УДК 519.6

# РЕКУРРЕНТНЫЙ МЕТОД СГЛАЖИВАНИЯ КРИВИЗНЫ ТРАЕКТОРИЙ В ЗАДАЧАХ ПЛАНИРОВАНИЯ ПУТИ ДЛЯ КОЛЕСНЫХ РОБОТОВ[1]

Р.Ф. Гилимьянов

Рассмотрена задача планирования пути для колесного робота, суть которой состоит в следующем. Управляемый вручную колесный робот проводится по желаемому пути, координаты которого измеряются GNSS-приемником. Для повторения пути в автоматическом режиме необходимо построить траекторию, удовлетворяющую определенным критериям гладкости и ограничениям на кривизну. Малые ошибки измерений координат точек могут существенно исказить кривизну построенной траектории, сделав ее непригодной в смысле управления. Предложен рекуррентный метод сглаживания кривизны кривой, состоящей из однородных кубических B-сплайнов, который можно применять при ограничениях на оперативную память и в режиме реального времени.

**Ключевые слова:** планирование пути, колесный робот, GNSS-навигация, приближение данных, B-сплайны, сглаживание кривизны, скользящее окно.

## ВВЕДЕНИЕ

Рассматривается следующая задача планирования пути для колесного робота, имеющего два режима управления — ручной и автоматический. Управляемый вручную колесный робот проводится по желаемому пути, координаты которого измеряются GNSS-приемником и сохраняются в качестве задания для дальнейшего повторения в автоматическом режиме. При этом возникает задача построения траектории колесного робота по заданному массиву точек на плоскости. Очевидна практическая важность такой задачи во многих областях, например, в сельском хозяйстве, когда трактору необходимо точно и неоднократно следовать вдоль криволинейной траектории для выполнения операций посадки, полива, внесения удобрений и т. п.

Построенная траектория должна удовлетворять ограничениям, наложенным на неголономную систему (колесный робот). Поскольку колесный робот не может двигаться в боковом направлении без проскальзывания, угол поворота передних колес и скорость их вращения ограничены, то траектория должна быть гладкой в смысле плавности графика кривизны, дважды непрерывно-дифференцируемой, т. е. принадлежать классу $C^2$, а также ее кривизна не должна превышать максимального значения $k_{\max} = 1/R_{\min}$, соответствующего минимальному радиусу $R_{\min}$ разворота колесного робота. Также важно уметь эффективно вычислять расстояние от робота до траектории. Под гладкостью траектории и кривой в смысле плавности графика кривизны в данной статье будем понимать следующее. Кривая гладкая, если график ее кривизны непрерывен, имеет соответствующий знак и представляет собой кусочно-монотонную функцию с как можно меньшим числом участков монотонности [1, 2].

Существуют множество способов представления пути, например, с помощью последовательности прямых линий и дуг окружностей, полиномиальных сплайнов, клотоид, или обобщенных спиралей Корню. Один из таких способов основывается на аппроксимации траектории с помощью однородных кубических B-сплайнов [3] (в данной статье также вводится функция квазирасстояния

[1] Работа выполнена при финансовой поддержке Программы № 15 ОЭММПУ РАН и государственной программы поддержки ведущих научных школ РФ (НШ-1676.2008.1).

для оценки отклонения робота от целевой кривой). В результате такой аппроксимации получается $C^2$-гладкая кривая. Однако малые ошибки GNSS-измерений координат точек приводят к значительному искажению кривизны траектории. Кривизна хаотически меняется и может превышать максимальное значение $k_{\max}$. На практике это приводит к ухудшению качества управления.

Помимо планирования пути для колесных роботов, гладкие кривые важны и используются во многих других сферах, например, при планировании траектории *робота-манипулятора или траектории* перемещения инструмента на станке с ЧПУ, в системах автоматизированного проектирования и компьютерной графики.

Существуют различные методы получения гладких сплайновых кривых, например, с помощью сглаживающих сплайнов (smoothing spline) [4—7] и фэринга (fairing) [1, 2, 8—12]. Широкое распространение получили следующие два подхода. В первом из них процесс построения кривой совмещен с ее сглаживанием, в другом подходе сначала строится аппроксимирующая или интерполирующая кривая, а затем производится ее сглаживание.

