УДК 004.021
DOI: 10.15827/0236-235X.130.257-265
Дата подачи статьи: 27.01.20
2020. Т. 33. № 2. С. 257–265

Модифицированный алгоритм построения карты занятости по облаку точек от нескольких лидаров

И.О. Шепель [1,2], *инженер-исследователь, аспирант, info@ictis.sfedu.ru*

[1] *Научно-конструкторское бюро вычислительных систем, г. Таганрог, 347936, Россия*
[2] *Институт компьютерных технологий и информационной безопасности Южного федерального университета, г. Таганрог, 347922, Россия*

В работе рассматривается проблема построения модели проходимости окружающего пространства по данным от нескольких сенсоров, установленных на роботизированной платформе. Эта проблема является ключевой для решения задачи автономного движения без использования какой-либо априорной информации о среде. Применение нескольких сенсоров обусловлено физическими ограничениями датчиков и необходимостью уменьшить размер слепых зон вокруг автономных роботизированных платформ. Целью работы является качественное улучшение алгоритма синтеза карты окружающего пространства путем добавления возможности обработки данных от нескольких независимых источников данных с их последующим комплексированием.

В статье представлен алгоритм построения карты занятости для системы технического зрения, состоящей из нескольких разнесенных в пространстве лидаров. Предложен подход к объединению данных от различных сенсоров без требования аппаратной синхронизации данных. Такое объединение происходит на алгоритмическом уровне и не накладывает ограничений на количество сенсоров или на физическую природу их данных. Оно позволяет обрабатывать облака точек от сенсоров независимо друг от друга, тем самым уменьшая вычислительную сложность по сравнению с обработкой объединенного облака точек. Кроме того, представлено дополнение алгоритма трассировки лучей, которое учитывает размещение нескольких лидаров на роботизированной платформе и объединяет поля зрения этих датчиков, что позволяет получить более полную модель проходимости окружающей среды. Описанный модифицированный алгоритм построения адаптивного буфера безопасности вокруг препятствий на карте занятости дает возможность планировать траекторию на равном удалении от объектов в сложных сценариях.

Ключевые слова*: карта занятости, лидар, облако точек, комплексирование данных, обнаружение препятствий, автономное движение.*

В настоящее время развитие вычислительной техники, а также распространение множества роботизированных мобильных платформ сделали возможным промышленное использование роботов для решения целого ряда задач. При этом одним из важных требований к подобным системам является способность автономно двигаться, планируя траекторию движения и избегая препятствий без вмешательства оператора.

Существует несколько наиболее распространенных подходов к нахождению препятствий по данным сенсоров технического зрения. В представленном исследовании в качестве основного сенсора используются лидары, позволяющие измерять расстояния до удаленных объектов окружающей среды с помощью лазерного излучения. Одним из подходов для решения задачи детектирования препятствий является построение карт занятости [1]. Карта занятости – это модель проходимости окружающей среды, представленная в виде дискретной решетки, каждая из ячеек которой помечена как препятствие, свободная или ячейка с неизвестной проходимостью. Подобные карты могут строиться по данным от различных сенсоров, таких как радары, стереокамеры и лидары. Имеется множество различных надстроек над представленным в работе [1] базовым алгоритмом синтеза карты проходимости. Эти надстройки позволяют выделять нависающие структуры (например, перекрытия мостов, ветви деревьев, навесы, под которыми можно свободно перемещаться) [2], варьировать размер одиночной ячейки карты для более точного определения границ препятствий [3], строить ограничивающие параллелепипеды объектов [4] и т.д. Однако в этих работах не рассматриваются конструктивные ограничения, накладываемые установкой сенсоров на *роботизированную платформу* (РП). При этом даже самые современные лидары обладают доста-

точно низким вертикальным разрешением, и поле зрения датчика сильно зависит от способа его установки. Так, с помощью пересечения сканирующих плоскостей двух лидаров авторы [5] решают сложную задачу нахождения отрицательных препятствий – ям, оврагов, крутых склонов и т.п.

В данной работе предлагается подход к решению проблемы комплексирования информации от различных сенсоров технического зрения и построения общей модели окружающего РП мира. В основу алгоритма, описанного далее, взят алгоритм из [6]. В статье также представлены дополнения к этому алгоритму, позволяющие учитывать расположение лидара на платформе и его поле зрения при трассировке лучей. Кроме того, карта занятости структурно разделена на единовременную, которая строится по каждому новому сообщению с данными от сенсора (в данном случае по облаку точек от лидара), и на накапливаемую, объединяющую данные от этих единовременных карт. Каждая единовременная карта получает данные от различных сенсоров, благодаря чему комплексирование данных происходит на уровне этих карт занятости, а не облаков точек.

