соответствует выбору базовой линии с номером k, а значение 0 соответствует случаю, когда базовая линия не используется. Обозначим $s = (s_1, s_2,\ \ldots,\ s_m)^T$. Для рассматриваемых бинарных переменных также будем использовать обозначение $s_{ij},\ i,\ j = 1, \ldots, n$, $s_{ij} \equiv s_{ji}$, где i и j – номера навигационных антенн A_i и A_j, которые соответствующая базовая линия соединяет. Пусть W — множество всех навигационных антенн, закрепленных на рассматриваемом твердом теле. Обозначим как $\left|\bar{V}\right|$ мощность некоторого множества антенн $\bar{V}$. Отметим, что выбор базовых линий, образующих остовное дерево, позволяет задействовать максимальное число антенн (а значит и навигационных измерений) без внутренних циклов, вносящих избыточность. Задача выбора базовых линий может быть математически сформулирована следующим образом.

Задача 3.1 (выбора базовых линий). *Найти*

$$\min_{s} \operatorname{cond}\left(Y \operatorname{diag}(s)\, Y^T\right) \tag{3.9}$$

при ограничениях

$$s_k \in \{0;\ 1\},\ k = 1,\ \ldots,\ m, \tag{3.10}$$

$$\sum_{k=1}^{m} s_k = n - 1, \tag{3.11}$$

$$\sum_{j=1,\ j\neq i}^{n} s_{ij} \geqslant 1,\ i = 1, \ldots,\ n, \tag{3.12}$$

где $s_k \equiv s_{ij}$ если базовая линия k соединяет антенны A_i и A_j. Если также потребовать, чтобы составленный из них граф являлся остовным деревом, требуется добавить хотя бы одно из следующих двух условий:

$$\forall \bar{V} \subset W, \bar{V} \neq \varnothing, \bar{V} \neq W \qquad \sum_{A_i \in \bar{V},\ A_j \notin \bar{V}} s_{ij} \geqslant 1, \tag{3.13}$$

$$\forall \bar{V} \subset W, \bar{V} \neq \varnothing, \bar{V} \neq W \qquad \sum_{A_i \in \bar{V},\ A_j \in \bar{V},\ i<j} s_{ij} \leqslant \left|\bar{V}\right| - 1. \tag{3.14}$$

Условие (3.13) означает, что из любого непустого подмножества антенн, не совпадающего с W, выходит хотя бы одна базовая линия. Это, в частности, гарантирует отсутствие внутренних циклов в конфигурации базовых линий. Условие (3.14) гарантирует отсутствие внутренних циклов другим способом: в

любом множестве $\bar{V}$ не может быть базовых линий больше, чем число вершин. Для любого допустимого решения задачи 3.1 выполнено соответствие

$$Y \operatorname{diag}(s)\, Y^T = X_0 X_0^T, \tag{3.15}$$

где X_0 — матрица размерности $3 \times (n-1)$ выбранных базовых линий (их направляющих векторов) в системе координат (x_0, y_0, z_0).

Отметим, что если для одной из базовых линий выбрать противоположно направленный вектор, т.е. изменить знаки элементов одного из столбцов матрицы X_0 на противоположные, то матрица $X_0 X_0^T$ не изменится. Поэтому два противоположных направления векторов базовых линий вносят одинаковый вклад в обусловленность матрицы $X_0 X_0^T$.

3.2 Оценка вычислительной сложности

Задача 3.1 вне зависимости от выбора дополнительных ограничений (3.13) или (3.14) является невыпуклой. При этом ее решение предполагает перебор C_m^{n-1} всевозможных комбинаций базовых линий. По теореме Кэли о числе остовных деревьев при введении ограничения (3.13) или (3.14) число возможных конфигураций базовых линий будет n^{n-2}. За приемлемое время точно задачу выбора базовых линий можно решить только для определенного небольшого числа антенн. Так, например, для $n = 8$ необходимо рассматривать уже 1184040 всевозможных конфигураций (без учета изолированности вершин), а общее число остовных деревьев в этом случае 262144. Это обстоятельство обуславливает необходимость использования методов оценки оптимальной конфигурации базовых линий, отличных от перебора.

Размерность матрицы, от которой определяется число обусловленности в (3.9), составляет 3×3. Помимо n бинарных ограничений (3.10), задача 3.1 содержит $n+1$ линейное ограничение. При введении дополнительного требования выбора конфигурации в виде основного дерева, число линейных ограничений возрастает до $2^n + n - 1$.

Однако, стоит отметить, что для часто используемой трехантенной системы, описание которой можно найти в работе [35], число возможных

конфигураций базовых линий равно трем, и для решения задачи 3.1 можно использовать полный перебор.

3.3 Предлагаемый метод решения

3.3.1 Релаксация ограничений

Применим линейную релаксацию к бинарным условиям задачи 3.1. Получаем следующую релаксированную задачу.

Задача 3.2. *Найти*

$$\min_s \operatorname{cond}\left(Y \operatorname{diag}(s) Y^T\right) \tag{3.16}$$

при ограничениях (3.11)—(3.12) *и*

$$s_k \in [0;\ 1],\ k = 1,\ \ldots,\ m, \tag{3.17}$$

а также (3.13) *или* (3.14) *в случае поиска остовного дерева.*

Заметим, что для произвольной симметричной положительно определенной матрицы $A \succ 0$ число обусловленности равно отношению ее максимального собственного числа $\lambda_{\max}(A)$ к минимальному $\lambda_{\min}(A)$ [94; 95]:

$$\operatorname{cond}(A) = \lambda_{\max}(A) / \lambda_{\min}(A). \tag{3.18}$$

Предположим, что матрица $A(s)$ является аффинной комбинацией симметричных положительно определенных матриц A_0, A_1, $\ldots$, A_n:

$$A(s) = A_0 + A_1 s_1 + A_2 s_2 + \cdots + A_n s_n, \tag{3.19}$$

а матрица $B(s)$ также является аффинной комбинацией симметричных матриц, причем ограничения задачи представимы в виде линейного матричного неравенства $B(s) \succeq 0$. Тогда, как следует из работ [3; 96], задачу минимизации максимального собственного числа $A(s)$

$$\min_s \lambda_{\max}(A(s)) \tag{3.20}$$

при $B(s) \succeq 0$ можно записать в виде поиска

$$\min_{s,\ \lambda} \lambda \tag{3.21}$$

при ограничениях $B(s) \succeq 0$, $\lambda I - A(s) \succeq 0$. Аналогично задача максимизации минимального собственного числа $A(s)$

$$\max_{s} \lambda_{\min}(A(s)) \tag{3.22}$$

при $B(s) \succeq 0$ представима в виде поиска

$$\min_{s,\ \lambda} \lambda \tag{3.23}$$

при ограничениях $B(s) \succeq 0$, $A(s) - \lambda I \succeq 0$.

Объединяя эти два результата, можно записать задачу 3.2 в виде поиска

$$\min_{s,\gamma,\mu} \gamma \tag{3.24}$$

при ограничениях (3.11)—(3.12) (а также (3.13) или (3.14) в случае поиска остовного дерева), (3.17) и

$$\gamma\mu I - Y \operatorname{diag}(s) Y^T \succeq 0, \tag{3.25}$$

$$Y \operatorname{diag}(s) Y^T - \mu I \succeq 0, \tag{3.26}$$

$$\mu > 0. \tag{3.27}$$

3.3.2 Метод масштабирования

Применим к задаче 3.2 метод масштабирования, описанный в работе [3]. Введем дополнительные переменные $\nu = 1/\mu$, $c_i = s_i/\mu$, $i = 1, 2, \ldots, m$. Как и для переменных s_i, будем также использовать двойные индексы: c_{ij} соответствует базовой линии между антеннами A_i и A_j. Отметим, что

$$Y \operatorname{diag}(s) Y^T = \sum_{i=1}^{m} s_i Y_i Y_i^T = \mu \sum_{i=1}^{m} c_i Y_i Y_i^T, \tag{3.28}$$

где Y_i – столбец матрицы Y, соответствующий базовой линии с номером $i = 1, 2, \ldots, m$. Таким образом, мы получаем следующую задачу оптимизации.

Задача 3.3. *Найти*

$$\min_{c_1,\ldots,c_m,\gamma,\ \nu} \gamma \tag{3.29}$$

при ограничениях

$$\gamma I - \sum_{i=1}^{m} c_i Y_i Y_i^T \succeq 0, \tag{3.30}$$

$$\sum_{i=1}^{m} c_i Y_i Y_i^T - I \succeq 0, \tag{3.31}$$

$$0 \leqslant c_k \leqslant \nu,\ k = 1,\ \ldots,\ m, \tag{3.32}$$

$$\sum_{k=1}^{m} c_k = (n-1)\,\nu, \tag{3.33}$$

$$\sum_{j=1,\quad j\neq i}^{n} c_{ij} \geqslant \nu,\ i = 1, \ldots,\ n. \tag{3.34}$$

Чтобы полученный граф являлся остовным деревом, нужно добавить хотя бы одно из условий:

$$\forall \bar{V} \subset W,\ \bar{V} \neq \varnothing,\ \bar{V} \neq W \qquad \sum_{A_i \in \bar{V},\ A_j \notin \bar{V}} c_{ij} \geqslant \nu, \tag{3.35}$$

$$\forall \bar{V} \subset W,\ \bar{V} \neq \varnothing,\ \bar{V} \neq W \qquad \sum_{A_i \in \bar{V},\ A_j \in \bar{V},\ i<j} c_{ij} \leqslant (\left|\bar{V}\right| - 1)\nu. \tag{3.36}$$

В задаче 3.3 целевая функция является линейной по переменным, условия (3.32)—(3.36) являются линейными, а (3.30) и (3.31) представляют собой линейные матричные неравенства. Таким образом, это задача полуопределенного программирования. Для ее решения могут быть использованы вычислительно-эффективные методы решения.

Связь между оптимальными решениями задачи полуопределенного программирования 3.3 $\widetilde{c}_1,\ \ldots,\ \widetilde{c}_m,\ \widetilde{\gamma},\ \widetilde{\nu}$ и релаксированной задачи 3.2 $\widetilde{s}_1, \ldots, \widetilde{s}_m$, как следует из замены переменных, задается формулой

$$\widetilde{s}_i = \widetilde{c}_i / \widetilde{\nu},\ i = 1,\ \ldots,\ m. \tag{3.37}$$

3.3.3 Итерационный метод

Для задачи 3.2 также можно применить итерационный метод минимизации числа обусловленности с заданной точностью, описанный в работах [94; 95]. Его можно описать следующим образом. Сначала выбираются конечные верхняя $\overline{\kappa}$ и нижняя $\underline{\kappa}$ границы для значения числа обусловленности κ, $\underline{\kappa} \leqslant \kappa \leqslant \overline{\kappa}$. На каждой итерации решается для $v = (\overline{\kappa} + \underline{\kappa})/2$ следующая задача оптимизации.

Задача 3.4. *Найти*

$$\min_{s} \lambda_{\max}(Y \operatorname{diag}(s) Y^T) - v\lambda_{\min}(Y \operatorname{diag}(s) Y^T) \tag{3.38}$$

при ограничениях (3.11)—(3.12), (3.17), *а также* (3.13) *или* (3.14) *в случае поиска остовного дерева.*

Если оптимальное значение критерия положительное, то обновляем нижнюю границу, полагая $\underline{\kappa} = v$, в противном случае – верхнюю, полагая $\overline{\kappa} = v$. Итеративное решение проводится до тех пор, пока разность $\underline{\kappa} - \overline{\kappa}$ не достигнет заданного порога точности.

Поскольку $\lambda_{\max} \geqslant \lambda_{\min}$, то из (3.18) следует, что для положительно определенной матрицы число обусловленности всегда больше либо равно единице. Тогда на первой итерации можно положить, что $\underline{\kappa} = 1$. Для определения $\overline{\kappa}$ можно использовать значение $\operatorname{cond}\left(X_0 X_0^T\right)$ для любой допустимой конфигурации из $n - 1$ базовой линии.

Задача 3.4 с использованием соотношений между критериями оптимизации (3.20)—(3.23) может быть записана в виде следующей задачи полуопределенного программирования, для которой v также является заданным параметром.

Задача 3.5. *Найти*

$$\min_{s,\, \lambda_1,\, \lambda_2} \lambda_1 - v\lambda_2 \tag{3.39}$$

при ограничениях (3.11)—(3.12), (3.17),

$$\lambda_1 I - Y \operatorname{diag}(s) Y^T \succeq 0, \tag{3.40}$$

$$Y \operatorname{diag}(s) Y^T - \lambda_2 I \succeq 0, \tag{3.41}$$

а также (3.13) *или* (3.14) *в случае поиска остовного дерева.*

Размерности задач 3.3 и 3.5 схожи. Отличие заключается в том, что итерационный метод позволяет напрямую получить значения вектора переменных s для оптимального релаксированной задачи 3.2. Однако он явным образом требует решения нескольких задач полуопределенного программирования. Сходимость итерационного метода обусловлена тем, что он использует деление отрезка пополам: число итераций определяется только точностью определения числа обусловленности и выбором его верхней и нижней оценки.

3.3.4 Алгоритм выбора базовых линий

Поскольку из-за использования релаксации бинарного условия получаются вещественные значения переменных $s_k \in [0, 1]$, их можно использовать в качестве весов w_k (например, $w_k = 1 - s_k$ или $w_k = 1/(s_k + 0.25)^2$) для алгоритма построения остовного дерева с минимальной суммой весов, например, для алгоритма Прима, изложенного в работе [97]. В случае, если ищется не обязательно остовное дерево, результат можно получить как решение соответствующей задачи целочисленного линейного программирования поиска

$$\min_{s} \sum_{k=1}^{m} w_j s_j \tag{3.42}$$

при ограничениях (3.11)—(3.12) и $s_k \in \{0, 1\}$, $k = 1, \ldots, m$.

Алгоритм выбора базовых линий можно представить следующим образом.

Алгоритм 3.1 (выбора базовых линий).

1. Определить положение всех n навигационных антенн в системе координат, связанной с телом.
2. Определить матрицу всевозможных базовых линий Y.
3. Решить релаксированную задачу оптимизации 3.2 для матрицы Y в помощью метода масштабирования либо итерационного метода. Оптимальное решение релаксированной задачи — набор значений $\widetilde{s}_k$, $k = 1, \ldots, m$.
4. Перейти от вещественных значений $\widetilde{s}_k \in [0, 1]$ к бинарным $\hat{s}_k \in \{0, 1\}$, $k = 1, \ldots, m$, с помощью одного из предложенных выше методов оценки.

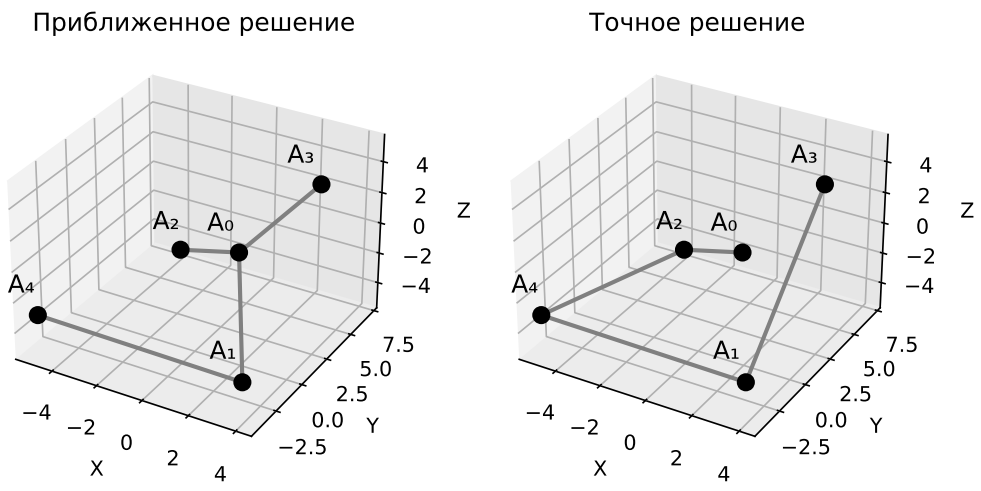


Рисунок 3.2 — Результат выбора базовых линий.

5. Выбрать базовые линии, которым соответствуют $\hat{s}_k = 1$, $k = 1, \ldots, m$. Заметим, что подстановкой значений $\widetilde{s}_k$ в целевую функцию (3.16) можно получить нижнюю оценку числа обусловленности.

3.4 Результаты экспериментов

Поскольку оба метода — итерационный и масштабирования — решают одну и ту же релаксированную задачу оптимизации 3.2, а алгоритм не предназначен для работы в системе реального времени, требуется оценить точность работы алгоритма по одному из методов минимизации числа обусловленности.

Рассмотрим пример (рисунок 3.2), для которого приближенное решение по предложенному алгоритму отличается от точного. Для вычислений использовался метод масштабирования. Приближенное и точное решения показаны на рисунке 3.2. Координаты антенн $A_0 = (1,\, 2,\, 0)\,,\ \ A_1 = (3,\, -2,\, -5)\,,\ \ A_2 = (-4,\, 7,\, -5)\,,\ \ A_3 = (4,\, 3,\, 5)\,,\ \ A_4 = (-5,\, -4,\, -3)$. В этом случае $n = 5$ навигационных антенн и $m = 10$ всевозможных базовых линий. Число островных деревьев 625, что позволяет получить для сравнения точное решение. В таком случае предложенный метод дает решение $\hat{s}$, для которого $\operatorname{cond}\left(Y \operatorname{diag}(\hat{s})\, Y^T\right) \approx 2{,}62$. Точный метод, основанный на переборе всевоз-

можных вариантов (исчерпывающий поиск), дает решение s^*, для которого $\mathrm{cond}\left(Y \mathrm{diag}\left(\hat{s}\right) Y^T\right) \approx 2{,}08$. Числа обусловленности не отличаются значительно.

Экспериментальная оценка точности на 1000 различных тестах для случаев $n = 4$ и $n = 5$ проводилась с помощью итерационного алгоритма. Число итераций алгоритма было равно 5. Для решения задач полуопределенного программирования использовался алгоритм SCS, приведенный в работах [75] и [76]. Проводилось сравнение с точным решением, полученным путем перебора (для $n = 4$ и $n = 5$ его можно получить за приемлемое время). Результаты представлены в таблице 2. Под κ_{opt} подразумевается число обусловленности матрицы для точного решения, κ_{approx} – для приближенного. В таблице приведены как процент от общего числа различных тестов с полным совпадением оптимального и приближенного решения, так и $\max(\kappa_{\mathrm{approx}}/\kappa_{\mathrm{opt}})$ — максимальное отношение числа обусловленности для приближенного решения к числу обусловленности для точного решения.

Таблица 2 — Результаты теста алгоритма выбора базовых линий.

n	поиск остовного дерева		поиск графа (общий вид)	
	% совпадений	$\max(\kappa_{\mathrm{approx}}/\kappa_{\mathrm{opt}})$	% совпадений	$\max(\kappa_{\mathrm{approx}}/\kappa_{\mathrm{opt}})$
4	92,4	6,92	92,5	3,59
5	65,8	3,21	61	3,95

Из результатов эксперимента видно, что есть определенный процент совпадения с точным решением для приведенных значений n, при этом даже в случае неточного решения число обусловленности матрицы получается примерно того же порядка, что и в точном решении. Существенным было бы отличие числа обусловленности на несколько порядков.

3.5 Выводы к главе 3

Рассмотрена задача выбора базовых линий при определении относительной ориентации твердого тела в пространстве методами спутниковой навигации с помощью решения задачи Wahba. В качестве критерия оптимизации выбрано число обусловленности матрицы $X_0 X_0^T$, где X_0 — матрица выбранных

базовых линий в системе координат, связанной с телом. Данная задача относится к комбинаторной оптимизации. Ее точное решение предполагает перебор всевозможных вариантов. Если имеется большое количество закрепленных на твердом теле антенн, то перебор требует значительного времени вычислений.

Предложена линейная релаксация бинарных условий задачи. Релаксированная задача может быть решена, например, одним из двух существующих методов: масштабирования переменных или итерационным. Сформулирован приближенный алгоритм выбора базовых линий с использованием решения релаксированной задачи. Проведено экспериментальное исследование, подтверждающее работоспособность алгоритма. Предложенный метод может быть использован для оценки оптимального выбора базовых линий при определении ориентации твердого тела с помощью методов ГНСС навигации. Данные результаты опубликованы в работах [38; 45; 46].

Глава 4. Задача планирования путей колесных роботов, покрывающих заданную поверхность

4.1 Постановка задачи

В точном земледелии предполагается использование спутниковой навигации, геоинформационных систем, мониторинга с помощью различных сенсоров в целях совершенствования технологий выращивания сельскохозяйственных культур. Эффективное использование собранных данных позволяет, например, сократить время выполнения работ, площадь задействованных участков земли, количество необходимых ресурсов.

