

Федеральное государственное автономное образовательное учреждение
высшего профессионального образования
«Московский физико-технический институт
(государственный университет)»
Факультет управления и прикладной математики
Кафедра информатики

Диссертация допущена к защите
зав. кафедрой

_____ Петров И.Б.

«_____» _____ 2016 г.

ДИССЕРТАЦИЯ на соискание ученой степени МАГИСТРА

**Тема: Разработка и анализ методов восстановления карты
проходимости на основе показаний датчиков измерения
расстояния**

Направление: 111111 – Название направления

Магистерская программа: 111111 – Название программы

Выполнил студент гр. 1111/1 _____ Шепелев Д.А.

Научный руководитель,

к. ф.-м. н. _____ Николаев Д.П.

Рецензент,

д. ф.-м. н., в. н. с. _____ ФИО рецензента

Оглавление

Введение	4
Глава 1. Методы восстановления карт проходимости, основанные на обратной и прямой моделях.	7
1.1. Восстановление карты проходимости на основе обратной модели сенсора	7
1.1.1. Описание алгоритма	7
1.1.2. Обратная модель сенсора	10
1.1.3. Недостатки метода картирования с обратной моделью . .	10
1.2. Восстановление карты проходимости на основе прямой модели сенсора	12
1.2.1. Прямая модель сонара	12
1.2.2. Картирование с прямой моделью	14
Глава 2. Картирование методом стохастического градиента . . .	16
2.1. Функционал правдоподобия карты проходимости	16
2.2. Алгоритм картирования стохастическим градиента	18
2.3. Работа в режиме реального времени	20
Глава 3. Картирование методом градиентного спуска	21
3.1. Модель сонара	21
3.1.1. Функция правдоподобия	21
3.1.2. Вычисление весовых коэффициентов	24
3.2. Построение карты проходимости методом градиентного спуска .	26
Глава 4. Результаты	29
4.1. Генерация синтетических данных с помощью прямой модели . .	29
4.2. Детали реализации	30

4.2.1.	Диаграмма направленности сонара	30
4.2.2.	Диаграмма чувствительности сонара	31
4.2.3.	Расположение сонаров во время эксперимента	32
4.3.	Численные эксперименты и результаты	33
4.3.1.	Алгоритм картирования методом градиентного спуска . .	33
4.3.2.	Алгоритм картирования методом стохастического гради- ента	35
Заключение		38
Список литературы		40
Приложение А. Восстановленные карты проходимости		42

Введение

Карта проходимости является одним из основных способов описания окружения робота. Чаще всего такая карта задается в виде сетки, состоящей из квадратных ячеек одинакового размера, каждая из которых содержит информацию о проходимости соответствующего участка территории. Например, территория может быть занята или свободна, или каждая клетка может содержать вероятность того, что она содержит препятствие. Подобные карты проходимости, построенные на основе данных с сонаров, уже использовались в 1989 году [1, 2].

В основном карты проходимости используются для задач навигации. Для построения этих карт могут использоваться различные датчики: стереопары (с использованием плотных алгоритмов стереосопоставления) [3, 4], лазерные дальномеры [5], сонары [1, 6, 7]. Лидары обеспечивают высокое качество измерений, однако являются дорогостоящими. Используя стереопару можно восстановить информацию о проходимости, однако её использование предполагает подходящие условия освещения. Сонары значительно дешевле и доступнее лидаров, и могут использоваться при любом освещении. Их недостатком является низкая точность, зашумленность и большой угловой разброс измерений. Также существуют поверхности, которые рассеивают ультразвук или отражают сигналы сонаров только под углами, близкими к прямому. Всё вышеуказанное значительно затрудняет процесс восстановления карты с использованием сонаров, однако существует множество методов, которые различными способами компенсируют эти недостатки.

Ставший на сегодняшний день уже традиционным метод картирования на основе *обратной модели сенсора* [1] представляет карту проходимости в виде сетки, каждая ячейка которой содержит вероятность занятости соответствующей территории. В работе [1] вероятность занятости отдельных клеток оценивается независимо от других, что позволяет быстро обновлять информацию о занятости каждой клетки и строить карту проходимости на лету. Простота ал-

горитма и возможность работы в режиме реального обеспечили ему широкое приложение в робототехнике. Однако предположение о независимости ячеек, которое используется в данном подходе, может приводить к артефактам на результирующей карте проходимости при некоторых сценариях работы.

Альтернативой упомянутому выше методу является нахождение полной карты проходимости, которая максимально правдоподобно объясняет сразу все показания сонаров. Похожие подходы используются, например, в работах [7–9]. Карта проходимости в работе [7] находится путем ЕМ-оптимизации [10], в статье [9] перебираются все возможные локальные конфигурации карты для нахождения максимально правдоподобной. Подобные методы опираются на так называемую *прямую модель* сонара. Проблема большинства алгоритмов, основанных на прямой модели, заключается в невозможности их имплементации для работы в режиме реального времени, а также необходимости больших вычислительных ресурсов для поиска оптимальной конфигурации карты. Требование работы в режиме реального времени к эффективному алгоритму построения карты проходимости является достаточно важным, так как часто такие модули являются неотъемлемой частью систем навигации робота.

Хотя и карты проходимости в виде сетки до сих пор остаются достаточно наглядным и популярным способом представления окружения, но из-за строгой дискретизации пространства неизбежна потеря точности и информации о форме поверхностей препятствий. Подобными недостатками не обладают карты, представляющие препятствия в виде геометрических фигур [11, 12]. Однако подобные подходы имеют два существенных недостатка: они находят только границы между занятыми и незанятыми областями, но не указывают явно проходимость той или иной клетки. В работе [13] предлагается метод, позволяющий объединить достоинства обоих подходов, однако он является вычислительно сложным и вряд ли возможна real-time реализация.

Целью данной работы является разработка алгоритмов картирования с прямой моделью сонара, которые лишены недостатков методов, основанных на

обратных моделях [1, 6], и при этом возможна имплементация разработанных методов для работы в режиме реального времени, в отличие от [7–9, 13]. В результате этой работы созданы два метода, восстанавливающие карту проходимости с помощью сонаров. Результаты картирования сравниваются с традиционным методом, описанным в [1].

Глава 1

Методы восстановления карт проходимости, основанные на обратной и прямой моделях.

В этой главе рассматриваются два подхода построения карт проходимости: на основе обратной модели сенсора [1] и прямой модели [7]. Идея алгоритма из [1] будет подробно изложена в этой главе. Прямая модель, предложенная в работе [7], в дальнейшем будет использована в главе 2, поэтому также будет рассмотрена подробно.

1.1. Восстановление карты проходимости на основе обратной модели сенсора

1.1.1. Описание алгоритма

Введем необходимые в дальнейшем обозначения. Пусть m_i - клетка карты проходимости m . Будем считать, что каждая клетка m_i - бинарная случайная величина, принимающая два значения: {свободная, занятая}. Наблюдением сенсора z_t будем называть измерение и позу датчика в момент времени t , где это измерение было получено. Вместо того, чтобы напрямую решать задачу картирования, будем искать вероятность занятости некоторой карты m при наблюдениях z_1, \dots, z_T . Будем обозначать эту вероятность следующим образом $p(m|z_1, \dots, z_T) \equiv p(m|z_{1,T})$.

Основная проблема в том, что карта проходимости m принадлежит пространству большой размерности. Чтобы обойти эту проблему при оценке $p(m|z_{1,T})$, вводится предположение о том, что клетки карты m_i - случайные, независимые величины. Тогда

$$p(m|z_{1,T}) = \prod_i p(m_i|z_{1,T}) \quad (1.1)$$

Таким образом, достаточно понять, как можно оценить вероятность занятости клетки i при известных наблюдениях $z_{1,T}$. Разложим $p(m_i|z_t)$ по правилу Байеса

$$p(m_i|z_{1,t}) = \frac{p(z_t|m_i, z_{1,t-1})p(m_i|z_{1,t-1})}{p(z_t|z_{1,t-1})} \quad (1.2)$$

В предположении статичности окружения, ясно, что наблюдение z_t не зависит от предыдущих наблюдений, при условии известной карты проходимости m

$$p(z_t|m, z_{1,t-1}) = p(z_t|m) \quad (1.3)$$

Выражение (1.3) действительно верное при предположении о статичности окружения. Однако далее это утверждение усиливается: *наблюдение z_t не зависит от предыдущих измерений при заданном состоянии клетки m_i , в независимости от состояний соседних клеток.*

$$p(z_t|m_i, z_{1,t-1}) = p(z_t|m_i) \quad (1.4)$$

Подставив (1.4) в (1.2), снова воспользуемся правилом Байеса

$$p(m_i|z_{1,t}) = \frac{p(z_t|m_i)p(m_i|z_{1,t-1})}{p(z_t|z_{1,t-1})} = \frac{p(m_i|z_t)p(z_t)p(m_i|z_{1,t-1})}{p(m_i)p(z_t|z_{1,t-1})} \quad (1.5)$$

Формула (1.5) выписана для оценки вероятности занятости клетки m_i . Аналогичное выражение можно получить для оценки вероятности того, что ячейка m_i является свободной

$$p(\overline{m_i}|z_{1,t}) = \frac{p(\overline{m_i}|z_t)p(z_t)p(\overline{m_i}|z_{1,t-1})}{p(\overline{m_i})p(z_t|z_{1,t-1})} \quad (1.6)$$

Поделив (1.5) на (1.6) получим

$$\frac{p(m_i|z_{1,t})}{p(\overline{m}_i|z_{1,t})} = \frac{p(m_i|z_t) p(\overline{m}_i) p(m_i|z_{1,t-1})}{p(\overline{m}_i|z_t) p(m_i) p(\overline{m}_i|z_{1,t-1})} \quad (1.7)$$

Заметим, что $p(\overline{m}_i) = 1 - p(m_i)$. Поэтому, переписав (1.7) в виде log-odds $l(m_i) = \log \frac{p(m_i)}{1-p(m_i)}$, окончательно получаем выражение итеративной оценки для $l(m_i|z_{1,t})$

$$l(m_i|z_{1,t}) = l(m_i|z_t) + l(m_i|z_{1,t-1}) - l(m_i) \quad (1.8)$$

Алгоритм 1: Картирование с обратной моделью сенсора

```

/* Инициализация */
for all  $m_i$  in  $m$  do
    |  $l_i = \log \frac{p(m_i)}{1-p(m_i)}$ 
end

/* Рекурсивное обновление log-odds */
for all  $z_t$  do
    | for all  $m_i$  in  $m$  do
    | |  $l_i += \log \frac{p(m_i|z_t)}{1-p(m_i|z_t)} - \log \frac{p(m_i)}{1-p(m_i)}$ 
    | end
end

/* Получение вероятностей из log-odds */
for all  $m_i$  in  $m$  do
    |  $p(m_i|z_{1,T}) = 1 - e^{-l_i}$ 
end

```

В (1.7) вероятность $p(m_i)$ выражает наши априорные представления о карте, обычно её полагают равной 0.5, считая что какой-либо информации о занятости каждой клетки карты нам неизвестно ничего определенного.

