Об алгоритме привязки координат объекта к графу дорог

И.Н. Полякова, С.Ю. Бурдуковская

***Аннотация* — В статье рассматривается алгоритм привязки координат объекта к графу дорог. Подобные алгоритмы применяются во многих приложениях. Такие алгоритмы могут вносить некоторые погрешности, что затрудняет определять точное местоположение объекта.**

В данной работе исследуются причины неточностей алгоритма, использующего ранее существующие подходы, и предлагается алгоритм, который устраняет некоторые неточности и временные задержки. Представленный алгоритм имеет оценки для сходства, близости и ориентации. Для каждой из оценок рассмотрены способы, которые позволяют алгоритму определять расположение на графе более точно и быстро. Представлены формулы для поиска точного расположения GPS – координаты, которые используются в алгоритме.

Все полученные результаты сравнивались с эталонным решением. Алгоритм показал наибольшее улучшение - 90% - которое произошло во время поворота. Ранее максимальная ошибка положения была 88 м, тогда как решение нового алгоритма отличалось только на 8 м от эталонной траектории. Проведенные исследования показали, что данные, полученные от GPS-приемника, совпадают практически везде с графом дорог. Предлагаемый алгоритм показал улучшения в среднем на 50%. Алгоритм успешно обрабатывает даже относительно плохие данные GPS.

***Ключевые слова*— Алгоритм привязки координат к графу дорог, координаты объекта, навигационные системы, программная реализация алгоритма, траектория движения объекта.**

Статья получена 23 мая 2021.

Полякова И.Н., Московский государственный университет имени М.В. Ломоносова (email: polyakova@cs.msu.ru)
Бурдуковская С.Ю., магистрант, Московский государственный университет имени М.В. Ломоносова (email: beswently@gmail.com)

I. ВВЕДЕНИЕ

Граждане пользуются услугами навигаторов, которые выбирают оптимальные маршруты движения. Возможность широкого использования навигационных систем существует благодаря тому, что почти все

электронные устройства используют глобальную навигационную спутниковую систему для определения местоположения в любой точке земной поверхности с применением специальных навигационных или геодезических приемников. Координаты, полученные с такого устройства, определяются с определенной погрешностью и шумом. Это зависит от погрешности самого датчика и от таких факторов, как ландшафт, скорость движения, количество и положение спутников.

Во избежание некорректного отображения объектов, в навигационных системах применяется алгоритм привязки координат (map matching algorithm) к графу дороги. В нашем случае граф - это последовательность хронологически упорядоченных пространственных точек: $p_1 \rightarrow p_2 \rightarrow ... \rightarrow p_n$, полученных от непрерывно движущегося объекта. Каждая точка p_i состоит из пары координат $<x_i, y_i>$, отметки времени t_i, скорости spd_i (необязательно) и заголовка θ_i (необязательно), то есть: $p_i = <x_i, y_i, t_i, spd_i, \theta_i>$ [1]. Алгоритм привязки получает на вход координаты, полученные из GPS - приёмника, обрабатывает их и выдает последовательность рёбер (координат) дорожного графа, которая своей геометрией максимально близко повторяет входные данные и соотносит их к дороге на цифровой карте [2]. Алгоритм должен корректно работать даже тогда, когда координаты, полученные от спутниковой системы навигации [3], и координаты дороги на цифровой карте имеют неточности и ошибки. Следует заметить, что электронная карта для алгоритма привязки представляет собой граф, на котором отмечены координаты известных путей, дорог, улиц, объектов. Скажем, городская улица при реализации алгоритма представляет собой граф с ее координатами. В этом случае привязка координат к цифровой карте означает нахождение координат на этом графе. Существует три вида алгоритмов

привязки координат пользователя к электронным картам [4, 5]. В данной работе используется привязка координат пользователя к фиксированному маршруту.

II. ПОИСК ПРИЧИН НЕТОЧНОСТЕЙ СОПОСТАВЛЕНИЯ КООРДИНАТ

Одна из причин неточности сопоставления координат - отсутствие фильтрации выбросов. Основной причиной такого поведения является влияние многолучевого распространения на сигнал GPS. Пример нерегулярного скачка в позиционировании GPS представлен на рисунке 1. Линия, соединяющая точки C_2, C_3, C_5 и C_6, обычно параллельна дуге L_0. Однако точка C_4 не соответствует этому шаблону. Пик или выброс, какой отображает точка C_4, может привести к ошибкам сопоставления, особенно если C_4 расположена ближе к дуге L_1. Если точка C_4 была бы оценена на основе вычисленного направления от C_3 - C_4, она должна была бы соответствовать дуге L_1.

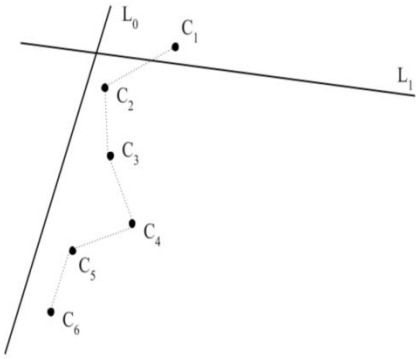

Рис. 1. Пики, вызванные ошибочными местоположениями GPS

Если происходит значительное изменение направления линий C_{t-1}-C_t, отображение точки C_t завершается только после того, как точки C_{t+1} и C_{t+2} были обнаружены и проанализированы. На рис.1 линии между точками C_3 и C_4 демонстрируют резкое изменение от ранее наблюдаемых направлений. Следовательно, точка C_4 будет отображена только после того, как будут рассчитаны направления для линий C_4-C_5 и C5-C_6. Задержка в две секунды (при условии, что позиция пользователя вычисляется каждую секунду) незначительна для пользователя, но очень полезна для надежности процедуры сопоставления. Эта задержка позволяет алгоритму проверять, соответствует ли вычисленное направление для линии GPS между точками C_3 - C_4 направлению линий C_4 - C_5 и C_5 - C_6. Если направление линии C_3 - C_4 не соответствует шаблону, установленному линиями, предшествующими C_3, и линиями, следующими за C_4, положение точки C_4 считается выбросом.