Алгоритм построения карты занятости по данным от лидара

Упрощенная схема алгоритма представлена на рисунке 1.

На вход алгоритма подается массив с трехмерными координатами точек в метрах (так называемое облако точек). Этот массив генерируется сенсором, в данном случае лидаром. Затем происходит генерация единовременной карты проходимости:

– все окружающее РП пространство разделяется на квадратные ячейки одинакового размера;

– трехмерные координаты точек облака распределяются в соответствующие им двумерные ячейки (при распределении не используется вертикальная координата);

– в каждой ячейке принимается решение о наличии в ней препятствия;

– на двумерной карте с отмеченными препятствиями и известным положением лидара производится трассировка его лучей; все ячейки в порядке распространения луча до первого препятствия помечаются как свободные от препятствий, а после – как ячейки с неизвестным типом.

При этом дискретная двумерная решетка неподвижна относительно поверхности Земли, а платформа свободно перемещается по ней. Решетка описывается следующим набором параметров: ширина, длина и разрешение – длина стороны квадратной ячейки. Так как синтез модели проходимости должен происходить в реальном масштабе времени, объем памяти для хранения карты должен быть жестко ограничен. Поэтому используется механизм так называемых сворачиваемых карт [7]. Эта модель в памяти компьютера реализуется как тор, в то время как в реальном мире земная поверхность замещается картой (рис. 2), то есть верхняя граница карты совпадает с нижней, а граница слева – с границей справа. Для определения ячейки, в которую попадает точка, используется следующая формула:

$$map\left[\left|\frac{rem\left(\frac{x}{W}\right)}{dx}\right|\right]\left[\left|\frac{rem\left(\frac{y}{H}\right)}{dy}\right|\right] = cell(x, y), \quad (1)$$

где *rem* – операция деления по модулю; *cell*(*x*, *y*) – ячейка карты, соответствующая точке с координатами *x*, *y*; *W*, *H* – соответственно ширина и высота карты; *dx*, *dy* – размеры ячейки по ширине и высоте; *map* – карта

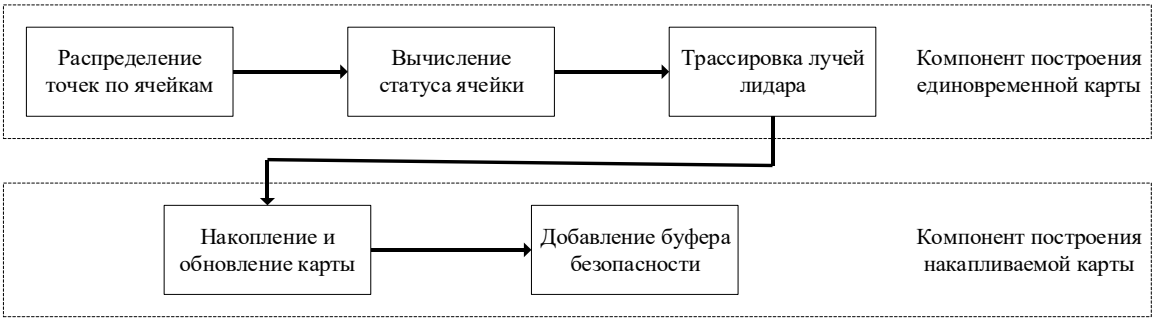


Рис. 1. Упрощенная схема алгоритма построения карты занятости

Fig. 1. The algorithm simplified diagram for building an occupancy grid

занятости. Для удобства и простоты представления и карта, и ячейки в ней квадратные. Размер стороны ячейки характеризует разрешающую способность карты. Так, препятствия с площадью проекции на поверхность Земли, меньшей площади одной ячейки, также будут занимать на карте занятости всю ячейку целиком. Для городской и сельской местности авторы [6] выбрали длину ячейки, равную 0,2 м. Размер самой карты выбирается исходя из физических ограничений самого датчика и размера ячейки так, чтобы хотя бы одна точка попадала в ячейку.