1.1.2. Обратная модель сенсора

Величину $p(m_i|z_t)$ называют *обратной моделью сенсора* (*inverse sensor model*), выражающую вероятность занятости клетки m_i при известном наблюдении z_t . Эта модель называется обратной, так как она обратна процессу измерения – расстояние до объекта в такой модели определяется показанием сонара, хотя в реальности наблюдение сонара определяется расстоянием до препятствия.

Заметим, что в таком простейшем виде обратная модель напрямую не подразумевает зависимость от соседних клеток. Т.е. о состоянии клетки можно сделать выводы основываясь только на наблюдениях, независимо от состояния клеток-соседей. В этом важном для метода допущении и кроется основная проблема при картировании с использованием сонаров, когда гипотеза о независимости клеток недопустима.

1.1.3. Недостатки метода картирования с обратной моделью

Выполнение предположения (1.4) является в некоторых случаях существенным, и его невыполнение влечет к артефактам на карте проходимости. Рассмотрим в качестве примера ситуацию, когда для картирования используется идеальные сонары (без ошибки измерений). Сонар имеет достаточно широкий луч (область видимости, диаграмма направленности), который часто представляется в виде конуса, пересекающий множество клеток (рисунок 1.1). Измерение сонара говорит о следующем - на конце конуса должно находиться препятствие, которое должно хорошо объяснять полученное измерение, при этом вероятность существования других препятствий внутри этого конуса невелико.

На рисунке 1.1 изображены два сонара (A , B), области видимости которых пересекаются в нескольких клетках. Для сонара A эти клетки принадлежат области препятствий, а для B - свободной от препятствий территории. В результате работы алгоритма мы получим, противоречивую информацию о за-

нятости этих клеток: одно измерение говорит о том, что эти ячейки должны быть заняты, другое - что свободны. В этом примере легко понять, что для того чтобы хорошо объяснить наблюдения сонаров A и B , клетки на пересечении конусов видимости должны быть свободны, так как есть другие клетки хорошо объясняющие эти измерения (Рис 1.1 (в)). Однако эта важная дополнительная информация не используется методом, в силу предположения о независимости клеток.

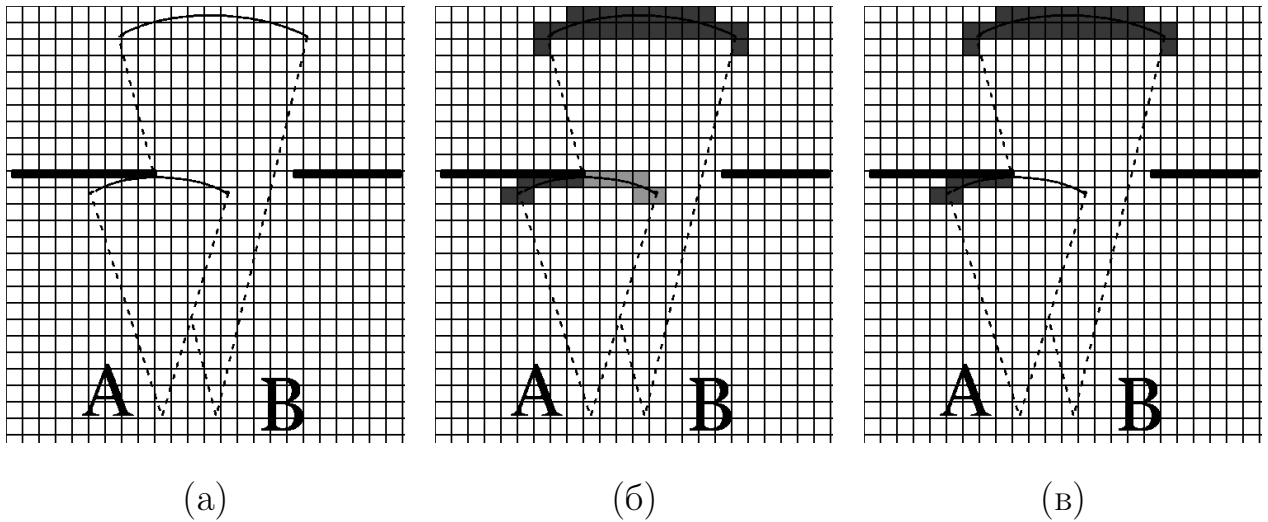


Рис. 1.1. Сонары A и B , области видимости которых пересекаются в нескольких клетках (а). Карта проходимости для этого случая, которая будет построена алгоритмом с обратной моделью сонара (б). Верная карта проходимости для этого случая (в).

Для случая лазерных дальномеров, у которых очень узкая область видимости, эта проблема практически не проявляется.

Таким образом, можно сделать вывод о том, что использование данного метода может приводить к восстановлению такой карты проходимости, которая в некоторых случаях ошибочно объясняет наблюдения сонаров. В работе [7] предложен другой подход к картированию, лишенный описанных в этой части проблем.

1.2. Восстановление карты проходимости на основе прямой модели сенсора

Величина $p(z|m)$ представляет собой вероятностное распределение наблюдений сенсора z при некоторой известной карте проходимости m . По аналогии с обратной моделью $p(m|z)$, будем называть $p(z|m)$ *прямой моделью сенсора* (*forward model*). Прямая модель показывает на сколько правдоподобно наблюдение z объяснено картой проходимости m . Далее приводится подробное описание прямой модели сонара из [7], так как в дальнейшем она будет использована одним из методов, предлагаемых в этой работе.

1.2.1. Прямая модель сонара

Предполагается, что сонар выдает измерения r из отрезка $[R_{min}, R_{max}]$. Измерение r может быть получено в результате двух сценариев:

1. **Случайный выброс.** С вероятностью p_{rand} сонар выдаёт случайное значение дальности, распределенное равномерно на $[R_{min}, R_{max}]$. Этот случай описывает возможные ошибочные измерения сенсора, которые могут получиться в результате переотражений, зашумлений другими сонарами и т.д.
2. **Обычный случай.** С вероятностью p_{hit} некоторое ближайшее препятствие, которое находится в области видимости сонара, может отразить волну, и таким образом сенсор с некоторой ошибкой вернёт расстояние до него. С вероятностью $1 - p_{hit}$ это препятствие может остаться незамеченным датчиком, но волна в конце концов может отразиться от какого-нибудь следующего препятствия. Если же все препятствия внутри области видимости останутся незамеченными сонаром, то в качестве измерения вернётся максимальное R_{max} .

В качестве пояснения рассмотрим пример, изображенный на рисунке 1.1. На этом рисунке видно, что самое близкое препятствие не лежит в конусе видимости сонара, поэтому в обычном случае оно не влияет на $p(z|m)$. С вероятностью p_{rand} сонар выдаст ошибочное измерение. Пусть d_1 и d_2 расстояния до первого и второго препятствия в области видимости сонара соответственно. С вероятностью $(1 - p_{rand})p_{hit}$ сонар обнаружит первое препятствие и вернёт $d_1 + e$, где e - некоторая ошибка. Однако с вероятностью $(1 - p_{rand})(1 - p_{hit})$ первое препятствие не будет замечено сенсором. Аналогично с вероятностью $(1 - p_{rand})(1 - p_{hit})p_{hit}$ будет обнаружено второе препятствие. С вероятностью $(1 - p_{rand})(1 - p_{hit})^2$ сенсор вернет максимально возможное измерение R_{max} .

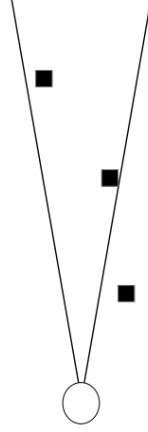


Рис. 1.1. Пример сонара и возможных положений препятствий.

Теперь опишем эту модель формально. Пусть внутри области видимости сонара находятся K препятствий, отсортированных в порядке возрастания дистанции d_k . Через $\{c_*, c_0, c_{1,K}\}$ будем обозначать множество различных сценариев работы сонара, через c_* - случайный выброс, c_0 - измерение R_{max} .