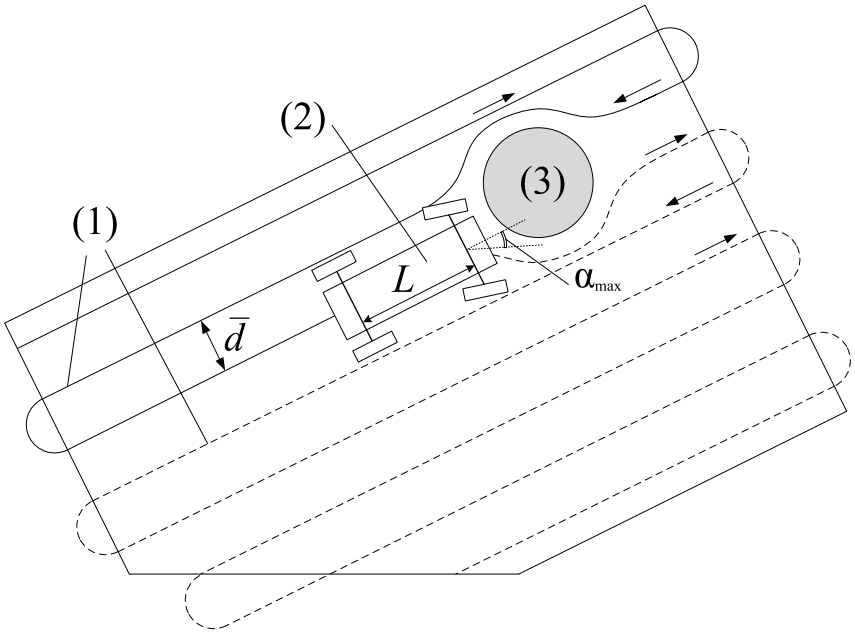


Рисунок 4.1 — Реализация путей (1), спланированных с учетом препятствия (3), колесным роботом (2). L — расстояние между осями, $\alpha_{\max}$ — максимальный эффективный угол поворота передних колес.

Одной из задач планирования работ на поле в точном земледелии является построение путей автономных сельскохозяйственных роботов, покрывающих заданный участок ландшафта (рисунок 4.1). Предполагается, что рельеф поля достаточно пологий. При планировании путей, покрывающих поле, как правило, требуется минимизировать перекрытие соседних рядов и полностью

исключить пропуски в покрытии (за исключением областей вблизи препятствий, которые могут обрабатываться отдельно).

Построение путей, полностью покрывающих заданную поверхность (англ. *Complete Coverage Path Planning*), встречается в различных приложениях, например, в точном земледелии [20—22; 98], роботизированной окраске [99] и очистке [18] поверхностей. В точном земледелии для решения этой задачи обычно используются почти параллельные траектории. Их построение может проводиться относительно некоторого начального пути. Предположим, что ширина рабочего инструмента сельскохозяйственной машины $\bar{d}$. На расстоянии $\bar{d}$ от исходного пути с одной стороны (или с обеих, если это возможно) строится следующий путь, который далее используется в качестве начального. Затем процедура повторяется, пока все поле не будет покрыто рядами.

Спланированные пути должны быть реализуемы. Колесные роботы, использующие механизм руления поворотом передних колес, не могут реализовывать траектории с нормальной кривизной, превышающей пороговое значение, определяемое характеристиками машины. Во всех точках спланированной траектории движения должно выполняться условие на нормальную кривизну u

$$\|u\| < u_{\max}, \tag{4.1}$$

где $u_{\max}$ – максимальная реализуемая нормальная кривизна траектории, определяемая формулой

$$u_{\max} = \frac{\mathrm{tg}\alpha_{\max}}{L}, \tag{4.2}$$

где L — расстояние между передней и задней осями робота, $\alpha_{\max}$ — максимальный эффективный угол поворота передних колес [100]. Если у робота два передних колеса, то углы их поворота могут быть различны. Поэтому в формуле (4.2) используется эффективный угол поворота. Колесные роботы с дифференциальным приводом способны совершить разворот на месте, в силу чего выполнение условия (4.1) для их траекторий не требуется. Заметим, что в (4.1) используется именно нормальная кривизна пути, а не трехмерная. Разницу между этими характеристиками можно проиллюстрировать на простом примере: преодолеваемый без поворота колес путь в гору, рельеф которой в вертикальном разрезе представляет собой сегмент окружности, имеет ненулевую трехмерную кривизну. При этом поворот колес определяется нормальной кривизной пути.

При планировании путей должны учитываться находящиеся на поле препятствия, которые могут быть как подвижными (динамическими), так и неподвижными (стационарными). Динамические препятствия могут изменять свое местоположение во время проведения работ. Это могут быть, например, животные и другие сельскохозяйственные машины. В автономных роботах траектории объезда подвижных препятствий планируются в процессе движения с использованием информации, собираемой в режиме реального времени стереокамерами, лидарами и другими сенсорами. Перепланирование траектории требует вычислительных ресурсов, которые, как правило, ограничены. Поэтому учет стационарных препятствий следует проводить до начала выполнения работ роботов. Траектории объезда также должны удовлетворять ограничению (4.1).

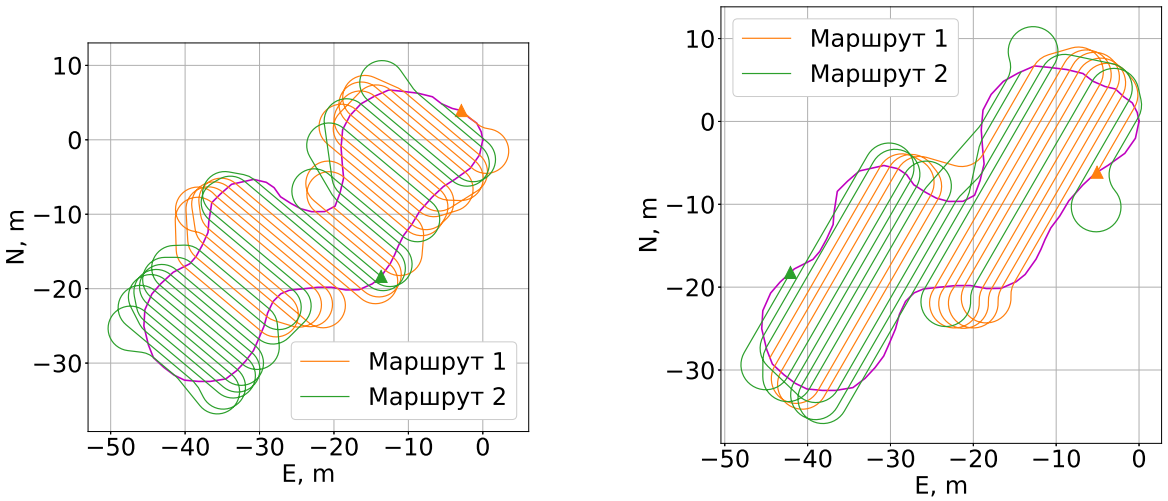
4.2 Существующие методы построения траекторий, покрывающих заданную поверхность

4.2.1 Построение путей, покрывающих плоскую поверхность

Если поверхность является плоской, либо с достаточно хорошей точностью может быть описана плоскостью, то можно использовать прямолинейный начальный путь. Полное покрытие поля может быть обеспечено параллельными путями с расстоянием $\bar{d}$ между каждой парой соседних из них.

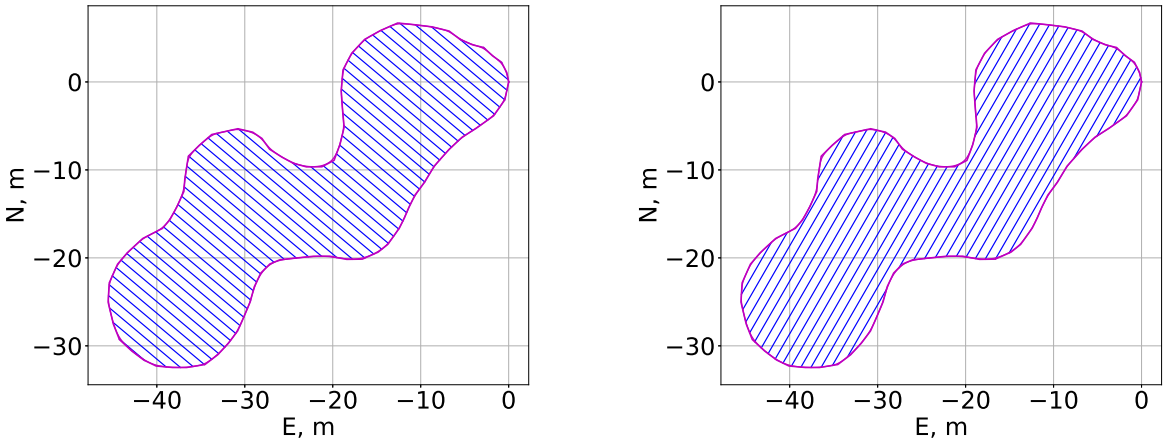
Предположим, что для описания путей используется плоская декартова система координат, ось x направлена на восток и y — на север. Начальный путь можно полностью задать положением одной из его точек (x_0, y_0) и азимутом φ — углом с направлением на север.

Простейший алгоритм построения путей, приведенный, например, в работе [22], заключается в следующем. Относительно прямой, проходящей через (x_0, y_0), строятся параллельные пути на кратных $\bar{d}$ расстояниях так, чтобы все поле было покрыто. Затем производится обрезка путей на границе поля. Таким образом, покрытие поля полностью задается с помощью начальной точки (x_0, y_0) и азимута φ.



а) б)

Рисунок 4.3 — Маршруты движения двух роботов для а) $\varphi = -50°$, б) $\varphi = 30°$.



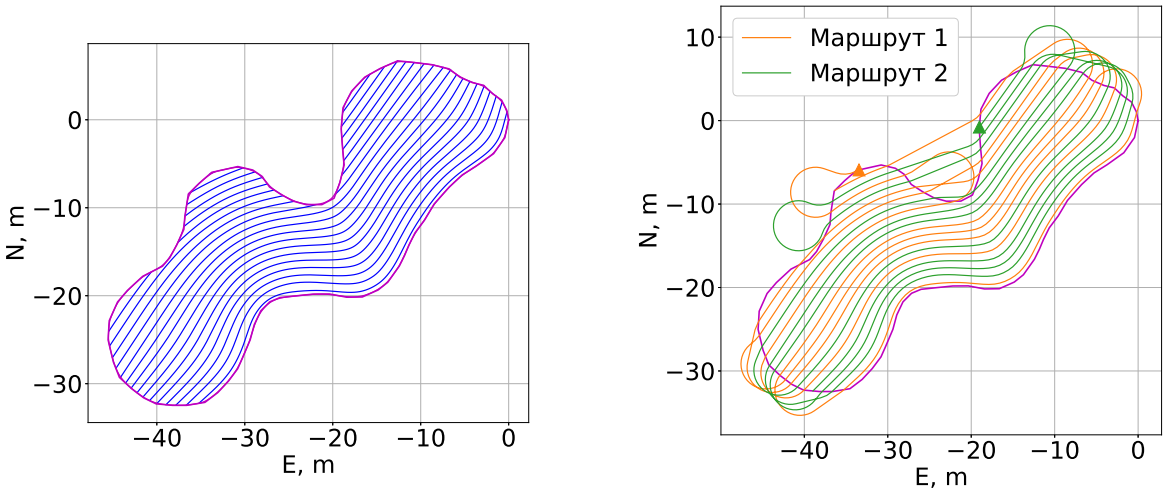
а) б)

Рисунок 4.2 — Покрытие поля параллельными путями для а) $\varphi = -50°$, б) $\varphi = 30°$.

В зависимости от выбора (x_0, y_0) и φ получается разное число точек разворота для переходов на соседние ряды. Рассмотрим примеры построения путей, показанные на рисунке 4.2. Точка (x_0, y_0) для них одинаковая, а азимуты φ различны и составляют $-50°$ и $30°$. На рисунке 4.3 показаны возможные маршруты движения при одновременном выполнении работ двумя роботами. В случае $\varphi = 30°$ длины маршрутов $D_1 = 546{,}3$ м и $D_2 = 544$ м а в случае $\varphi = -50°$ значительно больше, $D_1 = 670{,}8$ м и $D_2 = 671{,}3$ м. Такое значительное отличие длин маршрутов связано с большим отличием количества точек разворота.

С одной стороны, можно оптимизировать число точек разворота, варьируя (x_0, y_0) в некоторой окрестности радиусом $\bar{d}/2$ и азимут φ в диапазоне $[-90°; 90°)$. С другой стороны, можно использовать криволинейный начальный путь. В ряде случаев удобно выбирать в качестве начального пути участок границы поля, как это сделано в примере на рисунке 4.4, где длины маршрутов равны $D_1 = 557{,}3$ м и $D_2 = 556{,}6$ м. Заметим, что в данном примере длины путей после маршрутизации получились незначительно больше, чем с прямолинейным начальным путем при $\varphi = 30°$.

Для полей сложной формы к лучшим результатам может привести разбиение поля на участки, для каждого из которых покрытие строится отдельно. Постановки и методы решения таких задач рассмотрены в работе [17] по построению путей в пахотном земледелии.



а) б)

Рисунок 4.4 — Покрытие поля путями (а) и маршруты движения (б) двух роботов для криволинейного начального пути.

При построении покрытия с использованием криволинейного начального пути необходимо учитывать, что максимальная кривизна каждого следующего пути может оказаться больше, чем у предыдущего. Для машин с механизмом руления поворотом передних колес это может привести к нарушению условия (4.1), а значит, к нереализуемым траекториям. В таком случае ограничения на кривизну не позволяют обработать поле без пропусков. Кроме того, независимо от способа руления и параметра $u_{\max}$ (бесконечного в случае машины с дифференциальным приводом), при большой кривизне пути может образоваться самопересечение следующего (по построению) пути. Пример такого построения

приведен на рисунке 4.5. В литературе этот эффект называется «ласточкин хвост» [23; 24; 101]. Причина его возникновения заключается в том, что для криволинейной траектории нормали длиной $\bar{d}$ могут пересекаться. Как правило, при устранении самопересечения образуется траектория с разрывной первой производной по параметру. Все это значительно усложняет построение покрытия с использованием криволинейного начального пути.

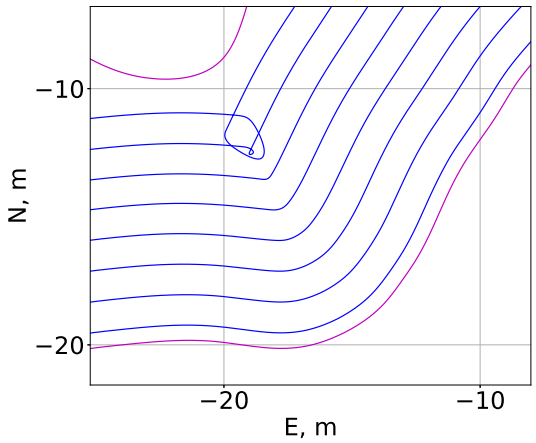


Рисунок 4.5 — Возникновение особенностей «ласточкин хвост» при построении покрытия поля путями.

4.2.2 Построение путей, покрывающих пологую поверхность

Построение покрывающих путей для трехмерного ландшафта заметно отличается от случая плоского поля. Как замечено в работе [22], если при построении покрывающих путей по неровной поверхности использовать простое проецирование двумерных путей на поверхность поля, то часто возникают перекрытия и пропуски в покрытии. Это связано с тем, что расстояние между путями в таком случае не сохраняется и может превысить ширину $\bar{d}$ рабочего инструмента сельскохозяйственной машины. В той же работе предложено проводить расчет трехмерных путей с помощью пересечения цилиндров с радиусом $\bar{d}$ вокруг исходного пути и поверхности. Такой подход позволяет избежать перекрытия и пропусков в покрытии для ряда случаев.

В работе [23] рассмотрена задача построения параллельных кривых по дифференцируемой криволинейной поверхности, заданной параметрически. Предложен метод расчета решением систем дифференциальных уравнений. Отметим, что среди предложенных авторами примеров также есть такие, в которых возникает особенность «ласточкин хвост». Таким образом, самопересечение соседнего пути может возникнуть и на криволинейной поверхности.

В пахотном земледелии целевые функции, позволяющие оценивать покрытие пологого поля, могут учитывать число разворотов, время разворота между рядами, водную эрозию почвы, кривизну получаемых путей. В работе [20] рассмотрены способы построения таких целевых функций. Задачи поиска начальной кривой, позволяющей получить наилучшее (по заданному критерию) покрытие трехмерного поля, также рассмотрены в работе [21].

4.2.3 Учет препятствий

В работах, посвященных учету препятствий в задачах планирования путей, обычно рассматривается либо декомпозиция поля на участки, не содержащие внутренних недоступных для проезда областей, либо расчет траектории их объезда.

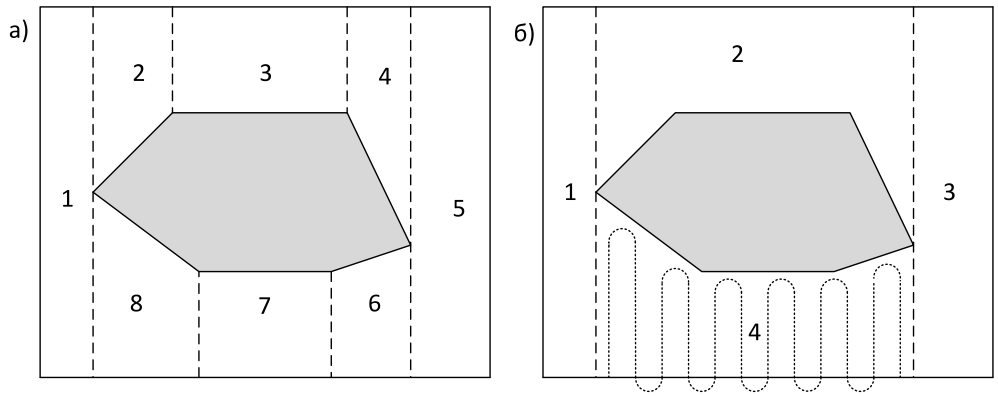


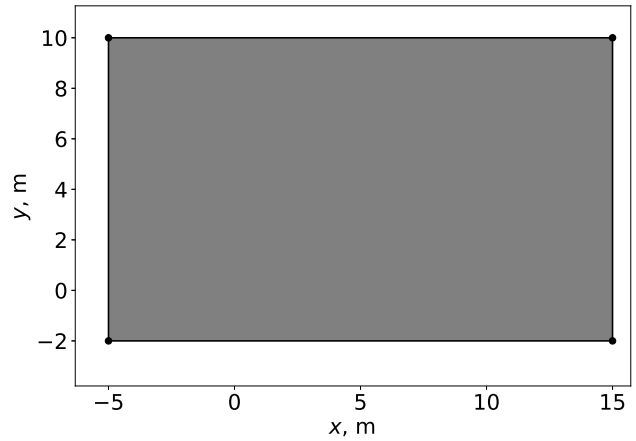
Рисунок 4.6 — Трапециевидная декомпозиция (а) и декомпозиция бустрофедон (б).

Рассмотрим случай плоского поля. Предположим, что на поле находятся препятствия, форма которых может быть описана выпуклым полигоном. Простейшим способом разделения рабочей области на участки, покрытие которых может быть построено без учета этого препятствия, является трапециевидная

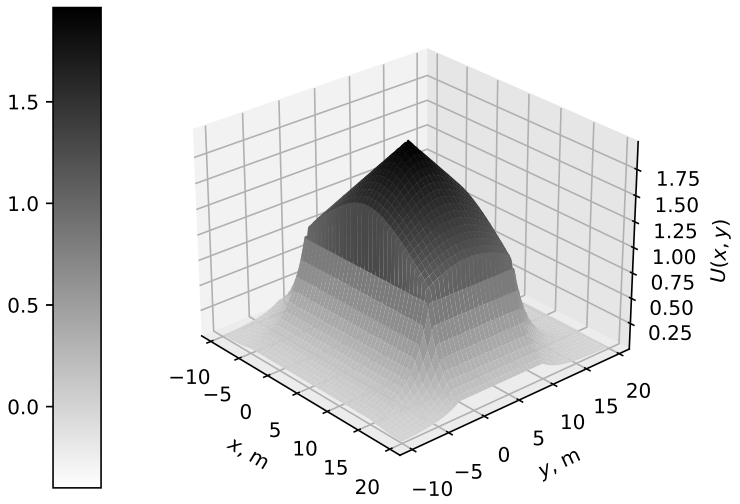
декомпозиция [26]. Пример трапециевидной декомпозиции показан на рисунке 4.6а. Препятствие отображено в виде закрашенного полигона. Внешний контур показывает границу поля, которое требуется обработать. Пунктирными линиями показан результат декомпозиции, а получившиеся участки поля без препятствий пронумерованы. Для каждого участка начальный путь вертикален относительно графика. В трапециевидной декомпозиции каждая вершина полигона препятствия задает границу участка. Участки получаются, как правило, в форме трапеций, откуда и возникает название метода. Отметим, что если у полигона препятствия достаточно много вершин, то и число участков может оказаться большим. Сократить их количество позволяет другой способ — декомпозиция бустрофедон [27]. Его название восходит от техники письма, в которой направление следования букв чередуется в зависимости от четности строки. В декомпозиции бустрофедон участки имеют такую форму, что они могут быть обработаны с помощью движения по двум чередующимся противоположным направлениям. Пример показан на рисунке 4.6б. Участки также пронумерованы, для участка 4 показан возможный способ обхода рядов. Число участков для трапециевидной декомпозиции — восемь, а для декомпозиции бустрофедон оказалось достаточным выделения четырех участков, при этом некоторые из них можно объединить. Обобщением метода бустрофедон для неполигональных препятствий является декомпозиция Морзе [25; 28].

Одним из способов построения траекторий объезда препятствий являются алгоритмы поиска на графе. Для этих алгоритмов граф включает в себя заранее заданные возможные изменения направления движения. Например, по проезду некоторого заданного расстояния может приниматься решение двигаться прямо либо повернуть колеса на максимальный угол влево или вправо. Исчерпывающий поиск на графе часто невозможно провести за приемлемое время. В связи с этим актуально выбирать порядок обхода ветвей графа. Направление поиска по графу может быть выбрано, например, детерминированными алгоритмами A* [29] и D* [30], либо стохастическим алгоритмом RRT [102; 103]. В работе [104] рассмотрено применение алгоритма A* для динамического перепланирования и стабилизации движения по траекториям.