Еще одна защита от неправильного сопоставления, связанного со скачками, состоит в том, чтобы вычислить оставшееся расстояние от последней сопоставленной точки до конца на текущей согласованной дуге L_i. Если это расстояние больше некоторого порога, то точка C_t GPS будет соответствовать текущей дуге L_i.

III. ПОСТАНОВКА ЗАДАЧИ ДЛЯ ПРЕДЛАГАЕМОГО АЛГОРИТМА

Предлагаемый алгоритм основан на комбинации нескольких методов - геометрическом [6, 7], топологическом [8, 9], весовом [10, 11]. Комбинация методов позволяет устранить существующие недостатки отдельно взятых конкретных методов.

Исходные данные

- Предыдущее состояние объекта (может отсутствовать в начале движения).
  - C_L' - предыдущая скорректированная координата, находящаяся на отрезке дороги
  - V_L' - предыдущий вектор скорости вдоль отрезка дороги
  - V_L'' - позапрошлый вектор скорости вдоль отрезка дороги;
- Текущее местоположение.
  - C_{GPS} - Текущая GPS-точка.

Заметим, что L_m – отрезок, соединяющий точки с соседними номерами m, m+1 в одной ветке (N_m, N_{m+1}). Ветка - связная часть графа. Отрезки разных веток, имеющие точки с одинаковыми координатами, считаются связанными. Точки принадлежат дорожным веткам и имеют порядковый номер в ветке. Это значит, что ветка не может иметь циклов или быть разрывной.

Требуется найти

- Местонахождение объекта на отрезке дороги - C_L - линейные координаты (ссылка на отрезок, километраж по участку дороги) и скорректированные гео-координаты (широта, долгота).
- Скорость и направление движения - V_L - гео-вектор вдоль отрезка дороги

IV. ОПИСАНИЕ АЛГОРИТМА

Требуется вычислить координату и найти отрезок L, на котором она находится. Для координат используем систему ECEF,

имеющийся алгоритм пересчета из GPS-координат и алгоритм пересчета в линейные координаты (километраж) и обратно. Затем найдем скорость V_L.

Опишем поиск координаты.

Простой начальный случай (предыдущее состояние отсутствует) (см. рис 2)

Шаг 1: Сформировать $L_\Delta = \{L_{\Delta i}\}$- вспомогательные опорные отрезки, геометрия которых находится внутри окружности с центром в C_{GPS} и радиусом Δ=200м (дельта-окрестность), где $L_{\Delta i}$ - это либо отрезок, хотя бы один из концов которого находится внутри окружности, либо это отрезок, пересекающий окружность (см. Рис. 3, а).

Шаг 2: Если $L_\Delta = Ø$, то выдать сообщение об ошибке привязки и перейти к Шагу 1.

Шаг 3: Среди всех L_Δ найти отрезки $L_{Near} = \{L_{Near\,i}\}$, ближайшие к C_{GPS} с допуском Θ = 50 м.

Конец алгоритма

Для ускорения вычислений, можно отсечь отрезки и даже ветки целиком, если их габариты заведомо далеко от C_{GPS} (Рис. 3, б).

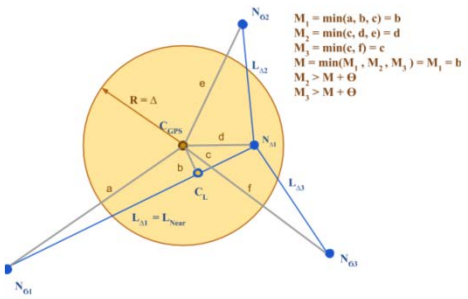

Рис. 2. Простой начальный случай

Обозначим $N_\Delta = \{N_{\Delta i}\}$ - опорные точки в дельта-окрестности (Δ=200 м) от текущей GPS-координаты C_{GPS}. Обозначим $N_б$ – точки, не входящие в Δ-окрестность, но являющиеся концами найденных отрезков. При расчете близости следует сначала использовать M = минимальное расстояние среди (1) расстояний от точки C_{GPS} до концов отрезка и (2) длины перпендикуляра к отрезку, если перпендикуляр может пройти и через точку, и закончиться на отрезке (а не снаружи его). Назовем D расстоянием от точки C_{GPS} до отрезка L_{Near}. На картинке минимальным из расстояний a,b,c,d,e,f является расстояние b, т.е. M = b. К минимальному расстоянию добавляем Θ и включаем в множество L_{Near} отрезки, расстояние до которых от точки C_{GPS} не больше (D + Θ). Если L_{Near} есть множество из одного элемента, а предыдущее состояние отсутствует, то это простой начальный случай. Искомый элемент C_L находится на противоположном от C_{GPS} конце минимального отрезка M (в данном случае b).