1. Пусть реализовался случай, когда измерение было порождено событием c_k , $k \in \{0, \dots, K\}$

$$p(z|m, c_k) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{1}{2} \frac{(z-d_k)^2}{\sigma^2}} \quad (1.9)$$

2. Если реализовался случай c_* , то

$$p(z|m, c_*) = \frac{1}{R_{max}} \quad (1.10)$$

Таким образом, распределение $p(z|m)$ является смесью распределений

$$p(z|m) = \sum_{c_i \in \{c_*, c_{0,K}\}} p(z|m, c_k) p(c_k) \quad (1.11)$$

Из рассуждений выше запишем априорную вероятность $p(c_k)$

$$p(c_k) = \begin{cases} p_{rand}, & k = * \\ (1 - p_{rand})(1 - p_{hit})^K, & k = 0 \\ (1 - p_{rand})(1 - p_{hit})^{k-1} p_{hit}, & k > 0 \end{cases} \quad (1.12)$$

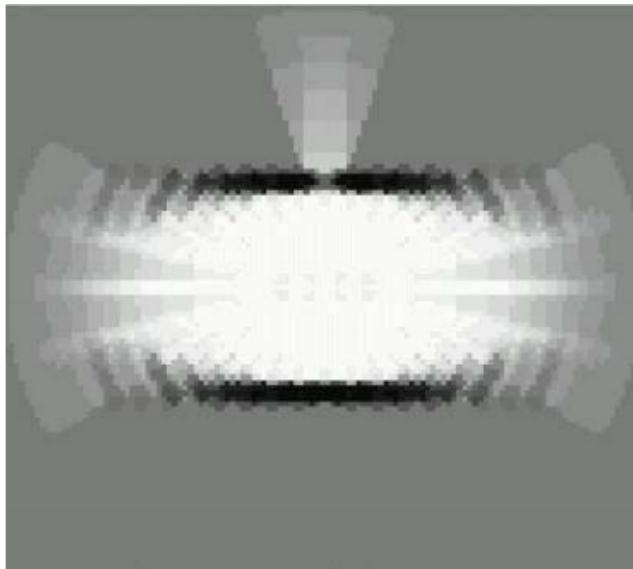
Окончательно получаем

$$\begin{aligned} p(z|m) &= \frac{1}{R_{max}} p_{rand} \\ &+ \sum_{i \in \{1, \dots, K\}} \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{1}{2} \frac{(z-d_k)^2}{\sigma^2}} (1 - p_{rand})(1 - p_{hit})^{k-1} p_{hit} \\ &+ \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{1}{2} \frac{(z-R_{max})^2}{\sigma^2}} (1 - p_{rand})(1 - p_{hit})^K \end{aligned} \quad (1.13)$$

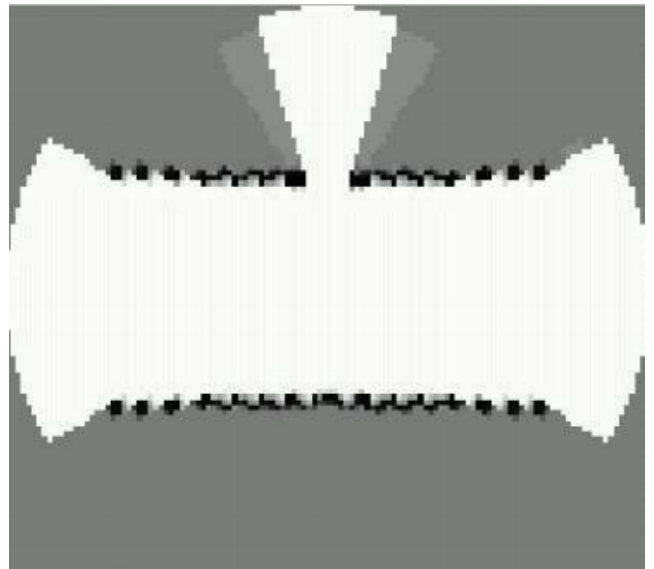
1.2.2. Картирование с прямой моделью

В работе [7] описанная прямая модель сонара используется для построения карты проходимости при помощи ЕМ-алгоритма [10]. В отличие от картирования на основе обратной модели, алгоритмы с прямой моделью сохраняют зависимости между клетками карты, что позволяет лучше восстанавливать карту проходимости. На рисунке 1.2 видно, что метод из работы [7] восстановил дверной проем, когда как традиционный метод [1] - нет.

Как уже отмечалось раньше основной недостаток метода [7] заключается в том, что он не может быть имплементирован для работы в режиме реального времени и требуют больших вычислительных ресурсов для поиска оптимальной



(a)



(б)

Рис. 1.2. Результаты картирования двери из работы [7]. (а) - результаты алгоритма на основе обратной модели, (б) - на основе ЕМ-алгоритма с прямой моделью

карты. Поэтому в этой работе предлагается иной метод на основе рассмотренной здесь прямой модели сонара, который допускает real-time реализацию.

Глава 2

Картирование методом стохастического градиента

Используя прямую модель Труна, мы предлагаем метод картирования, который использует преимущества прямой модели, при этом допускает реализацию, работающую в режиме реального времени. Как и раньше, через m будем обозначать карту проходимости. Через S - множество сонаров s . Через $o(m_i)$ будем обозначать занятость клетки m_i : $o(m_i) = 0$ - клетка проходима, $o(m_i) = 1$ - клетка непроходима.

Вначале введем функционал, состоящий из прямой модели сенсора и априорных представлений об окружении. Затем случайным градиентным спуском будем максимизировать величину этого функционала.

2.1. Функционал правдоподобия карты проходимости

Введем следующий функционал от m при заданных наблюдениях S :

$$\Phi(m, S) = \phi_{sonars}(m, S) + \phi_{occupancy}(m) + \phi_{borders}(m), \quad (2.1)$$

Рассмотрим составляющие функционала (2.1)

1. ϕ_{sonars} - распределение наблюдений сонаров z_s при заданной карте проходимости m .

$$\phi_{sonars}(m, S) = p(z_1, \dots, z_S | m) = \prod_{s \in S} p(z_s | m) \quad (2.2)$$

Эта часть функционала (2.1) показывает на сколько хорошо карта m объясняет показание сонаров $s \in S$. В работе в качестве модели сонара

$p(z_s|m)$ используется прямая модель Труна. Вместо (2.2) в окончательной формуле используется логарифм правдоподобия:

$$\phi_{sonars}(m, S) = \log p(z_1, \dots, z_S|m) = \sum_{s \in S} \log p(z_s|m) \quad (2.3)$$

2. $\phi_{occupancy}(m)$ отвечает за априорные знания о проходимости карты

$$\phi_{occupancy}(m) = \sum_{m_i} w_o o(m_i) \quad (2.4)$$

В зависимости от значения весового коэффициента w_o можно регулировать наше первоначальное представление о карте, без учета наблюдений сонаров. Например, при $w_o < 0$ и $\phi_{sonars}(m, S) = const$ пустая карта будет максимизировать $\Phi(m, S)$.

3. $\phi_{borders}(m)$ является суммой штрафов для каждой клетки, пропорциональный квадрату числа границ между проходимой и непроходимой областью этой ячейки

$$\phi_{borders}(m) = \sum_{m_i} w_b n^2(m_i) \quad (2.5)$$

Функционал (2.5), как и (2.4), отвечает за наши априорные знания о окружении и имеет простую интерпретацию. Естественнo считать что, если большинство соседей заняты, то и рассматриваемая клетка, скорее всего, непроходима. Аналогичную гипотезу можно сформулировать и для незанятых клеток. Поэтому, считая $w_b < 0$, за каждого соседа, который не согласуется с проходимостью, мы штрафует. Абсолютная величина коэффициента w_b позволяет регулировать относительный вклад в функционал (2.1).

Таким образом задача картирования сводится к задаче максимизации (2.1) по всем возможным картам проходимости

$$m^*(S) = \underset{m}{\operatorname{argmax}} \left(\phi_{sonars}(m, S) + \phi_{occupancy}(m) + \phi_{borders}(m) \right) \quad (2.6)$$

2.2. Алгоритм картирования стохастическим градиента

Оптимизационная задача (2.6) решается методом стохастического градиента. Каждый оптимизационный шаг состоит из следующих действий:

1. Случайным образом выбирается клетка m_i и значение проходимости $o(m_i)$ инвертируется

$$o^*(m_i) = 1 - o(m_i)$$

2. Для нового значения проходимости клетки m_i пересчитываются $\phi_{sonars}(m, S)$, $\phi_{occupancy}(m)$ и $\phi_{borders}(m)$.

В слагаемом $\phi_{sonars}(m, S) = \prod_{s \in S} p(z_s | m)$ меняются только те члены произведения, для которых инвертированная клетка лежит в области видимости сенсора. Поэтому можно достаточно быстро пересчитать новое значение $\phi_{sonars}^*(m, S)$.

В силу того, что слагаемые $\phi_{occupancy}(m)$ и $\phi_{borders}(m)$ являются суммами слагаемых, величина которых зависит только значения проходимости самой клетки и её ближайших соседей, поэтому ясно, что при инвертировании одной клетки можно быстро и понятным способом пересчитать новые значения $\phi_{occupancy}^*(m)$ и $\phi_{borders}^*(m)$.

3. Если $\Phi_{new}(m, S) = \phi_{sonars}^*(m, S) + \phi_{occupancy}^*(m) + \phi_{borders}^*(m) > \Phi(m, S)$, то сохраняем новое значение $o^*(m_i)$ сохраняя новое состояние, иначе возвращаемся в предыдущее состояние.

Для того чтобы избежать застревания в локальных минимумах, добавляется рандомизация сохранения нового состояния: инвертирование сохраняется с вероятностью p_{rand} , вне зависимости от величины $\Phi_{new}(m, S)$.

Алгоритм 2: Оффлайн версия картирования методом стохастического градиента

Data: m^0 - начальное состояние карты, S - множество наблюдений

сонаров, k - число оптимизационных шагов

Result: m - карта проходимости

инициализация;

begin

$m = m^0;$

$\phi_{sonars}(m, S), \phi_{occupancy}(m), \phi_{borders}(m);$

$\Phi(m, S) = \phi_{sonars}(m, S) + \phi_{occupancy}(m) + \phi_{borders}(m);$

end

for $i = 0; i < k; i = i + 1$ **do**

случайным образом выбирается клетка карты m_i ;

$o^*(m_i) = 1 - o(m_i);$

пересчитать $\phi_{sonars}^*(m, S), \phi_{occupancy}^*(m), \phi_{borders}^*(m);$

$\Phi^*(m, S) = \phi_{sonars}^*(m, S) + \phi_{occupancy}^*(m) + \phi_{borders}^*(m) > \Phi(m, S);$

if $\Phi^*(m, S) > \Phi(m, S)$ *или* $rand(0, 1) < p_{randomfactor}$ **then**

$m = m^* \quad \phi_{sonars}(m, S) = \phi_{sonars}^*(m, S);$

$\phi_{occupancy}(m) = \phi_{occupancy}^*(m);$

$\phi_{borders}(m) = \phi_{borders}^*(m);$

$\Phi(m, S) = \Phi^*(m, S);$

end

end

В этой работе имплементирована оффлайн версия алгоритма 2, которая на вход получает сразу все наблюдения сонаров, и затем оптимизирует карту проходимости.

2.3. Работа в режиме реального времени

Как уже говорилось ранее можно реализовать алгоритм, который работает в режиме реального времени. Далее предлагаются способы того, каким образом это можно сделать.