Другим способом построения траекторий объезда препятствий является деформация пути с применением штрафных функций. Предположим, что пути колесных роботов построены без учета препятствий. Требуется видоизменить часть путей так, чтобы они не пересекали препятствия. Построение



а)



б)

Рисунок 4.7 — Пример штрафной функции для задачи деформации пути
а) граница препятствия, б) штрафная функция $U(x, y)$.

штрафных функций (искусственных потенциальных полей) для препятствий выпуклой полигональной формы рассмотрено в работах [31; 32]. Штрафные функции должны, с одной стороны, содержать некоторый барьер, не позволяющий точкам пути пересекать недопустимые области, а с другой стороны, их форма внутри препятствия должна позволять выталкивать из него точки пути в процессе численного решения оптимизационной задачи. В связи с этим, как правило, данные штрафные функции являются невыпуклыми, что усложняет решение задачи. Пример штрафной функции $U(x, y)$ и соответствующего ей препятствия в плоскости (x, y) приведен на рисунке 4.7, оттенком дополнительно отображено значение $U(x, y)$. В [32] предложен метод деформации путей, представленных однородными кубическими B-сплайнами, с применением

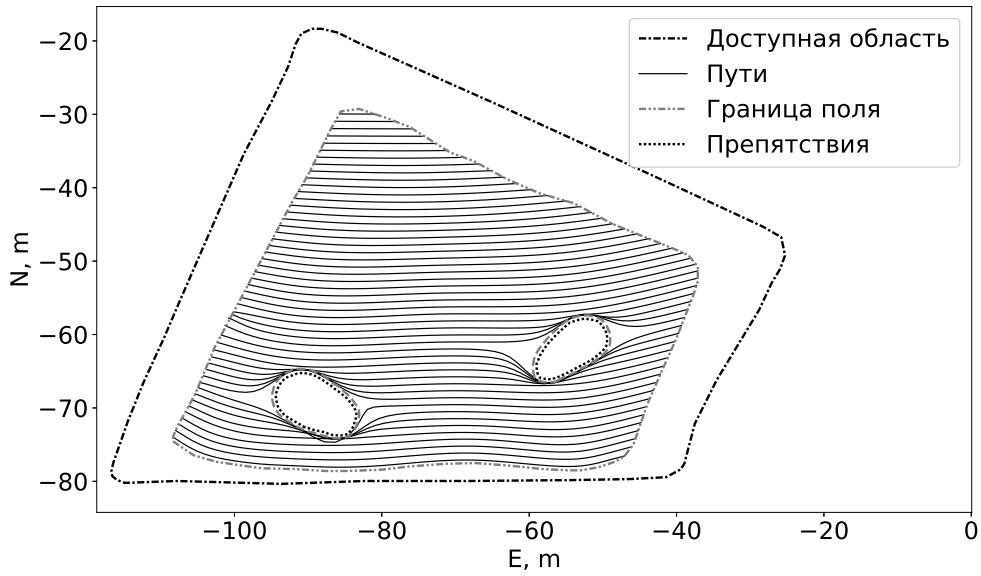


Рисунок 4.8 — Пример деформации покрывающих путей.

штрафных функций. В представленном алгоритме одновременно с деформацией происходит сглаживание кривизны получившегося пути [105].

Рассмотрим результат деформации путей, представленный на рисунке 4.8. Из данного примера видно, что в точном земледелии решение задачи деформации пути с применением барьерных функций $U(x,\ y)$ может привести к значительным пропускам в покрытии вблизи препятствий. В связи с этим требуется подход, позволяющий сокращать площадь необрабатываемой области.

4.3 Предлагаемый метод

4.3.1 Описание поверхности и путей

При работе с трехмерным ландшафтом можно использовать локальную правую декартову систему координат (x, y, z) ENU (от англ. *East, North, Up*). Начало координат выбирается в некоторой точке поля. Ось z направлена по нормали к эллипсоиду используемой модели Земли, ось x — на восток, ось y — на север, как показано на рисунке 4.9.

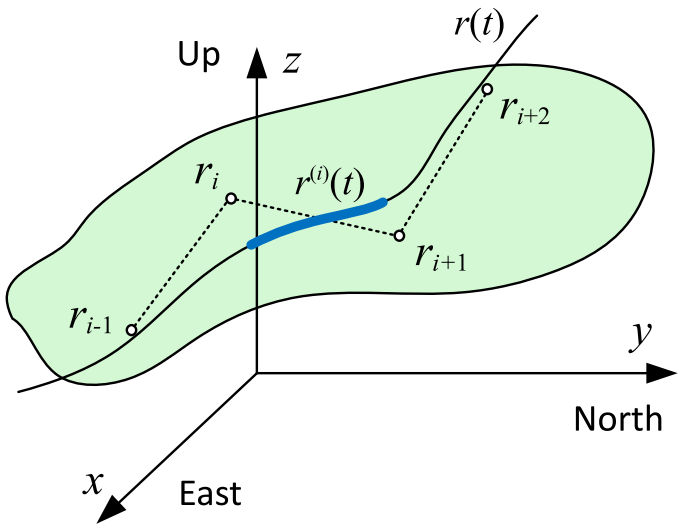


Рисунок 4.9 — Система координат ENU, сплайновая кривая $r(t)$, элементарный сплайн $r^{(i)}(t)$ и его контрольные точки.

Чтобы определить поверхность, по которой движутся роботы, необходимо задать функцию $z(x, y)$, возвращающую для известных координат x и y координату z точки на поверхности. На практике такую функцию задать точно не всегда возможно. В связи с этим используются интерполяционные методы. В работе [22] предлагается использовать билинейную или бикубическую интерполяцию. Эти два метода позволяют строить поверхность по данным о высотах в узлах равномерной сетки. В случае, если карта высот строится по данным ГНСС RTK-измерений, полученным при проезде робота по полю, то следует использовать методы, не требующие данных на равномерной сетке. Примерами являются двумерная B-сплайновая интерполяция [106] и геостатистические методы на основе кригинга [107]. Вид функции $z(x, y)$ и ее параметры определяют цифровую модель высот (англ. *digital elevation model, DEM*).

Для задания траекторий движения колесных роботов в большинстве приложений требуются гладкие функции. Примером являются однородные кубические B-сплайны, обладающие непрерывной первой и второй производной по параметру. Особенности их применения для колесных роботов рассмотрены в работе [33]. Сплайновая кривая $r(t) \in R^3$ полностью задается набором своих контрольных точек $r_1, r_2, \ldots, r_n \in R^3$. $r(t)$ состоит из набора элементарных B-сплайнов $r^{(i)}(t) \in R^3$, определяемых четырьмя соседними контрольными точками $r_{i-1}, r_i, r_{i+1}, r_{i+2} \in R^3$ по формуле

$$r^{(i)}(t) = b_0(t)r_{i-1} + b_1(t)r_i + b_2(t)r_{i+1} + b_3(t)r_{i+2}, \tag{4.3}$$

где t — параметр сплайна, $0 \leqslant t \leqslant 1$,

$$b_0(t) = \frac{(1-t)^3}{6}, \quad b_1(t) = \frac{4 - 6t^2 + 3t^3}{6},$$
$$b_2(t) = \frac{1 + 3t + 3t^2 - 3t^3}{6}, \quad b_3(t) = \frac{t^3}{6}. \tag{4.4}$$

Набор контрольных точек дополняется еще двумя точками

$$r_0 = 2r_1 - r_2,$$
$$r_{n+1} = 2r_n - r_{n-1}, \tag{4.5}$$

в результате чего концы сплайновой кривой $r(t)$ совпадают с r_1 и r_n. Элементарный сплайн $r^{(i)}(t)$ лежит в выпуклой оболочке своих четырех контрольных точек $r_{i-1}, r_i, r_{i+1}, r_{i+2}$. Сплайновая кривая $r(t)$ и ее контрольные точки могут лежать либо на поверхности, задаваемой цифровой моделью высот, либо вблизи нее. Отметим, что для случая плоского поля ($r(t) \in R^2$) формулы B-сплайнов имеют тот же вид.

4.3.2 Построение соседнего пути с возможным перекрытием

Рассмотрим построение соседней сплайновой кривой $\widetilde{r}(t)$ по заданной начальной кривой $r(t)$. Будем считать, что контрольные точки для $r(t)$ расположены на приблизительно равных расстояниях друг от друга $D_s \approx \|r_{i-1} - r_i\| \approx \|r_i - r_{i+1}\|$. Тогда максимальное отклонение контрольной точки r_i, $i = 1, \ldots, n$ от соответствующей точки кривой $r^{(i)}(0)$ не превышает $D_s^2 k_{\max}/6$, где $k_{\max}$ — максимальная кривизна траектории на данном участке (см. [108; 109]). Выбором D_s можно уменьшить значение этого отклонения.

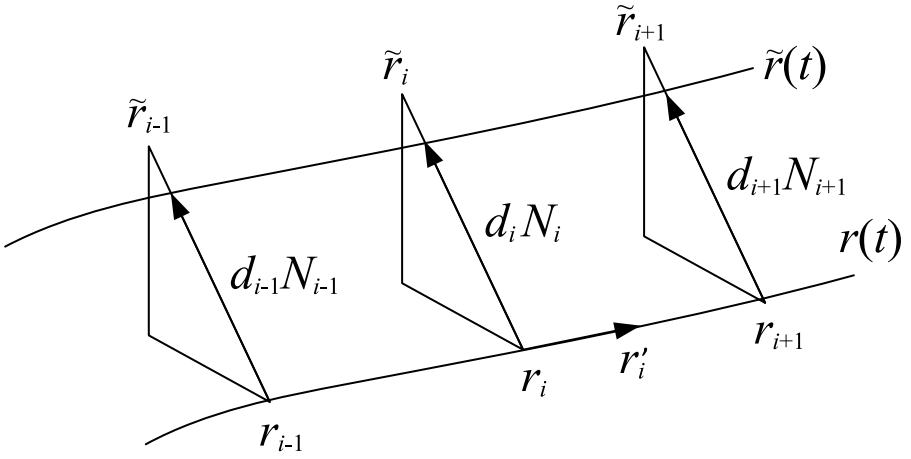


Рисунок 4.10 — Построение соседнего пути.

Обозначим как $\overline{N}_i^x$ и $\overline{N}_i^y$ первые две координаты вектора нормали к начальной кривой в точке $r^{(i)}(0)$, направленной в сторону построения следующего сплайна, такие, что длина вектора $(\overline{N}_i^x,\ \overline{N}_i^y,\ 0)$ равна $\bar{d}$. Определим $\overline{N}_i^z,\ i = 1, \ldots, n$ как

$$\overline{N}_i^z = z(x_i + \overline{N}_i^x,\, y_i + \overline{N}_i^y) - z(x_i,\, y_i). \tag{4.6}$$

Контрольные точки соседнего пути будем искать вдоль векторов $\overline{N}_i = (\overline{N}_i^x,\ \overline{N}_i^y,\ \overline{N}_i^z)$, отложенных из точек r_i. Введем единичные векторы $N_i = \overline{N}_i / \left\|\overline{N}_i\right\|$. Контрольные точки нового пути $\widetilde{r}_i$ зададим через перемененные d_i, $i = 1, \ldots n$, так, что $\widetilde{r}_i = r_i + d_i N_i$ (рисунок 4.10). Для дополнения сплайновой кривой положим $d_0 = d_{n+1} = \bar{d}$. Значение d_i соответствует смешению контрольной точки в единицах длины при построении соседнего пути. Поскольку мы должны исключить пропуски в покрытии, $d_i \leqslant \bar{d}$.

В то же время, если положить $d_i = \bar{d}$ для всех возможных i, то, как указано ранее, для следующего сплайна может нарушиться условие на кривизну (4.1), а также возникнуть самопересечение («ласточкин хвост»). По этой причине для обеспечения возможности спрямления разрешим небольшое перекрытие путей, не более $\eta\%$ по ширине. Пусть $\gamma = 1 - \eta/100$. Тогда для ограничения перекрытия нужно потребовать, чтобы $d_i \geqslant \gamma\bar{d}$.

4.3.3 Условия на нормальную кривизну траектории для плоского поля

Рассмотрим сначала случай плоского поля с декартовой системой координат (x, y). Для компонент вектора r_i будем использовать обозначения x_i, y_i, где $r_i = (x_i,\, y_i)$. Обозначим как $r^{(i)\prime}(t)$ и $r^{(i)\prime\prime}(t)$ первую и вторую производные элементарного сплайна по параметру. При движении в плоскости кривизна $k^{(i)}(t)$ трехмерной кривой $r^{(i)}(t)$ элементарного сплайна совпадает с нормальной кривизной $u^{(i)}(t)$. Можно сформулировать следующую лемму.

Лемма 4.1. *Для элементарного B-сплайна $r^{(i)}(t)$ максимальное значение $\|r^{(i)\prime\prime}(t)\|$, $0 \leqslant t \leqslant 1$ достигается при $t = 0$ или $t = 1$.*

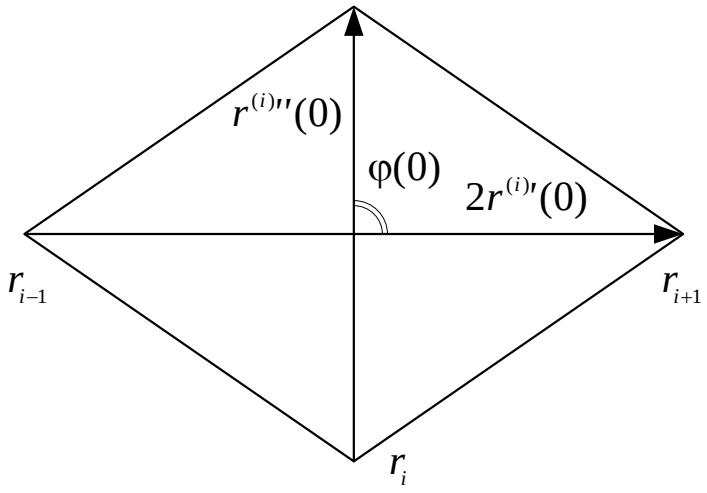


Рисунок 4.11 — Ромб, образующийся равномерно расположенными контрольными точками, и векторы $r^{(i)\prime}(t)$ и $r^{(i)\prime\prime}(t)$ при $t = 0$.

Доказательство. Из формул (4.3) и (4.4) следует, что

$$r^{(i)\prime\prime}(t) = (r_{i-1} - 2r_i + r_{i+1}) + t\,(-r_{i-1} - 2r_i + r_{i+1} + r_{i+2})\,. \tag{4.7}$$

Определим константы A, B, C и D как

$$A = x_{i-1} - 2x_i + x_{i+1}, \tag{4.8}$$

$$B = -x_{i-1} - 2x_i + x_{i+1} + x_{i+2}, \tag{4.9}$$

$$C = y_{i-1} - 2y_i + y_{i+1}, \tag{4.10}$$

$$D = -y_{i-1} - 2y_i + y_{i+1} + y_{i+2}. \tag{4.11}$$

Используя (4.7), (4.8), (4.9), (4.10) и (4.11), получаем

$$\|r^{(i)\prime\prime}(t)\|^2 = t^2\left(B^2 + D^2\right) + 2t(AB + CD) + A^2 + B^2. \tag{4.12}$$

Это означает, что $\|r^{(i)\prime\prime}(t)\|^2$ является выпуклой функцией от параметра сплайна t (прямой или параболой с ветвями вверх). Тогда хотя бы на одной из границ отрезка $[0;\ 1]$ будет достигаться максимальное значение $\|r^{(i)\prime\prime}(t)\|^2$, а значит и $\|r^{(i)\prime\prime}(t)\|$. □

Норма вектора трехмерной кривизны $k^{(i)}(t)$ в точке $r^{(i)}(t) \in R^2$ может быть определена по формуле

$$\|k^{(i)}(t)\| = \frac{\|r^{(i)\prime\prime}(t)\| \sin\varphi(t)}{\left\|r^{(i)\prime}(t)\right\|^2}, \tag{4.13}$$

где $\varphi(t)$ — угол между векторами $r^{(i)\prime}(t)$ и $r^{(i)\prime\prime}(t)$. При движении в плоскости кривизна трехмерной кривой $k^{(i)}(t)$ совпадает с нормальной кривизной $u^{(i)}(t)$.

Лемма 4.2 (необходимое условие реализуемости пути). *Если условия* $\|u^{(i)}(t)\| \leqslant u_{\max}$, $i = 1, 2, \ldots, n-1$ *выполняются для плоского однородного кубического B-сплайна с эквидистантными контрольными точками* r_i, $i = 0, 1, 2, \ldots, n+1$, *причем* $D_s = \|r_{i-1} - r_i\|$, $i = 1, 2, \ldots, n+1$, *то*

$$\|r_{i-1} - 2r_i + r_{i+1}\| \leqslant u_{\max} {D_s}^2 \tag{4.14}$$

для $i = 1, 2, \ldots, n$.

Доказательство. Поскольку контрольные точки сплайнов эквидистантны, векторы $r_{i-i} - r_i$ и $r_{i+1} - r_i$ образуют ромб (рисунок 4.11). Тогда для элементарного сплайна с индексом i угол $\varphi(0)$ является прямым и $\sin\varphi(0) = 1$. Из формулы (4.3) следует, что

$$r^{(i)\prime}(0) = -0{,}5\, r_{i-1} + 0{,}5\, r_{i+1}, \tag{4.15}$$

$$r^{(i)\prime\prime}(0) = (r_{i-1} - r_i) + (r_{i+1} - r_i). \tag{4.16}$$

В точках стыка элементарных сплайнов

$$\left\|u^{(i)}(0)\right\| = \frac{\left\|r^{(i)\prime\prime}(0)\right\|}{\left\|r^{(i)\prime}(0)\right\|^2} = \frac{4\,\|r_{i-1} - 2r_i + r_{i+1}\|}{\|r_{i+1} - r_{i-1}\|^2}. \tag{4.17}$$

Используя неравенство треугольника $\|r_{i+1} - r_{i-1}\| \leqslant 2D_s$, получаем условие (4.14). □

Если считать, что неравномерность расположения контрольных точек при переходе к соседнему пути искажается незначительно, то для новых контрольных точек можно потребовать выполнения условий

$$\left\| \begin{pmatrix} x_{i-1} + N^x_{i-1} d_{i-1} - 2(x_i + N^x_i d_i) + x_{i+1} + N^x_{i+1} d_{i+1} \\ y_{i-1} + N^y_{i-1} d_{i-1} - 2(y_i + N^y_i d_i) + y_{i+1} + N^y_{i+1} d_{i+1} \end{pmatrix} \right\| \leqslant u_{\max} D_s^2, \tag{4.18}$$

где $i = 1, 2, \ldots, n$.

Лемма 4.3 (достаточные условия). *Если для плоского однородного кубического B-сплайна с контрольными точками* $r_i + N_i d_i$, $i = 1, \ldots, n$, *и дополнением* (4.5) *выполнены условия*

$$\left\| \begin{pmatrix} x_{i-1} + N^x_{i-1} d_{i-1} - 2(x_i + N^x_i d_i) + x_{i+1} + N^x_{i+1} d_{i+1} \\ y_{i-1} + N^y_{i-1} d_{i-1} - 2(y_i + N^y_i d_i) + y_{i+1} + N^y_{i+1} d_{i+1} \end{pmatrix} \right\| \leqslant u_{\max} \hat{h}_i(d_{i-1},\, d_{i+1}), \tag{4.19}$$

$$4\hat{h}_i(d_{i-1},\, d_{i+1}) = (x_{i+1} - x_{i-1})^2 + 2(x_{i+1} - x_{i-1})(N^x_{i+1}d_{i+1} - N^x_{i-1}d_{i-1}) + \\ + (y_{i+1} - y_{i-1})^2 + 2(y_{i+1} - y_{i-1})(N^y_{i+1}d_{i+1} - N^y_{i-1}d_{i-1}), \quad (4.20)$$

где $i = 1, 2, \ldots, n$, *то для точек стыка элементарных сплайна норма вектора кривизны ограничена значением* $u_{\max}$ *:*

$$\|u^{(i)}(0)\| \leqslant u_{\max},\ i = 1,\, 2,\, \ldots, n-1. \quad (4.21)$$

Доказательство. В случае плоского сплайна вектор кривизны кривой $k(t)$ совпадает с вектором нормальной кривизны $u(t)$. Тогда евклидова норма вектора $u(t)$ может быть также определена по формуле (4.13). Векторы первой и второй производных в точках стыка элементарных сплайнов имеют вид

$$r^{(i)\prime}(0) = \frac{1}{2}\left(r_{i+1} + N_{i+1}d_{i+1} - r_{i-1} - N_{i-1}d_{i-1}\right), \quad (4.22)$$

$$r^{(i)\prime\prime}(0) = r_{i-1} + N^r_{i-1}d_{i-1} - 2(r_i + N^r_i d_i) + r_{i+1} + N^r_{i+1}d_{i+1}. \quad (4.23)$$

Заметим, что функция $\hat{h}_i(d_{i-1},\, d_{i+1})$, определяемая формулой (4.20), является оценкой снизу для $\left\|r^{(i)\prime}(0)\right\|^2$, не учитывающей квадратичные слагаемые по смещению:

$$\left\|r^{(i)\prime}(0)\right\|^2 = \hat{h}_i(d_{i-1},\, d_{i+1}) + \frac{1}{4}(N^x_{i+1}d_{i+1} - N^x_{i-1}d_{i-1})^2 + \frac{1}{4}(N^y_{i+1}d_{i+1} - N^y_{i-1}d_{i-1})^2. \quad (4.24)$$

Тогда из (4.19) следует, что

$$\left\|r^{(i)\prime\prime}(0)\right\| \sin\varphi(0) \leqslant u_{\max}\left\|r^{(i)\prime}(0)\right\|^2, \quad (4.25)$$

а значит, в точках стыка элементарных сплайнов условие (4.21) выполнено. □

Оценка квадрата нормы вектора первой производной на стыках элементарных сплайнов $\hat{h}_i(d_{i-1},\, d_{i+1})$ предполагает, что смешения контрольных точек малы. В случае построения соседнего пути можно предположить, что малыми будут величины $\bar{d} - d_i$. Тогда можно вести следующую лемму, аналогичную лемме 4.3, использующую другую нижнюю оценку для $\left\|r^{(i)\prime}(0)\right\|^2$.

