УДК 004.896:621.865

А. В. Марченко¹, Ю. А. Гунченко², д.т.н., А. Е. Козлов¹, к.т.н.¹Одеський національний морський університет, г. Одеса²Одеський національний університет ім. Мечникова, г. Одеса

ЛОКАЛИЗАЦИЯ АВТОНОМНОГО РОБОТА С ПОМОЩЬЮ МЕТОДА ФИЛЬТРА ЧАСТИЦ

В статье рассмотрена актуальная научно-практическая задача – определение местоположения (локализация) автономного робота на местности. Проведен анализ метода фильтра частиц на основе фильтра Байеса, для практической реализации предложен алгоритм локализации. Учтены особенности сенсоров-дальномеров, влияние на них шумов, отклонений и отказов. Моделирование алгоритма показало его адекватность и возможность применения в робототехнике.

Ключевые слова: робот, локализация, фильтр частиц, дальномер, положение, ресэмплинг (resampling).

Постановка проблемы

Создание автономных роботов, которые могут самостоятельно решать поставленные задачи, является актуальной проблемой робототехники.

Мобильная робототехника – это одна из наиболее динамично развивающихся областей. Создаются алгоритмы, позволяющие осуществить переход от мобильных роботов, дистанционно управляемых оператором, к роботам способным выполнять движение в автоматическом режиме, без постоянного контроля.

Локализация – одна из наиболее фундаментальных задач восприятия в робототехнике, поскольку знание местонахождения объектов и самого действующего субъекта является основой успешной физического взаимодействия. Существуют три разновидности задачи локализации с возрастающей сложностью: отслеживание траектории, глобальная локализация, задача похищения. Во всех случаях предполагается, что робот имеет модель окружающей среды, то есть карту [1].

Целью работы является анализ и синтез алгоритма для глобальной локализации мобильного робота с использованием вероятностного подхода.

Эта задача является актуальной, потому что служит для обучения робота движению и ориентации на местности.

1. Постановка задачи локализации и фильтр Байеса

Задачей локализации является вычисление распределения вероятностей состояния системы в каждый момент времени, основываясь на информации от сенсоров и системы управления робота. В отличие от других известных методов – фильтр

Байеса представляет собой наиболее общий механизм, который позволяет рекурсивно вычислять вероятностное распределение на основании наблюдений и управления [2].

Состояние системы оценивается в дискретные моменты времени, которые будем обозначать: t , $t+1$, $t+2$, ... Для моделирования данной задачи введем следующие случайные переменные:

x_t – вектор описывающий состояние системы в момент времени t , или в нашем случае, местоположение и ориентация робота в пространстве;

z_t – вектор описывающий данные с сенсоров;

u_t – вектор управления, в нашем случае описывающий действия робота, например, вращения колес, или поворот вокруг своей оси.

Процесс движения робота в пространстве может быть промоделирован с помощью случайного процесса, который представлен в графическом виде на рис. 1.

Рисунок 1 – Графический вид представления процесса движения робота в пространстве

Стрелки показывают зависимости между случайными переменными. Как видим, состояние си-

стемы (положение робота) в момент времени t зависит от состояния в предыдущий момент $t-1$ и от действий робота. Само состояние системы x является неизвестным ($x_0, x_1, x_2 \dots x(t)$), и единственное что мы можем наблюдать это переменные u и z . Информация от сенсоров z , в свою очередь, зависит от положения робота и от окружения, в котором робот находится (которое мы будем называть картой).

Иными словами, будем рассматривать следующие условные распределения вероятностей:

$p(x_t | u_t, x_{t-1})$ – распределение вероятностей, которое описывает, как система эволюционирует во времени – модель движения.

$p(z_t | x_t)$ – распределение вероятностей, которое описывает, как измерения сенсоров соответствуют положению робота – модель измерений.

Тогда задачей локализации является вычисление, в каждый момент времени t , следующего распределения вероятностей:

$$p(x_t | z_{1:t}, u_{1:t}, x_{0:t-1}) = p(x_t | z_t, u_t, x_{t-1}) \quad (1)$$

В (1) мы сразу сделали важное допущение о том, что положение робота в следующий момент времени зависит только от текущего положения и от текущей информации от сенсоров, но не зависит от предшествующих положений или предшествующих показателей сенсоров. Это допущение – свойство Маркова, и является очень важным для эффективного решения задачи.

Для определения алгоритма решения задачи, обозначим $bel(x_t) = p(x_t | z_{1:t}, u_{1:t})$ распределение вероятностей, которое характеризует знания робота о своем местоположении в момент времени t . Фильтр Байеса позволяет рекурсивно вычислить $bel(x_t)$ на основании $bel(x_{t-1})$, а также данных о перемещении и данных с сенсоров.