В режиме реального времени оптимизация ведется одновременно с получением новых данных. Ясно, что если при долгом и непрерывном сборе данных, в определенный момент времени количество наблюдений превысит тот их объем, который возможно обрабатывать в режиме реального времени. Поэтому необходимо выбирать лишь ту часть данных, которые будут алгоритмом в процессе оптимизации. Здесь предлагаются следующие способы выбора наблюдений для использования в оптимизации:

- Использовать скользящее окно и рассматривать последние N_{max} измерений - таким образом мы гарантируем, что каждый пересчет $\phi_{sonars}(m, S)$ не превысит $T_{max} = \Delta_t N_{max}$, где Δ_t - время пересчета одного сонара.
- Для каждой клетки хранить номера измерений сонаров, которые содержат её в поле зрения. Ограничение числа привязанных к каждой ячейке измерений значением N_{max} гарантирует, что для каждой ячейки пересчет $\phi_{sonars}(m, S)$ будет занимать времени не более $T_{max} = \Delta_t N_{max}$. Предлагается выбирать N_{max} последних наблюдений.
- Будем использовать трехмерную сетку пространства (x, y, ϕ) , каждый узел этой сетки хранит список наблюдений, которые принадлежат области пространства $(\Delta x, \Delta y, \Delta \phi)$, соответствующей этому узлу. Тогда в каждом узле можно хранить N_{max} последних измерений или даже эти измерения некоторым образом фильтровать. При этом масштаб этой сетки, может не совпадать с масштабом карты проходимости.

Глава 3

Картирование методом градиентного спуска

В этой главе предлагается другой метод построения карты проходимости. В отличие от других методов картирования с прямой моделью, в которых задача сводится к задаче оптимизации в конечномерных пространствах большой размерности, предлагаемый метод решает эту задачу в бесконечномерном пространстве. Для этого нужно переформулировать задачу картирования.

Представим клетки карты проходимости m как переменные x_i , которые принимают значения от 0 до 1 - это значение характеризует степень занятости ячейки. Как и раньше будем искать такую карту m^* , которая наилучшим образом объясняет наблюдения сонаров, и для этого введем функцию, которая описывает, насколько хорошо карта m объясняет измерения сонаров. В этом методе используется прямая модель сонара, отличная от модели Труна. Далее следует описание используемой модели.

3.1. Модель сонара

3.1.1. Функция правдоподобия

Измерение сонара r несет в себе не только информацию о том, что на расстоянии r скорее всего находится препятствие, но и о том, что на расстоянии меньше r препятствия маловероятны. Поэтому каждое наблюдение сонара r можно интерпретировать следующим образом:

- В некоторой области, расположенной на расстоянии примерно r от сонара, располагается препятствие такого размера, чтобы сонар получил от него отражение на таком расстоянии.
- В области перед сонаром на расстоянии меньше r , препятствия определен-

ных размеров, которые могли стать причиной показания сонара с меньшим r , отсутствуют.

Через $\Omega_{free}(m, z)$ будем обозначать множество переменных x_i , которые соответствуют клеткам карты, находящимся внутри области видимости сонара, где не должно быть препятствий в соответствии с наблюдением $z = (x, y, \phi, r)$. Множество переменных x_i , которые соответствуют ячейкам карты, располагающимся в луче сонара там, где должно находиться препятствие, будем обозначать как $\Omega_{occ}(m, z)$.

В работе предполагается, что показание сонара определяется только суммарной (как будет показано ниже, взвешенной) площадью препятствий внутри областей Ω_{free} и Ω_{occ} , и не зависит от взаимного расположения занятых клеток в них. Ясно, что на самом деле, на измерение сонара сильно влияет направление нормали к поверхности препятствия. Действительно, легче всего обнаружить то препятствие, поверхность которого перпендикулярна лучу сонара. Кроме того, сонары легче детектируют одно препятствие, расположенное в его поле зрения, чем несколько препятствий аналогичной формы и суммарной площади, которые рассредоточены в области луча. Данные эффекты в работе не рассматриваются.

Далее рассмотрим функцию правдоподобия $\psi(m, z)$, которая показывает, насколько хорошо карта m объясняет показание сонара z . $\psi(m, z)$ состоит из двух слагаемых: $\psi_{free}(m, z)$ отвечает за отсутствие препятствий в области $\Omega_{free}(m, z)$, за наличие препятствий в области $\Omega_{occ}(m, z)$ — $\psi_{occ}(m, z)$. Далее для наблюдения сонара z_s и карты m будем коротко записывать

$$\psi^s = \psi_{free}^s + \psi_{occ}^s \quad (3.1)$$

Пусть $Z = \{z_1, \dots, z_n\}$ - множество всех наблюдений сонаров. Тогда $\Psi(m, Z)$ - функция правдоподобия всех измерений сонаров. В этом алгоритме считается, что $\Psi(m, Z)$ является аддитивной относительно каждого наблюдения z_s , все

показания датчиков имеют одинаковую достоверность и входят в $\Psi(m, Z)$ с равным весом. Поэтому значение функции правдоподобия ψ^s должно принимать одинаковые максимальные и минимальные значения для различных сонаров. Постулируется, что для всех $s \in \{1, \dots, n\}$ значения ψ_{free}^s и ψ_{occ}^s принимают значения от 0 до 1. Таким образом

$$\Psi(m, Z) = \sum_{s \in Z} \psi_{free}^s + \psi_{occ}^s \quad (3.2)$$

Теперь рассмотрим, каким образом должно формироваться значение ψ_{free}^s для области Ω_{free}^s . Ясно, что занятые клетки внутри Ω_{free}^s противоречат наблюдению сонара. Однако, если чувствительность датчика низкая, то одна занятая ячейка не является достаточным основанием полагать, что показание сенсора не объяснено картой m . Таким образом внутри Ω_{free}^s может быть некоторое количество занятых клеток, которое зависит от чувствительности сонара. При этом не все ячейки в Ω_{free}^s являются равнозначными: состояние тех клеток, которые лежат ближе к сонару, должно вносить больший вклад в ψ_{free}^s , так как отражение импульса от них более вероятно, чем от более удаленных занятых клеток. Поэтому, чтобы получить оценку эффективного количества препятствий в Ω_{free}^s , рассмотрим взвешенную сумму

$$X = \sum_{x_i \in \Omega_{free}^s} x_i \omega_i \quad (3.3)$$

где ω_i - вес, определяющий, насколько важной является ячейка x_i . Если взвешенная сумма X меньше некоторого порогового значения X_{free}^s , то наблюдение z_s считается хорошо объясненным. Если же сумма X превышает X_{free}^s , то начисляется штраф, увеличивающийся с возрастанием X . Будем считать, что ψ_{free}^s является кусочно линейной функцией вида

$$\psi_{free}^s = \begin{cases} \alpha_{free} X, & X < X_{free}^s \\ \alpha_{free} X_{free}^s + \beta_{free} (X - X_{free}^s), & X \geq X_{free}^s \end{cases} \quad (3.4)$$

Учитывая ограничение $\psi_{free}^s(1) = 1$, получаем

$$\beta_{free} = \frac{1 - \alpha_{free} X_{free}^s}{1 - X_{free}^s} \quad (3.5)$$

Параметр $\alpha_{free} > 0$ должен быть достаточно малым, но отличным от нуля. Это нужно для того, чтобы был ненулевой штраф при значениях $X \leq X_{free}^s$, когда измерение сонара хорошо объяснено в области Ω_{free}^s , и, таким образом, метод градиентного спуска мог бы сойтись.

Аналогичным образом, будем считать, что, при ограничении $\psi_{occ}^s(1) = 0$ и малом $\alpha_{occ} > 0$, ψ_{occ}^s является кусочно линейной функцией вида

$$\psi_{occ}^s = \begin{cases} 1 - \beta_{occ} X, & X < X_{occ}^s \\ 1 - \beta_{occ} X_{occ}^s - \alpha_{occ} (X - X_{occ}^s), & X \geq X_{occ}^s \end{cases} \quad (3.6)$$

$$\beta_{occ} = \frac{1 - \alpha_{occ} + \alpha_{occ} X_{occ}^s}{X_{occ}^s} \quad (3.7)$$

3.1.2. Вычисление весовых коэффициентов

Далее через $X_{threshold}$ будем обозначать X_{occ} и X_{free} . Для расчета порогов $X_{threshold}$ и весов ω_i необходимо знать чувствительность сонара. Обычно эта информация предоставляется в спецификации производителя, которая получается следующим образом: используется набор столбов различного диаметра, каждый из столбов помещается в разные точки диаграммы направленности сенсора. Таким образом для каждого из столбов определяется геометрическое место точек, в которых этот столб детектируется сонаром. Пример такой спецификации, использованный в данной работе, приведен в [14]. Можно обобщить это определение чувствительности сонара. *Диаграммой чувствительности сонара* назовем функцию $d(x, y)$, определённую следующим образом: для каждой точки (x, y) в системе координат сонара, $d(x, y)$ задает минимальный диаметр препятствия d , которое обнаруживается датчиком в этой точке. Для реального сонара диаграмма чувствительности $d(x, y)$ может быть построена эмпири-

чески, либо приближенно на основе спецификации. В этой работе в качестве диаграммы чувствительности используется $s(x, y)$ - минимальная площадь препятствия, обнаруживаемая сонаром в точке (x, y) , построенная на основе [14].

Теперь опишем, каким образом находятся веса ω_i , при известной диаграмме чувствительности $d(x, y)$. Пусть площадь одной клетки равна s . Тогда для расчёта весов ω_i и порога $X_{threshold}$ можно использовать следующий алгоритм:

1. Сначала происходит расчет предварительных весов: для каждой ячейки x_i выбирается вес

$$\omega_i^0 = \min\left(\frac{s}{s(x, y)}, 1\right) \quad (3.8)$$

То есть вес равен отношению площади клетки к минимальной обнаруживаемой сонаром площади в этой точке, но если площадь ячейки больше, чем эта минимальная площадь, то вес берётся равным 1. Тогда можно сказать, что сонар обнаружит препятствие, если $X = \sum_{x_i \in \Omega} x_i \omega_i$ будет не меньше 1.

2. Чтобы значения функций ψ_{free} и ψ_{occ} были заключены между 0 и 1, необходимо чтобы X принимала значения от 0 до 1. Для этого мы проводим нормировку

$$\omega_i = \frac{\omega_i^0}{\sum \omega_i^0} \quad (3.9)$$

Тогда порог $X_{threshold}$ берётся равным

$$X_{threshold} = \frac{1}{\sum \omega_i^0} \quad (3.10)$$

Теперь у нас есть все составляющие для того чтобы построить карту проходимости.