Лемма 4.4 (достаточные условия). *Если для плоского однородного кубического B-сплайна с контрольными точками* $r_i + N_i d_i$, $i = 1, \ldots, n$, *и*

дополнением (4.5) *выполнены условия*

$$\left\| \begin{pmatrix} x_{i-1} + N^x_{i-1} d_{i-1} - 2(x_i + N^x_i d_i) + x_{i+1} + N^x_{i+1} d_{i+1} \\ y_{i-1} + N^y_{i-1} d_{i-1} - 2(y_i + N^y_i d_i) + y_{i+1} + N^y_{i+1} d_{i+1} \end{pmatrix} \right\| \leqslant u_{\max} \hat{l}_i(d_{i-1},\, d_{i+1}), \tag{4.26}$$

$$\begin{aligned} 4\hat{l}_i(d_{i-1},\, d_{i+1}) &= \left(x_{i+1} + \bar{d}N^x_{i+1} - x_{i-1} - \bar{d}N^x_{i-1}\right)^2 + \\ &+ 2\left(x_{i+1} + \bar{d}N^x_{i+1} - x_{i-1} - \bar{d}N^x_{i-1}\right)\left(N^x_{i-1}(\bar{d} - d_{i-1}) - N^x_{i+1}(\bar{d} - d_{i+1})\right) + \\ &+ \left(y_{i+1} + \bar{d}N^y_{i+1} - y_{i-1} - \bar{d}N^y_{i-1}\right)^2 + \\ &+ 2\left(y_{i+1} + \bar{d}N^y_{i+1} - y_{i-1} - \bar{d}N^y_{i-1}\right)\left(N^y_{i-1}(\bar{d} - d_{i-1}) - N^y_{i+1}(\bar{d} - d_{i+1})\right), \end{aligned} \tag{4.27}$$

где $i = 1, 2, \ldots, n$, *то для точек стыка элементарных сплайна норма вектора кривизны ограничена значением* $u_{\max}$ *:*

$$\|u^{(i)}(0)\| \leqslant u_{\max},\ i = 1,\, 2,\, \ldots,\, n-1. \tag{4.28}$$

Доказательство. Как и для леммы 4.3, в случае плоского сплайна нормы векторов нормальной кривизны и трехмерной кривизны совпадают, векторы первой и второй производных в точках стыка элементарных сплайнов определяются формулами (4.22) и (4.23). При этом

$$4\left\|r^{(i)\prime}(0)\right\|^2 = \|r_{i+1} + N_{i+1}d_{i+1} - r_{i-1} - N_{i-1}d_{i-1}\|^2. \tag{4.29}$$

Обозначим $q_i = \bar{d} - d_i$, $i = 0, \ldots, n+1$. Тогда

$$4\left\|r^{(i)\prime}(0)\right\|^2 = \left\|r_{i+1} + N_{i+1}(\bar{d} - q_{i+1}) - r_{i-1} - N_{i-1}(\bar{d} - q_{i+1})\right\|^2. \tag{4.30}$$

Раскрывая скобки, получаем

$$4\left\|r^{(i)\prime}(0)\right\|^2 = \left\|r_{i+1} + \bar{d}N_{i+1} - r_{i-1} - \bar{d}N_{i-1} + N_{i-1}q_{i-1} - N_{i+1}q_{i+1}\right\|^2. \tag{4.31}$$

Формула (4.31) может быть представлена в виде

$$4\left\|r^{(i)\prime}(0)\right\|^2 = \left\| \begin{pmatrix} x_{i+1} + \bar{d}N^x_{i+1} - x_{i-1} - \bar{d}N^x_{i-1} + N^x_{i-1}q_{i-1} - N^x_{i+1}q_{i+1} \\ y_{i+1} + \bar{d}N^y_{i+1} - y_{i-1} - \bar{d}N^y_{i-1} + N^y_{i-1}q_{i-1} - N^y_{i+1}q_{i+1} \end{pmatrix} \right\|^2. \tag{4.32}$$

Тогда

$$\begin{aligned}4\left\|r^{(i)\prime}(0)\right\|^2 =&\left(x_{i+1}+\bar{d}N^x_{i+1}-x_{i-1}-\bar{d}N^x_{i-1}\right)^2+\\&+2\left(N^x_{i-1}q_{i-1}-N^x_{i+1}q_{i+1}\right)\left(x_{i+1}+\bar{d}N^x_{i+1}-x_{i-1}-\bar{d}N^x_{i-1}\right)+\\&+\left(N^x_{i-1}q_{i-1}-N^x_{i+1}q_{i+1}\right)^2+\\&+\left(y_{i+1}+\bar{d}N^y_{i+1}-y_{i-1}-\bar{d}N^y_{i-1}\right)^2+\\&+2\left(N^y_{i-1}q_{i-1}-N^y_{i+1}q_{i+1}\right)\left(y_{i+1}+\bar{d}N^y_{i+1}-y_{i-1}-\bar{d}N^y_{i-1}\right)+\\&+\left(N^y_{i-1}q_{i-1}-N^y_{i+1}q_{i+1}\right)^2.\end{aligned}\tag{4.33}$$

Используя формулу (4.27), получаем, что

$$\left\|r^{(i)\prime}(0)\right\|^2=\hat{l}_i(d_{i-1},\,d_{i+1})+\frac{1}{4}\left(N^x_{i-1}q_{i-1}-N^x_{i+1}q_{i+1}\right)^2+\frac{1}{4}\left(N^y_{i-1}q_{i-1}-N^y_{i+1}q_{i+1}\right)^2.\tag{4.34}$$

Заметим, что $\hat{l}_i(d_{i-1},\,d_{i+1})$ — нижняя оценка для квадрата нормы первой производной в начале i-ого элементарного сплайна. Из (4.26) следует, что

$$\left\|r^{(i)\prime\prime}(0)\right\|\sin\varphi(0)\leqslant u_{\max}\left\|r^{(i)\prime}(0)\right\|^2.\tag{4.35}$$

Тогда в точках стыка элементарных сплайнов условие (4.28) выполнено. □

Заметим, что множества допустимых значений условий (4.18) и (4.19) являются конусами второго порядка. Их правая часть является константой либо линейной функцией по переменным d_{i-1} и d_{i+1}. Из леммы 4.1 следует, что числитель дроби (4.13) принимает максимальное значение на границах элементарного сплайна. Длина вектора $r^{(i)\prime}(t)$ остается примерно постоянной величиной для элементарного сплайна и отражает соответствие между малым изменением длины траектории и соответствующим ему изменении параметра сплайна t. В связи с этим для практической реализации можно требовать лишь выполнения достаточного условия на кривизну в точках стыка элементарных сплайнов.

4.3.4 Условия на нормальную кривизну траектории для пологого поля

Для пологого поля длины векторов нормальной кривизны и трехмерной кривизны могут отличаться. Предположим, что ландшафт является достаточ-

но пологим: в окрестности радиусом $\max(D_s, \bar{d})$ его поверхность можно описать наклонной плоскостью, т.е. главные кривизны поверхности достаточно малы. Тогда значения $\|u^{(i)}(t)\|$ можно оценить модулем вектора кривизны сплайна, формирующегося при проецировании четверки контрольных точек на эту наклонную плоскость. Пусть r'_i — единичный вектор касательной для точки $r^{(i)}(0)$ на исходной кривой. Введем двумерную декартову систему координат $(\widetilde{x},\ \widetilde{y})$, оси которой $\widetilde{x}$ и $\widetilde{y}$ сонаправлены с векторами r'_i и N_i соответственно. В системе $(\widetilde{x},\ \widetilde{y})$ можно записать следующие условия, соответствующие (4.18), (4.19) и (4.34).

$$\left\| \begin{pmatrix} \widetilde{x}_{i-1} + \widetilde{N}^x_{i-1} d_{i-1} + \widetilde{x}_{i+1} + \widetilde{N}^x_{i+1} d_{i+1} \\ \widetilde{y}_{i-1} + \widetilde{N}^y_{i-1} d_{i-1} - 2d_i + \widetilde{y}_{i+1} + \widetilde{N}^y_{i+1} d_{i+1} \end{pmatrix} \right\| \leqslant u_{\max} D_s^2, \tag{4.36}$$

$$\left\| \begin{pmatrix} \widetilde{x}_{i-1} + \widetilde{N}^x_{i-1} d_{i-1} + \widetilde{x}_{i+1} + \widetilde{N}^x_{i+1} d_{i+1} \\ \widetilde{y}_{i-1} + \widetilde{N}^y_{i-1} d_{i-1} - 2d_i + \widetilde{y}_{i+1} + \widetilde{N}^y_{i+1} d_{i+1} \end{pmatrix} \right\| \leqslant u_{\max} \widetilde{h}_i(d_{i-1},\, d_{i+1}), \tag{4.37}$$

$$\begin{aligned} 4\widetilde{h}_i(d_{i-1},\, d_{i+1}) &= (\tilde{x}_{i+1} - \widetilde{x}_{i-1})^2 + 2(\widetilde{x}_{i+1} - \tilde{x}_{i-1})(\widetilde{N}^x_{i+1} d_{i+1} - \widetilde{N}^x_{i-1} d_{i-1}) + \\ &+ (\widetilde{y}_{i+1} - \widetilde{y}_{i-1})^2 + 2(\widetilde{y}_{i+1} - \widetilde{y}_{i-1})(\widetilde{N}^y_{i+1} d_{i+1} - \widetilde{N}^y_{i-1} d_{i-1}), \end{aligned} \tag{4.38}$$

$$\left\| \begin{pmatrix} \widetilde{x}_{i-1} + \widetilde{N}^x_{i-1} d_{i-1} + \widetilde{x}_{i+1} + \widetilde{N}^x_{i+1} d_{i+1} \\ \widetilde{y}_{i-1} + \widetilde{N}^y_{i-1} d_{i-1} - 2d_i + \widetilde{y}_{i+1} + \widetilde{N}^y_{i+1} d_{i+1} \end{pmatrix} \right\| \leqslant u_{\max} \widetilde{l}_i(d_{i-1},\, d_{i+1}), \tag{4.39}$$

$$\begin{aligned} 4\widetilde{l}_i(d_{i-1},\, d_{i+1}) &= \left(\widetilde{x}_{i+1} + \bar{d}\widetilde{N}^x_{i+1} - \widetilde{x}_{i-1} - \bar{d}\widetilde{N}^x_{i-1}\right)^2 + \\ &+ 2\left(\widetilde{x}_{i+1} + \bar{d}\widetilde{N}^x_{i+1} - \widetilde{x}_{i-1} - \bar{d}\widetilde{N}^x_{i-1}\right)\left(\widetilde{N}^x_{i-1}(\bar{d} - d_{i-1}) - \widetilde{N}^x_{i+1}(\bar{d} - d_{i+1})\right) + \\ &+ \left(\widetilde{y}_{i+1} + \bar{d}\widetilde{N}^y_{i+1} - \widetilde{y}_{i-1} - \bar{d}\widetilde{N}^y_{i-1}\right)^2 + \\ &+ 2\left(\widetilde{y}_{i+1} + \bar{d}\widetilde{N}^y_{i+1} - \widetilde{y}_{i-1} - \bar{d}\widetilde{N}^y_{i-1}\right)\left(\widetilde{N}^y_{i-1}(\bar{d} - d_{i-1}) - \widetilde{N}^y_{i+1}(\bar{d} - d_{i+1})\right), \end{aligned} \tag{4.40}$$

где для каждого элементарного сплайна константы $(\widetilde{x}_{i-1},\, \widetilde{y}_{i-1})$ и $(\widetilde{x}_{i+1},\, \widetilde{y}_{i+1})$ — координаты в $(\widetilde{x},\ \widetilde{y})$ соседних проецируемых точек исходной кривой r_{i-1} и r_{i+1}, а $(\widetilde{N}^x_{i-1},\, \widetilde{N}^y_{i-1})$ и $(\widetilde{N}^x_{i+1},\, \widetilde{N}^y_{i+1})$ — проекции на плоскость $(\widetilde{x},\ \widetilde{y})$ векторов нормалей N_{i-1} и N_{i+1}. Их можно определить по формулам

$$(\widetilde{x}_{i-1},\, \widetilde{y}_{i-1}) = (\langle\, r_{i-1} - r_i, r'_i\rangle,\, \langle\, r_{i-i} - r_i, N_i\rangle)\,, \tag{4.41}$$

$$(\widetilde{x}_{i+1},\, \widetilde{y}_{i+1}) = (\langle\, r_{i+1} - r_i, r'_i\rangle,\, \langle\, r_{i+1} - r_i, N_i\rangle)\,, \tag{4.42}$$

$$\left(\widetilde{N}^x_{i-1},\, \widetilde{N}^y_{i-1}\right) = (\langle\, N_{i-1}, r'_i\rangle,\, \langle\, N_{i-1}, N_i\rangle)\,, \tag{4.43}$$

$$\left(\widetilde{N}^x_{i+1},\, \widetilde{N}^y_{i+1}\right) = (\langle\, N_{i+1}, r'_i\rangle,\, \langle\, N_{i+1}, N_i\rangle)\,. \tag{4.44}$$

Важно отметить, что координаты, определенные по формулам (4.41) — (4.44) можно использовать лишь для точки i: у каждой точки своя наклонная плоскость $(\widetilde{x},\ \widetilde{y})$ и для других точек эти проекции могут отличаться. В формулах (4.41) — (4.44) индексы $i-1$ и $i+1$ обозначает лишь предыдущую и следующую контрольную точку.

4.3.5 Задача конического программирования второго порядка построения покрытия поля путями

Новые соседние пути предлагается получать как решение следующей задачи конического программирования второго порядка.

Задача 4.1 (построения соседнего пути в виде SOCP). *Найти*

$$\min_{u,d_0,\dots d_{n+1}} \beta\frac{\|u\|}{u_{\max}} - (\bar{d})^{-1}\sum_{i=0}^{n+1} d_i, \tag{4.45}$$

где $u=(u_1,u_2,\dots,u_n)$, *при ограничениях* (4.39) *для* $i=1,\dots,n$,

$$\gamma\bar{d} \leqslant d_i \leqslant \bar{d},\ i=0,1,\dots,n+1, \tag{4.46}$$

$$0 \leqslant u_i \leqslant u_{\max},\ i=1,\dots,n, \tag{4.47}$$

$$\left\| \begin{pmatrix} \widetilde{x}_{i-1}+\widetilde{N}^x_{i-1}d_{i-1}+\widetilde{x}_{i+1}+\widetilde{N}^x_{i+1}d_{i+1} \\ \widetilde{y}_{i-1}+\widetilde{N}^y_{i-1}d_{i-1}-2d_i+\widetilde{y}_{i+1}+\widetilde{N}^y_{i+1}d_{i+1} \end{pmatrix} \right\| \leqslant u_i D_s^2,\ i=1,\dots,n. \tag{4.48}$$

В задаче 4.1 параметр β ($\beta \geqslant 0$) является безразмерным и отвечает за компромисс между спрямлением траекторий и перекрытием соседних дорожек. После того, как путь получен, он параметризуется эквидистантными контрольными точками. Затем относительно него строится следующий путь. Процедура повторяется до тех пор, пока все поле не будет покрыто. После того, как все пути построены, производится их обрезка на границе поля.

Если задача 4.1 оказалась несовместной, то это означает, что указанным методом при данном начальном пути и параметре β построение путей невозможно. В таком случае нужно либо выбрать другую начальную кривую, либо, уменьшая γ, разрешить большее по ширине перекрытие дорожек. Значение $\gamma=0$ всегда отвечает совместной задаче при корректном задании параметров.

4.3.6 Деформация путей для учета препятствий

Предположим, что покрытие поля путями построено без учета препятствий. Требуется деформировать пересекающие препятствия пути так, чтобы они стали реализуемыми. Направления N_i поиска новых положений контрольных точек выберем точно также, как и для построения соседнего пути (рисунок 4.12). d_i будут отвечать смещениям контрольных точек вдоль этих направлений. Для дополнения сплайновой кривой положим $d_0 = d_{n+1} = 0$.

Множества допустимых значений ограничений на кривизну (4.37) являются конусами второго порядка. Правая часть этих условий линейно зависит от значений переменных d_i, $i = 1, \ldots, n$. Это позволяет сформулировать задачу деформации пути с ограничением на кривизну в виде следующей задачи конического программирования второго порядка.

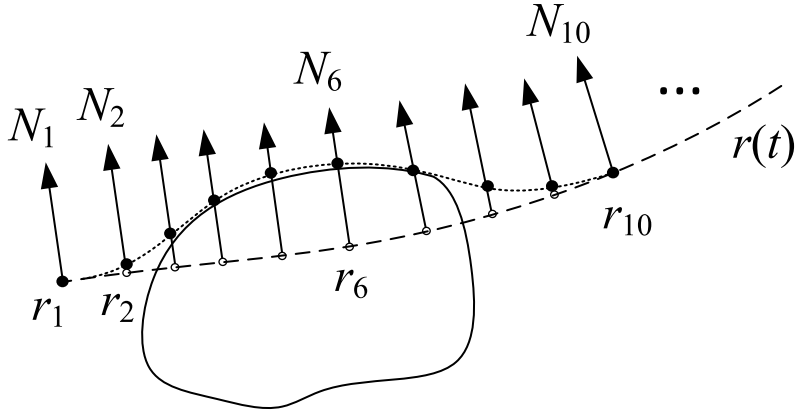


Рисунок 4.12 — Сплайновая кривая $r(t)$ до (штриховая линия) и после (пунктирная линия) деформации пути, контрольные точки r_i, направления поиска N_i. Контур препятствия отображен сплошной линией.

Задача 4.2 (деформации пути в виде SOCP). *Найти*

$$\min_{q} \; q_{n+1} \tag{4.49}$$

при ограничении (4.37) *при* $q_i = d_i$ *для* $i = 1, \ldots, n$,

$$\left\| w^{\mathrm{T}} q \right\| \leqslant q_{n+1}, \tag{4.50}$$

$$\overline{d_i} \leqslant q_i \leqslant \overline{\overline{d_i}}, \; i = 1, \; \ldots, \; n, \tag{4.51}$$

где q_i *— i-я компонента вектора* $q \in R^{n+1}$.

Для оптимального решения задачи первые n компонент вектора q равны смещениям контрольных точек, которые требуется провести: $d_i = q_i$, $i = 1, \ldots, n$. Значения $\overline{d_i}$ и $\overline{\overline{d_i}}$, $i = 1, \ldots, n$ определяют геометрические ограничения на отклонения контрольных точек вдоль направления N_i, обусловленные препятствиями и границами поля. При этом для каждого препятствия выбирается направление обхода. Первые n компонент вектора $w \in R^{n+1}$ являются фиксированными значениями весовой функции, которые могут быть выбраны как одинаковыми, так и зависящими от расположения контрольных точек относительно препятствий. Последняя компонента вектора w равна нулю: $w_{n+1} = 0$. Критерий оптимизации (4.49) в сочетании с условием (4.50) позволяет минимизировать смещения контрольных точек и сократить пропуски в покрытии, образующиеся при объезде препятствий из-за ограничения на нормальную кривизну пути.

На практике для определения значений $\overline{d_i}$ и $\overline{\overline{d_i}}$, $i = 1, \ldots, n$ можно использовать представление границ препятствий и контура поля замкнутыми ломаными линиями, проекции которых на плоскость локального горизонта будут представлять собой многоугольники. В плоскости локального горизота поиск $\overline{d_i}$ и $\overline{\overline{d_i}}$ сводится к выбору направления обхода каждого препятствия и решению задачи о пересечении прямой и многоугольника.

Задача оптимизации 4.2 является выпуклой и допускает вычислительно эффективные методы решения. Другой вычислительной особенностью данной задачи является то, что вывод о совместности или несовместности ограничений может быть сделан в процессе решения. Отметим, что предлагаемый подход не нацелен на поиск направлений обхода препятствий, а служит инструментом построения путей в заданной выпуклой области, определенной ограничениями (4.51). Если задача деформации пути оказалась несовместной, а допустимый путь в заданных границах существует, то это может быть обусловлено невозможностью выполнения условий на кривизну при смещении точек вдоль выбранных направлений. Невозможность выполнения ограничений на кривизну может быть обусловлена тем, что при больших смещениях нарушается равномерность расположения контрольных точек. Кроме того, проекции на плоскость (x, y) направлений поиска N_i могут пересекаться, что приводит к решениям с самопересекающимися траекториями, для которых, как правило, нарушается условие на кривизну. Перечисленные проблемы можно устранить применением следующего алгоритма деформации пути, основанного на ослаб-

лении ограничений задачи с последующим представлением пути равномерно расположенными контрольными точками.

Алгоритм 4.1 (деформации пути для учета препятствий).

1. Решается задача 4.2 при исходном ограничении $u_{\max}$. Если решение найдено, то производится перепараметризация траектории равномерно расположенными контрольными точками. Задача 4.2 решается повторно для текущего расположения контрольных точек. Если решение найдено, то оно возвращается в качестве ответа.
2. Решается задача 4.2 для текущего расположения контрольных точек для максимального значения нормальной кривизны $u_{\max} + \Delta$, $\Delta > 0$. Если оптимальное решение найдено, то производится перепараметризация траектории равномерно расположенными контрольными точками и выполняется возврат на шаг 1. Если не найдено, то значение Δ увеличивается, после чего повторяются действия шага 2.