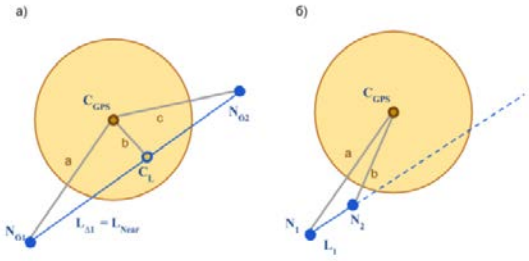

Рис. 3. Отрезки, габариты которых пересекают (а) и не пересекают (б) окружность

Если же L_{Near} есть множество из нескольких элементов, то ситуация усложняется. Далее рассматриваются случаи, когда будет несколько L_{Near} и несколько C_L, и возможно есть предыдущее состояние локомотива.

Сложный начальный случай (см. Рис. 4.)

Шаг 1: Если в Θ-окрестности несколько подходящих C_L, а предыдущее состояние объекта отсутствует, то приближенно выбрать любую C_L среди точек с минимальным расстоянием от C_{GPS}.

Шаг 2: Если же в Θ-окрестности несколько подходящих C_L, и есть предыдущее состояние объекта, то это общий случай, который рассматривается далее.

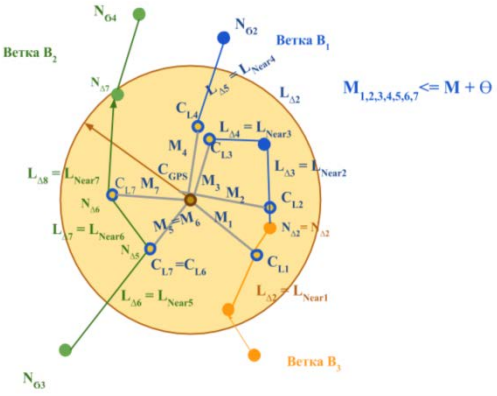

Рис. 4. Сложный начальный случай

Общий случай (расширяет сложный начальный случай)

В общем случае, у нас есть предыдущее состояние объекта (C_L', V_L', V_L''), с помощью которого можно улучшить точность привязки координаты.

Шаг 1: Вычислить (как и в простом начальном случае) множество L_{Near}, и множество $C_{L.}$

Шаг 2: Присвоить каждому отрезку L_{Neari} (а не точке C_{Li}) значение веса отрезка W.

Шаг3: Выбрать отрезок с наибольшим весовым значением $L_W = L_{Neari}$. Если у всех отрезков L_{Neari} значение веса равно 0, то применить алгоритм начального случая.

Шаг 4: Найти на отрезке $L_П$ точку C_L (лучшая из приблизительных точек C_{Li} будет $C_{L\ Best}$ с максимальным весовым коэффициентом), среднюю между $C_L{}' + S_K$ и $C_{L\ Best}$.

Конец алгоритма.

Весовое значение W вычисляется по формуле (1)

$$W = W_{AZ} + W_D + W_I + K_i \quad (1)$$

$$W_{AZ} = C_{az} * \cos^{n}{}_{az}(\Delta AZ) \quad (2)$$

$$W_D = C_D - a * D^{n}{}_{D} \quad (3)$$

$$W_I = C_I * \cos^{n}{}_{I}(\Delta AZ), \quad (4)$$

где:

W - общий балл

W_{AZ} - вес для сходства в ориентации,

W_D - вес для близости точки к линии

W_I - вес для пересечения, если применимо,

K_i – влияние предыдущих скоростей.

ΔAZ - разница между азимутами линии GPS и оцененной сетевой дуги.

C_i - постоянная, составляющая максимальный балл.

Заметим, что можно, снижая точность вычислений, ускорить вычисления по общему алгоритму следующим образом. Если в Ω-окрестности ($\Omega = 50м < \Delta$) точки C_{GPS} есть L_Δ, которые находятся в той же самой ветке, что и прошлое линейное положение объекта, то допускаем $L_{Near} = L_\Delta$ той же ветки, а при вычислении веса допускаем, что $W_D = W_I = 1$ (const).

V. МОДИФИКАЦИЯ АЛГОРИТМА

Модифицированный алгоритм сопоставления карты состоит из двух отдельных методов. Первый называется InitialMapping(), включает простой и сложный начальные случаи. Он необходим для начального определения местоположения пользователя на дорожном пути и основан на геометрическом подходе. На основании начального сопоставления может быть выполнен последующий топологический анализ - второй метод алгоритма сопоставления карт (Map()). Он включает общий случай (см. п.IV). Данная часть заменяет сложный случай старого алгоритма. Это основная часть алгоритма сопоставления, которая использует топологическое обоснование и весовую схему для сопоставления точки GPS C^t с отрезками L сети N. Алгоритм Map() применяется только после того, как начальное совпадение было найдено с помощью процедуры InitialMapping().

Применение InitialMapping():

1. При получении первой точки GPS C_0.
2. Когда расстояние между новой точкой GPS C_t и точкой C_{t-1} превышает предварительно выбранный допуск на расстояние.
3. При невозможности успешного отображения конкретной точки GPS с помощью основного алгоритма сопоставления карт. В таком случае процедура сбрасывается для запуска совершенно новой задачи сопоставления.

Большое расстояние между двумя последовательными точками GPS может возникнуть, если приемник временно прекращает отслеживать сигнал GPS из-за препятствий или из-за других кратковременных механических неисправностей.

InitialMapping() реализует следующие этапы для определения начального соответствия C_t и L (рис.5).