Алгоритм 3: Алгоритм вычисления весов

Data: z_s -наблюдение, $s(x, y)$ и s - диаграмма чувствительности и

площадь одной клетки карты

Result: $(\vec{\omega}_{free}^s, \vec{\omega}_{occ}^s, X_{free}^s, X_{occ}^s)$, где $\vec{\omega}_{free}^s, \vec{\omega}_{occ}^s$ - вектора весов

переменных x_i в соответствии с наблюдением z_s , X_{free}^s, X_{occ}^s -

пороговые значения

for $al\ k \in \{free, occ\}$ **do**

$\vec{\omega}_{free}^s = \vec{0};$

$\vec{\omega}_{occ}^s = \vec{0};$

for $all\ x_i\ in\ \Omega_k^s$ **do**

$\vec{\omega}_k^s[x_i] = \min\left(\frac{s}{s(x,y)}, 1\right);$

end

$S = \sum_{x_i} \vec{\omega}_k^s[x_i];$

$X_k = \frac{1}{S};$

for $all\ x_i\ in\ \Omega_k^s$ **do**

$\vec{\omega}_k^s[x_i] = \frac{\omega_i^0}{S};$

end

end

3.2. Построение карты проходимости методом градиентного спуска

Для восстановления карты проходимости минимизируется $\Psi(m, Z)$, используя метод градиентного спуска. Обратим внимание, что все наши рассуждения о модели сонаров касались карт проходимости, в которых переменные x_i принимают только значения 0 или 1. Оптимизация без ограничений может привести к

значительным искажениям карты проходимости, когда x_i принимают значения больше 1 или меньше 0. Поэтому, вводятся дополнительные члены регуляризации в $\Psi(m, Z)$, ограничивающие значения переменных x_i между 0 и 1. Похожий способ борьбы с артефактами может быть найден, например, в [15].

Чтобы заключить значения проходимости клеток карты между 0 и 1 вводится следующая регуляризация ($c_1 > 0$)

$$R_1^i(x_i) = \begin{cases} c_1(x_i - 1), & x_i > 1 \\ 0, & x_i \in [0, 1] \\ -c_1x_i, & x_i < 0 \end{cases} \quad (3.11)$$

Ещё одно регуляризационное слагаемое, которое штрафует за близость значения переменной к 0.5, вводится для того, чтобы x_i принимали значения ближе к 0 или 1 ($c_2 > 0$)

$$R_2^i(x_i) = \begin{cases} c_2x_i, & x_i \in [0, 0.5] \\ c_2(1 - x_i), & x_i \in [0.5, 1] \\ 0, & \text{иначе} \end{cases} \quad (3.12)$$

Таким образом, получаем регуляризацию

$$R^i(x_i) = R_1^i(x_i) + R_2^i(x_i) = \begin{cases} c_1(x_i - 1), & x_i > 1 \\ c_2x_i, & x_i \in [0, 0.5] \\ c_2(1 - x_i), & x_i \in [0.5, 1] \\ -c_1x_i, & x_i < 0 \end{cases} \quad (3.13)$$

Доопределим производные для всех z_s и x_i

$$\begin{aligned} \frac{\partial}{\partial x_i} \psi_{free}^s(X_{free}^s) &= \frac{\partial}{\partial x_i} \psi_{occ}^s(X_{occ}^s) = 0 \\ \frac{\partial}{\partial x_i} R^i(0) &= \frac{\partial}{\partial x_i} R^i(1) = \frac{\partial}{\partial x_i} R^i(0.5) = 0 \end{aligned} \quad (3.14)$$

Предположим что начальное приближение m^0 такое, что все $x_i = 0.5$, тогда из-за $\frac{\partial}{\partial x_i} R^i(0.5) = 0$, соответствующие x_i тех клеток, которые не попали внутрь диаграммы направленности ни одного сонара, останутся равны 0.5.

Таким образом, задача построения оптимальной карты переходит в следующую задачу минимизации

$$m^*(Z) = \operatorname{argmin}_m \left(\sum_{x_i} R^i(x_i) + \sum_{z_s \in Z} (\psi_{free}^s + \psi_{occ}^s) \right) \quad (3.15)$$

В работе предложенный алгоритм использовался только для обработки уже собранных данных. Скорость работы алгоритма позволяет использовать его в режиме реального времени, однако для работы в таком режиме (и для работы в течение неограниченного времени) необходимо осуществлять фильтрацию измерений, например, используя методы из [2.3](#).

Глава 4

Результаты

4.1. Генерация синтетических данных с помощью прямой модели

В процессе разработки алгоритма картирования необходим некоторый массив данных, который позволит эффективно исследовать свойства алгоритма и сравнивать его с другими методами в одних и тех же сценариях. Однако сбор таких данных с реального робота представляет большие трудности:

- Необходимо знать положение робота в пространстве в каждый момент времени, и следовательно определение положения робота каждый раз при измерении, требует либо специального полигона, либо наличие инструментов, которые позволят быстро и точно это сделать. Ясно, что это не всегда возможно.
- Свойства сонара сильно отличаются в зависимости от окружения, поэтому для полноты исследования того или иного метода картирования требуется возможность проводить эксперименты в различных условиях. На реальном роботе это далеко не всегда возможно.

В этой работе предлагается простой метод, который позволяет быстро получать экспериментальные синтетические данные. Ядром этого метода является прямая модель сонара (1.13).

На вход алгоритму подаётся некоторая карта проходимости, например в формате картинки, с заданным масштабом. Затем можно задать траекторию движения робота и параметры прямой модели сонара, которые позволяют простым образом моделировать различные сценарии работы сонара. Каждая клетка на входной картинке является препятствием определенного размера. Задавая

положение робота на этой карте, мы имитируем измерение сонара следующим образом:

1. С вероятностью p_{rand} сонар выдает случайное значение дальности, распределенное равномерно на $[R_{min}, R_{max}]$.
2. Находим в области видимости ближайшее не рассмотренное препятствие. С вероятностью p_{hit} препятствие считается замеченным сонаром, дистанция от сонара до этой клетки с нормальным шумом выдается в качестве измерения.
3. С вероятностью $1 - p_{hit}$ препятствие считается незамеченным сонаром, оно помечается как рассмотренное и алгоритм возвращается к предыдущему шагу.
4. Если все препятствия остались незамеченными, то выдается максимальное измерение R_{max} .

Таким образом, наш метод решает указанные выше проблемы и позволяет быстро создать необходимый массив наблюдений сонаров.

4.2. Детали реализации

4.2.1. Диаграмма направленности сонара

Реальную диаграмму направленности можно было увидеть в [14]. При реализации области видимости сонара использовалось предположение о том, что она является выпуклой областью. В качестве приближения реальной диаграммы направленности используется выпуклый многоугольник, заданный множеством координат своих вершин $V = \{(x_v, y_v)\}$, упорядоченных таким образом, что контур, который они образуют, является правильно ориентированной кривой. Тогда алгоритм определения того, лежит ли точка (x, y) внутри луча сенсора выглядит следующим образом

1. пусть нужно проверить лежит ли точка P внутри области видимости сонара
2. для каждой вершины из V , кроме первой из этого множества, обозначив за $a = V_i - V_0$ и $b = P - V_i$, по порядку проверяем выполнение следующего условия

$$a_x b_y - b_x y_a \geq 0$$

Если условие выше выполняется, продолжаем проверку. Не выполнение этого условия означает, что точка P не принадлежит области видимости датчика.

3. На последнем шаге проверяем это же условие, считая что $a = V_0 - V_{last}$ и $b = P - V_{last}$

Приближение формы луча использованной в работе приведена на рисунке 4.1

4.2.2. Диаграмма чувствительности сонара

Диаграмма чувствительности, которая используется в алгоритме, описанном в Главе 3, была получена из [14]. Вследствии дискретности карты прохо-

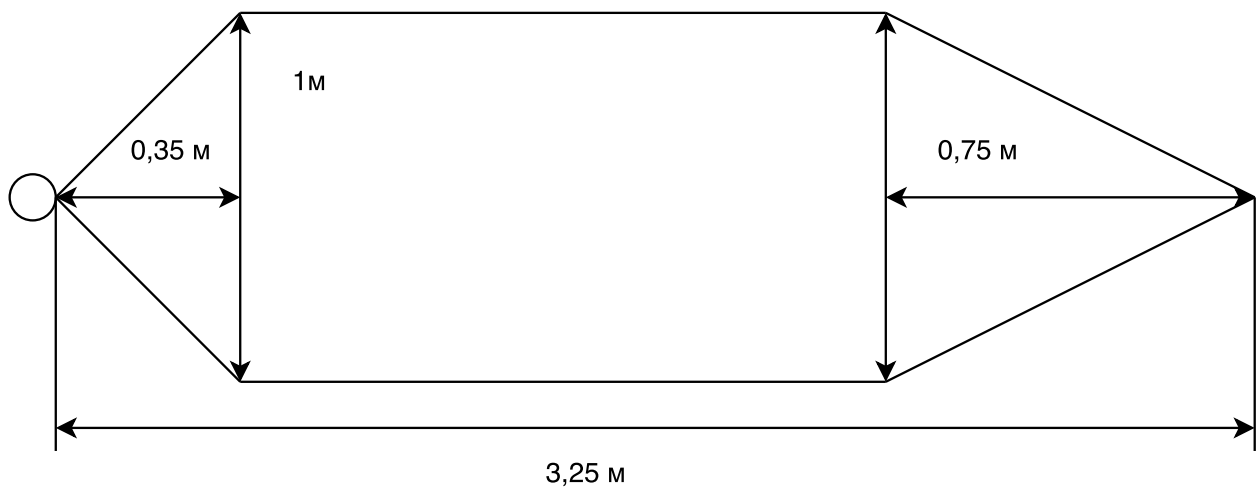


Рис. 4.1. Приближение диаграммы направленности сонаров, использованных в экспериментах.