Из-за периодической структуры карты не существует взаимно-однозначного соответствия между координатами ячеек решетки и координатами РП: при движении платформа будет циклически перемещаться по одним и тем же ячейкам. Поэтому вокруг РП выделяется область интереса, которая перемещается вместе с платформой (рис. 2). При этом ячейки, о проходимости которых принимается решение в текущий момент и на которых затем будет строиться траектория движения, находятся только внутри этой области. В процессе движения РП некоторые ячейки выходят из области интереса и данные, находящиеся в них, удаляются из карты. Чтобы область интереса при движении не пересекалась сама с собой, то есть не ссылалась дважды на одну и ту же ячейку сворачиваемой карты, ее размер выбирается исходя из неравенства $l_{roi} < H/\sqrt{2}$, где l_{roi} – длина стороны области интереса; H – длина кратчайшей стороны сворачиваемой карты. Центр обновляемой области интереса совпадает с положением

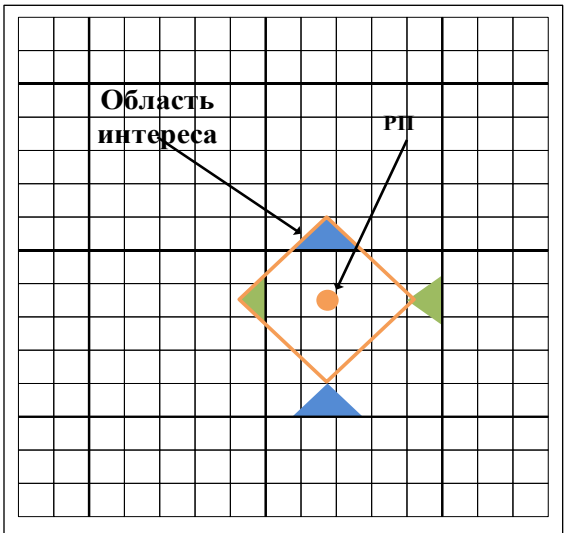


Рис. 2. Модель сворачиваемой карты

Fig. 2. The wrappable map model

платформы, и при ее движении эта область перемещается по карте вместе с платформой.

После распределения точек в каждой ячейке вычисляется ее статус – занята она препятствием или свободна [6]. Для этого проверяются следующие условия:

– наличие положительных препятствий

$$N > 1 \,\&\, p_z^N - p_z^1 \geq \tau_b, \quad (2)$$

– наличие нависающих препятствий

$$\exists j : p_z^{j+1} - p_z^j > h_{rob} \,\&\, p_z^j - p_z^1 < \tau_b, \quad (3)$$

где N – количество точек в ячейке; p_z^N, p_z^1, p_z^j – высота последней, первой и j-й точек в массиве соответственно; τ_b – пороговое значение высоты для присвоения ячейке статуса препятствия; h_{rob} – минимальная высота, необходимая для проезда робота.

Положительными препятствиями называют вертикально протяженные объекты, непроходимые для РП, нависающими – различные тоннели, перекрытия мостов, ветви деревьев и т.д., под которыми РП может свободно перемещаться. Ячейки, в которых выполняется (2) и не выполняется (3), помечаются в единовременной карте как препятствия.

Карта с размеченными препятствиями подается на вход алгоритма трассировки лучей, который воспроизводит физический принцип работы лидара. Лидар излучает электромагнитные импульсы в определенных направлениях, которые затем отражаются от различных объектов окружающей среды. Все пространство, пройденное этими импульсами до момента отражения, свободно от препятствий. Для этого в соответствии с расположением сенсора на платформе на карту помещается виртуальный лидар, от которого и строятся траектории распространения лучей. Все ячейки на таких траекториях обходятся в порядке распространения луча (рис. 3). До первого препятствия ячейки помечаются как свободные, а после – как ячейки с неизвестным типом проходимости (далее – неизвестные). Таким образом, алгоритм трассировки используется, чтобы размечать ячейки, в которые не попало достаточное количество точек облака, чтобы считать эти ячейки препятствием. Этот алгоритм позволяет решить проблему низкой плотности данных от лидаров, связанную с их низким вертикальным разрешением.