Алгоритм 4.1 на каждом этапе своей работы предполагает выбор изменения кривизны Δ. От того, как этот выбор произведен, зависит, сколько итераций алгоритма потребуется выполнить, прежде чем решение задачи будет найдено.

4.4 Маршрутизация

После построения покрытия поля нужно определить маршрут движения одного или нескольких роботов по полученным рядам, позволяющий произвести обработку за минимальное время. Предположим, что на поле одновременно начинают работу M роботов, движущиеся с постоянной скоростью. Требуется найти такой порядок обхода N рядов, чтобы время выполнения всех работ было минимально.

Задача 4.3 (маршрутизации нескольких роботов). *Найти*

$$\min_{p \in P} \max_{m \in \{1,\dots M\}} D_m(p), \tag{4.52}$$

где m — порядковый номер робота, $m = 1, \dots, M$, p — перестановка порядка обхода рядов, P — множество всевозможных перестановок порядка обхода рядов, $D_m(p)$ — длина маршрута робота m для порядка обхода p.

Информация о направлении движения по ряду включается в порядок обхода. Основную сложность в решении задачи 4.3 составляет то, что мощность множества P быстро растет при увеличении числа рядов N. В итоге перебор всевозможных перестановок часто оказывается нереализуем за приемлемое время. В случае одного робота задача маршрутизации может быть представлена в виде задачи коммивояжера (англ. *travelling salesman problem, TSP*) без возврата в исходную точку пути, а в случае нескольких роботов — в виде задачи VRP (от англ. *vehicle routing problem*) [110].

В процессе решения задачи маршрутизации требуется определять длину разворота между концами рядов. Рассмотрим случай плоского поля. Если пути реализуются машиной с дифференциальным приводом, то кратчайший по длине разворот может быть осуществлен по прямой. Если используется машина с механизмом руления поворотом передних колес, то кривизна путей должна быть ограничена условием (4.1). Задача построения разворота с одного ряда на другой эквивалента поиску кратчайшей траектории, кривизна которой ограничена условием (4.1), позволяющей из граничной точки одного ряда попасть на граничную точку другого, причем конечная и начальная ориентации робота заданы. Предположим сначала, что движение задним ходом не используется. В работе Л. Э. Дубинса [111] доказано, что искомая траектория состоит из отрезков прямых и сегментов окружностей (радиусом $1/u_{\max}$) и может быть описана одним из шести слов «lsl», «lsr», «rlr», «lrl», «rsr», «rsl», где «l» означает движение с колесами, максимально вывернутыми влево, «r» – с колесами, максимально вывернутыми вправо, «s» – по прямой. Таким образом, для поиска оптимального разворота достаточно рассмотреть шесть траекторий (если они реализуемы). Машину, реализующую такие траектории, в литературе называют машиной Дубинса. В работе [112] анализируется выбор оптимальной траектории из приведенных шести. В работе Д. Ридса и Л. Шеппа [113] рассматриваются оптимальные по длине траектории, учитывающие возможность движения задним ходом (машина Ридса-Шеппа). Однако оптимальные траектории, как правило, содержат скачкообразные изменения кривизны. Динамика приводов при повороте колес у реальных сельскохозяйственных роботов может не позволять точную реализацию оптимальных траекторий, которую способны описать машины Дубинса или Ридса-Шеппа. В связи с этим может оказаться актуальным построение траекторий с непрерывно меняющейся кривизной, рассмотренное, например, в работе [114]. Поскольку сельскохозяйственные поля

часто достаточно пологие, то для них можно проецировать траектории разворота, построенные в плоскости локального горизонта. Отметим, что выбор способа маневра может также быть обусловлен геометрическими ограничениями — неподвижными и подвижными препятствиями.

Предположим, что длины разворотов определены и требуется найти маршруты движения одного или нескольких роботов. Поскольку точное решение задачи маршрутизации получить за приемлемое время невозможно, требуются приближенные методы. На практике для этого часто используются метаэвристические алгоритмы. Алгоритмы маршрутизации сельскохозяйственных машин, основанные на методе имитации отжига (англ. *simulated annealing, SA*), предложены, в частности, в работах [115; 116]. Метод SA основан на физической аналогии: в отжиге металлов при понижении температуры снижается вероятность перехода атомов в состояния с большей энергией. Приведем алгоритм, который был построен на основе SA для проведения вычислительных экспериментов данной работы. Алгоритм предназначен для маршрутизации двух роботов ($M = 2$), но он может быть обобщен на большее количество машин. Обозначим $E(p)$ («энергия» в терминах SA) — длина самого длинного маршрута для заданной перестановки порядка обхода рядов:

$$E(p) = \max_{m \in \{1,\dots M\}} D_m(p). \tag{4.53}$$

Поясним, как можно интерпретировать перестановку рядов p с позицией разделителя s на примере, изображенном на рисунке 4.13. Начальные точки маршрутов обозначены как S_1 и S_2, а конечные — как F_1 и F_2. Предполагается, что ограничение на кривизну путей такое, что для перехода на соседний ряд необходимо совершить Ω-образный разворот, а на дальние ряды — U-образный. Пусть $p = (1, 3, 5, 4, 2, 6, 8, 10, 7, 9)$ (число рядов $M = 10$), $s = 4$. Последовательные элементы вектора, начиная с $s = 4$ до последнего относятся к маршруту второго робота ($p_2 = (2, 6, 8, 10, 7, 9)$), а оставшиеся — первого робота ($p_1 = (1, 3, 5, 4)$). Для простых форм поля, для которых в длинных разворотах от одного края до другого нет необходимости, p_1 задает всего два маршрута, отличающиеся тем, с какой из двух граничных точек первого ряда начинается обход. Аналогично p_2 задает только два порядка обхода. Для каждого робота выберем кратчайшее направление обхода. Значение энергии $E(p)$ для заданного p будем определять следующим образом.

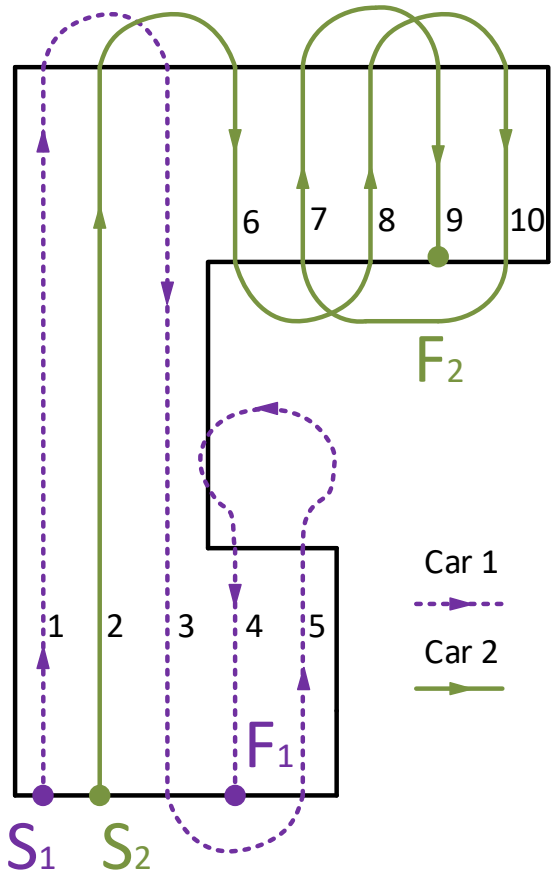


Рисунок 4.13 — Интерпретация перестановки рядов в задаче маршрутизации.

Алгоритм 4.2 (расчета значения энергии $E(p)$).

1. Выбрать начальную позицию разделителя $s = \lfloor N/2 \rfloor$.
2. Определить заданные перестановкой p и позицией разделителя s минимальные длины маршрутов D_1 и D_2 для первой и второй машины соответственно. Первые s компонент относятся к маршруту первого робота и оставшиеся $(M - s)$ — второго.
3. Если $D_1 > D_2$, то $s := s - 1$ иначе $s := s + 1$.
4. Если предыдущее изменение позиции разделителя s было в противоположном направлении (увеличении либо уменьшении на 1), то выбрать $E(p) = \max(D_1, D_2)$, иначе вернуться на шаг 2.

Приведем алгоритм SA. В ходе работы алгоритма SA изменяется параметр эвристики T («температура») от начального значения T_0 до минимального $T_{\min}$. Если новое найденное решение p имеет энергию $E(p)$ больше, чем у предыдущего, то переход к этому решению осуществляется с вероятностью, зависящей от текущей температуры. Если новое решение имеет меньшую энергию, то переход к нему реализуется всегда, не зависимо от текущей температуры.

Алгоритм 4.3 (маршрутизации на основе SA).

1. Выбрать начальный вектор p_0, например, $p_0 = (1, 2, \ldots, N)$, начальную температуру $T = T_0$ и минимальную температуру $T_{\min}$. Вычислить $E(p_0)$. Для счетчика итераций положить $iter := 1$.
2. Цикл SA.
   а) Выбрать случайные (псевдослучайные) значения i и j, $i < j$ и получить перестановку $\widetilde{p}$ с помощью инверсии порядка элементов вектора p_0 от i-ой до j-ой компоненты.
   б) Если $E(p_0) > E(\widetilde{p})$, то выбрать $p_0 = \widetilde{p}$, иначе выбрать $p_0 = \widetilde{p}$ с вероятностью $P = \exp(-(E(\widetilde{p}) - E(p_0))/T)$.
   в) Положить $iter := iter + 1$, $T := 0{,}1T_{\min}/iter$. Если $T \leqslant T_{\min}$, то выйти из цикла SA.
3. Выбрать из цикла SA решение p с минимальным значением $E(p)$.

Другой метаэвристикой, которую можно использовать для приближенного решения задачи маршрутизации, является метод муравьиной колонии (англ. *ant colony optimization, ACO*). Он основан на наблюдении за путями муравьев до источника пищи. На более коротких путях муравьи при возвращении оставляют большее количество феромона, чем на длинных. Феромонные тропы позволяют ориентироваться другим муравьям. Кроме того, феромон испаряется с течением времени. Примеры применения ACO к задачам маршрутизации можно найти, например, в работах [117—119].

Приведем алгоритм на основе метода ACO, который был разработан для проведения вычислительных экспериментов данной работы. Пронумеруем граничные точки рядов $i = 1, 2, \ldots, 2N$ так, что $(i \mod N)$ — номер ряда. Если $i \leqslant N$, то точка i соответствует началу ряда, иначе — концу ряда. Разворот из точки i в точку j будем обозначать как $i \to j$. Предположим, что для всех разворотов известна длина d_{ij}, а также значение a_{ij}, равное единице, если разворот разрешен, и нулю иначе, $a_{ii} = 0$. Переменными являются матрица концентрации феромона τ_{ij} и матрица текущего накопленного обновления феромона $\Delta\tau_{ij}$. Параметрами ACO являются максимальное число итераций $I_{\max}$, число итераций между последовательными обновлениями матрицы концентраций I_{upd}, степень влияния количества феромона на выбор пути α, степень влияния длин разворотов β, коэффициент испарения феромона ρ, количество феромона Q. Предположим, что роботы (муравьи) движутся с единичной ско-

ростью так, что время движения численно равно пройденному пути. Порядок обхода рядов может быть определен по следующему алгоритму.

Алгоритм 4.4 (маршрутизации на основе ACO).

1. Инициализировать начальные позиции роботов s_k, $k = 1, \ldots, M$ (выбранными значениями от 1 до $2N$), текущий рекорд $L = \infty$, текущую матрицу концентрации феромона τ_{ij} (константами), матрицу текущего обновления феромона $\Delta\tau_{ij}$ (нулями). Следующие операции (шаги 2–7) повторить $I_{\max}$ раз.
2. Установить текущие позиции p_k роботов (муравьев) на старт: $p_k := s_k$, $k = 1, \ldots, M$. Назначить обработку первого ряда для каждого робота.
3. Пока все ряды не обработаны, выполнять следующий цикл.
   а) Выбрать робота k, который раньше всех закончит назначенные ему работы.
   б) Для каждого возможного перехода $p_k \to j$ данного робота назначить вес по формуле $w_{p_kj} := {\tau_{p_kj}}^{\alpha} / d_{p_kj}^{\beta}$.
   в) Выбрать для робота k следующую точку маршрута j с вероятностью, прямо пропорциональной w_{p_kj}. Назначить обработку ряда с граничной точкой j.
4. Определить время одновременного выполнения всех работ на данной итерации алгоритма L_{iter}.
5. Если $L_{iter} < L$, то $L := L_{iter}$ и сохранить текущую последовательность обхода рядов.
6. Для всех реализованных на данной итерации переходов $i \to j$ выполнить
$$\Delta\tau_{ij} := \Delta\tau_{ij} + \frac{Q}{L_{iter} - LB + \varepsilon}, \tag{4.54}$$
где ε — малый параметр (например, $\varepsilon = 0{,}001$), LB — нижняя оценка времени одновременного выполнения всех работ (или длины самого длинного маршрута).
7. Если номер итерации кратен I_{upd}, то для всех переходов $i \to j$ выполнить обновление матрицы концентрации: $\tau_{ij} := \tau_{ij} + \Delta\tau_{ij}$.

Несмотря на то, что алгоритм является приближенным, разница между значениями получившегося времени выполнения работ и нижней оценки является гарантированной точностью решения. Остается только определить способ вычисления нижней оценки LB.

Рассмотрим случай одного робота с началом маршрута в точке S. Введем бинарные переменные x_{ij}, такие, что значения $x_{ij} = 1$ соответствуют тому, что переход $i \to j$ реализован, и $x_{ij} = 0$ — не реализован. Пусть $\bar{G}$ — множество всех граничных точек рядов. Рассмотрим следующую задачу целочисленного линейного программирования.

Задача 4.4 (маршрутизации в виде MILP). *Найти*

$$\min_{x_{ij},\, i=1,\ldots,2N,\, j=1,\ldots,2N} \sum_{i=1}^{2N}\sum_{j=1}^{2N} d_{ij}a_{ij}x_{ij} \tag{4.55}$$

при ограничениях

$$x_{ij} \in \{0,\ 1\},\ \ i,j = 1,\ \ldots,\ 2N, \tag{4.56}$$

$$0 \leqslant x_{ij} \leqslant a_{ij},\ \ i,j = 1,\ \ldots,\ 2N, \tag{4.57}$$

$$\sum_{i=1}^{2N} a_{ij}x_{ij} = 1,\ \ \forall j \in \{1,\ \ldots,\ 2N\}/S, \tag{4.58}$$

$$\sum_{i=1}^{2N} a_{ij}x_{ij} \leqslant 1,\ \ j = 1,\ \ldots,\ 2N, \tag{4.59}$$

$$x_{i(i+N)}a_{i(i+N)} + x_{(i+N)i}a_{(i+N)i} = 1,\ \ i = 1,\ \ldots,\ N, \tag{4.60}$$

$$\sum_{i\in\bar{G}\setminus G}\ \sum_{j\in G} a_{ij}x_{ij} \geqslant 1,\ \ \forall G \subset \bar{G},\, G \neq \varnothing,\, G \neq \bar{G},\, G \cap S = \varnothing. \tag{4.61}$$

Критерий (4.62) минимизирует длину пути и отвечает длине маршрута. Неравенство (4.57) ограничивает возможные развороты с учетом матрицы разрешенных переходов a_{ij}. Условие (4.58) отвечает за то, что в каждую точку, кроме начальной S, можно прийти только один раз, а (4.59) — за то, что из каждой точки можно выйти только один раз или остановить движение в ней. Ограничение (4.60) гарантирует, что каждый ряд будет обработан по одному из двух направлений. Выполнение (4.61) требуется для отсутствия внутренних циклов. Вычислительная сложность решения задачи 4.4 не позволяет использовать ее для маршрутизации с большим числом рядов. Рассмотрим линейную релаксацию задачи 4.4.

Задача 4.5 (линейная релаксация задачи 4.4). *Найти*

$$\min_{x_{ij},\, i=1,\ldots,2N,\, j=1,\ldots,2N} \sum_{i=1}^{2N}\sum_{j=1}^{2N} d_{ij}a_{ij}x_{ij} \tag{4.62}$$

при ограничениях (4.57)—(4.61).

Значение критерия оптимизации для решения задачи 4.5 можно использовать как нижнюю оценку длины маршрута LB. Релаксированная задача относится к линейному программированию. При этом условие (4.60) гарантирует, то LB больше суммарной длины всех рядов на поле.

В заключении отметим, что существуют частные случаи задачи маршрутизации, для которых решение может быть получено точно. Один из таких примеров — это планирование работ опрыскивателя с дозаправками (рассмотрен в работе [47]), который допускает точное решение с помощью жадного алгоритма.

4.5 Результаты экспериментов

4.5.1 Полевые эксперименты

На рисунке 4.14 представлены результаты построения путей для полевого эксперимента, проведенного с помощью трех одинаковых автономных колесных роботов с механизмом руления поворотом передних колес. Местность представляла собой небольшой склон от насыпи дороги. Начала маршрутов отмечены треугольниками. Максимальное значение нормальной кривизны для роботов составляло $u_{\max} = 0{,}33$ м$^{-1}$, ширина дорожек $\bar{d} = 1$ м. Данные о поверхности были получены с помощью RTK измерений сантиметровой точности (с применением базовой станции дифференциальных поправок) в процессе движения одного из роботов в ручном режиме. Для цифровой модели высот использовалась двумерная B-сплайновая интерполяция. Покрытие путями построено решением задачи 4.1, маршруты движения — с помощью алгоритма на основе метода имитации отжига. Длины маршрутов $D_1 = 105{,}5$ м, $D_2 = 106{,}6$ м и $D_3 = 105{,}1$ м. Они отличаются по длине не более, чем на 1,5 м, что составляет 1,4% от длины самого длинного маршрута. Полевой эксперимент доказал реализуемость полученных путей. На рисунке 4.15 представлена фотография с квадрокоптера, полученная в процессе проведения эксперимента и фрагмент интерфейса наблюдателя.

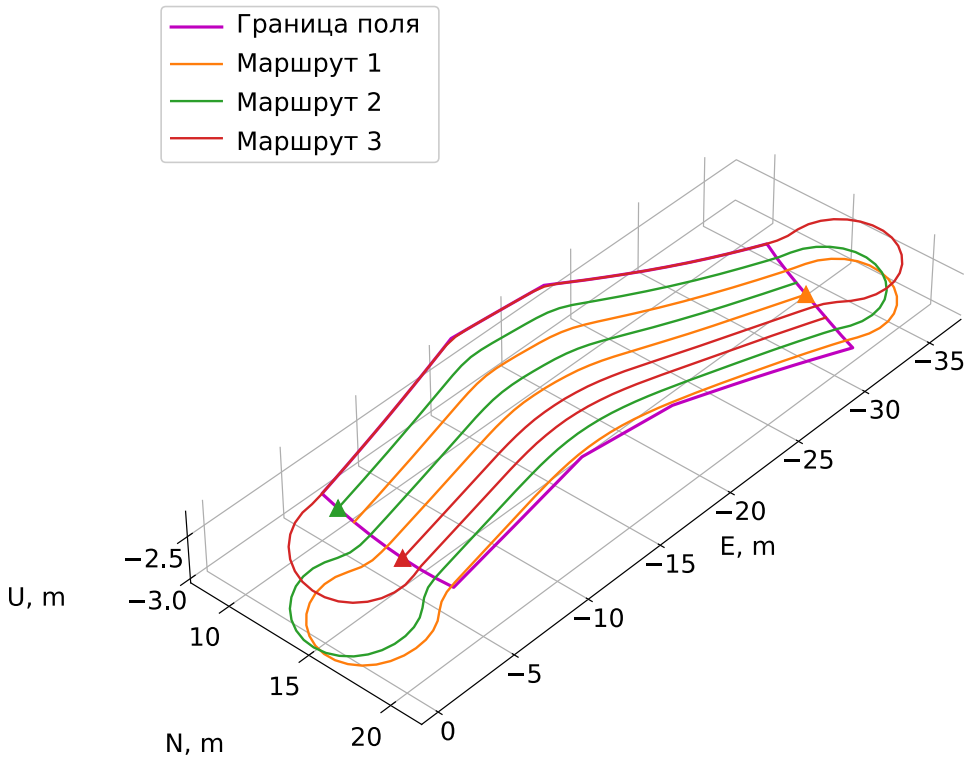


Рисунок 4.14 — Пути по трехмерной поверхности в полевом эксперименте с тремя роботами.

На рисунке 4.16 показан пример решения задачи деформации путей для учета трех препятствий, $u_{\max} = 0{,}4$ м$^{-1}$, $\bar{d} = 1$ м. Данные о покрытии собраны с реального поля с помощью RTK-измерений. Сначала без учета препятствий было построено покрытие поля путями. Затем для каждого пути, пересекающего хотя бы одно препятствие была решена задача 4.2. Маршрутизация проводилась с использованием алгоритма на основе метода муравьиных колоний для одного робота с использованием переднего и заднего хода на разворотах. Длина получившегося маршрута $D_1 = 2868{,}5$ м. Также получена нижняя оценка длины маршрута с помощью линейной релаксации задачи 4.4. Она составила $LB = 2837{,}3$ м.

Рисунок 4.15 — Фотография с квадракоптера и фрагмент интерфейса наблюдателя в полевом эксперименте.