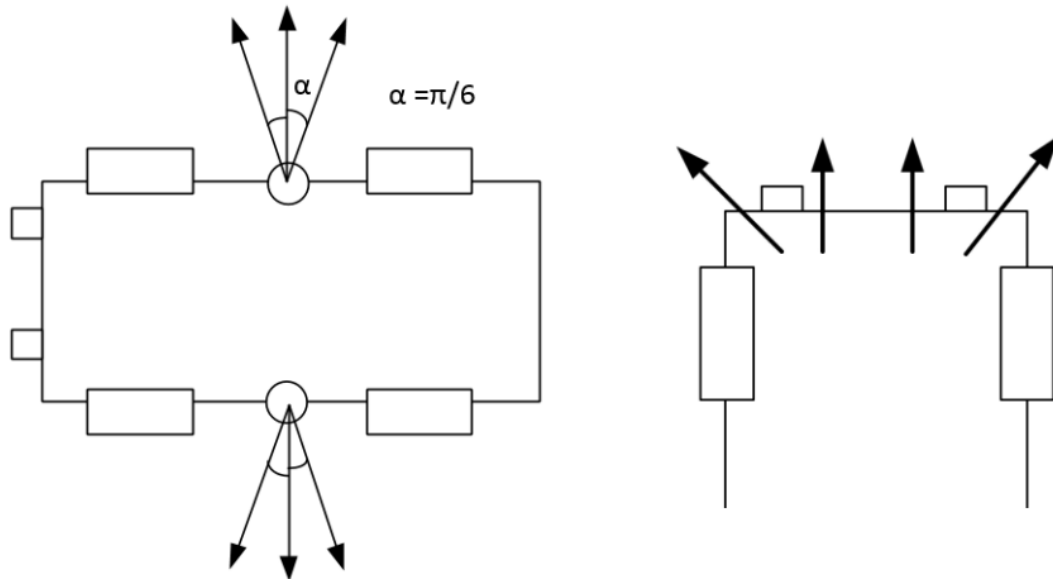


Рис. 4.2. Расположение сонаров на роботе для эксперимента с использованием синтетических (слева) и реальных (справа) данных.

мости, можно дискретизировать функцию $s(x, y)$.

Рассмотрим пару (Ω, s) , где Ω - некоторой области, а s - минимальная площадь препятствия, которое обнаруживается сонаром в этой области. Таким образом, $s(x, y)$ можно представить как множество пар $\tilde{s} = \{(\Omega_i, s_i)\}$. Будем считать, что Ω_i может быть представлена таким же образом, как и область видимости. В зависимости от масштаба карты, можно строить различные приближения \tilde{s} для диаграммы чувствительности. На рисунке 4.8 можно увидеть \tilde{s} , которое используется в работе.

4.2.3. Расположение сонаров во время эксперимента

На рисунке 4.2 показано расположение сонаров для экспериментов с использованием синтетических (слева) и реальных (справа) данных. Для экспериментов на синтетических данных с каждой стороны робота (слева и справа) располагаются по три сонара, при этом два датчика развернуты от центрального под углом в 30 градусов. На реальном роботе в передней части расположены четыре сонаара, два крайних сонаара развернуты от центральных под углом 45 градусов.

4.3. Численные эксперименты и результаты

В этой части главы описаны проведенные численные эксперименты и результаты разработанных в данной работе алгоритмов. Полученные карты проходимости сравниваются с результатами картирования методом с обратной моделью сенсора.

4.3.1. Алгоритм картирования методом градиентного спуска

Диаграмма чувствительности сонара, использованная в численных экспериментах, приведена на рисунке 4.8. Области Ω_{free} и Ω_{occ} определяются следующим образом: область Ω_{occ} содержит все точки области видимости сонара, расстояние до которых лежит в диапазоне $[r - \Delta r, r + \Delta r]$. Область Ω_{free} содержит все точки луча сонара, расстояние до которых меньше $r - \Delta r$.

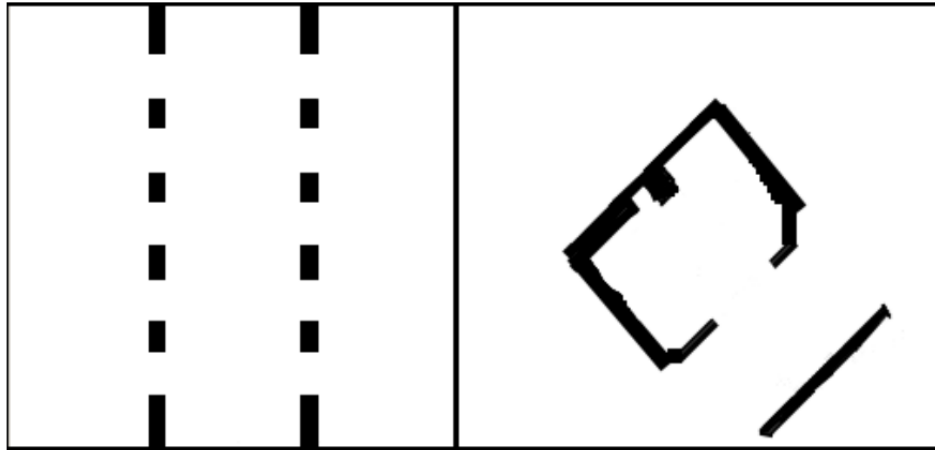


Рис. 4.3. Территории, на которых проводилось картирование (слева – искусственная территория, справа – реальная территория).

Для численных экспериментов выбраны два набора данных. Первый набор данных синтезирован на основе искусственной карты, которые были получены в результате процедуры описанной в 4.1, представленной на слева на рисунке 4.3, и все показания сонаров являются точными. В этом эксперименте робот движется по прямой снизу вверх (по центру карты). Результаты восстановления карты проходимости на синтетических данных представлены на рисунке 4.4. Как вид-

но из рисунка, картирование предложенным методом значительно лучше восстанавливает дверные проемы чем традиционный метод. Для экспериментов использовались следующие параметры: $\alpha_{free} = 0.05$, $\alpha_{occ} = 0.01$, $c_1 = 5$, $c_2 = 1$.

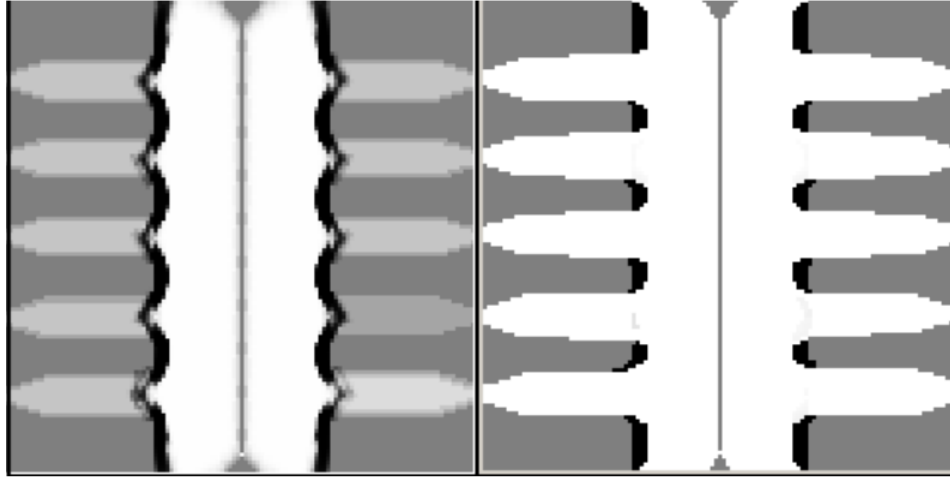


Рис. 4.4. Результаты восстановления карты проходимости на основе синтетических данных с помощью алгоритма картирования с обратной моделью (слева) и методом градиентного спуска (справа). Масштаб карты 5 сантиметров на одну клетку карты. Размер картируемой территории 6 метров на 6 метров.

Второй набор данных собран с использованием настоящего робота, сонары на котором расположены как показано на рис. 4.2 (справа). При этом робот осуществляет полное вращение на месте в комнате, карта которой представлена справа на рисунке 4.3. Показания распределены неравномерно по углу (поскольку вращение осуществлялось с непостоянной скоростью), некоторые показания сонаров являются выбросовыми, а также позиции сонаров измерены не совсем точно. Этот эксперимент позволяет определить, какой эффективностью обладает предложенный алгоритм на реальных данных в сравнении с методом картирования с обратной моделью. Результаты восстановления представлены на рисунке 4.5. Вследствие неравномерности снятых измерений, а также особенностей используемых методов, оба метода не смогли восстановить точную карту проходимости, однако метод на основе градиента сумел успешно восстановить дверное отверстие, в то время как традиционный метод не справился с этой задачей.

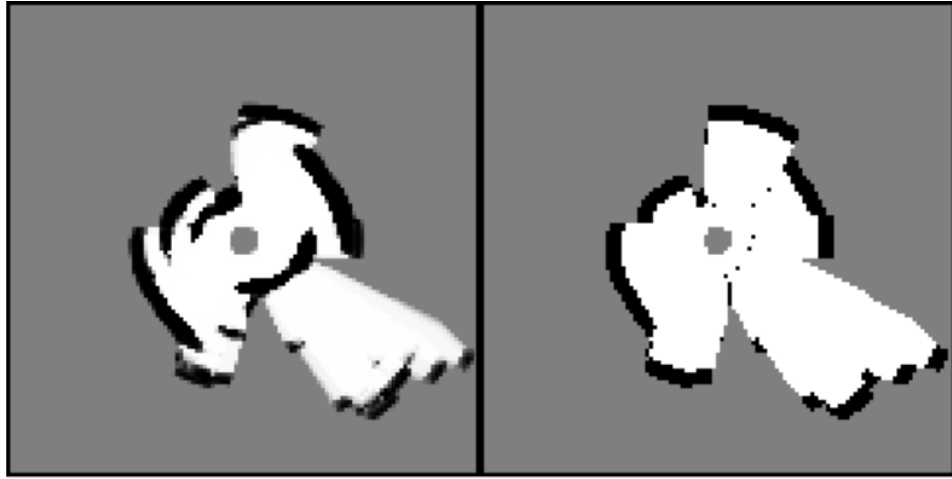


Рис. 4.5. Результаты восстановления карты проходимости на основе реальных данных с помощью алгоритма картирования с обратной моделью (слева) и методом градиентного спуска (справа). Масштаб карты 5 сантиметров на одну клетку карты. Размер картируемой территории 6 метров на 6 метров.