Ко второму подходу относится метод, представленный в работе [12]. В его основе лежит задача квадратичного программирования, сложность и время решения которой нелинейно возрастает с ростом числа точек. Для повышения эффективности в настоящей работе предлагается перейти от задачи условной минимизации к задаче безусловной минимизации, для решения которой строится рекуррентная схема. На основе полученной схемы разработаны методы со скользящим окном, позволяющие сглаживать кривизну длинных траекторий при ограничениях на оперативную память и в режиме реального времени с небольшой задержкой. Помимо всего перечисленного, последнее может быть также полезным в задачах группового управления (formation control). В настоящей статье приводятся результаты применения этих методов к данным реальных GNSS-измерений, а также результаты сравнения с предыдущим методом, описанным в работе [12].

## 1. ЗАДАЧА УСЛОВНОЙ МИНИМИЗАЦИИ

Введем обозначения и кратко повторим основные моменты работы [12]. Пусть есть набор равноточных GNSS-измерений $r_1, r_2, \ldots, r_n$, $r_i \in R^2$, т. е. предполагается, что шумы измерений имеют одинаковую дисперсию. Пусть расстояние между любыми двумя соседними контрольными точками приблизительно одинаково: $\|r_{i+1} - r_i\| \approx \|r_{j+1} - r_j\|$, где $\|\cdot\|$ — евклидова норма вектора. Искомая траектория аппроксимируется однородными кубическими B-сплайнами, каждый из которых строится по четверке контрольных точек следующим образом

$$r^{(i)}(t) = R_i M T(t), \quad R_i = [r_{i-1}, r_i, r_{i+1}, r_{i+2}],$$

$$M = \frac{1}{6}\begin{bmatrix} 1 & -3 & 3 & -1 \\ 4 & 0 & -6 & 3 \\ 1 & 3 & 3 & -3 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad T(t) = \begin{bmatrix} 1 \\ t \\ t^2 \\ t^3 \end{bmatrix}, \quad t \in [0,1],$$

где $t$ — параметр сплайна.

Небольшой шум измерений существенно влияет на кривизну, определяемой формулой $k(t) = \|\dot r(t) \times \ddot r(t)\|/\|\dot r(t)\|^3$. Как будет видно из результатов численного эксперимента, кривизна траектории, построенной таким образом по исходным данным имеет неудовлетворительный вид. Колесному роботу с ограниченными углом и скоростью поворота передних колес сложно точно повторить траекторию с такой кривизной в автоматическом режиме.

В случае равномерно расположенных точек норма вектора первой производной по параметру сплайна $\|\dot r(t)\|$ примерно постоянна, и кривизна кривой пропорциональна норме второй производной сплайна: $k \sim \|\ddot r(t)\|$. Последняя является непрерывной функцией на каждом элементарном сплайне и в точке соединения с соседним сплайном принимает значения $\ddot r^{(i)}(0) = \ddot r^{(i-1)}(1) = r_{i-1} - 2r_i + r_{i+1}$. Производная кривизны $\dot k \sim \|\dddot r(t)\|$ постоянна на протяжении одного элементарного сплайна $\dddot r^{(i)} = -r_{i-1} + 3r_i - 3r_{i+1} + r_{i+2}$ и имеет разрыв в точке соединения с соседним сплайном $\Delta \dddot r_i \equiv \dddot r^{(i)} - \dddot r^{(i-1)} = r_{i-2} - 4r_{i-1} + 6r_i - 4r_{i+1} + r_{i+2}$. Эти скачки третьей производной в точках соединения двух смежных сплайнов влияют на характер графика кривизны.