1. Получить координату C_0.
2. Сформировать $\mathbf{L}_\Delta = \{\mathbf{L}_{\Delta i}\}$- вспомогательные опорные отрезки, геометрия которых хоты бы частично находится внутри окружности с центром в C_{GPS} и радиусом Δ=200 м
3. Если $L_\Delta = Ø$, то вывести ошибку привязки.
4. Среди всех L_Δ найти отрезки $\mathbf{L_{Near}} = \{\mathbf{L_{Neari}}\}$, ближайшие к C_{GPS} с допуском $\boldsymbol{\Theta} = 50$ м.
5. Если L_{Near} есть множество из одного элемента, а предыдущее состояние отсутствует, то искомый C_L находится на противоположном от C_{GPS} конце минимального отрезка M (в данном случае b).
6. Если L_{Near} есть множество из нескольких элементов, тогда рассматриваются случаи, когда будет несколько L_{Near} и несколько C_L, и возможно есть предыдущее состояние.

$\mathbf{N}_\Delta = \{\mathbf{N}_{\Delta i}\}$ - опорные точки в дельта-окрестности (Δ=200 м) от текущей GPS-координаты C_{GPS}, $\mathbf{N_б}$ – точки, не входящие в Δ-окрестность, но являющиеся концами найденных отрезков.

Алгоритм Map() состоит из следующих этапов (рис. 6):

1. Получить следующую точку C_t, имея после применения InitialMapping() C_{t-1}.
2. Сформировать отрезок между точками C_{t-1} и C_t.
3. Оценить близость и ориентацию линии GPS к текущему согласованному сегменту дороги L_i. (см. п. VI).
4. Если очередная точка C_t не отображается на текущий сегмент L_i, найти другой сегмент дороги, L_{i+1}, который либо подключен к L_i, либо находится ниже по потоку от L_i. (Дуга L_{i+1} также выбирается на основе той же схемы оценки близости и ориентации -см. Рис. 9).

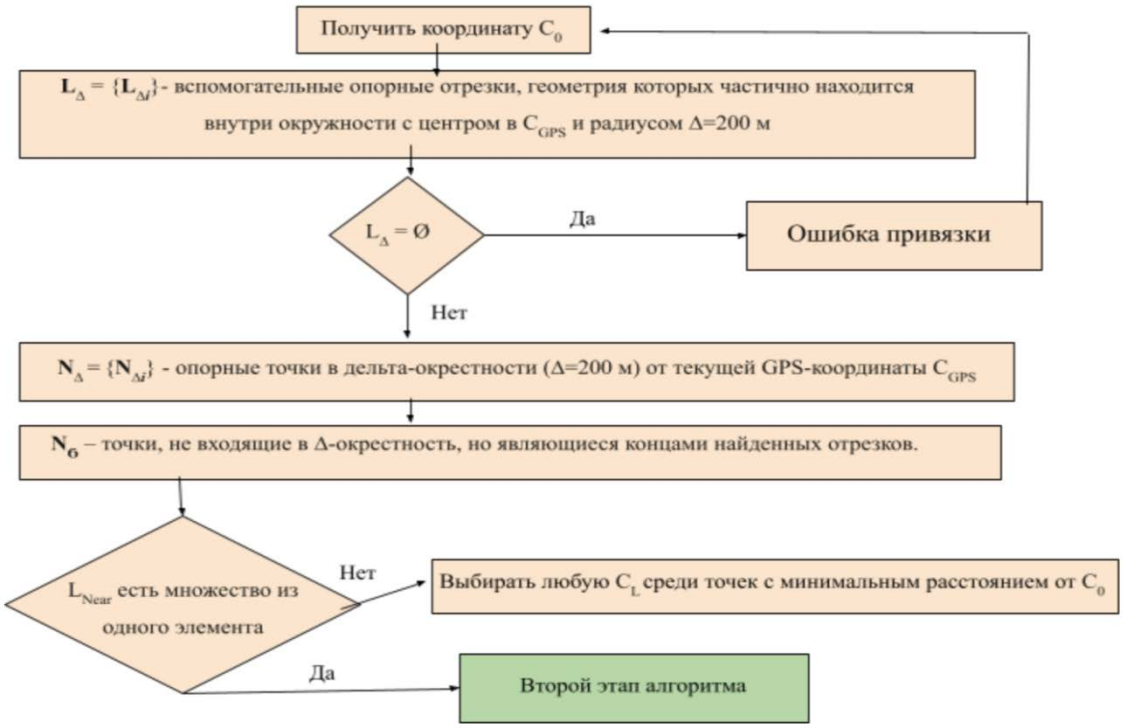

Рис. 5. Схема работы InitialMap()

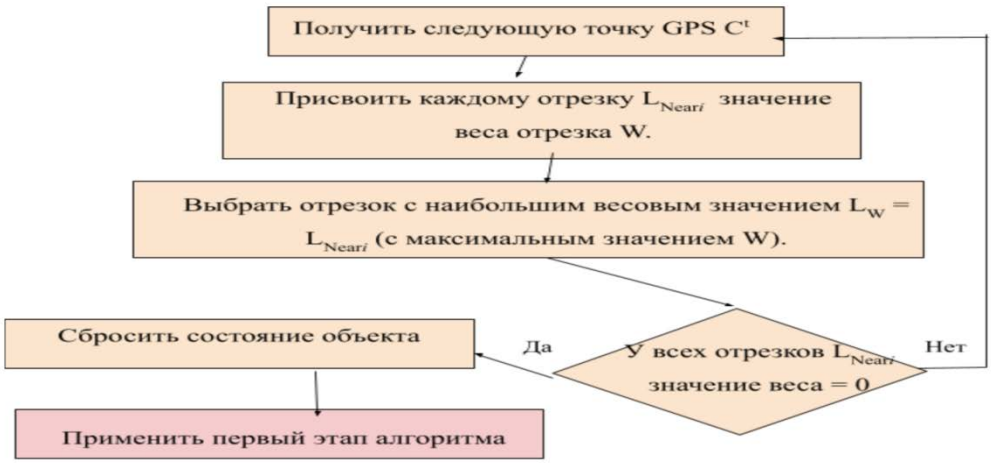

Рис. 6. Схема работы метода Map()