В общем виде модель алгоритма фильтрации Байеса выглядит следующим образом:

$$\overline{bel}(x_t) = \int p(x_t | u_t, x_{t-1}) bel(x_{t-1}) dx; \quad (2)$$

$$bel(x_t) = \gamma p(z_t | x_t) \overline{bel}(x_t). \quad (3)$$

В приведенной выше модели можно выделить два важных компонента:

Шаг 1: выражение (2) является прогнозированием положения робота после выполнения действия u_t .

Шаг 2: по выражению (3) прогноз уточняется с помощью данных наблюдений.

Изначальное распределение $bel(x_0)$ представляет наше знание об изначальном положении робота. Например, в случае полного отсутствия каких-либо знаний мы можем использовать равномерное распределение по всему пространству состояний.

Фильтр Байеса (2, 3) предоставляет нам общий механизм для рекурсивной оценки положения робота. Модель движения и модель измерений являются исходными данными, и обычно моделируются на этапе создания робота с учетом особенностей движителей и сенсоров. Модель измерения также содержит информацию об окружении робота, например, карту местности или помещения. К сожалению интегрирование в (2) и умножение в (3) в явном виде возможно только для очень простых моделей движения и моделей сенсоров, либо для полностью дискретных сред. Поэтому на практике применяется множество приближенных методов, одним из которых является перспективный метод фильтрации частиц.

2. Метод фильтрации частиц

Основная идея метода фильтрации частиц состоит в представлении апостериорного распределения положения робота $bel(x_t)$, с помощью конечного числа семплов, вместо того чтобы представлять его распределение в параметрическом виде (например экспоненциальной функции в случае нормального распределения) [3]. Такое представление является приближенным, но благодаря непараметрическому виду позволяет представлять намного более сложные распределения.

Обозначим $X = \{x^{(1)}, x^{(2)}, \dots, x^{(N)}\}$ множество из N частиц. Каждая частица представляет конкретную гипотезу о положении робота в данный момент. Также присвоим каждой частице вес $w^{(1)}, w^{(2)}, \dots, w^{(N)}$, который будет обозначать, насколько данная гипотеза о положении робота правдоподобна в данный момент.

Тогда псевдокод алгоритма фильтрации частиц для локализации робота выглядит следующим образом:

particle_filter(X_{t-1}, u_t, z_t):

for $n = 1$ to N :

$x_t^{(n)} = move(x_{t-1}^{(n)}, u_t)$ // Шаг 1

$w_t^{(n)} = sensor(x_t^{(n)}, z_t, карта)$ // Шаг 2

end for

$$X_t = \emptyset$$

for $n = 1$ to N :

вытянуть $x_t^{(i)}$ с вер. $w^{(i)}$ из X_{t-1} // Шаг 3

$$X_t = X_t + x_t^{(i)}$$

end for

return X_t

Шаг 1 является шагом прогнозирования. На этом шаге информация о перемещениях робота применяется к каждой из частиц, с помощью функции *move()*. Таким образом, мы используем новую информацию о движении, чтобы обновить гипотезы о положении робота.

Возьмём, например, частицу $x_t = \{10, 10, 90^\circ\}$, которая описывает гипотезу о том, что координаты робота (10, 10) и он ориентирован строго на восток (0° это север). Допустим теперь, что мы знаем, что робот проехал вперед 10 метров и повернулся влево на 30° . Тогда новые координаты частицы, или наша новая гипотеза о положении робота, будет следующая $x_{t+1} = \{20, 10, 60^\circ\}$.

После этого на **Шаге 2** вес каждой частицы вычисляется на основании того как хорошо согласуются данные от сенсоров, карта и положение частицы. Другими словами, вес частицы, должен быть пропорционален вероятности того что при данной карте, в положении x_t могут быть получены измерения z_t . Тут важно понимать, что измерения z_t получены от реального робота, в том месте, где он действительно расположен в данный момент. Далее мы пытаемся соотнести эти измерения с положением каждой частицы, и на основании этого получаем её вес. Например, допустим после предыдущего шага, частица x_{t+1} оказалась в самом углу комнаты на нашей карте. Также, предполагаем, наш робот оборудован лазерным дальномером, измерения z_t с которого, в данный момент времени определяется, что ближайший объект находится на значительном расстоянии в 10м от робота. Таким образом, для частицы x_{t+1} (которая находится в углу комнаты, а значит близко от стен), данные измерений плохо соотносятся с её положением. Из чего мы можем сделать вывод что данная, частица представляет маловероятную гипотезу о положении робота. Таким образом, вес w_{t+1} этой частицы будет очень маленьким.