4.3.2. Алгоритм картирования методом стохастического градиента

Эксперименты проводились на том же наборе данных, что и в 4.3.1. Результаты представлены на рисунках 4.6 и 4.7. Так как метод картирования стохастическим градиентом не может предоставлять промежуточную информацию о проходимости клетки, в отличие от метода картирования с обратной моделью, результирующие данные представлены в виде бинарных рисунков, где черный пиксель соответствует занятой клетке, белый - свободной клетке. Результаты картирования традиционным методом преобразованы в такой вид следующим образом: $o(m_i) \geq 0.5$ - клетка считалась занятой, $o(m_i) < 0.5$ - ячейка свободна. Для экспериментов использовались следующие параметры: $p_{rand} = 0.05$, $p_{hit} = 0.75$, $w_o = 0.05$, $w_b = 0.005$, $p_{randomfactor} = 0$.

Восстановленная карта проходимости на основе синтетических данных представлена на рисунке 4.6. Видно что, картирование методом стохастического градиента хорошо справляется с задачей восстановления дверных проемов, и восстанавливает карту проходимости точнее традиционного метода.

Результаты восстановления на реальных данных представлены на рисунке 4.7. Предложенный метод не смог восстановить точную карту проходимости,

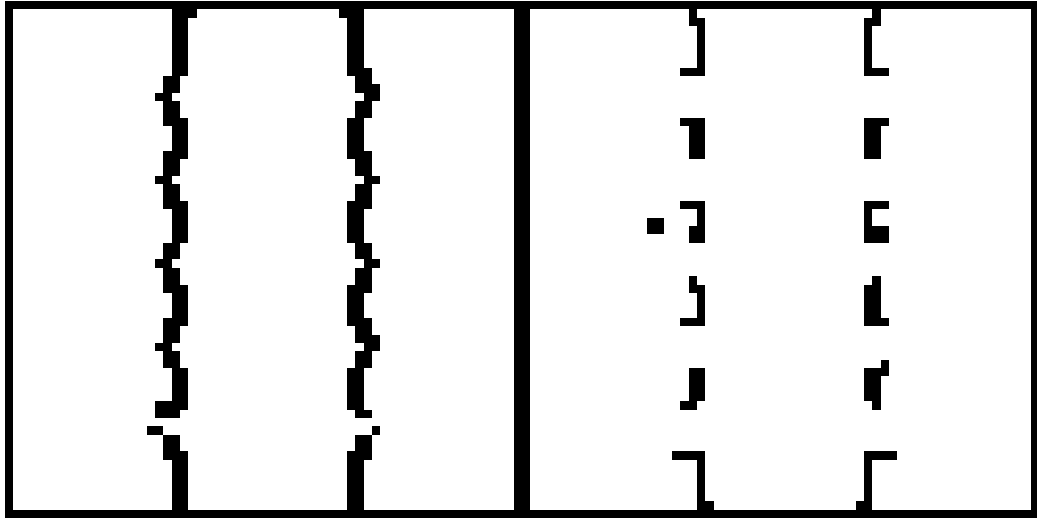


Рис. 4.6. Результаты восстановления карты проходимости на основе синтетических данных с помощью алгоритма картирования с обратной моделью (слева) и методом стохастического градиента (справа). Масштаб карты 10 сантиметров на одну клетку карты. Размер картируемой территории 6 метров на 6 метров.

однако частично восстановил дверное отверстие, в отличии от метода картирования с обратной моделью, который совершенно провалил эту задачу.

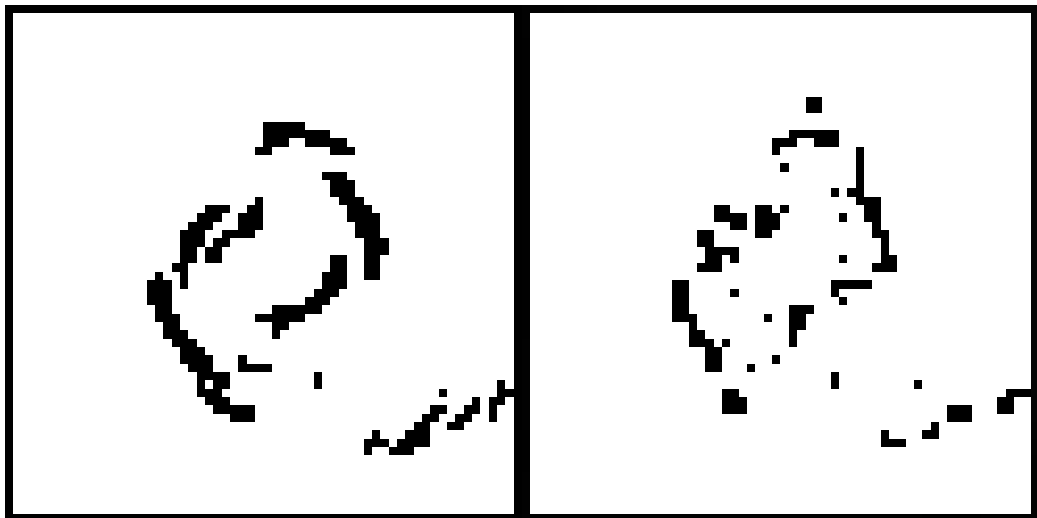


Рис. 4.7. Результаты восстановления карты проходимости на основе реальных данных с помощью алгоритма картирования с обратной моделью (слева) и методом стохастического градиента (справа). Масштаб карты 10 сантиметров на одну клетку карты. Размер картируемой территории 6 метров на 6 метров.

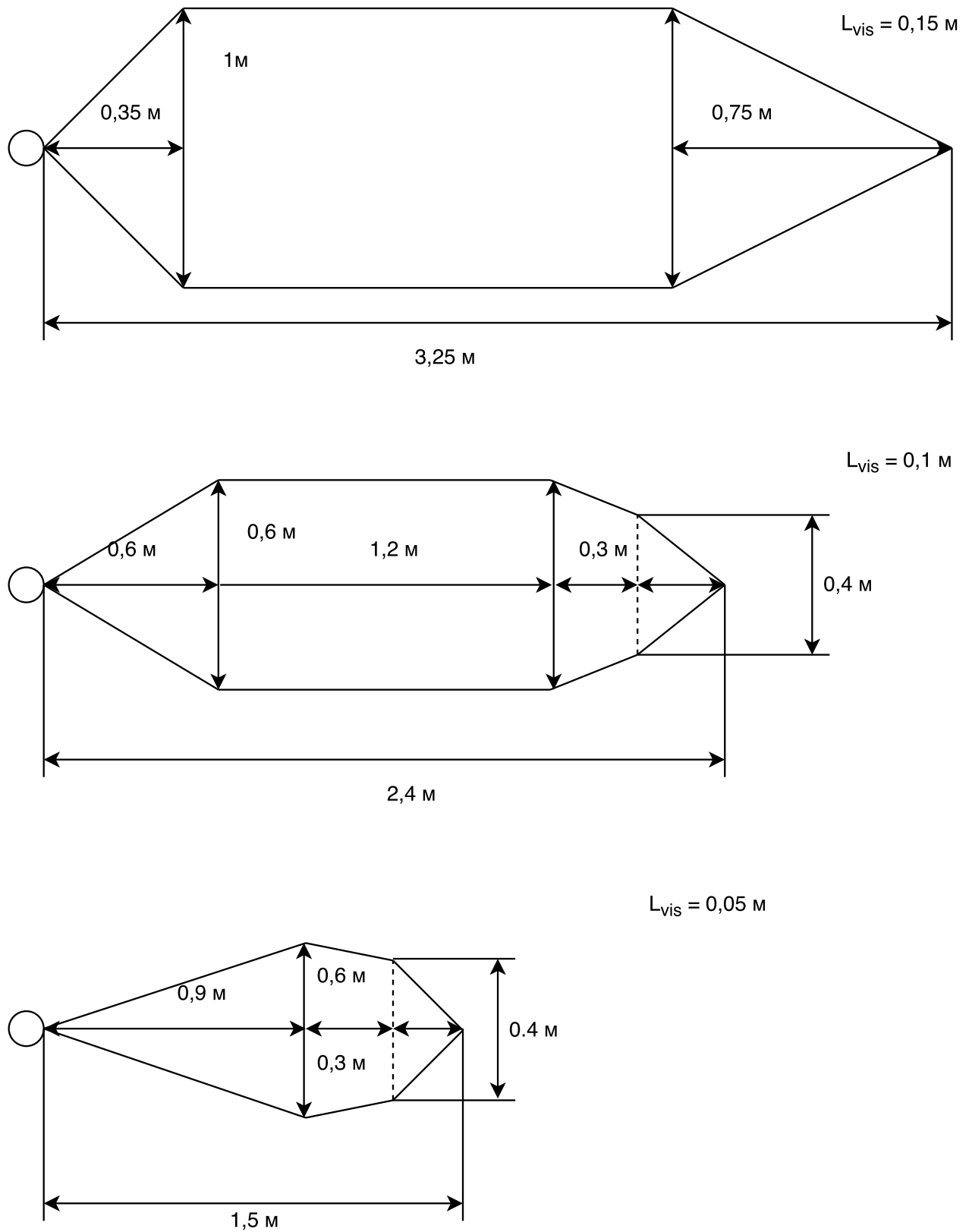


Рис. 4.8. Приближение диаграммы чувствительности сонаров, использованных в экспериментах. Она состоит из 3 областей, для каждой из которых задано $L_{vis} = \sqrt{s}$ - минимальная площадь препятствия, которое детектируется сонаром в каждой точке соответствующей области видимости.

Заключение

В работе предложены два алгоритма для восстановления карты проходимости с помощью сонаров.

Первый основан на методе стохастического градиента. Метод не использует предположения о независимости ячеек карты проходимости и способен работать в режиме реального времени. Проведенны численные эксперименты на реальных и синтетических данных, которые показали, что в ряде случаев предложенный метод способен восстанавливать карты значительно точнее, чем традиционный метод, основанный на обратной модели сонара. Также в работе предложены способы того, как разработанный метод можно использовать в режиме реального времени для продолжительного картирования территории, когда количество данных, получаемых роботом, непрерывно растет.

Второй алгоритм при восстановлении использует метод градиентного спуска. Предложена простая прямая модель сонара, которая описывает насколько хорошо наблюдения сонара объяснены имеющейся картой проходимости. Основным преимуществом предложенной модели над стандартными является отсутствие предположения о независимости значений проходимости соседних ячеек карты проходимости.