VI. ПРОВЕДЕННЫЕ ОЦЕНКИ

Метод характеризуется тремя оценками:
- оценка близости
- оценка ориентации
- оценка сходства

Оценка близости и ориентации

Первая и самая простая – оценка близости точки C_t к дуге L_i [12]. Это вычисление кратчайшего (или перпендикулярного) расстояния от C_t до L_i. Если на самом деле точка C_t связана с дугой L_i, то можно ожидать, что точка C_t близка к L_i. Однако предположение, что правильное совпадение найдено только из-за геометрического факта,

что дуга является ближайшей к точке, ошибочно. Тот факт, что дуга находится ближе всего к точке, свидетельствует о том, что кандидатура на сопоставление должна рассматриваться, но не о том, что правильное сопоставление определенно найдено.

Вторая оценка позволяет определить, пересекаются ли линия, соединяющая точки C_t и C_{t-1}, и дуга L_i. Если пересечение происходит и линии следуют в одном и том же направлении, вероятно, было найдено правильное совпадение. И наоборот, если две линии почти перпендикулярны друг другу, то очень маловероятно, что было найдено правильное совпадение. В обоих случаях линии должны пересекаться на сегменте дуги и между точками C_t и C_{t-1}, а не на продолжениях ни одной из этих линий. Если пересечение происходит на продолжении любой из этих линий, критерий пересечения недействителен. На рис.7 проиллюстрированы различные случаи пересечения. Пересечения a и b показывают правильные совпадения. Пересечение c неверно из-за большого угла между пересекающимися линиями C_2, C_3 и L_2. Пересечение d приходится на продолжение линий C_3, C_4 и L_1, и поэтому решение о том, находится ли точка C_4 на дуге L_1 или нет, не должно основываться на том факте, что эти линии пересекаются где-то в пространстве.

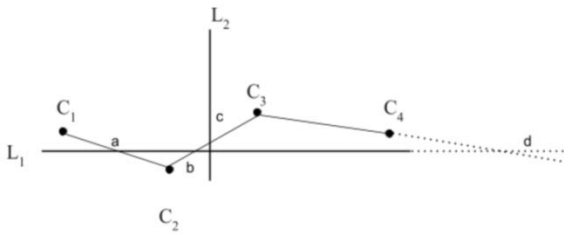

Рис. 7. Пересечение линий GPS и сегментов дорог.

Направление линии между последовательными точками GPS и направление линии между двумя узлами сетевой дуги должны быть одинаковыми [13]. Для оценки сходства можно использовать несколько метрик ориентации, таких как угол или азимут. В предлагаемом алгоритме используется критерий подобия азимута, чтобы определить, может ли отрезок быть кандидатом на совпадение. Преимущество вычисления азимута вместо простого угла между линиями состоит в том, что азимут имеет диапазоны от 0 до 2, и, следовательно, он различает 2 параллельные линии, которые указывают в противоположных направлениях.

Поскольку описанный подход к решению проблемы сопоставления карт заключался в том, что доступна только информация о координатах, был рассчитан азимут линии GPS. Азимут дуги A не доступен из других источников и поэтому должен быть рассчитан. Азимут части линии от C_1 до C_2 может быть вычислен по формуле

$$\mathbf{Az}_{\mathbf{P^1,P^2}} = \tan^{-1}\frac{\mathbf{x_2 - x_1}}{\mathbf{y_2 - y_1}}$$

Оценка сходства

Чтобы определить, какой отрезок L_i должен быть сопоставлен с точкой C_t, создана система весов, основанная на критериях сходства [14, 15]. Взвешивающая система оценивает несколько подходящих дуг для правильного соответствия, вычисляя оценку вероятности на основе различных критериев, которые обсуждались ранее. Этими критериями являются: (а) перпендикулярное расстояние точки GPS от сегмента дуги, (б) степень параллелизма между линией GPS и дугой дорожной сети, (в) угол пересечения (если таковой существует).

Весовое значение W вычисляется по формуле (1). Выбирая для этих параметров различные значения из формул (2), (3), (4), можно контролировать количество веса, которое дается конкретной характеристике соответствия. Например, выбор большего значения веса для C_{AZ} по сравнению с C_D означает, что сходство в ориентации важнее, чем близость. Выбор более высокого значения для n_D, чем для n_{AZ}, означает, что соответствующий вес будет уменьшаться быстрее, когда точка GPS находится дальше от дуги L_i. И наоборот, поскольку ориентация линий расходится друг от друга. Использование косинуса при вычисления весов, связанных с азимутом, и применения нечетного числа для мощности ni не только уменьшает вес при увеличении ΔAZ, но также вводит отрицательный вес, если ΔAZ больше 90. Соответственно, общий вес совпадения будет уменьшаться для линий, которые указывают в противоположных направлениях, и вес большого расстояния может быть компенсирован весом несовместимой ориентации.