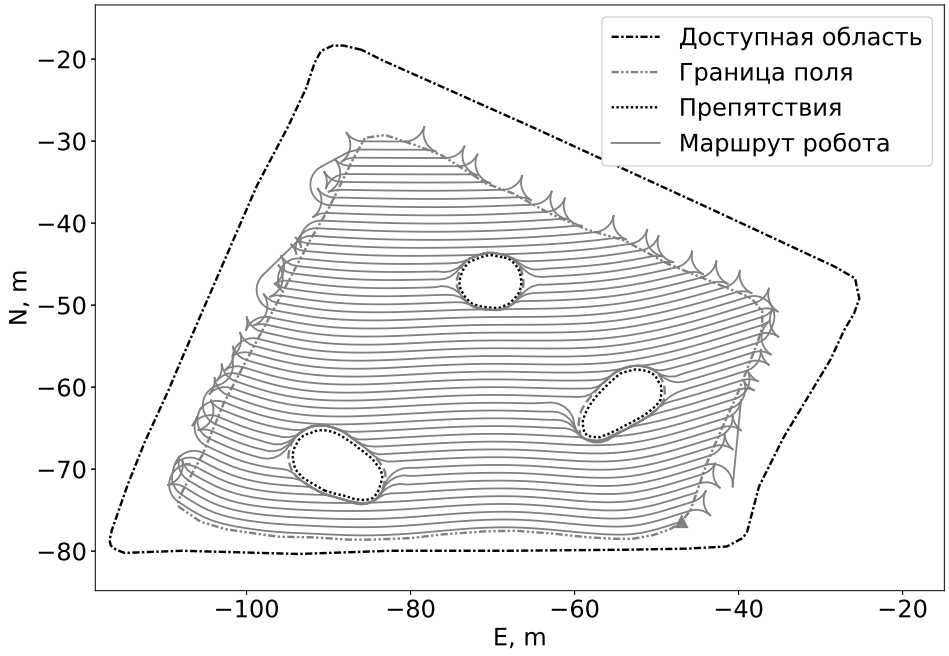


Рисунок 4.16 — Пример деформации путей для поля с тремя препятствиями на основе задачи SOCP.

На рисунке 4.17 приведено построение путей для четырех роботов для поля длиной около 170 метров, $\bar{d} = 2$ м, $u_{\max} = 0{,}4$ м$^{-1}$. Границей препятствия на данном рисунке обозначена область, которую не может посетить рабочая точка

робота. Цифровая модель высот для данного примера построена с использованием билинейной интерполяции по данным на сетке. Маршруты получены с помощью алгоритма на основе метода муравьиных колоний. Разница между протяженностями маршрутов не превысила 2,8% длины максимального из четырех.

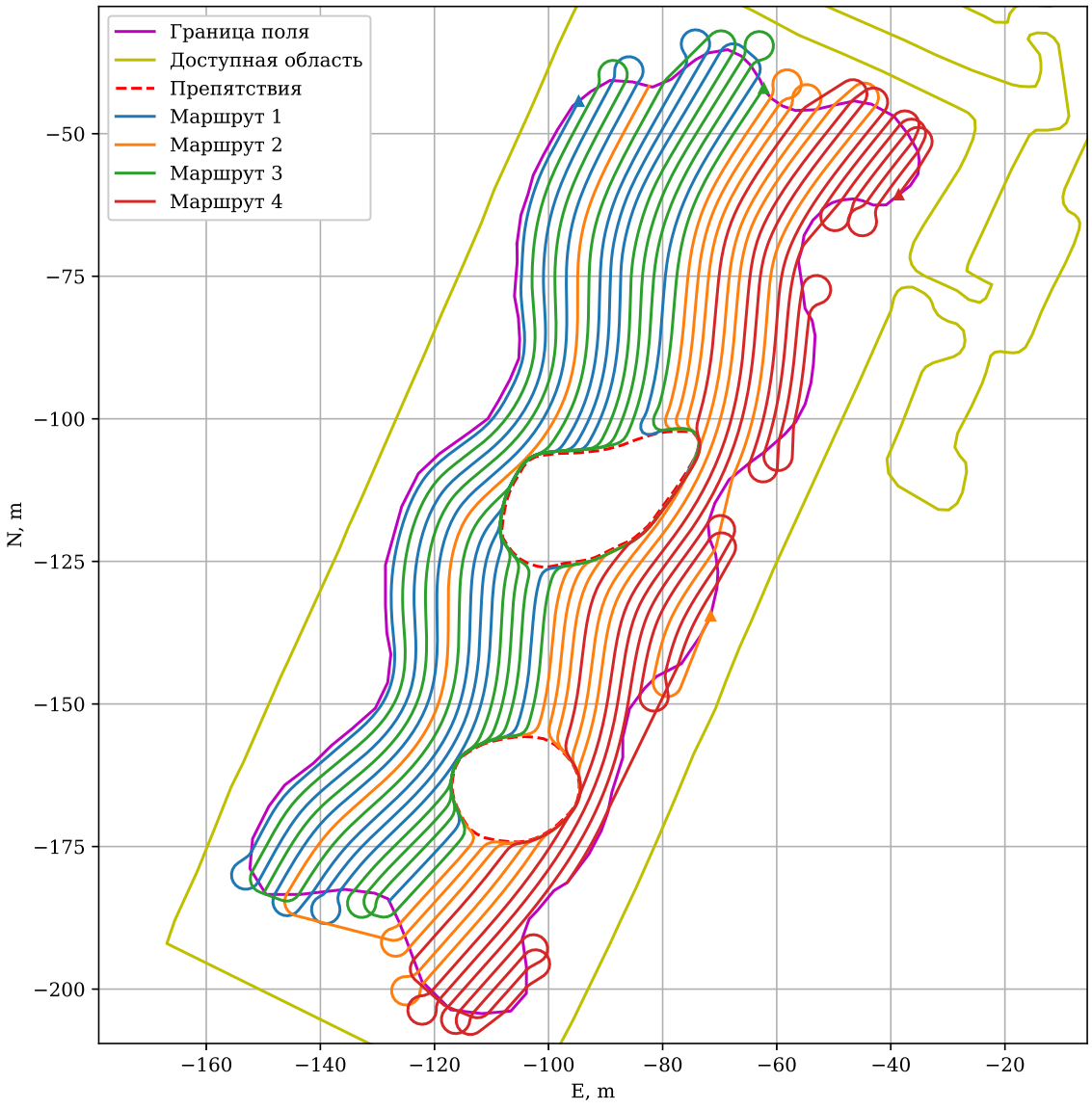


Рисунок 4.17 — Пути четырех роботов для полевого эксперимента.

4.5.2 Спрямление путей за счет перекрытия полос движения

Следующий пример иллюстрирует эффект спрямления полос движения техники за счет их перекрытия при выборе двух разных значений параметра γ в задаче 4.1. Результаты приведены на рисунках 4.18 и 4.19. Контур получен с реального поля с использованием спутниковой карты. В качестве начального пути выбран участок границы поля.

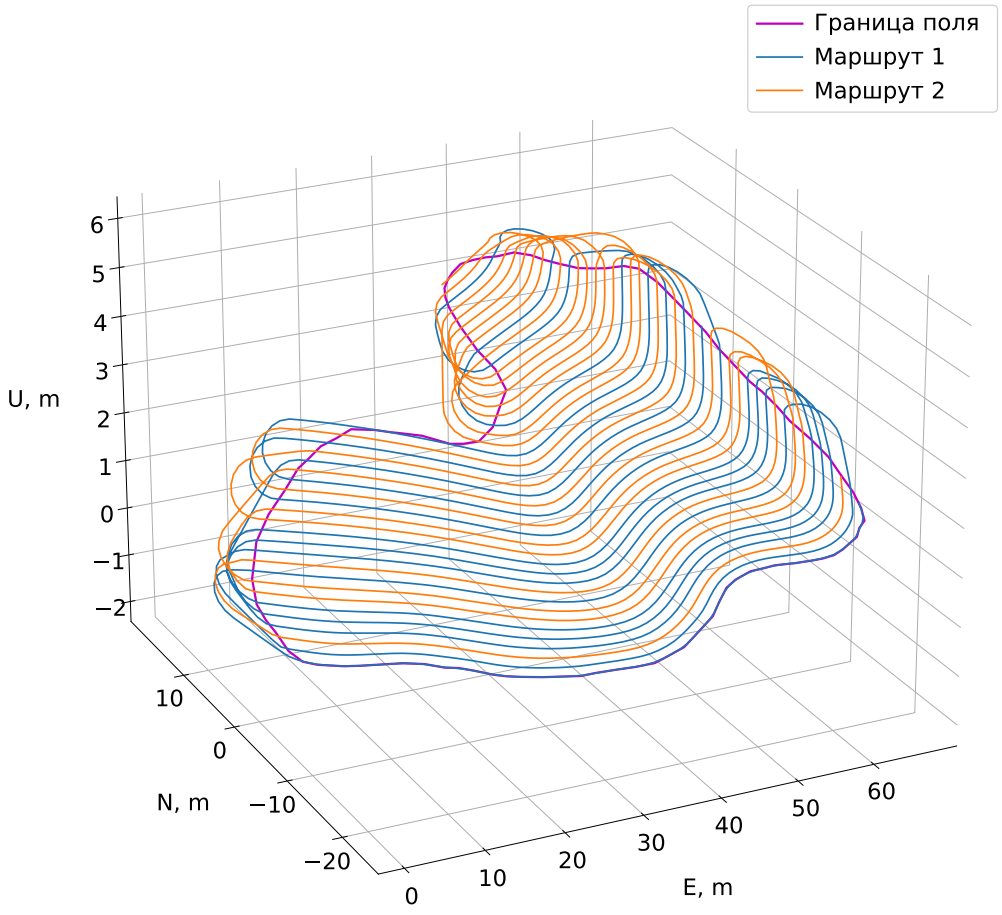


Рисунок 4.18 — Пример спрямления путей при $\gamma = 0{,}95$ (максимальное перекрытие дорожек 5% от их ширины).

В обоих случаях $\bar{d} = 1$ м, $u_{\max} = 0{,}33$ м$^{-1}$, $\beta = (0{,}01 \text{ м} \cdot u_{\max})^{-1}$. Маршруты получены с помощью алгоритма на основе метода имитации отжига. Для случая $\gamma = 0{,}95$ длины маршрутов $D_1 = 1322{,}1$ м и $D_2 = 1321{,}9$ м, для $\gamma = 0{,}75$ — $D_1 = 1475{,}7$ м и $D_2 = 1477{,}2$ м. Видно, что целевая функция (4.45) позволяет спрямлять траектории при достаточно больших β, причем этот спрямление

ограничено допустимым перекрытием соседних дорожек. Отметим, что в данном примере невозможно построение путей без перекрытия соседних дорожек (при $\gamma = 1$).

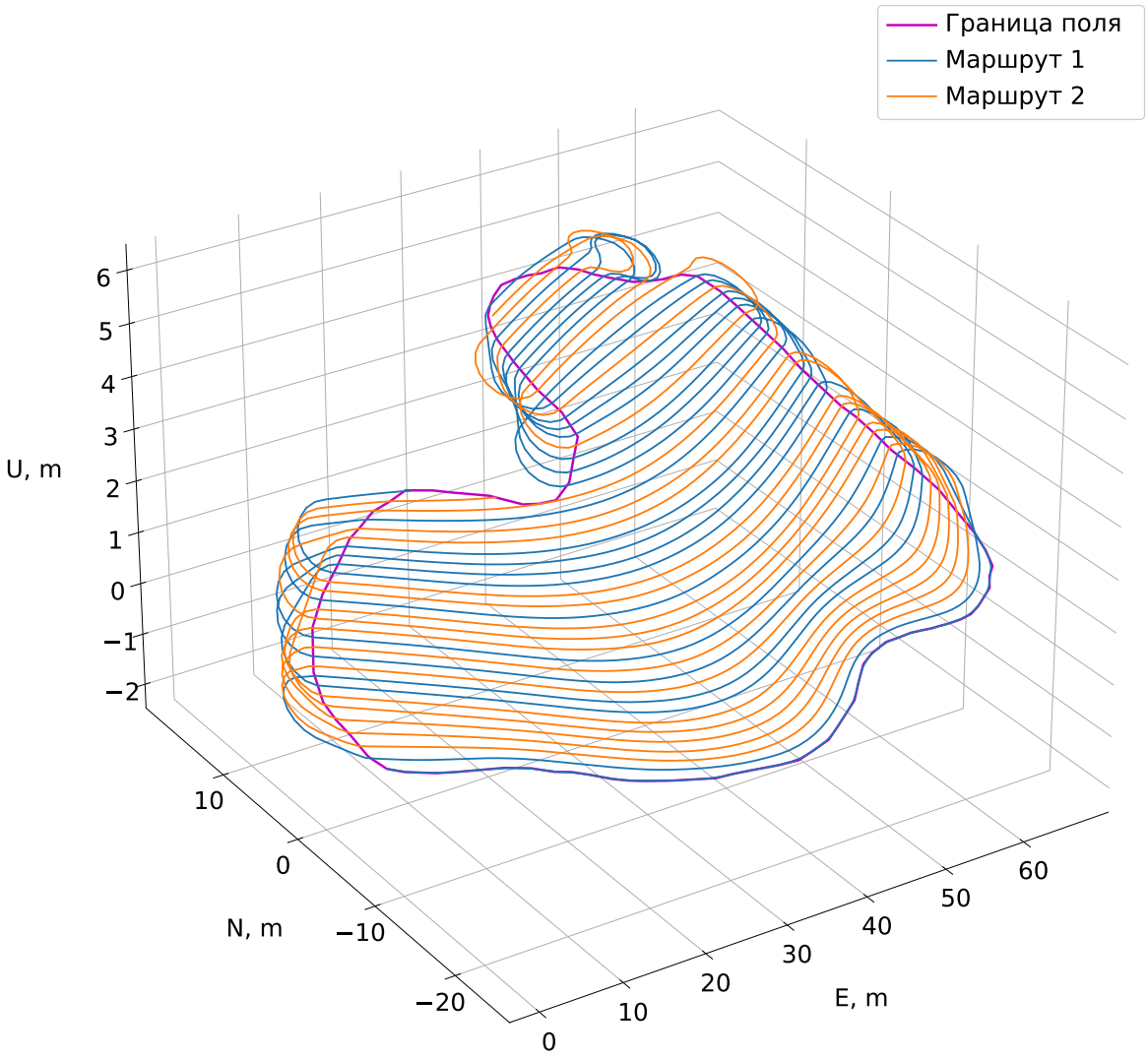


Рисунок 4.19 — Пример спрямления путей при $\gamma = 0{,}75$ (максимальное перекрытие дорожек 25% от их ширины).

В примере на рисунке 4.20 выбраны те же параметры для другого поля. Построение путей с помощью выбранного начального пути в виде участка границы поля здесь нереализуемо без перекрытия. Длины маршрутов здесь $D_1 = 1159{,}8$ м и $D_2 = 1158{,}6$ м.

Заметим, что о невозможности построения покрытия для данного γ свидетельствует несовместность ограничений задачи 4.1 при построении одного

из путей. Большинство численных методов позволяют установить эту несовместность в процессе решения. В таком случае следует выбирать либо другой начальный путь, либо другой параметр перекрытия γ. При этом параметр β должен быть достаточно большим, чтобы спрямление наблюдалось.

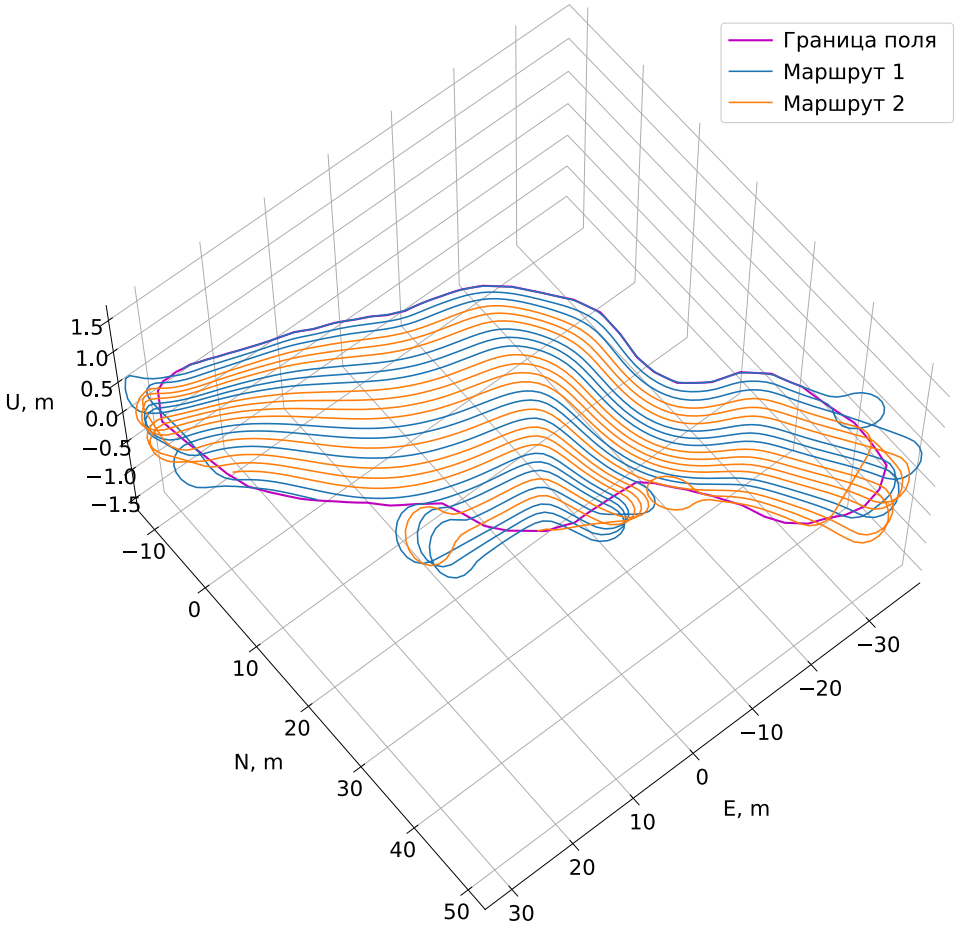


Рисунок 4.20 — Построение путей с перекрытием.

4.5.3 Сравнение методов деформации путей для учета препятствий

На рисунке 4.21 приведен пример решения задачи деформации путей на основе задачи конического программирования второго порядка 4.2 для учета одного препятствия невыпуклой формы. Ширина дорожек составляет один метр, $u_{\max} = 0{,}4$ м$^{-1}$. Покрытие поля путями сначала построено без учета препятствий, а затем решена задача деформации 4.2 для каждого из путей, пересекающего границу препятствия. Веса контрольных точек выбраны единичными ($w_i = 1$). Параметр Δ в алгоритме деформации пути 4.1 выбирался

в соответствии с формулой

$$\Delta = (u_{\max}^{-1} - 0{,}09m)^{-1} - u_{\max}, \tag{4.63}$$

где m — номер итерации алгоритма деформации пути на шаге 2, что соответствует уменьшению максимально допустимого радиуса нормальной кривизны на 0,09 м за одну итерацию.

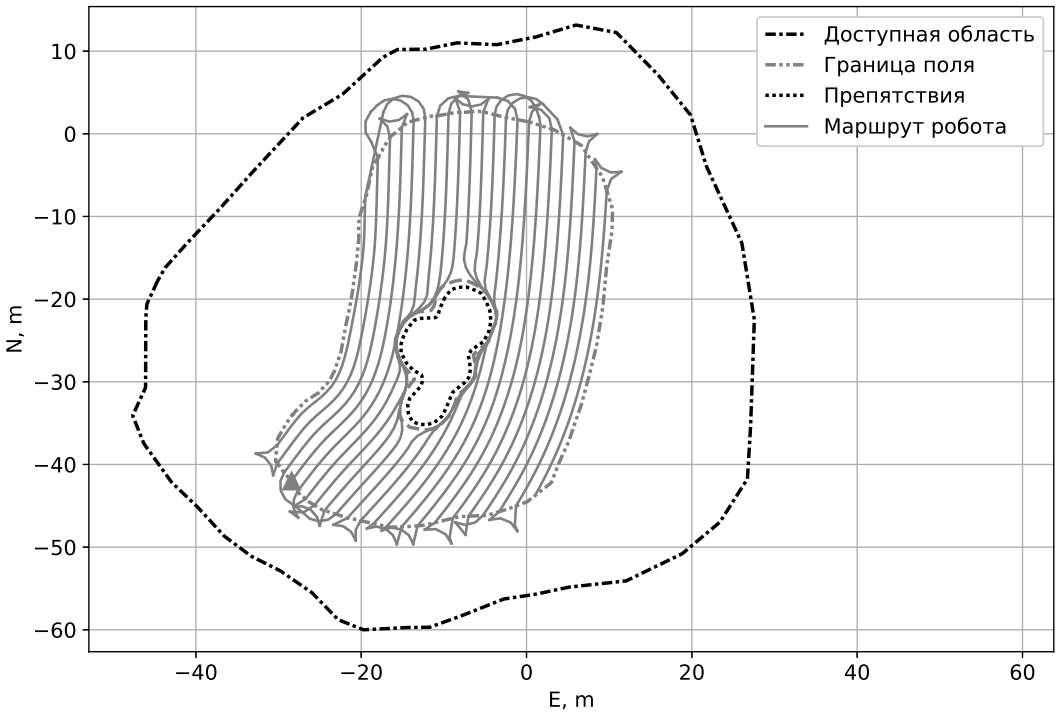


Рисунок 4.21 — Пример деформации путей на основе задачи конического программирования второго порядка.