В статье [12] предлагается путем малых (в пределах ошибки измерений $\delta$) вариаций $\varepsilon_i$ контрольных точек в перпендикулярном к кривой направлении (малый сдвиг вдоль кривой не сильно меняет ее форму) $\tilde r_i = r_i + N_i\varepsilon_i$ минимизировать проекции скачков третьей производной $(\Delta \dddot r_i, N_i)$, где $N_i$ — нормаль к B-сплайновой кривой в точке

соединения $i$-го и $(i + 1)$-го элементарных сплайнов. Запишем в вариациях проекции скачков

$$F_i(\varepsilon) = F_i(0) + (N_{i-2}, N_i)\varepsilon_{i-2} - 4(N_{i-1}, N_i)\varepsilon_{i-1} + \\ + 6\varepsilon_i - 4(N_{i+1}, N_i)\varepsilon_{i+1} + (N_{i+2}, N_i)\varepsilon_{i+2}, \quad (1)$$

где $F_i(0) \equiv (\Delta\ddot{r}_i, N_i) = (r_{i-2}, N_i) - 4(r_{i-1}, N_i) + 6(r_i, N_i) - 4(r_{i+1}, N_i) + (r_{i+2}, N_i)$.

Для вычисления значений $F_1$, $F_2$ и $F_{n-1}$, $F_n$ нужны дополнительные точки: две точки $r_{-1}$, $r_0$ в начале и две точки $r_{n+1}$, $r_{n+2}$ в конце траектории. Для них положим $\varepsilon_{-1} = \varepsilon_0 = \varepsilon_{n+1} = \varepsilon_{n+2} = 0$. О том, как выбирать дополнительные граничные точки и на что этот выбор влияет, см. работу [12].

Введем векторные обозначения $\varepsilon = [\varepsilon_1, ..., \varepsilon_n]^T$, $F(\varepsilon) = [F_i(\varepsilon), ..., F_n(\varepsilon)]^T$ и перепишем выражение (1) в матричной форме

$$F(\varepsilon) = F(0) + C\varepsilon, \quad (2)$$

где $C$ — пятидиагональная симметричная матрица

$$C = \begin{bmatrix} 6 & -4n_{12} & n_{13} & 0 & ... \\ -4n_{21} & 6 & -4n_{23} & n_{24} & ... \\ n_{31} & -4n_{32} & 6 & -4n_{34} & ... \\ 0 & n_{42} & -4n_{43} & 6 & ... \\ \vdots & \vdots & \vdots & \vdots & \ddots \end{bmatrix},$$

$n_{ij} = (N_i, N_j)$ — скалярное произведение двух нормалей. Таким образом, минимизация скачков третьих производных свелась к минимизации квадратичного функционала $\|F(\varepsilon)\|^2$. Возводя выражение (2) в квадрат и отбрасывая свободный член, получаем квадратичный функционал

$$\Phi_c(\varepsilon) = \frac{1}{2}\varepsilon^T H_c\varepsilon + f^T\varepsilon, \quad H_c = C^TC, \quad f = C^TF(0),$$

и приходим к следующей задаче квадратичного программирования с простыми ограничениями

$$\min_{\varepsilon} \Phi_c(\varepsilon),\ \varepsilon \in R^n, \quad -\delta \leqslant \varepsilon_i \leqslant \delta. \quad (3)$$

С ростом числа точек нелинейно возрастает сложность данной задачи и время ее решения. В общем случае задача квадратичного программирования является *NP*-сложной. В нашем случае, при положительно определенной матрице $H_c$, задача (3) решается за полиномиальное время и требует выполнений около $O(n^pL)$ арифметических операций (см., например, книгу [13] и приведенные там ссылки), где $L$ — длина входа задачи (величина, определяющая число двоичных символов, необходимых для записи входной информации задачи), $p > 1$, например, при решении задачи методом внутренней точки $p = 3{,}5$.