Пропускание дуг

Топологически обоснованный алгоритм согласования предполагает, что в момент завершения пользователем путешествия по дуге L_i следующая «перемещаемая» дуга L_{i+1}, должна начинаться с конечного узла L_i [9, 16,

17]. Следовательно, если точка C_t не может быть отображена на L_i, она должна быть отображена на L_{i+1}. Однако иногда это предположение не соответствует действительности. Например, сеть может включать в себя очень короткую дугу, по которой пользователь движется очень быстро и не имеет возможности сделать наблюдение местоположения по этой дуге. То же самое может произойти, если сигнал GPS заблокирован на несколько секунд, а когда определение позиции возобновлено, пользователь уже находится на дуге L_{i+2} (см. Рис. 8).

Рис. 8. Пропуск дуги в процессе сопоставления.

Проблема пропуска дуги может быть решена двумя различными способами [18]. Первое и самое простое решение - перезапустить процесс сопоставления с помощью InitialMapping() всякий раз, когда возникает ситуация пропуска дуги. То есть, если точка P_t не может быть сопоставлена на текущей дуге L_i или на любой другой дуге L_{i+1}, которая связана с дугой L_i, то надо перезапустить процесс сопоставления, и C_t станет C_0. Недостаток этого простого решения состоит в том, что уже известная информация о местоположении пользователя теряется в процессе инициализации, и пользователь должен быть перемещен относительно всей сети N.

Другое решение проблемы пропуска дуги - попытка отобразить точку C^t на все дуги сети, которые связаны с конечными узлами всех возможных L_{i+1}. То есть вместо поиска совпадения только для дуг, которые связаны с конечной точкой текущей дуги, можно искать совпадение для дуг, которые связаны с конечными точками дуг, которые связаны с концом точки текущей дуги. Это может увеличить потребность в топологическом учете, но в то же время ограничит следующую подходящую дугу в окрестности текущей.

VII. ПРОГРАММНАЯ РЕАЛИЗАЦИЯ АЛГОРИТМА

Для визуальной составляющей алгоритма привязки используется платформа Unity3D [19, 20, 21], которая позволяет создавать 3D приложения реального времени, работающие на различных устройствах. Непосредственное программирование скриптов осуществлялось в редакторе Visual Studio на языке C# [22, 23, 24]. Данный редактор тесно интегрирован с платформой Unity, что позволяет использовать существующие инструменты Visual Studio для Unity, такие как редактирование, отладка. Это позволяет тратить меньше времени на выполнение простых задач, обеспечивает повышение производительности.

VIII. ЗАКЛЮЧЕНИЕ

Процедура сопоставления карт апробирована с помощью GPS-наблюдений, проведенных на маршруте, охватывающем различные условия эксплуатации. Обоснование выбора набора данных состояло в том, чтобы проверить тщательность процедуры в неблагоприятных условиях. Проведены преднамеренно введенные отключения GPS продолжительностью 15-20 секунд для проверки производительности во время отключений, фокусируясь на таких областях, как крутые повороты на дороге, переезды, светофоры, которые представляют собой наиболее сложные сценарии для предлагаемого алгоритма. Другой сложной особенностью этой траектории была ее переменная скорость с частыми остановками и внезапными ускорениями. Скорость объекта постоянно менялась из-за переездов, светофоров и остановок. Максимальная ошибка положения (в метрах) для всех отключений показана на диаграмме 1.

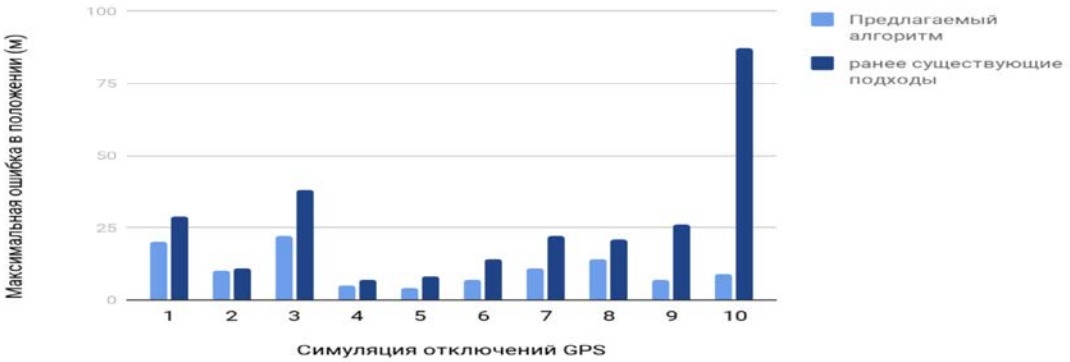

Диаграмма 1. Максимальные ошибки в положении на тестируемом маршруте

Полученные данные сравнивались с эталонным решением. Можно заметить, что предложенный алгоритм показал улучшение во всех десяти отключениях GPS по сравнению с ранее существовавшими подходами (см. Диаграмму 1). Во время отключения №10 алгоритм показал наибольшее улучшение - 90%, которое произошло во время поворота. Ранее максимальная ошибка положения была 88 м, тогда как решение нового алгоритма отличалось только на 8 м от эталонной траектории. При отключении GPS ранее наблюдалась максимальная погрешность позиционирования 22 м против 11 м в модифицированном алгоритме, что является улучшением на 50%. Проведенные исследования показали, что данные, полученные от GPS-приемника, практически везде совпадают с графом дорог. Алгоритм хорошо работает даже с относительно плохими данными GPS.