Построение карты по данным от нескольких лидаров

Алгоритм, приведенный в данной работе, дополняет алгоритм из [6] возможностью ра-

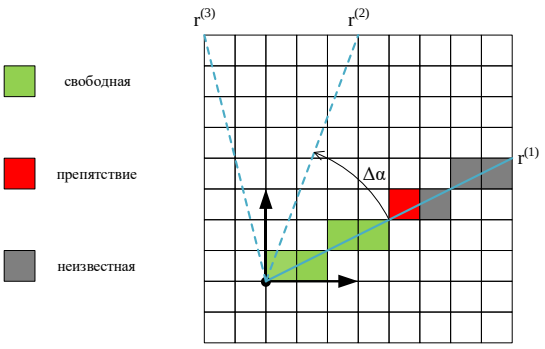


Рис. 3. Трассировка лучей виртуального лидара на дискретной решетке

Fig. 3. The virtual lidar ray tracing on a discrete grid

боты с несколькими лидарами. Существуют два подхода комплексирования данных от нескольких лидаров [8]. Первый подход – это комплексирование на уровне облаков точек, при котором массивы точек от всех лидаров объединяются и затем подаются на вход для дальнейшей обработки. Второй подход предполагает обработку каждого облака точек отдельно, а объединяются собственно результаты работы алгоритма на данных от разных сенсоров. В случае первого подхода сообщения от разных лидаров нуждаются во временной синхронизации. В противном случае они будут записаны в разные моменты времени, что при движении РП приводит к большим погрешностям в определении координат препятствия. Такой эффект наблюдается даже при работе с единственным круговым сканирующим лидаром. Кроме того, не все существующие сканирующие лазерные дальномеры поддерживают возможность аппаратной синхронизации.

По этим причинам комплексирование данных от различных источников производится на уровне карт занятости, а не облаков точек. На рисунке 1 представлена упрощенная схема алгоритма синтеза единовременной и накапливаемой карт. Процессы построения единовременной и накапливаемой карт выполняются раздельно соответствующими программными компонентами. Единовременные карты строятся независимо для каждого лидара, а затем объединяются в одну общую накапливаемую модель (рис. 4).

Для такого объединения карт необходимо точно знать позицию и ориентацию РП в пространстве в момент съемки облака точек от каждого из лидаров. Для этого компоненты построения единовременной карты сохраняют сообщения от *инерциально-навигационной спутниковой системы* (ИНСС) на борту РП синхронно с сообщением от лидара. Также сообщения от ИНСС используются при обновлении накапливаемой карты, чтобы нивелировать расстояние, пройденное РП за время обработки облака точек компонентом построения единовременной карты.

Алгоритм трассировки лучей

В [6] алгоритм трассировки лучей используется для маркировки свободных и неизвестных ячеек. Ячейки обходятся в порядке распространения луча и до ячейки-препятствия помечаются как свободные, а после – как неизвестные. В случае с единственным лидаром, размещенным на РП, эти зоны вне поля прямой видимости с неизвестным типом проходимости могут занимать большую область карты, что повышает риск необнаружения в них препятствия. Несколько сенсоров помогают уменьшить размер и количество слепых зон, однако требуют более сложного алгоритма маркировки ячеек. Для объединения данных от нескольких сенсоров в одной карте предлагается следующий подход.

Все единовременные карты и накапливаемая карта имеют общую систему координат с центром, привязанным к положению робота. При этом трассировка лучей в каждой из единовременных карт происходит из ячейки, в которой находится соответствующий лидар. Для получения точной информации о взаимном положении лазерных дальномеров и ИНСС производится их калибровка [9]. При трассировке лучей от нескольких лидаров возможна ситуация, при которой одной и той же ячейке будет присвоен разный статус в различных единовременных картах. В этом случае приоритет маркировки следующий: 1 – ячейка-препятствие, 2 – свободная ячейка, 3 – неизвестная ячейка.

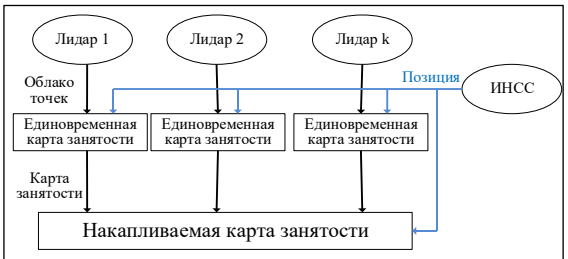


Рис. 4. Схема передачи данных с несколькими лидарами

Fig. 4. Data transfer scheme with multiple lidars

Таким образом, количество неизвестных из-за перекрытия области видимости ячеек карты уменьшается, и при этом не теряется информация о препятствиях в таких зонах. Карты, синтезированные по данным от нескольких и от одного источника, продемонстрированы на рисунке 5. Темно-серым цветом выделена область с неизвестными ячейками, серым – свободная область, наблюдаемая только одним лидаром (точки белого цвета), светло-серым – свободная область, видимая двумя лидарами (белые и синие точки). Черными ячейками выделены препятствия.