## 2. ЗАДАЧА БЕЗУСЛОВНОЙ МИНИМИЗАЦИИ

Для повышения эффективности предлагается перейти от задачи условной минимизации к задаче безусловной минимизации введением штрафа за большие вариации контрольных точек.

$$\Phi(\varepsilon) = \Phi_c(\varepsilon) + \frac{1}{2}\gamma\varepsilon^T\varepsilon = \frac{1}{2}\varepsilon^T H\varepsilon + f^T\varepsilon,$$
$$H = H_c + \gamma I,$$

где $I$ — единичная матрица размера $n \times n$, $\gamma$ — параметр, в данной работе выбираемый экспериментально. Исходная задача свелась к безусловной минимизации квадратичного функционала

$$\min_{\varepsilon} \Phi(\varepsilon), \quad \varepsilon \in R^n. \quad (4)$$

Ее решение сводится к решению системы линейных уравнений

$$H\varepsilon = -f^T \quad (5)$$

с симметричной положительно определенной матрицей $H$, что позволяет компактно хранить ее в памяти и эффективно решить систему, разложив матрицу $H$ на произведение нижней и верхней треугольных ленточных матриц $H = LL^T$ и методом прямой и обратной прогонки решить ленточные треугольные системы

$$Ly = -f^T, \quad (6)$$

$$L^T\varepsilon = y. \quad (7)$$

При нижней ширине $p$ ленты матрицы $L$, равной 4, решение системы (5) требует выполнение около $46n$ операций сложений и умножений, а также $n$ вычислений квадратных корней [14]. Это гораздо меньшее число вычислений по сравнению с решением задачи квадратичного программирования (3).

## 3. РЕКУРРЕНТНАЯ СХЕМА

Построим рекуррентную схему решения задачи (4). Пусть есть измерения координат точек $r_1$, $r_2$, ..., $r_{k+1}$. Добавим две граничные точки $r_{-1}$, $r_0$ в начале траектории. В качестве двух конечных граничных точек в конце траектории возьмем точки $r_k$, $r_{k+1}$ (напомним, что координаты граничных точек не варьируются). По данным измерениям построим матрицы $C^{k-1}$, $H^{k-1}$ и $L^{k-1}$ размера

$(k-1) \times (k-1)$ и векторы $F^{k-1}(0)$, $f^{k-1}$, $y^{k-1}$, $\varepsilon^{k-1}$ размера $k-1$.

Рассмотрим, как они изменятся, когда появится новое измерение $r_{k+2}$ и граничными конечными точками станут точки $r_{k+1}$, $r_{k+2}$.

- К вектору $F(0)$ добавится новый элемент $F_k^k(0)$.
- К матрице $C$ добавятся строка и столбец: $C^k = \begin{bmatrix} C^{k-1} & c_k \\ c_k^T & c_{kk} \end{bmatrix}$, $c_k = [0, ..., 0, c_{k-2,k}, c_{k-1,k}]^T$, $c_{kk} = 6$.
- У вектора $f$ изменятся два последних элемента и добавится новый $f_k^k$: $f^k = [(\tilde{f}^{k-1})^T, f_k^k]^T$, $\tilde{f}^{k-1}$ — это вектор $f^{k-1}$, к последним двум компонентам которого прибавлены компоненты вектора $F_k^k(0)[c_{k-2,k}, c_{k-1,k}]^T$, $f_k^k = [F_{k-2}^k(0), F_{k-1}^k(0), F_k^k(0)][c_{k-2,k}, c_{k-1,k}, c_{kk}]^T$.
- У матриц $H$ и $L$ изменится правый нижний блок размером $2 \times 2$ и добавятся строка и столбец: $H^k = \begin{bmatrix} \tilde{H}^{k-1} & h_k \\ h_k^T & h_{kk}+\gamma \end{bmatrix}$, где матрица $\tilde{H}^{k-1}$ — это матрица $H^{k-1}$, к правому нижнему углу которой прибавлена матрица $D = \begin{bmatrix} d_1 & d_3 \\ d_2 & d_4 \end{bmatrix} = \begin{bmatrix} c_{k-2,k}^2 & c_{k-2,k}c_{k-1,k} \\ c_{k-2,k}c_{k-1,k} & c_{k-1,k}^2 \end{bmatrix}$, $d_2 = d_3$, $[h_k^T \; h_{kk}] = [c_k^T \; c_{kk}]C^k(1:k, k-4:k)$.
- $L^k = \begin{bmatrix} \tilde{L}^{k-1} & 0 \\ l_k^T & l_{kk} \end{bmatrix}$, где $\tilde{L}^{k-1}$ представляет собой матрицу $L^{k-1}$ за исключением трех элементов $\tilde{L}_{k-2,k-2}^{k-1} = \sqrt{L_{k-2,k-2}^{k-1} + d_1}$, $\tilde{L}_{k-1,k-2}^{k-1} = (L_{k-2,k-2}^{k-1} L_{k-1,k-2}^{k-1} + d_2)/\tilde{L}_{k-2,k-2}^{k-1}$, $\tilde{L}_{k-1,k-1}^{k-1} = \sqrt{(L_{k-1,k-2}^{k-1})^2 - (\tilde{L}_{k-1,k-2}^{k-1})^2 + (L_{k-1,k-1}^{k-1})^2 + d_4}$. Имея $\tilde{L}^{k-1}$ — разложение Холецкого матрицы $\tilde{H}^{k-1}$ и строку $[h_k^T, h_{kk}+\gamma]$, можно досчитать разложение $L^k$ и найти строку $[l_k^T \; l_{kk}]$.
- У вектора решения $y$ системы (6) изменятся два последних элемента и добавится новый $y_k^k$, все их можно найти за три шага прямой прогонки $y^k = [(y^{k-1}(1:k-3))^T, y_{k-2}^k, y_{k-1}^k, y_k^k]^T$.
- Вектор решения $\varepsilon$ системы (7) изменится полностью. Вектор $\varepsilon^k$ получается полной обратной прогонкой.