БЛАГОДАРНОСТИ

Авторы выражают благодарность профессору Сергею Юрьевичу Соловьеву за ценные замечания и полезные советы, сделанные в ходе подготовки данной статьи.

БИБЛИОГРАФИЯ

[1] Lianxia Xi, Quan Liu, Minghua Li, Zhong Liu, “Map Matching Algorithm and Its Application,” College of Computer Science and Technology, Soochow University, SuZhou 215006, P. R. China, State Key Laboratory for Novel Software Technology, Nanjing University, Nanjing 210093, P. R. China, vol. 7, 2007.

[2] Greenfeld, J.S. “Matching GPS observations to locations on a digital map,” - Washington: In proceedings of the 81st Annual Meeting of the Transportation Research Board. 2002.

[3] Hofmann-Wellenhoff, B., Lichtenegger H. and Collins J. GPS: Theory and Practice, 4th Edition, Springer-Verlag, New York, 1997.

[4] Joshua S. Greenfeld. “Matching GPS Observations to Locations on a Digital Map. - Washington,” DC: Department of Civil and Environmental Engineering, New Jersey Institute of Technology, 2002. p.13

[5] Pereira, F.C., Costa, H. and Pereira, N.M. “An off-line map-matching algorithm for incomplete map databases,” European Conference of Transport Research Institutes (ECTRI), 2009. p. 124

[6] Pingfu Chao, Yehong Xu, Wen Hua, and Xiaofang Zhou. “A Survey on Map-Matching Algorithms, Australia: School of Information Technology and Electrical Engineering,” The University of Queensland, 2009. p.12

[7] Zheng, K., Zheng, Y., Xie, X., Zhou, X. “Reducing uncertainty of low-sampling-rate trajectories” IEEE 28th International Conference on Data Engineering. 2012. pp.1144–1155.

[8] Chen, W., Yu, M., Li, Z.-L. Chen, Y.-Q. “Integrated vehicle navigation system for urban applications” Proceedings of the 7th International Conference on Global Navigation Satellite Systems (GNSS), 2003, pp. 15-22.

[9] Quddus M.A., Ochieng W.Y., Zhao L., Noland R.B. “A general map matching algorithm for transport telematics applications”, GPS Solutions, 2003. pp. 157-167.

[10] Gustafsson, F., Gunnarsson, F., Bergman, N. Forssell, U, Jansson, J., Karlsson, R., Nordlund. P. “Particle filters for positioning, navigation, and tracking”, IEEE Transactions on Signal Processing. 2002, pp. 50, 425- 435.

[11] Krakiwsky, E. J., Harris C. B., Wong R. V. C. “A Kalman Filter for Integrating Dead Reckoning, Map Matching and GPS Positioning”, Proceedings of IEEE Position Location and Navigation Symposium. 1988, pp. 39-46.

[12] Bernstein D., Kornhauser A. An introduction to map matching for personal navigation assistants [Электронный ресурс]. – Электрон. дан. - URL: http://www.njtude.org/reports/mapmatchintro.pdf. (Дата обращения: 19 февраля 2020).

[13] White, C.E., Bernstein, D., Kornhauser, A.L. “Some map matching algorithms for personal navigation assistants”, Transportation Research Part C 8, 2000. pp. 91-108.

[14] Blazquez, C.A., Vonderohe, A.P. “Simple map-matching algorithm applied to intelligent winter maintenance vehicle data”, Transportation Research Record 1935, 2005. pp .68-76.

[15] Yin H., Wolfson O. “A weight-based map matching method in moving objects databases”, Scientific and Statistical Database Management, 2004. pp. 437-438.

[16] Meng, Y. “Improved Positioning of Land Vehicle in ITS Using Digital Map and Other Accessory Information”, 2006.

[17] Taylor, G., Blewitt, G., Steup, D., Corbett, S., Car, A. “ Road reduction filtering for GPS-GIS navigation”, 2001, pp. 193-207.

[18] Ochieng, W.Y., Quddus, M.A. and Noland, R.B. “Map matching in complex urban road networks”, Brazilian Journal of Cartography (Revista Brasileira de Cartografia), 55(2), 1-18.

[19] Ларкович С.Н. “Unity на практике. Создаем 3D-игры и 3D-миры.” - М.: Наука и Техника, 2019. - 272 с.

[20] Jeremy Gibson Bond. “Introduction to Game Design, Prototyping, and Development: From Concept to Playable Game with Unity and C#, Addison-Wesley Professional”, 2017, vol. 928.
[21] Thorn A. “Unity Animation Essentials” - Packt Publishing Ltd, 2015, 176 p.
[22] Troelsen A., Japikse P. “Pro C# 7 With .NET and .NET Core”, Apress, 2017, vol. 1328 .
[23] Gittleman A. “Computing with C# and the .NET Framework”, 2011, Jones & Bartlett Learning, vol. 756.
[24] Albahari J., Albahari B. “C# 7.0 Pocket Reference”, 2017, O'Reilly, vol. 240.

About the algorithm for linking the coordinates of an object to the road graph

Irina Polyakova, Svetlana Burdukovskaya

Abstract --- The article deals with algorithm for linking the coordinates of an object to a road graph. Similar algorithms are used in every application due to the widespread use of electronic devices. Such algorithms may have some inaccuracies, which makes it difficult to determine the exact location of the object.

In this paper, we search for the reasons for the inaccuracies of the algorithm using previously existing approaches, and propose an algorithm that eliminates some inaccuracies and time delays. The presented algorithm has estimates of similarity, proximity, and orientation. For the estimates, methods are considered that allow you to determine the location on each graph more accurately and quickly. The formulas for finding the exact GPS-coordinates which are used in this algorithm are presented.