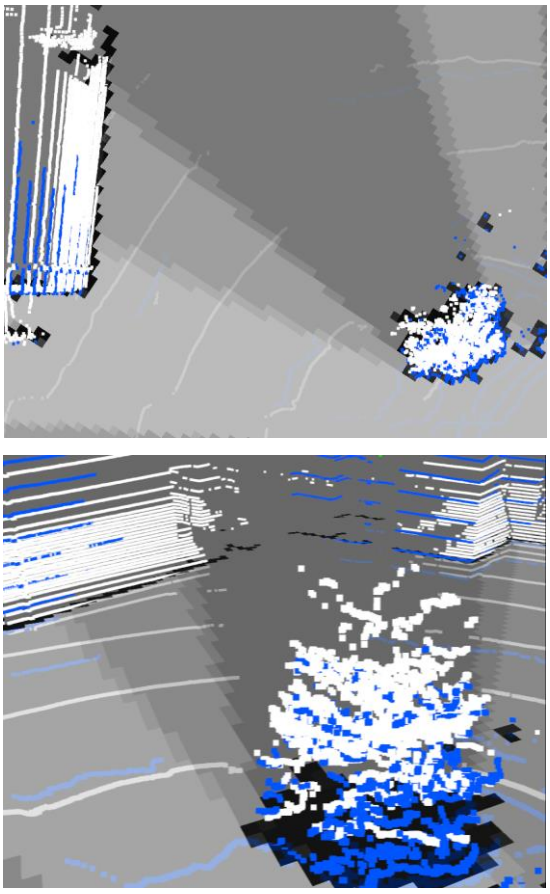


Рис. 5. Результат работы алгоритма трассировки для нескольких лидаров на реальной сцене

Fig. 5. The ray tracing algorithm result for several lidars on a real scene

Адаптивный буфер безопасности

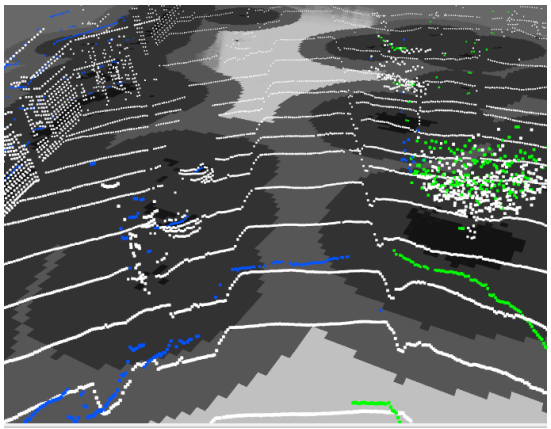
При решении задач определения препятствий движению РП и планирования траектории необходимо не только точно локализовать положение препятствий, но и учитывать габариты платформы. Наиболее широко используемым решением является добавление буфера безопасности – искусственного расширения препятствий. Радиус области, на которую расширяется препятствие, зависит от габаритов РП и обычно выбирается равным половине от наибольшего измерения платформы. Основная проблема такого подхода заключается в том, что при объезде препятствия кратчайшая, а следовательно, оптимальная, траектория движения будет вплотную прилегать к этому буферу. В реальных условиях при движении в узких проходах (например, в тоннелях) подобный маршрут движения может оказаться небезопасным. При этом наиболее безопасной будет равноудаленная от препятствий траектория движения. Для этого вводится так называемый адаптивный буфер безопасности. Он состоит из двух частей: жесткого буфера, где запрещены планирование траекторий и движение, и мягкого буфера, где запрещено планирование и разрешено движение. Изначально траектория строится в обход обоих буферов, но при движении изменяется поле зрения сенсоров, что приводит к появлению новых препятствий. В подобных ситуациях траектория перестраивается, только если пересекает жесткий буфер. Этот буфер строится как непроходимая область вокруг каждой ячейки-препятствия шириной, равной половине наибольшего габарита РП. Мягкий буфер строится вокруг жесткого по следующему алгоритму:

– карта занятости с добавленным на нее жестким буфером бинаризуется;