Для возможности вести прямую прогонку и получать решение $y$ системы (6) при появлении нового измерения достаточно иметь не более пяти последних элементов векторов $F(0)$, $f$ и правый нижний блок размера $5 \times 5$ матриц $C$, $H$ и $L$. Для получения $\varepsilon$ нужно иметь всю матрицу $L$ и вектор $y$.

Построенная рекуррентная схема получения матрицы $L$ и вектора $y$ позволяет решать задачи большой размерности при ограничениях на оперативную память следующим образом. Получая новые измерения координат точек, ведем прямую прогонку и записываем матрицу $L$ и вектор $y$ на накопитель, пока не достигнем конца траектории. Затем делаем полную обратную прогонку, считывая $L$ и $y$ в обратном направлении, находим $\varepsilon$ — искомое решение.

## 4. МЕТОДЫ СО СКОЛЬЗЯЩИМ ОКНОМ

Замечено, что для достаточно большого $k \leqslant n$ начальные элементы векторов $\varepsilon^{k-1}$ и $\varepsilon^k$ практически не отличаются друг от друга. Это объясняется тем, что у задач с матрицами $H^{k-1}$ и $H^k$ одинаковые граничные условия в начале и разные в конце. Поэтому вместо решения системы (5) с матрицей $H^k$ можно взять первые $k-l$ компонент решения $\varepsilon^{k-1}$ системы (5) с матрицей $H^{k-1}$, а оставшиеся компоненты $\varepsilon^k$ найти за $l$ шагов обратной прогонки системы (7) с матрицей $(L^k)^T$:

$$\varepsilon^k = [(\varepsilon^{k-1}(1:k-l))^T, \quad \varepsilon_{k-l+1}^k, ..., \varepsilon_k^k]^T.$$

Можно построить рекуррентный метод сглаживания кривизны траектории, в котором используется скользящее окно (sliding window) фиксированного размера $l \times l$, $l \geqslant p+1$, $p = 4$. Матрица $L$ и вектор $y$ хранятся в массивах размерности $l$. При заполнении массива делается сдвиг, в результате которого стирается первый элемент и освобождается последний. Схематически данный процесс показан на левом рис. 1. С появлением нового измерения согласно рекуррентной схеме вычисляются матрица $L$, вектор $y$ и заполняются соответствующие массивы. Если массивы не заполнены и траектория еще не закончилась, то обратная прогонка не делается. Иначе, делается $l$ шагов обратной прогонки и выдается в качестве результата первый элемент полученного решения $\varepsilon$, а если закончилась траектория — все $l$ элементов $\varepsilon$.