All the data obtained were compared with the reference solution. The algorithm showed the greatest improvement - 90% - that occurred during the turn. Previously, the maximum position error was 88 m, while the solution of the new algorithm differed only by 8 m from the reference trajectory. Studies have shown that the data received from the GPS receiver coincides almost everywhere with the road graph. The proposed algorithm showed improvements of an average of 50% compared to the algorithm, which is based on previously existing approaches. The algorithm works well even with relatively poor or busy GPS data.

The algorithm in question was implemented in C# using the platform .NET and Unity3D, as well as Visual Studio development environments.

***Keywords* — map matching algorithm, navigation systems, object coordinates, object trajectory, software implementation of the algorithm.**

REFERENCES

[1] Lianxia Xi, Quan Liu, Minghua Li, Zhong Liu, "Map Matching Algorithm and Its Application," College of Computer Science and Technology, Soochow University, SuZhou 215006, P. R. China, State Key Laboratory for Novel Software Technology, Nanjing University, Nanjing 210093, P. R. China, vol. 7, 2007.

[2] Greenfeld, J.S. "Matching GPS observations to locations on a digital map," - Washington: In proceedings of the 81st Annual Meeting of the Transportation Research Board. 2002.

[3] Hofmann-Wellenhoff, B., Lichtenegger H. and Collins J. GPS: Theory and Practice, 4th Edition, Springer-Verlag, New York, 1997.

[4] Joshua S. Greenfeld. "Matching GPS Observations to Locations on a Digital Map. - Washington," DC: Department of Civil and Environmental Engineering, New Jersey Institute of Technology, 2002. p.13

[5] Pereira, F.C., Costa, H. and Pereira, N.M. "An off-line map-matching algorithm for incomplete map databases," European Conference of Transport Research Institutes (ECTRI), 2009. p. 124

[6] Pingfu Chao, Yehong Xu, Wen Hua, and Xiaofang Zhou. "A Survey on Map-Matching Algorithms, Australia: School of Information Technology and Electrical Engineering," The University of Queensland, 2009. p.12

[7] Zheng, K., Zheng, Y., Xie, X., Zhou, X. "Reducing uncertainty of low-sampling-rate trajectories" IEEE 28th International Conference on Data Engineering. 2012. pp.1144–1155.

[8] Chen, W., Yu, M., Li, Z.-L. Chen, Y.-Q. "Integrated vehicle navigation system for urban applications" Proceedings of the 7th International Conference on Global Navigation Satellite Systems (GNSS), 2003, pp. 15-22.

[9] Quddus M.A., Ochieng W.Y., Zhao L., Noland R.B. "A general map matching algorithm for transport telematics applications", GPS Solutions, 2003. pp. 157-167.

[10] Gustafsson, F., Gunnarsson, F., Bergman, N. Forssell, U, Jansson, J., Karlsson, R., Nordlund. P. "Particle filters for positioning, navigation, and tracking", IEEE Transactions on Signal Processing. 2002, pp. 50, 425- 435.

[11] Krakiwsky, E. J., Harris C. B., Wong R. V. C. "A Kalman Filter for Integrating Dead Reckoning, Map Matching and GPS Positioning", Proceedings of IEEE Position Location and Navigation Symposium. 1988, pp. 39-46.

[12] Bernstein D., Kornhauser A. An introduction to map matching for personal navigation assistants URL: http:// www.njtude.org/reports/mapmatchintro.pdf. (Accessed: February 19, 2020).

[13] White, C.E., Bernstein, D., Kornhauser, A.L. "Some map matching algorithms for personal navigation assistants", Transportation Research Part C 8, 2000. pp. 91-108.

[14] Blazquez, C.A., Vonderohe, A.P. "Simple map-matching algorithm applied to intelligent winter maintenance vehicle data", Transportation Research Record 1935, 2005. pp .68-76.

[15] Yin H., Wolfson O. "A weight-based map matching method in moving objects databases", Scientific and Statistical Database Management, 2004. pp. 437-438.

[16] Meng, Y. - Hongkong: "Improved Positioning of Land Vehicle in ITS Using Digital Map and Other Accessory Information", 2006.

[17] Taylor, G., Blewitt, G., Steup, D., Corbett, S., Car, A. " Road reduction filtering for GPS-GIS navigation", 2001, pp. 193-207.

[18] Ochieng, W.Y., Quddus, M.A. and Noland, R.B. "Map matching in complex urban road networks", Brazilian Journal of Cartography (Revista Brasileira de Cartografia), 55(2), 1-18.

[19] Larkovich S. N. " Unity in practice. We create 3D games and 3D worlds."- Moscow: Nauka i Tekhnika, 2019 - 272 p.

[20] Jeremy Gibson Bond. "Introduction to Game Design, Prototyping, and Development: From Concept to Playable Game with Unity and C#, Addison-Wesley Professional", 2017, vol. 928.

[21] Thorn A. "Unity Animation Essentials" - Packt Publishing Ltd, 2015, 176 p.

[22] Troelsen A., Japikse P. "Pro C# 7 With .NET and .NET Core", Apress, 2017, vol. 1328 .

[23] Gittleman A. "Computing with C# and the .NET Framework", 2011, Jones & Bartlett Learning, vol. 756.

[24] Albahari J., Albahari B. "C# 7.0 Pocket Reference", 2017, O'Reilly, vol. 240.