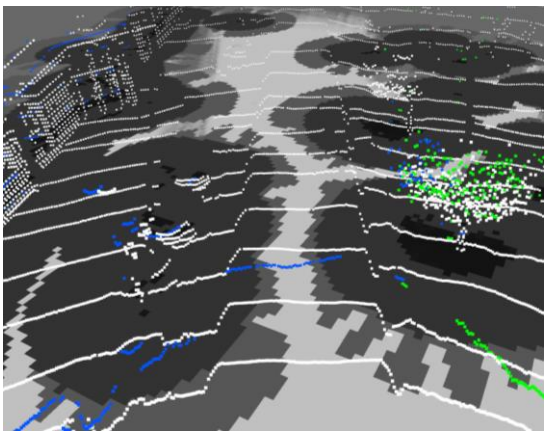
– по бинаризованному изображению выполняется дистантное преобразование;

– к полученному изображению применяется оператор Лапласа.

В результате работы алгоритма получаем мягкий буфер, который увеличивает расстояние от планируемой траектории до препятствий на заданное значение, а в случае пересечения с буфером от другого препятствия оставляет равноудаленную от этих двух препятствий зону. Результат работы алгоритма показан на рисунке 6. Слева и справа на сцене находятся естественные препятствия (кусты и деревья), вокруг которых строится буфер безопасности. Размеры жесткого и мягкого буферов соответственно равны 1,8 и 1,2 м. Белым цветом выделены свободные для перемещения ячейки, черным – препятствия, темно- и светло-серым со-



а)



б)

Рис. 6. Адаптивный буфер безопасности без оператора Лапласа (а) и с ним (б)

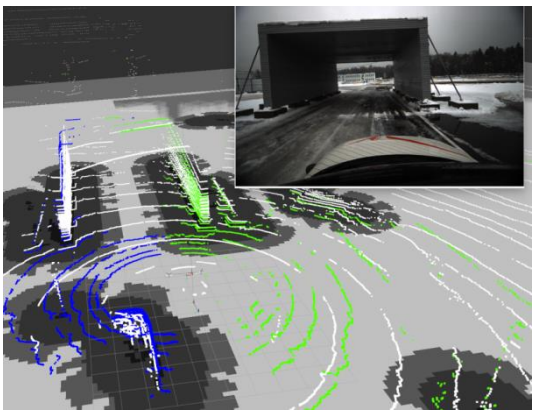
Fig. 6. Adaptive safety buffer without Laplace operator (a) and with it (б)

ответственно жесткий и мягкий буферы. Белым, синим и зеленым выделены также облака точек от различных лидаров. Если не используется оператор Лапласа, между препятствиями не существует свободных ячеек, по которым можно построить траекторию движения. При его применении пересечение буферов от двух препятствий вырезается и появляется свободная область, по которой возможно планировать траекторию. Подобный подход более подробно описан в [10].

Проведенные исследования

Разработка и начальное тестирование алгоритма проводились в среде симуляции CARLA на синтетических данных [11]. Дальнейшая проверка работы осуществлялась на реальных данных (рис. 7), заснятых на автомобиле Kia Soul с 3 встроенными лидарами Velodyne: 32-лучевым VLP32C на крыше и двумя 16-лучевыми VLP16, встроенными в крылья автомобиля (рис. 7), а также с бортовой ИНСС Atlans-C, с которой поступали данные о положении и ориентации автомобиля. При этом датчики в среде симуляции были размещены так же, как и на реальном автомобиле.

а)



б)

в)

Рис. 7. Карта занятости в среде CARLA (а) и на реальной сцене (б). Беспилотный автомобиль Kia Soul с системой технического зрения (в)

Fig. 7. Occupancy grid in the CARLA (a) and in the real scenario (б). Self-driving Kia Soul with a vision system (в)

Быстродействие программных компонент проверялось на двух наборах данных длиной в 3 минуты, один из которых был синтезирован с помощью среды CARLA, а другой снят в реальных условиях. Вычисления проводились на компьютере с процессором Intel Core i7-8700 3.2 ГГц и 32 ГБ оперативной памяти. Среднее время обработки единовременной карты для 16- и 32-лучевого лидаров составила 38 мс и 64 мс соответственно, в то время как синтез накапливаемой карты происходит за 16 мс. Такое быстродействие показывает, что и на реальных, и на модельных данных время обработки одного облака точек меньше, чем период между сообщениями с этими облаками (стандартный для лидаров Velodyne – 100 мс), что позволяет использовать программную реализацию алгоритма в реальном масштабе времени.