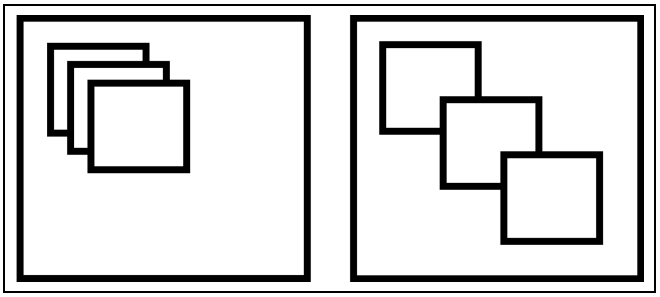

**Рис. 1. Схема методов со скользящим окном**

Предлагается следующая модификация, отличающаяся только способом проведения обратной прогонки и размером окна $w$, $w > l$. При появлении нового измерения также ведется прямая прогонка, но не делается обратная, пока не заполнятся массивы $L$ и $y$. Как только это произойдет, делается полная обратная прогонка ($w$ шагов) и выдаются первые $w - l$ компонент решения $\varepsilon$. Затем делается сдвиг, в результате которого стираются $w - l$ первых компонент массивов $L$, $y$ и $\varepsilon$. Схематически данный процесс показан на рис. 1 справа.

Если число измерений $n$, то при использовании предыдущего метода нужно выполнить $(n - l + 1)l$ шагов обратной прогонки, а модифицированного метода — около $n(1 + l/(w - l))$. Чем больше размер окна $w$ по сравнению с $l$, тем требуется меньшее число обратных прогонок, но тем с большой задержкой будем получать решение.

Данные методы со скользящим окном можно применять для сглаживания длинных траекторий при ограниченной оперативной памяти не только для постпроцессинговой обработки, но и в режиме реального времени с небольшой задержкой.

## 5. ЧИСЛЕННЫЕ ЭКСПЕРИМЕНТЫ

Предложенный метод сглаживания кривизны апробировался на многих траекториях, построенных по данным реальных навигационных измерений. В качестве колесного робота была взята автомашина, оборудованная спутниковыми антеннами, приемником, приводом поворота рулевых колес и датчиком угла поворота колес. Минимальный радиус разворота машины $R_{\min} \approx 5$ м ($k_{\max} \approx 0{,}2$ м$^{-1}$). В описываемом далее эксперименте машина, управляемая вручную, проводилась по желаемой траектории, координаты точек которой измерялись приемником в фазово-дифференциальном режиме со средним квадратичным отклонением горизонтальной ошибки измерения около 1,5 см.

На рис. 2 пунктирной линией изображена траектория, построенная по полученному набору точек $r_1,\ldots, r_n$, $n = 454$, а на рис. 3 тонкой линией изображена ее кривизна в зависимости от пройденного пути $s$. Как видно, кривизна траектории, построенной по исходным точкам, имеет осциллирующий характер и может превышать значение $k_{\max}$.

Были получены еще два набора точек путем сдвига исходных точек вдоль нормалей на величины $\varepsilon_{con}$ и $\varepsilon_{unc}$, найденные при решении задачи условной минимизации с параметром $\delta = 0{,}025$ м и задачи безусловной минимизации с параметром $\gamma = 0{,}001$. На рис. 2 изображены сплошной и штриховой линиями соответственно траектории, построенные по этим наборам точек. На рис. 3 толстой