Для того же примера получено решение задачи деформации путей на основе метода штрафных функций (искусственных потенциальных полей), которое отображено на рисунке 4.22. Из построения видно, что при применении метода штрафных функций образуются значительные необработанные участки, которые в проекции на плоскость локального горизонта расположены между границей препятствия и его выпуклой оболочкой. В тоже время целевая функция задачи конического программирования второго порядка 4.2 позволила сократить необработанную область вокруг невыпуклого препятствия, образующуюся из-за учета ограничений на нормальную кривизну.

Маршруты обхода рядов для данных примеров построены для одного робота с разворотами передним и задним ходом. Для оптимизации длины маршрута использован метод муравьиных колоний.

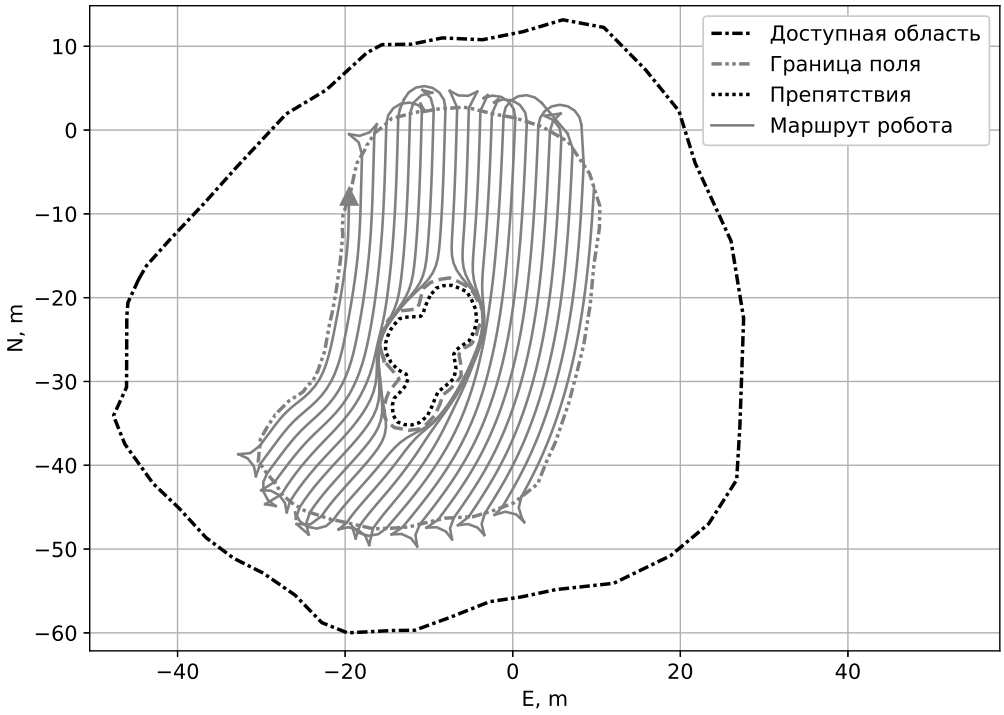


Рисунок 4.22 — Пример деформации путей с применением метода штрафных функций.

4.6 Выводы к главе 4

Рассмотрены возникающие в точном земледелии задачи планирования покрывающих заданную поверхность путей для колесных роботов. Обзор существующих подходов показал актуальность разработки методов построения покрытия поля путями при ограничении на их нормальную кривизну, а также деформации этих путей для учета препятствий.

Предложен метод построения покрытия поля путями для случая пологого трехмерного ландшафта с препятствиями, использующий задачи конического программирования второго порядка, которые являются выпуклыми и допускают вычислительно эффективные методы решения. Для задач маршрутизации роботов по полученным путям приведены приближенные алгоритмы на основе метода имитации отжига и метода муравьиных колоний. Также для случая одного робота приведен способ вычисления гарантированной оценки точности решения задачи маршрутизации с применением линейного программирования. Перечисленные результаты опубликованы в работах [35; 36; 40—42; 47—50].

Заключение

В работе рассмотрено решение с применением полуопределенного программирования ряда задач, возникающих в спутниковой навигации и в точном земледелии.

Основные результаты диссертационной работы следующие.

1. Разработаны метод и алгоритм приближенного решения задачи выбора оптимального рабочего созвездия из не более чем m навигационных спутников по критерию GDOP, основанные на решении релаксированных задач. Получена формализация релаксированных задач в виде задач полуопределенного программирования, сформулированы и доказаны леммы об их оптимальных решениях. Получена двусторонняя оценка точности приближенного решения. Работоспособность предложенного метода проверена в 12-часовом тесте, результаты которого показали высокую точность решения задачи, практическую применимость двусторонней оценки точности, приемлемую скорость решения на вычислительном устройстве.
2. Получены метод и алгоритм приближенного решения задачи выбора базовых линий при определении относительной ориентации твердого тела в пространстве. Вычислительные эксперименты на сгенерированных данных доказали применимость метода.
3. Получено решение возникающей при применении колесных роботов в точном земледелии задачи планирования путей ограниченной нормальной кривизны. Сформулированы и доказаны леммы о необходимых и достаточных условиях на кривизну пути при размещении контрольных точек B-сплайна. На основе задач конического программирования второго порядка предложены методы построения путей, покрывающих заданную поверхность, и их деформации для учета препятствий. За счет того, что в основе этих методов лежат задачи выпуклой оптимизации, реализующие их алгоритмы являются вычислительно эффективными, что подтверждено экспериментами с реальными ландшафтами. Полевые эксперименты доказали применимость результатов построения путей для планирования работ одного или нескольких автономных колесных роботов на одном поле.

Список сокращений и условных обозначений

Сокращения, относящиеся к спутниковой навигации и геодезии.

ГЛОНАСС — российская глобальная навигационная спутниковая система.

ГНСС — глобальные навигационные спутниковые системы.

DEM — цифровая модель высот (от англ. *digital elevation model*).

GDOP — параметр созвездия, отражающий геометрическое снижение точности позиционирования (от англ. *geometric dilution of precision*).

GPS — глобальная навигационная спутниковая система, принадлежащая США (от англ. *Global Positioning System*).

ECEF — глобальная геоцентрическая система координат (от англ. *earth-fixed coordinate system*).

ENU — система координат локального горизонта (от англ. *east, north and up*).

RTK — режим кинематики в реальном времени, в котором местоположение объекта определяется с высокой (сантиметровой) точностью с помощью фазовых ГНСС измерений и поправок базовых станций (от англ. *real time kinematic*).

Классы задач математического программирования.

LP — линейное программирование (англ. *linear programming*).

SDP — полуопределенное программирование (англ. *semidefinite programming*).

SOCP — коническое программирование второго порядка (англ. *second-order cone programming*).

MILP — частично-целочисленное линейное программирование (англ. *mixed integer linear programming*).

MISDP — частично-целочисленное полуопределенное программирование (англ. *mixed integer semidefinite programming*).

MISOCP — частично-целочисленное коническое программирование второго порядка (англ. *mixed integer second-order cone programming*).

Методы, алгоритмы и библиотеки программ для решения задач оптимизации.

ACO — алгоритм муравьиной колонии (англ. *ant colony optimization*).

ADMM — метод множителей переменного направления (англ. *alternating direction method of multipliers*).

CVX, CVXPY — библиотеки программ для Matlab и Python, реализующие управляемое выпуклое программирование и позволяющие задать ряд задач выпуклой оптимизации в форме, необходимой для их решения распространенными пакетами программ (такими как SCS, ECOS и др.).

DCP — управляемое выпуклое программирование (англ. *disciplined convex programming*).

ECOS — пакет встраиваемых программ для решения задач конического программирования второго порядка (англ. *Embedded conic solver*).

SA — алгоритм имитации отжига (англ. *simulated annealing*).

SCS — одна из библиотек программ для решения задач конического программирования (от англ. *Splitting conic solver*).

Задачи и алгоритмы планирования путей.

RRT — алгоритм планирования траектории движения на основе исследования пространства допустимых состояний путем генерации случайных узлов с соединением их в деревья (англ. *rapidly exploring random tree*).

TSP — задача коммивояжера (англ. *travelling salesman problem*).

VRP — задача маршрутизации одной или нескольких машин, которым требуется посетить несколько точек назначения (англ. *vehicle routing problem*).

Математические обозначения.

R^n — пространство n-мерных векторов с действительными компонентами.

$\mathbb{S}^n$ — пространство вещественных симметричных матриц размера $n \times n$.

$\mathbb{Z}$ — множество целых чисел.

I_n — единичная матрица размера $n \times n$.

$\mathrm{diag}(x)$ — диагональная матрица с элементами вектора x на диагонали.

$\mathrm{trace}(A)$ — сумма диагональных элементов (след) квадратной матрицы A.

$\mathrm{cond}(A)$ — число обусловленности матрицы A.

$\mathrm{rank}(A)$ — ранг матрицы A.

A^T — результат транспонирования матрицы A.

$\langle \cdot, \cdot \rangle$ — скалярное произведение (матричное для матриц и евклидово для векторов).

$\| \cdot \|_F$ — матричная норма Фробениуса.

$\| \cdot \|$ — евклидова норма вектора.

$A \succ 0$ — условие положительной определенности матрицы A.

$A \succeq 0$ — условие неотрицательно определенности матрицы A.

$\lambda_{\max}(A)$ — максимальное собственное значение матрицы A (симметричной).

$\lambda_{\min}(A)$ — минимальное собственное значение матрицы A (симметричной).

epi f — надграфик функции f.

$a := b$ — присвоение переменной a значения b.

$\lfloor a \rfloor$ — округление a к меньшему целому числу.

$\max\limits_x f(x)$ — максимальное значение функции $f(x)$, где x – переменная.

$\min\limits_x f(x)$ — минимальное значение функции $f(x)$, где x – переменная.

mod — деление по модулю.

Список литературы

1. *Wahba G.* A Least Squares Estimate of Satellite Attitude // SIAM Review. — 1965. — Vol. 7, no. 3. — P. 409—409. — DOI: 10.1137/1007077.
2. *Leick A.*, *Rapoport L.*, *Tatarnikov D.* GPS Satellite Surveying. — Hoboken, NJ, USA : John Wiley & Sons, Inc, 2015. — P. 840. — DOI: 10.1002/9781119018612.
3. *Boyd S.*, *El Ghaoui L.*, *Feron E.*, *Balakrishnan V.* Linear Matrix Inequalities in System and Control Theory. — Society for Industrial, Applied Mathematics, 1994. — DOI: 10.1137/1.9781611970777.
4. *Поляк Б. Т.*, *Хлебников М. В.*, *Рапопорт Л. Б.* Математическая теория автоматического управления: учебное пособие. — Москва : ЛЕНАНД, 2019. — 500 с.
5. *Wu S.-P.*, *Boyd S.*, *Vandenberghe L.* FIR filter design via semidefinite programming and spectral factorization // Proceedings of 35th IEEE Conference on Decision and Control. Vol. 1. — 1996. — P. 271—276. — DOI: 10.1109/CDC.1996.574313.
6. *Wu S.-P.*, *Boyd S.*, *Vandenberghe L.* FIR Filter Design via Spectral Factorization and Convex Optimization // Applied and Computational Control, Signals, and Circuits: Volume 1 / ed. by B. N. Datta. — Boston, MA : Birkhäuser Boston, 1999. — P. 215—245. — DOI: 10.1007/978-1-4612-0571-5_5.
7. *Wang N.*, *Yang L.* Semidefinite Programming for GPS // Proceedings of the 24th International Technical Meeting of the Satellite Division of The Institute of Navigation (ION GNSS 2011) / ed. by O. Portland. — 2011. — P. 2120—2126.
8. *Rapoport L.* Usage of SDP Relaxation in Some GNSS Navigation Problems // DEStech Transactions on Computer Science and Engineering, 2018 IX International Conference on Optimization and Applications (OPTIMA 2018) (Supplementary Volume). — 2018. — P. 325—335. — DOI: 10.12783/dtcse/optim2018/27943.

9. *Meng F.*, *Zhu B.*, *Wang S.* A new fast satellite selection algorithm for BDS-GPS receivers // IEEE Workshop on Signal Processing Systems, SiPS: Design and Implementation. — 2013. — P. 371—376. — DOI: 10.1109/SiPS.2013.6674535.

10. *Meng F.*, *Wang S.*, *Zhu B.* Research of fast satellite selection algorithm for multi-constellation // Chinese Journal of Electronics. — 2016. — Vol. 25, no. 6. — P. 1172—1178. — DOI: 10.1049/cje.2016.10.009.

11. *Park C.-W.*, *How J. P.* Quasi-optimal satellite selection algorithm for real-time applications // 14th International Technical Meeting of the Satellite Division of the Institute of Navigation (ION GPS 2001), Salt Lake City, UT. — 2001. — P. 3018—3028.

12. *Liu M.*, *Fortin M.-A.*, *Landry Jr. R.* A recursive quasi-optimal fast satellite selection method for GNSS receivers // 22nd International Technical Meeting of the Satellite Division of the Institute of Navigation 2009, ION GNSS 2009. Vol. 5. — 2009. — P. 3022—3032.

13. *Li G.*, *Xu C.*, *Zhang P.*, *Hu C.* A modified satellite selection algorithm based on satellite contribution for GDOP in GNSS. 176 LNEE. — 2012. — P. 415—421. — (Lecture Notes in Electrical Engineering ; VOL. 1). — DOI: 10.1007/978-3-642-31507-7_67.

14. *Wei J.*, *Ding A.*, *Li K.*, *Zhao L.*, *Wang Y.*, *Li Z.* The satellite selection algorithm of GNSS based on neural network // Lecture Notes in Electrical Engineering. — 2016. — Vol. 388. — P. 115—123. — DOI: 10.1007/978-981-10-0934-1_11.

15. *Cohen C. E.* Attitude Determination // Global Positioning System: Theory and Applications, Volume II. — Washington DC : American Institute of Aeronautics, Astronautics, 1996. — P. 519—538. — DOI: 10.2514/5.9781600866395.0519.0538.

16. *Рапопорт Л. Б.* Применение метода полуопределенной релаксации для определения ориентации твердого тела в пространстве // Проблемы управления. — 2019. — Вып. 5. — С. 79—83. — DOI: 10.25728/pu.2018.5.10. — Перевод:

Rapoport L. B. Application of the Method of Semidefinite Relaxation for Determining the Orientation of a Solid Body in Space // Automation and Remote Control. — 2019. — Vol. 80, no. 4. — P. 773—780. — DOI: 10.1134/S0005117919040131.

17. *Jin J.*, *Tang L.* Optimal Coverage Path Planning for Arable Farming on 2D Surfaces // Transactions of the ASABE. — St. Joseph, MI, 2010. — Vol. 53, no. 1. — P. 283—295. — DOI: 10.13031/2013.29488.

18. *De Carvalho R.*, *Vidal H.*, *Vieira P.*, *Ribeiro M.* Complete coverage path planning and guidance for cleaning robots // IEEE International Symposium on Industrial Electronics. Vol. 2. — 1997. — P. 677—682. — DOI: 10.1109/ISIE.1997.649051.

19. *Rodrigues R. T.*, *Aguiar A. P.*, *Pascoal A.* A coverage planner for AUVs using B-splines // 2018 IEEE/OES Autonomous Underwater Vehicle Workshop (AUV). — 2018. — P. 1—6. — DOI: 10.1109/AUV.2018.8729760.

20. *Jin J.*, *Tang L.* Coverage path planning on three-dimensional terrain for arable farming // Journal of Field Robotics. — 2011. — Vol. 28, no. 3. — P. 424—440. — DOI: 10.1002/rob.20388.

21. *Hameed I. A.* Intelligent coverage path planning for agricultural robots and autonomous machines on three-dimensional terrain // Journal of Intelligent and Robotic Systems: Theory and Applications. — 2014. — Vol. 74, no. 3/4. — P. 965—983. — DOI: 10.1007/s10846-013-9834-6.

22. *Hameed I. A.*, *La Cour-Harbo A.*, *Osen O. L.* Side-to-side 3D coverage path planning approach for agricultural robots to minimize skip/overlap areas between swaths // Robotics and Autonomous Systems. — 2016. — Vol. 76. — P. 36—45. — DOI: 10.1016/j.robot.2015.11.009.

23. *Gálvez A.*, *Iglesias A.*, *Puig-Pey J.* Computing parallel curves on parametric surfaces // Applied Mathematical Modelling. — 2014. — Vol. 38, no. 9/10. — P. 2398—2413. — DOI: 10.1016/j.apm.2013.10.042.

24. *Phan N. D. M.*, *Quinsat Y.*, *Lavernhe S.*, *Lartigue C.* Scanner path planning with the control of overlap for part inspection with an industrial robot // The International Journal of Advanced Manufacturing Technology. — 2018. — Vol. 98, no. 1—4. — P. 629—643. — DOI: 10.1007/s00170-018-2336-8.

25. *Galceran E.*, *Carreras M.* A survey on coverage path planning for robotics // Robotics and Autonomous Systems. — 2013. — Vol. 61, no. 12. — P. 1258—1276. — DOI: 10.1016/j.robot.2013.09.004.

26. *Latombe J.-C.* Robot Motion Planning. — Boston, MA : Springer US, 1991. — DOI: 10.1007/978-1-4615-4022-9.

27. *Choset H.*, *Pignon P.* Coverage Path Planning: The Boustrophedon Cellular Decomposition // Field and Service Robotics / ed. by A. Zelinsky. — London : Springer London, 1998. — P. 203—209. — DOI: 10.1007/978-1-4471-1273-0_32.

28. *Acar E. U.*, *Choset H.*, *Rizzi A. A.*, *Atkar P. N.*, *Hull D.* Morse Decompositions for Coverage Tasks // The International Journal of Robotics Research. — 2002. — Vol. 21, no. 4. — P. 331—344. — DOI: 10.1177/027836402320556359.

29. *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. — Vol. 4, no. 2. — P. 100—107. — DOI: 10.1109/TSSC.1968.300136.

30. *Stentz A.* Optimal and efficient path planning for partially-known environments // Proceedings of the 1994 IEEE International Conference on Robotics and Automation. Vol. 4. — 1994. — P. 3310—3317. — DOI: 10.1109/ROBOT.1994.351061.

31. *Chuang J.-H.* Potential-based modeling of three-dimensional workspace for obstacle avoidance // Proceedings IEEE International Conference on Robotics and Automation. Vol. 3. — 1993. — P. 19—24. — DOI: 10.1109/ROBOT.1993.291927.

32. *Гилимьянов Р. Ф.*, *Рапопорт Л. Б.* Метод деформации пути в задачах планирования движения роботов при наличии препятствий // Проблемы управления. — 2012. — Т. 1. — С. 70—76. — Перевод:
Gilimyanov R. F., Rapoport L. B. Path Deformation Method for Robot Motion Planning Problems in the Presence of Obstacles // Automation and Remote Control. — 2013. — Vol. 74, no. 12. — P. 2163—2172. — DOI: 10.1134/S0005117913120187.

33. *Пестерев А. В.*, *Гилимьянов Р. Ф.* Планирование пути для колесного робота // Труды Института системного анализа Российской академии наук. — 2006. — Т. 25. — С. 205—212.

34. *Рапопорт Л. Б.*, *Тормагов Т. А.* Применение методов выпуклой релаксации для оптимизации множества навигационных спутников // Проблемы управления. — 2019. — № 4. — С. 65—71. — DOI: 10.25728/pu.2019.4.7. — Перевод:
Rapoport L. B., Tormagov T. A. Relaxation Methods for Navigation Satellites Set Optimization // Automation and Remote Control. — 2020. — Vol. 81, no. 9. — P. 1711—1721. — DOI: 10.1134/S0005117920090106.

35. *Тормагов Т. А.*, *Генералов А. А.*, *Шавин М. Ю.*, *Рапопорт Л. Б.* Задачи управления движением автономных колесных роботов в точном земледелии // Гироскопия и навигация. — 2022. — Т. 30, 1 (116). — С. 39—60. — Перевод:
Tormagov T. A., Generalov A. A., Shavin M. Y., Rapoport L. B. Motion Control of Autonomous Wheeled Robots in Precision Agriculture // Gyroscopy and Navigation. — 2022. — Vol. 13, no. 1. — P. 23—35. — DOI: 10.1134/S2075108722010072.

36. *Тормагов Т. А.* Метод деформации путей ограниченной кривизны для колесных роботов в точном земледелии на основе конического программирования второго порядка // Автоматика и телемеханика. — 2024. — № 2. — С. 46—59. — DOI: 10.31857/S0005231024020037. — Перевод:
Tormagov T. A. Path Deformation Method with Constraints on Normal Curvature for Wheeled Robots in Precision Agriculture Based on Second-Order Cone Programming // Automation and Remote Control. — 2024. — Vol. 85, no. 2. — P. 134—143. — DOI: 10.31857/S0005117924020037.

37. *Rapoport L.*, *Tormagov T.* Using of the SDP relaxation method for optimization of the satellites set chosen for positioning // Proceedings of the 31st International Technical Meeting of the Satellite Division of the Institute of Navigation, ION GNSS+ 2018. — Institute of Navigation, 2018. — P. 3812—3820. — DOI: 10.33012/2018.15994.

38. *Rapoport L.*, *Tormagov T.* Attitude determination with multiple antennas using SDP relaxation // Proceedings of the 31st International Technical Meeting of the Satellite Division of the Institute of Navigation, ION GNSS+ 2018. —

Institute of Navigation, 2018. — P. 3859—3867. — DOI: 10.33012/2018.16035.

39. *Rapoport L.*, *Tormagov T.* An Approximate Solution of a GNSS Satellite Selection Problem Using Semidefinite Programming // Communications in Computer and Information Science. 1145 CCIS. — 2020. — P. 137—149. — DOI: 10.1007/978-3-030-38603-0_11.