Представленная в статье модификация алгоритма построения карты занятости позволяет обрабатывать данные от нескольких различных лидаров в реальном масштабе времени, а также учитывать взаимное расположение датчиков на роботизированной платформе, за счет чего уменьшается слепая зона. Кроме того, представлена модификация, дающая возможность синтезировать на карте буфер безопасности, позволяющий РП планировать траекторию в узких местах. Модифицированный алгоритм был протестирован как на модельных, так и на реальных данных.

Дальнейшее развитие подхода

Предложенный алгоритм комплексирует данные от разных источников на уровне единовременных карт проходимости, но при этом алгоритм построения таких карт не универсален, а зависит от физической природы обрабатываемых данных. Для дальнейшего развития такого подхода к комплексированию необходимо разработать алгоритмы построения карт занятости по облакам точек от стереокамер и по радарным данным. Эти три типа сенсоров наиболее часто используются в робототехнических системах. Облака точек от стереокамер отличаются большей плотностью, чем лидарные, но намного более низкой точностью измерения дальности, в то время как радары миллиметрового диапазона характеризуются меньшим разрешением и плотностью возвращаемых данных.

Исследование выполнено при финансовой поддержке РФФИ, проект № 19-07-00570.

Экспериментальные исследования проведены совместно с Научно-конструкторским бюро вычислительных систем (г. Таганрог).

Литература

1. Elfes A. Using occupancy grids for mobile robot perception and navigation. Computer, 1989, no. 6, pp. 46–57.

2. Souza A., Gonçalves L.M.G. Occupancy-elevation grid: an alternative approach for robotic mapping and navigation. Robotica, 2016, no. 11, pp. 2592–2609.

3. Häselich M., Arends M., Wojke N., Neuhaus F., Paulus D. Probabilistic terrain classification in unstructured environments. Robotics and Autonomous Systems, 2013, no. 10, pp. 1051–1059.

4. Himmelsbach M., Mueller A., Luttel T., Wunsche H.J. LIDAR-based 3D object perception. Proc. of 1st Intern. Workshop on Cognition for Technical Systems. 2008. URL: https://mafiadoc.com/lidar-based-3d-object-perception-semantic-scholar_59a692091723dd0b40ac9c7c.html (дата обращения: 27.12.2019).

5. Shang E., An X., Li J., He H. A novel setup method of 3D LIDAR for negative obstacle detection in field environment. Proc. 17th Intern. IEEE Conf. ITSC, 2014, pp. 1436–1441.

6. Jaspers H., Himmelsbach M., Wuensche H.J. Multi-modal local terrain maps from vision and LiDAR. Proc. IV Intelligent Vehicles Symposium, IEEE, 2017, pp. 1119–1125.

7. Kelly A., Stentz A. Rough terrain autonomous mobility. Pt. 2: An active vision, predictive control approach. Autonomous Robots, 1998, vol. 5, no. 2, pp. 163–198.

8. Mertz C., Navarro-Serment L.E., MacLachlan R., Rybski P., Steinfeld A., Suppe A., Urmson C., Vandapel N., Hebert M., Thorpe C. Moving object detection with laser scanners. J. Field Robotics, 2013, no. 1, pp. 17–43.

9. Taylor Z., Nieto J. Motion-based calibration of multimodal sensor extrinsics and timing offset estimation. Transactions on Robotics, 2016, no. 5, pp. 1215–1229.

10. Чернухин Ю.В., Бутов П.А. Синтез тормозных квазиполей препятствий для бортовой системы автономного планирования траектории движения малогабаритных мобильных роботов // Инженерный

вестник Дона. 2014. № 2. URL: http://ivdon.ru/ru/magazine/archive/n2y2014/2382 (дата обращения: 27.12.2019).

11. Dosovitskiy A., Ros G., Codevilla F., Lopez A., Koltun V. CARLA: An open urban driving simulator. Proc. CORL, 2017, arXiv:1711.03938 [cs.LG].

Software & Systems
DOI: 10.15827/0236-235X.130.257-265

Received 27.01.20
2020, vol. 33, no. 2, pp. 257–265

Modified algorithm for building an occupancy grid from multi-lidar point cloud

I.O. Shepel [1,2], *Research Engineer, Postgraduate Student, info@ictis.sfedu.ru*

[1] *Scientific Design Bureau of Computing Systems, Taganrog, 347936, Russian Federation*
[2] *Institute of Computer Technologies and Information Security of Southern Federal University, Taganrog, 347922, Russian Federation*

Abstract. The paper considers building a possibility model problem for the environment using data from several sensors installed on a robotic platform. This is the key problem for solving the autonomous movement problem without using any a priori information. Several sensors usage due to the sensor physical limitations and to reduce blind spots around autonomous robotic platforms. The work aim is to improve occupancy grid building by adding the ability to process data from several independent sources.