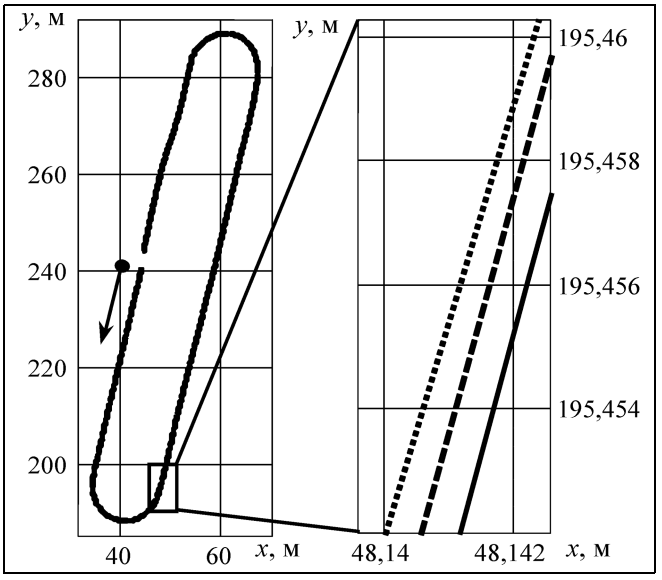


**Рис. 2. Траектории, построенные по разным наборам точек (справа показан в увеличенном масштабе фрагмент траекторий)**

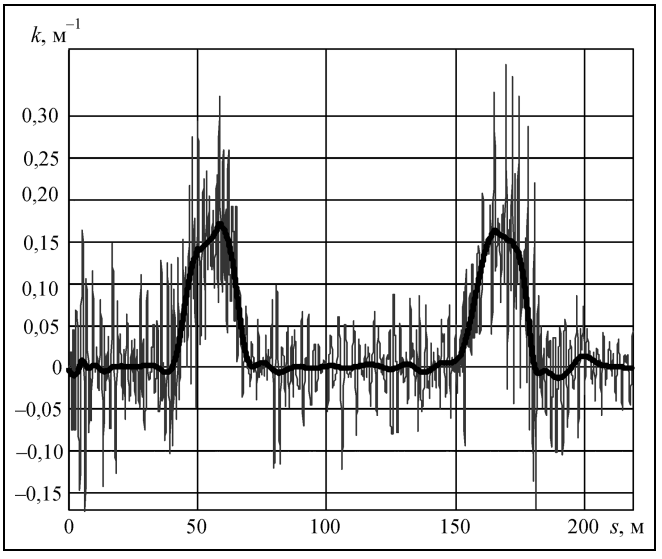


**Рис. 3. Зависимость кривизны траекторий от пройденного пути**

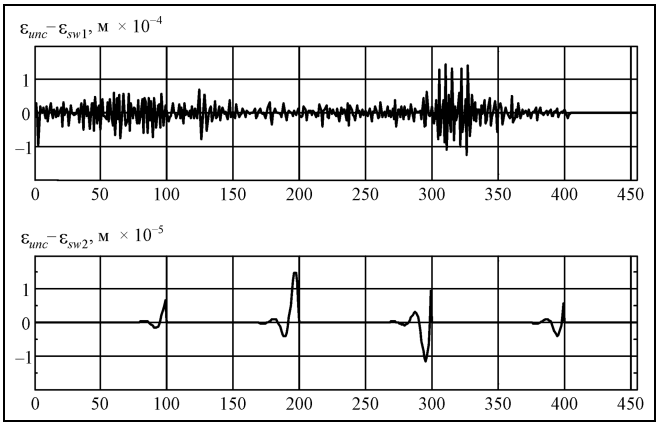


**Рис. 4. Разность решений (по оси абсцисс отложен номер компоненты вектора)**

линией изображена кривизна только второй из них, поскольку графики кривизны обеих траекторий не сильно отличаются. Ни одна из компонент вектора полученного решения $\varepsilon_{con}$ по абсолютному значению не превысила 0,025 м, и только 6 из 454 компонент $\varepsilon_{unc}$ превысили данное значение, причем максимальная из них равна 0,036 м.

Применение метода со скользящим окном $l = 50$ и модифицированного метода с $w = 150$ и $l = 50$ дало решения $\varepsilon_{sw1}$ и $\varepsilon_{sw2}$ практически не отличающие от решения $\varepsilon_{unc}$, (рис. 4). Соответствующие графики кривизны не приводятся, так как они практически совпадают с графиком, изображенным на рис. 3 жирной линией.