40. *Rapoport L.*, *Generalov A.*, *Shavin M.*, *Tormagov T.* Navigation and Control Problems in Precision Farming // 2021 28th Saint Petersburg International Conference on Integrated Navigation Systems (ICINS). — 2021. — 9470810. — DOI: 10.23919/ICINS43216.2021.9470810.

41. *Tormagov T.*, *Rapoport L.* Coverage Path Planning for 3D Terrain with Constraints on Trajectory Curvature Based on Second-Order Cone Programming // Advances in Optimization and Applications. Communications in Computer and Information Science, 1514. — 2021. — P. 258—272. — DOI: 10.1007/978-3-030-92711-0_18.

42. *Tormagov T.*, *Rapoport L.* Coverage Path Planning with Constraints on Normal Curvature for a Three-dimensional Terrain with Obstacles // 2022 16th International Conference on Stability and Oscillations of Nonlinear Control Systems (Pyatnitskiy's Conference). — 2022. — P. 1—2. — DOI: 10.1109/STAB54858.2022.9807549.

43. *Рапопорт Л. Б.*, *Тормагов Т. А.* Использование метода полуопределённой релаксации для оптимизации состава спутниковых навигационных сигналов // Материалы 14-й Международной конференции «Устойчивость и колебания нелинейных систем управления» (конференция Пятницкого). — 2018. — С. 346—349.

44. *Тормагов Т. А.*, *Рапопорт Л. Б.* Метод выбора GNSS спутников на основе SOCP // Труды 61-й Всероссийской научной конференции МФТИ. 19-25 ноября 2018 года. Радиотехника и компьютерные технологии. — 2018. — С. 41—42.

45. *Рапопорт Л. Б.*, *Тормагов Т. А.* Оптимизация выбора базовых линий в задаче определения ориентации твердого тела // Труды 13-го Всероссийского совещания по проблемам управления (ВСПУ XIII, Москва, 2019). — 2019. — С. 465—469.

46. *Тормагов Т. А.*, *Рапопорт Л. Б.* Метод выбора базовых линий на основе полуопределенного программирования в задаче определения ориентации твердого тела // Материалы VI Международной научно-практической конференции «Системы управления, сложные системы: моделирование, устойчивость, стабилизация, интеллектуальные технологии», посвященной 100-летию со дня рождения профессора А.А. Шестакова (Елец, 16-17 сентября 2020 г.) — 2020. — С. 27—32.

47. *Тормагов Т. А.* Задача планирования заправок опрыскивателя на одном поле // Труды 63-й Всероссийской научной конференции МФТИ 23–29 ноября 2020 года. Радиотехника и компьютерные технологии. — 2020. — С. 27—28.

48. *Рапопорт Л. Б.*, *Генералов А. А.*, *Тормагов Т. А.*, *Шавин М. Ю.* Задачи навигации и управления движением в точном земледелии // В сборнике: XXVIII Санкт-Петербургская международная конференция по интегрированным навигационным системам. — 2021. — С. 352—359.

49. *Тормагов Т. А.* Планирование покрывающих путей при ограничении на кривизну траектории на основе задачи конического программирования второго порядка // Труды 64-й Всероссийской научной конференции МФТИ. 29 ноября – 03 декабря 2021 г. Радиотехника и компьютерные технологии. — 2021. — С. 65—67.

50. *Тормагов Т. А.*, *Рапопорт Л. Б.* Построение покрывающих путей с ограничением на нормальную кривизну для трехмерного ландшафта с препятствиями // Устойчивость и колебания нелинейных систем управления (конференция Пятницкого): Материалы XVI Международной научной конференции (1-3 июня 2022 г., Москва). — 2022. — С. 433—436.

51. Handbook on Semidefinite, Conic and Polynomial Optimization. Vol. 166 / ed. by M. F. Anjos, J. B. Lasserre. — Boston, MA : Springer US, 2012. — (International Series in Operations Research & Management Science). — DOI: 10.1007/978-1-4614-0769-0.

52. *Нестеров Ю. Е.* Введение в выпуклую оптимизацию / под ред. Б. Т. Поляка, С. А. Назина. — М. : МЦНМО, 2010. — С. 280.

53. *Horn R. A.*, *Johnson C. R.* Matrix Analysis. — Cambridge University Press, 1985. — DOI: 10.1017/CBO9780511810817.

54. *Жадан В. Г.* Методы оптимизации. Ч. III. Дополнительные главы : учебное пособие. — М. : МФТИ, 2017. — С. 244.

55. *Boyd S.*, *Vandenberghe L.* Convex Optimization. — Cambridge : Cambridge University Press, 2004. — DOI: 10.1017/CBO9780511804441.

56. Handbook of Semidefinite Programming. Vol. 27 / ed. by H. Wolkowicz, R. Saigal, L. Vandenberghe. — Boston, MA : Springer US, 2000. — P. 654. — (International Series in Operations Research & Management Science). — DOI: 10.1007/978-1-4615-4381-7.

57. *Nesterov Y.* Lectures on Convex Optimization. Vol. 137. — Cham : Springer International Publishing, 2018. — P. 589. — (Springer Optimization and Its Applications). — DOI: 10.1007/978-3-319-91578-4.

58. *Воронцова Е. А.*, *Хильдебранд Р. Ф.*, *Гасников А. В.*, *Стонякин Ф. С.* Выпуклая оптимизация : учебное пособие. — М. : МФТИ, 2021. — С. 364.

59. *Lobo M. S.*, *Vandenberghe L.*, *Boyd S.*, *Lebret H.* Applications of second-order cone programming // Linear Algebra and Its Applications. — 1998. — Vol. 284, no. 1—3. — P. 193—228. — DOI: 10.1016/S0024-3795(98)10032-0.

60. *Gally T.*, *Pfetsch M. E.*, *Ulbrich S.* A framework for solving mixed-integer semidefinite programs // Optimization Methods and Software. — 2018. — Vol. 33, no. 3. — P. 594—632. — DOI: 10.1080/10556788.2017.1322081.

61. *Goemans M. X.*, *Williamson D. P.* Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming // Journal of the ACM. — 1995. — Vol. 42, no. 6. — P. 1115—1145. — DOI: 10.1145/227683.227684.

62. *Юдин Д. Б.*, *Немировский А. С.* Информационная сложность и эффективные методы решения выпуклых экстремальных задач // Экономика и математические методы. — 1976. — Т. 12, № 2. — С. 357—369.

63. *Шор Н. З.* Метод отсечения с растяжением пространства для решения задач выпуклого программирования // Кибернетика. — 1977. — № 1. — С. 94—95.

64. *Nesterov Y.*, *Nemirovskii A.* Interior-Point Polynomial Algorithms in Convex Programming. — Philadelphia : SIAM, 1994. — P. 405. — DOI: 10.1137/1.9781611970791.

65. *Vandenberghe L.*, *Boyd S.* Semidefinite Programming // SIAM Review. — 1996. — Vol. 38, no. 1. — P. 49—95. — DOI: 10.1137/1038003.

66. *Boyd S.* Distributed Optimization and Statistical Learning via the Alternating Direction Method of Multipliers // Now Foundations and Trends. — 2010. — Vol. 3, no. 1. — P. 1—122. — DOI: 10.1561/2200000016.

67. *Поляк Б. Т.*, *Щербаков П. С.* Рандомизированный метод решения задач полуопределенного программирования // Стохастическая оптимизация в информатике. — 2006. — Т. 2. — С. 123—127.

68. *Yamashita M.*, *Fujisawa K.*, *Fukuda M.*, *Kobayashi K.*, *Nakata K.*, *Nakata M.* Latest Developments in the SDPA Family for Solving Large-Scale SDPs // Handbook on Semidefinite, Conic and Polynomial Optimization. — Boston, MA : Springer US, 2012. — P. 687—713. — DOI: 10.1007/978-1-4614-0769-0_24.

69. *Yamashita M.*, *Fujisawa K.*, *Kojima M.* Implementation and evaluation of SDPA 6.0 (Semidefinite Programming Algorithm 6.0) // Optimization Methods and Software. — 2003. — Vol. 18, no. 4. — P. 491—505. — DOI: 10.1080/1055678031000118482.

70. *Yamashita M.*, *Fujisawa K.*, *Kojima M.* SDPARA: Semidefinite programming algorithm parallel version // Parallel Computing. — 2003. — Vol. 29, no. 8. — P. 1053—1067. — DOI: 10.1016/S0167-8191(03)00087-5.

71. *Nakata K.*, *Yamashita M.*, *Fujisawa K.*, *Kojima M.* A parallel primal–dual interior-point method for semidefinite programs using positive definite matrix completion // Parallel Computing. — 2006. — Vol. 32, no. 1. — P. 24—43. — DOI: 10.1016/j.parco.2005.07.002.

72. *Toh K. C.*, *Todd M. J.*, *Tütüncü R. H.* SDPT3 — A Matlab software package for semidefinite programming, Version 1.3 // Optimization Methods and Software. — 1999. — Vol. 11, no. 1—4. — P. 545—581. — DOI: 10.1080/10556789908805762.

73. *Tutuncu R. H.*, *Toh K. C.*, *Todd M. J.* Solving semidefinite-quadratic-linear programs using SDPT3 // Mathematical Programming. — 2003. — Vol. 95, no. 2. — P. 189—217. — DOI: 10.1007/s10107-002-0347-5.

74. *Sturm J. F.* Using SeDuMi 1.02, A Matlab toolbox for optimization over symmetric cones // Optimization Methods and Software. — 1999. — Vol. 11, no. 1—4. — P. 625—653. — DOI: 10.1080/10556789908805766.

75. *O'Donoghue B.*, *Chu E.*, *Parikh N.*, *Boyd S.* Conic Optimization via Operator Splitting and Homogeneous Self-Dual Embedding // Journal of Optimization Theory and Applications. — 2016. — Vol. 169, no. 3. — P. 1042—1068. — DOI: 10.1007/s10957-016-0892-3.

76. *O'Donoghue B.* Operator Splitting for a Homogeneous Embedding of the Linear Complementarity Problem // SIAM Journal on Optimization. — 2021. — Vol. 31, no. 3. — P. 1999—2023. — DOI: 10.1137/20M1366307.

77. *Domahidi A.*, *Chu E.*, *Boyd S.* ECOS: An SOCP solver for embedded systems // 2013 European Control Conference, ECC 2013. — 2013. — P. 3071—3076. — DOI: 10.23919/ECC.2013.6669541.

78. *Diamond S.*, *Boyd S.* CVXPY: A Python-embedded modeling language for convex optimization // Journal of Machine Learning Research. — 2016. — Vol. 17, no. 83. — P. 1—5.

79. *Lofberg J.* YALMIP : a toolbox for modeling and optimization in MATLAB // 2004 IEEE International Conference on Robotics and Automation. — Taipei : IEEE, 2004. — P. 284—289. — DOI: 10.1109/CACSD.2004.1393890.

80. *Grant M.*, *Boyd S.*, *Ye Y.* Disciplined Convex Programming // Global Optimization. — Boston : Kluwer Academic Publishers, 2006. — P. 155—210. — DOI: 10.1007/0-387-30528-9_7.

81. *Гилл Ф.*, *Мюррей У.*, *Райт М.* Практическая оптимизация. — Москва : Мир, 1985. — С. 509. — Пер. с англ. под ред. А. А. Петрова.

82. *Yarlagadda R.*, *Ali I.*, *Al-Dhahir N.*, *Hershey J.* GPS GDOP metric // IEE Proceedings: Radar, Sonar and Navigation. — 2000. — Vol. 147, no. 5. — P. 259—263. — DOI: 10.1049/ip-rsn:20000554.

83. *Teng Y.*, *Wang J.* New characteristics of geometric dilution of precision (GDOP) for multi-GNSS constellations // Journal of Navigation. — 2014. — Vol. 67, no. 6. — P. 1018—1028. — DOI: 10.1017/S037346331400040X.

84. *Doong S. H.* A closed-form formula for GPS GDOP computation // GPS Solutions. — 2009. — Vol. 13, no. 3. — P. 183—190. — DOI: 10.1007/s10291-008-0111-2.

85. *Teng Y.*, *Wang J.* A closed-form formula to calculate geometric dilution of precision (GDOP) for multi-GNSS constellations // GPS Solutions. — 2016. — Vol. 20, no. 3. — P. 331—339. — DOI: 10.1007/s10291-015-0440-x.

86. *Specht D.* A general regression neural network // IEEE transactions on neural networks / a publication of the IEEE Neural Networks Council. — 1991. — Vol. 2. — P. 568—76. — DOI: 10.1109/72.97934.

87. *Kihara M.*, *Okada T.* A Satellite Selection Method and Accuracy for the Global Positioning System // Navigation. — 1984. — Vol. 31, no. 1. — P. 8—20. — DOI: 10.1002/j.2161-4296.1984.tb00856.x.

88. *Жилинский В.*, *Гагарина Л.* Алгоритмы выбора навигационных космических аппаратов при решении навигационной задачи // Вестник Воронежского государственного технического университета. — 2021. — № 6. — С. 43—55. — DOI: 10.36622/VSTU.2021.17.6.006.

89. *Kong J. H.*, *Mao X.*, *Li S.* BDS/GPS satellite selection algorithm based on polyhedron volumetric method // 2014 IEEE/SICE International Symposium on System Integration, SII 2014. — 2014. — P. 340—345. — DOI: 10.1109/SII.2014.7028061.

90. *Sherman J.* Adjustment of an inverse matrix corresponding to changes in the elements of a given column or a given row of the original matrix // Annals of mathematical statistics. — 1949. — Vol. 20, no. 4. — P. 621. — DOI: 10.1214/aoms/1177729893.

91. *Swaszek P.*, *Hartnett R.*, *Seals K.*, *Swaszek R.* A temporal algorithm for satellite subset selection in multi-constellation GNSS // Proceedings of the 2017 International Technical Meeting of The Institute of Navigation, ITM 2017. — 2017. — P. 1147—1159. — DOI: 10.33012/2017.14904.

92. *Поляк Б. Т.*, *Щербаков П. С.* Робастная устойчивость и управление. — М. : Наука, 2002. — 303 с.

93. *Tahsin M.*, *Sultana S.*, *Reza T.*, *Hossam-E-Haider M.* Analysis of DOP and its preciseness in GNSS position estimation // 2nd International Conference on Electrical Engineering and Information and Communication Technology, iCEEiCT 2015. — 2015. — DOI: 10.1109/ICEEICT.2015.7307445.

94. *Chen X.*, *Womersley R. S.*, *Ye J. J.* Minimizing the Condition Number of a Gram Matrix // SIAM Journal on Optimization. — 2011. — Vol. 21, no. 1. — P. 127—148. — DOI: 10.1137/100786022.

95. *Lu Z.*, *Pong T. K.* Minimizing Condition Number via Convex Programming // SIAM Journal on Matrix Analysis and Applications. — 2011. — Vol. 32, no. 4. — P. 1193—1211. — DOI: 10.1137/100795097.

96. *Boyd S.*, *El Ghaoui L.* Method of centers for minimizing generalized eigenvalues // Linear Algebra and its Applications. — 1993. — Vol. 188/189. — P. 63—111. — DOI: 10.1016/0024-3795(93)90465-Z.

97. *Prim R. C.* Shortest Connection Networks And Some Generalizations // Bell System Technical Journal. — 1957. — Vol. 36, no. 6. — P. 1389—1401. — DOI: 10.1002/j.1538-7305.1957.tb01515.x.

98. *Oksanen T.*, *Visala A.* Coverage path planning algorithms for agricultural field machines // Journal of Field Robotics. — 2009. — Vol. 26, no. 8. — P. 651—668. — DOI: 10.1002/rob.20300.

99. *Atkar P. N.*, *Greenfield A.*, *Conner D. C.*, *Choset H.*, *Rizzi A. A.* Uniform coverage of automotive surface patches // International Journal of Robotics Research. — 2005. — Vol. 24, no. 11. — P. 883—898. — DOI: 10.1177/0278364905059058.

100. *Gilimyanov R.*, *Pesterev A.*, *Rapoport L.* Motion control for a wheeled robot following a curvilinear path // Journal of Computer and Systems Sciences International. — 2008. — Vol. 47, no. 6. — P. 987—994. — DOI: 10.1134/S1064230708060129.

101. *Kim T.* Constant cusp height tool paths as geodesic parallels on an abstract Riemannian manifold // Computer-Aided Design. — 2007. — Vol. 39, no. 6. — P. 477—489. — DOI: 10.1016/j.cad.2007.01.003.

102. *LaValle S. M.*, *Kuffner J. J.* Randomized Kinodynamic Planning // The International Journal of Robotics Research. — 2001. — Vol. 20, no. 5. — P. 378—400. — DOI: 10.1177/02783640122067453.

103. *Karaman S.*, *Walter M. R.*, *Perez A.*, *Frazzoli E.*, *Teller S.* Anytime Motion Planning using the RRT* // 2011 IEEE International Conference on Robotics and Automation. — IEEE, 2011. — P. 1478—1483. — DOI: 10.1109/ICRA.2011.5980479.

104. *Chichkanov I.*, *Shawin M.* Algorithm for Finding the Optimal Obstacle Avoidance Maneuver for Wheeled Robot Moving Along Trajectory // 2022 16th International Conference on Stability and Oscillations of Nonlinear Control Systems (Pyatnitskiy's Conference). — Moscow : IEEE, 2022. — P. 1—3. — DOI: 10.1109/STAB54858.2022.9807526.

105. *Гилимьянов Р. Ф.* Покомпонентный метод сглаживания кривизны пространственного пути, построенного по зашумленным измерениям в задачах планирования движения роботов // Проблемы управления. — 2011. — Вып. 6. — С. 66—72. — Перевод:
Gilimyanov R. F. Component-wise method of smoothing the curvature of spatial paths constructed by noisy measurements in robot motion planning problems // Automation and Remote Control. — 2013. — Vol. 74, no. 5. — P. 853—862. — DOI: 10.1134/S0005117913050093.

106. *Dierckx P.* An Algorithm for Surface-Fitting with Spline Functions // IMA Journal of Numerical Analysis. — 1981. — Vol. 1, no. 3. — P. 267—283. — DOI: 10.1093/imanum/1.3.267.

107. *Cressie N.* The origins of kriging // Mathematical Geology. — 1990. — Vol. 22, no. 3. — P. 239—252. — DOI: 10.1007/BF00889887.

108. *Gilimyanov R. F.*, *Pesterev A. V.*, *Rapoport L. B.* Smoothing curvature of trajectories constructed by noisy measurements in path planning problems for wheeled robots // Journal of Computer and Systems Sciences International. — 2008. — Vol. 47, no. 5. — P. 812—819.

109. *Pesterev A. V.*, *Rapoport L. B.*, *Gilimyanov R. F.* Global energy fairing of B-spline curves in path planning problems // 2007 Proceedings of the ASME International Design Engineering Technical Conferences and Computers and Information in Engineering Conference, DETC2007. 8 PART B. — 2008. — P. 1133—1139. — DOI: 10.1115/DETC2007-35306.

110. *Kilby P.*, *Shaw P.* Vehicle Routing // Handbook of Constraint Programming. — Elsevier, 2006. — P. 801—836. — DOI: 10.1016/S1574-6526(06)80027-1.

111. *Dubins L. E.* On Curves of Minimal Length with a Constraint on Average Curvature, and with Prescribed Initial and Terminal Positions and Tangents // American Journal of Mathematics. — 1957. — Vol. 79, no. 3. — DOI: 10.2307/2372560.

112. *Souères P.*, *Boissonnat J.-D.* Optimal trajectories for nonholonomic mobile robots // Robot Motion Planning and Control. — London : Springer-Verlag, 2006. — P. 93—170. — DOI: 10.1007/BFb0036072.

113. *Reeds J.*, *Shepp L.* Optimal paths for a car that goes both forwards and backwards // Pacific Journal of Mathematics. — 1990. — Vol. 145, no. 2. — P. 367—393. — DOI: 10.2140/pjm.1990.145.367.

114. *Sabelhaus D.*, *Röben F.*, *Meyer zu Helligen L. P.*, *Schulze Lammers P.* Using continuous-curvature paths to generate feasible headland turn manoeuvres // Biosystems Engineering. — 2013. — Vol. 116, no. 4. — P. 399—409. — DOI: 10.1016/j.biosystemseng.2013.08.012.

115. *Vahdanjoo M.*, *Zhou K.*, *Sørensen C. A. G.* Route Planning for Agricultural Machines with Multiple Depots: Manure Application Case Study // Agronomy. — 2020. — Vol. 10, no. 10. — P. 1608. — DOI: 10.3390/agronomy10101608.

116. *Conesa-Muñoz J.*, *Bengochea-Guevara J. M.*, *Andujar D.*, *Ribeiro A.* Route planning for agricultural tasks: A general approach for fleets of autonomous vehicles in site-specific herbicide applications // Computers and Electronics in Agriculture. — 2016. — Vol. 127. — P. 204—220. — DOI: 10.1016/j.compag.2016.06.012.

117. *Gambardella L.*, *Dorigo M.* Solving symmetric and asymmetric TSPs by ant colonies // Proceedings of IEEE International Conference on Evolutionary Computation. — IEEE, 1996. — P. 622—627. — DOI: 10.1109/ICEC.1996.542672.

118. *Dorigo M.*, *Maniezzo V.*, *Colorni A.* Ant system: optimization by a colony of cooperating agents // IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics). — 1996. — Vol. 26, no. 1. — P. 29—41. — DOI: 10.1109/3477.484436.

119. *Wang M.*, *Ma T.*, *Li G.*, *Zhai X.*, *Qiao S.* Ant Colony Optimization With an Improved Pheromone Model for Solving MTSP With Capacity and Time Window Constraint // IEEE Access. — 2020. — Vol. 8. — P. 106872—106879. — DOI: 10.1109/ACCESS.2020.3000501.