The paper presents an algorithm for constructing an occupancy grid for a technical vision system consisting of several spaced lidars. There is an approach to combining data from different sensors without requiring hardware data synchronization.

This merge occurs at the algorithmic level and does not impose any restrictions on the number of sensors or on the physical nature of their data.

It allows processing point clouds from sensors independently, thus reducing the computational complexity compared to processing a combined point cloud.

In addition, there is a ray-tracing algorithm extension, which takes into account the placement of several leaders on a robotic platform and combines the view fields for these sensors, which allows them to get a more complete model of the cross-country environment. The described modified algorithm for building an adaptive safety buffer around obstacles on the occupancy grid makes it possible to plan a trajectory at an equal distance from objects in complex scenarios.

Keywords: occupancy grid, lidar, point cloud, data merging, obstacle detection, autonomous movement.

Acknowledgements. *The reported study was with the financial support of the RFBR, project no. 19-07-00570.*

Experimental research were conducted with the Scientific-Design Bureau of Computing Systems (Taganrog).

References

1. Elfes A. Using occupancy grids for mobile robot perception and navigation. *Computer*, 1989, no. 6, pp. 46–57.

2. Souza A., Gonçalves L.M.G. Occupancy-elevation grid: An alternative approach for robotic mapping and navigation. *Robotica*, 2016, no. 11, pp. 2592–2609.

3. Häselich M., Arends M., Wojke N., Neuhaus F., Paulus D. Probabilistic terrain classification in unstructured environments. *Robotics and Autonomous Systems*, 2013, no. 10, pp. 1051–1059.

4. Himmelsbach M., Mueller A., Luttel T., Wunsche H.J. LIDAR-based 3D object perception. *Proc. of 1st Intern. Workshop on Cognition for Technical Systems*, 2008. Available at: https://mafiadoc.com/lidar-based-3d-object-perception-semantic-scholar_59a692091723dd0b40ac9c7c.html (accessed December 27, 2019).

5. Shang E., An X., Li J., He H. A novel setup method of 3D LIDAR for negative obstacle detection in field environment. *Proc. 17th Intern. IEEE Conf. ITSC*, 2014, pp. 1436–1441.

6. Jaspers H., Himmelsbach M., Wuensche H.J. Multi-modal local terrain maps from vision and LiDAR. *Proc. IV Intelligent Vehicles Symposium*, IEEE, 2017, pp. 1119–1125.

7. Kelly A., Stentz A. Rough terrain autonomous mobility. Pt. 2: An active vision, predictive control approach. *Autonomous Robots*, 1998, vol. 5, no. 2, pp. 163–198.

8. Mertz C., Navarro-Serment L.E., MacLachlan R., Rybski P., Steinfeld A., Suppe A., Urmson C., Vandapel N., Hebert M., Thorpe C. Moving object detection with laser scanners. *J. Field Robotics*, 2013, no. 1, pp. 17–43.

9. Taylor Z., Nieto J. Motion-based calibration of multimodal sensor extrinsics and timing offset estimation. *Transactions on Robotics*, 2016, no. 5, pp. 1215–1229.

10. Chernukhin Yu.V., Butov P.A. The synthesis of inhibitory quasi-fields of obstacles for SUGV on-board path planning system. *Eng. J. of Don*, 2014, no. 2. Available at: http://ivdon.ru/ru/magazine/archive/n2y2014/2382 (accessed December 27, 2019) (in Russ.).

11. Dosovitskiy A., Ros G., Codevilla F., Lopez A., Koltun V. CARLA: An open urban driving simulator. *Proc. Conf. on Robot Learning*, 2017, arXiv:1711.03938 [cs.LG].

Для цитирования

Шепель И.О. Модифицированный алгоритм построения карты занятости по облаку точек от нескольких лидаров // Программные продукты и системы. 2020. Т. 33. № 2. С. 257–265. DOI: 10.15827/0236-235X.130.257-265.

For citation

Shepel I.O. Modified algorithm for building an occupancy grid from multi-lidar point cloud. *Software & Systems*, 2020, vol. 33, no. 2, pp. 257–265 (in Russ.). DOI: 10.15827/0236-235X.130.257-265.