Проведенные эксперименты показали, что предложенный метод превосходит традиционный метод восстановления карты проходимости. Метод может быть имплементирован для работы в реальном времени: для этого достаточно имплементировать динамическое добавление и удаление членов из функции, которая минимизируется при поиске карты. Однако в данной работе подобные эксперименты не были проведены.

В Приложении [А](#) представлены другие карты проходимости, которые были восстановлены имплементированными в этой работе алгоритмами. В результате этой работы опубликованы статьи [??](#), [??](#). В будущем планируется имплементирование real-time версий предложенных алгоритмов, используя подходы предло-

женные в этой работе, и проведение более масштабных испытаний на реальных данных.

Список литературы

1. Moravec H. Sensor fusion in certainty grids for mobile robots. // AI magazine. 1988. Vol. 9(2). P. 61–.
2. Elfes A. Occupancy grids: a probabilistic framework for robot perception and navigation.: Ph.D. thesis / Department of Electrical and Computer Engineering, Carnegie Mellon University. 1989.
3. Lategahn H., Derendarz W., Graf T. et al. Occupancy grid computation from dense stereo and sparse structure and motion points for automotive applications. // In Intelligent Vehicles Symposium (IV) IEEE. 2010. P. 819–824.
4. Burger A. J. Occupancy Grid Mapping using Stereo Vision, master dissertation. 2015.
5. Moras J., Cherfaoui V., Bonnifait P. Credibilist occupancy grids for vehicle perception in dynamic environments. // In Robotics and Automation (ICRA), 2011 IEEE International Conference. 2011. P. 84–89.
6. Elfes A. Using occupancy grids for mobile robot perception and navigation. // Computer. 1989. Vol. 22(6). P. 46–57.
7. Thrun S. Learning Occupancy Grids With Forward Sensor Models. // Autonomous Robot. 2003. Vol. 15(2). P. 111–127.
8. Konolige K. Improved occupancy grids for map building. // Autonomous Robots 4. 1997. Vol. 4. P. 351–367.
9. Elfes A. Occupancy grids: A stochastic spatial representation for active robot perception. // arXiv:1304.1098. 2013.
10. Dempster A. P., Laird N. M., Rubin D. B. Maximum likelihood from incomplete data via the EM algorithm. // Journal of the royal statistical society. Series B (methodological). 1977. P. 1–38.
11. Latombe J.-C. Robot Motion Planning. Boston: Kluwer Academic Publishers, 1991.
12. Tardos J., Neira J., Newman P., Leonard J. Robust mapping and localization in

- indoor environments using sonar data. // The International Journal of Robotics Research. 2002. Vol. 21(4). P. 311–330.
13. Paskin M., Thrun S. Robotic mapping with polygonal random fields. // Proceedings of the Twenty-First Conference on Uncertainty in Artificial Intelligence (UAI2005). 2005. P. 450–458.
 14. LV MaxSonar EZ Series Data sheet. http://maxbotix.com/documents/LV-MaxSonar-EZ_Datasheet.pdf.
 15. Shvets E., Nikolaev D. Complex approach to long-term multi-agent mapping in low dynamic environments. // Eighth International Conference on Machine Vision. International Society for Optics and Photonics. 2015.

Приложение А

Восстановленные карты проходимости

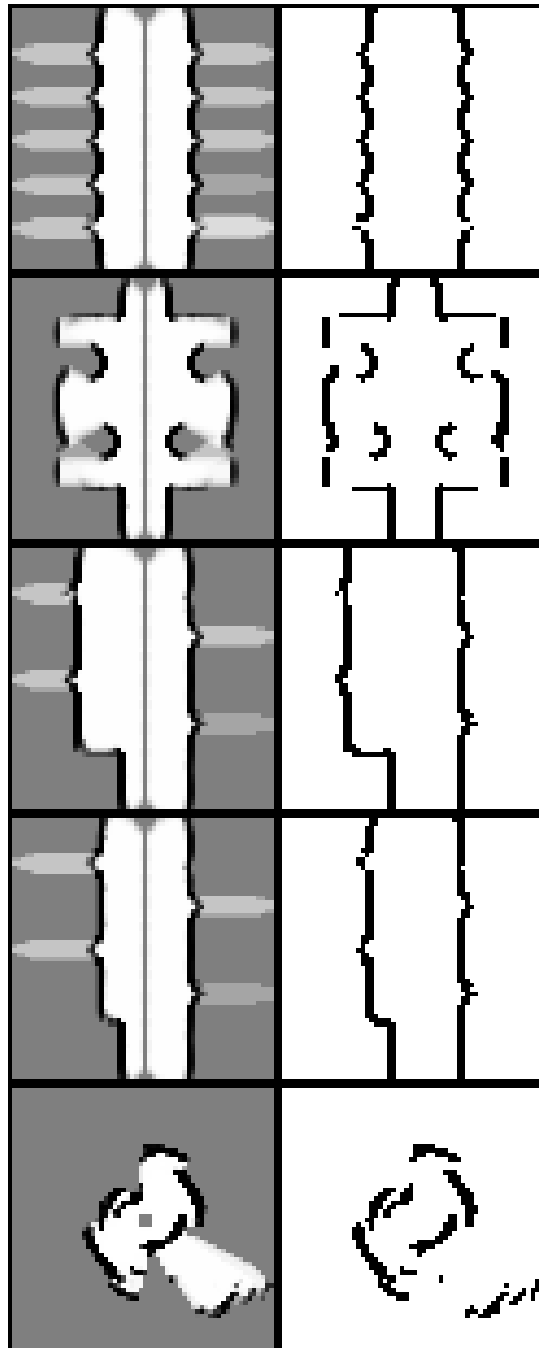


Рис. А.1. Результаты восстановления карты проходимости на основе синтетических и реальных данных с помощью алгоритма картирования на основе обратной модели сонара. На рисунке представлены обычная карта проходимости (слева) и бинаризированная версия (справа). Масштаб карты 10 сантиметров на одну клетку карты. Размер картируемых территорий 6 метров на 6 метров.

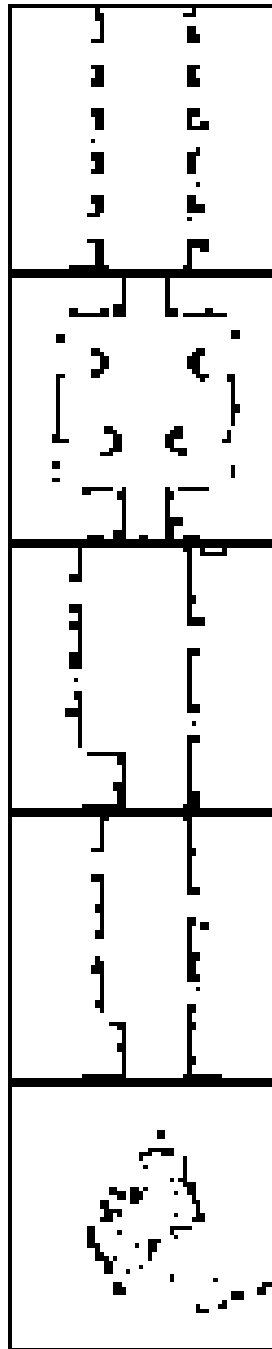


Рис. А.2. Результаты восстановления карты проходимости на основе синтетических и реальных данных с помощью алгоритма картирования на основе метода стохастического градиента. Масштаб карты 10 сантиметров на одну клетку карты. Размер картируемых территорий 6 метров на 6 метров.

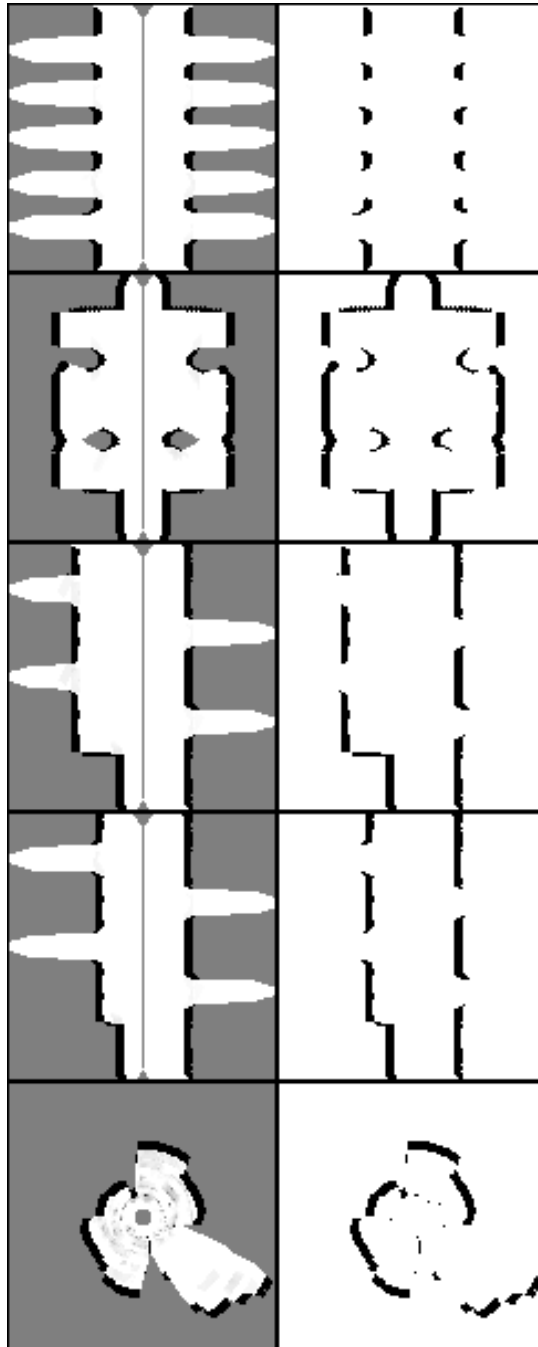


Рис. А.3. Результаты восстановления карты проходимости на основе синтетических и реальных данных с помощью алгоритма картирования на основе метода градиентного спуска. На рисунке представлены обычная карта проходимости (слева) и бинаризованная версия (справа). Масштаб карты 5 сантиметров на одну клетку карты. Размер картируемых территорий 6 метров на 6 метров.