Все траектории, изображенные на рис. 2, близки друг к другу, но только траектории, изображенные сплошной и штриховой линиями, имеют подходящую кривизну для проезда колесного робота вдоль нее в автоматическом режиме, что подтвердили отдельные эксперименты по стабилизации движения машины вдоль траектории в автоматическом режиме, при этом использовался закон управления, описанный в работе [15]. В качестве управления выступала скорость поворота рулевых колес. Во время движения вдоль траектории, построенной по исходным точкам, руль с максимальной скоростью постоянно поворачивался из стороны в сторону — машина пыталась повторить траекторию с быстроменяющейся кривизной. Такое управление неэффективно (машина следует вдоль траектории с большими боковыми отклонениями, расходуется много энергии на повороты колес) и может привести к поломке рулевого устройства. Движение вдоль траектории со сглаженной кривизной показало лучшие результаты.

## ЗАКЛЮЧЕНИЕ

Разработаны рекуррентные методы сглаживания кривизны траекторий в задачах большой размерности и при ограничениях на оперативную память. Методы, в которых используется скользящее окно, можно также применять в режиме реального времени с небольшой задержкой. Эффективность предложенных методов проиллюстрирована численными примерами их применения к траекториям, построенным по реальным данным GNSS-измерений.

## ЛИТЕРАТУРА


1. *Farin G., Sapidis N.* Curvature and the Fairness of Curves and Surfaces // IEEE Computer Graphics and Applications. — 1989. — Vol. 9, № 2. — P. 52—57.
2. *Sapidis N., Farin G.* Automatic Fairing Algorithm for B-Spline Curves // Computer-Aided Design. — 1990. — Vol. 22. — P. 121—129.
3. *Пестерев А.В., Гилимьянов Р.Ф.* Планирование пути для колесного робота // Проблемы вычислений в распределенной среде: распределенные приложения, коммуникационные системы, математические модели и оптимизация / Труды ИСА РАН. — Москва, 2006. — Т. 25. — С. 204—211.
4. *Schoenberg I.* Spline functions and the problem of graduation // Proc. Nat. Acad. Sci. — 1964. — Vol. 52. — P. 947—950.
5. *Reinsch C.* Smoothing by spline functions // Numer. Math. — 1967. — Vol. 10. — P. 177—183.
6. *De Boor C.* A Practical Guide to Splines. — New York: Springer-Verlag, 1978. — 392 p.
7. *Завьялов Ю.С., Квасов Б.И., Мирошниченко В.Л.* Методы сплайн-функций. — М.: Наука, 1980. — 352 с.
8. *Kjellander J.* Smoothing of Cubic Parametric Splines // Computer-Aided Design. — 1983. — Vol. 15(3). — P. 175—179.
9. *Farin G.* et al. Fairing Cubic B-Spline Curves // Computer Aided Geometric Design. — 1987. — Vol. 4. — P. 91—103.
10. *Eck M., Hadenfeld J.* Local Energy Fairing of B-spline Curves // Computing Supplement. — 1995. — Vol. 10. — P. 129—147.
11. *Xu S., Li W., Zhao G.* Target curvature based automatic fairing of planar b-spline curves // Computer Aided Geometric Design. — 2004. — Vol. 21(5). — P. 427—530.
12. *Гилимьянов Р.Ф., Пестерев А.В., Рапопорт Л.Б.* Сглаживание кривизны траекторий, построенных по зашумленным измерениям, в задачах планирования пути для колесных роботов // Изв. РАН. Теория и системы управления. — 2008. — Т. 47, № 5. — С. 152—159.
13. *Floudas C.A., Pardalos P.M.* Encyclopedia of Optimization. — New York: Springer-Verlag, 2009. — 4626 p.
14. *Golub G.H., Van Loan C.F.* Matrix Computations. — London: The John Hopkins University Press, 1996. — 728 p.
15. *Гилимьянов Р.Ф., Пестерев А.В., Рапопорт Л.Б.* Управление движением колесного робота в задаче следования вдоль криволинейного пути // Изв. РАН. Теория и системы управления. — 2008. — Т. 47, № 6. — С. 209—216.


*Статья представлена к публикации членом редколлегии В.Ю. Рутковским.*

**Гилимьянов Руслан Фаильевич** — мл. науч. сотрудник,
Институт проблем управления им. В.А. Трапезникова РАН,
г. Москва, ☎ (495) 926-52-00 доб. 11-49, 334-93-69,
✉ R.Gilimyanov@javad.com.