Цикл, содержащий **Шаг 3**, называется ресэмплингом (*resampling*) и является очень важным элементом метода фильтрации частиц. На этом шаге частицы фильтруются. Это делается путем выборки с возвратом N частиц из исходного мно-

жества, причем вероятность выбора конкретной частицы прямо пропорциональна её весу. После выборки вес частицы уже не имеет значения, и будет пересчитан при следующей итерации алгоритма. Этот шаг позволяет сформировать новое множество частиц, в котором информация о плотности вероятности представлена с помощью плотности частиц, а не их веса. После этого алгоритм готов к следующей итерации.

Метод фильтрации частиц позволяет эффективно представлять непараметрические распределения вероятностей любой сложности, а также использовать произвольные нелинейные модели движения и сенсоров. Это обусловило чрезвычайную популярность этого метода на практике. Тем не менее необходимо учитывать, что этот метод является приближенным и тщательный подбор параметров необходим для обеспечения его устойчивой работы.

Приближенность метода обусловлена следующими причинами:

- конечное число используемых частиц;
- случайность процесса ресэмплинга. В ходе ресэмплинга мы удаляем частицы и тем самым уменьшаем разнообразие гипотез;
- неточность априорного распределения. Так как метод, по сути, не генерирует новые гипотезы (частицы), а лишь обновляет либо удаляет существующие, то первоначальное распределение должно содержать частицы близкие к реальному положению робота. Эта проблема особенно актуальна в случае пространств состояний со многими измерениями и обычно требует увеличения числа используемых частиц.

3. Модель движения

На шаге 1 в предлагаемом алгоритме все частицы перемещаются в соответствии с моделью движения робота. Рассмотрим одну из моделей движения робота на плоскости, которая часто используется на практике. Положение робота описывается двумя координатами (x, y) и углом поворота θ . Допустим мы можем использовать датчики робота для получения информации о его относительном перемещении. Например, в реальном положении это может быть информация вида:

- Робот повернулся на 15°
- Робот проехал 15 м
- Робот повернулся на 30°

Каждое изменение положения робота мы можем описать с помощью трех компонентов: δ_{rot1} – начального поворота, δ_{trans} – линейного переме-

щения и δ_{rot2} – конечного поворота, как приведено на рис. 2.

Рисунок 2 – Изменение положение робота

Таким образом, $u_t = \langle \delta_{rot1}, \delta_{trans}, \delta_{rot2} \rangle$. Тогда алгоритм движения робота можно записать в следующем виде:

move(x_{t-1}, u_t)

$$\tilde{\delta}_{rot1} = \delta_{rot1} + N(\alpha_1)$$

$$\tilde{\delta}_{trans} = \delta_{trans} + N(\alpha_2)$$

$$\tilde{\delta}_{rot2} = \delta_{rot2} + N(\alpha_3)$$

$$x' = x + \tilde{\delta}_{trans} \cos(\theta + \tilde{\delta}_{rot1})$$

$$y' = y + \tilde{\delta}_{trans} \sin(\theta + \tilde{\delta}_{rot1})$$

$$\theta' = \theta + \tilde{\delta}_{rot1} + \tilde{\delta}_{rot2}$$

return $x_t = \{x', y', \theta'\}$

Как видим, модель движения использует известные тригонометрические формулы для вычисления конечного положения робота. Однако, важным моментом являются первые три строки, где мы применяем случайный шум ко всем входным параметрам. Это позволяет учесть неточность датчиков и приводов робота. На рис. 3 показано распределение конечного положения робота для различных показателей шума.

Рисунок 3 – Распределение конечного положения робота с учетом показателей шума

4. Модель сенсора

На шаге 2 алгоритма мы используем модель измерений робота для вычисления веса каждой

частицы в данный момент времени. Модель измерений представляет собой функцию, которая на основании данных от сенсоров z_t и положения частицы, вычисляет вес частицы. Для этого модель сенсоров использует информацию о карте и определяет насколько вероятны текущие измерения для каждой частицы x_i .

Вид модели движения и вектора z_t во многом зависит от вида сенсора, используемого роботом. Например, в случае если робот оборудован стереокамерой, z_t может представлять собой пару изображений, а модель сенсоров пытается распознать объекты на этих изображениях и сопоставить их с реальными объектами на карте. Модель сенсоров проектируется на этапе разработки робота и должна быть тщательно откалибрована и протестирована.

Рассмотрим модель сенсора на примере дальномера – датчика, который определяет расстояние до ближайшего объекта на карте в определенном направлении. Допустим, робот оборудован K дальномерами, которые равномерно расположены по окружности. Тогда вектор измерений $z_t = \{z_t^1, z_t^2, \dots, z_t^K\}$ будет содержать K значений.

Вектор z_t получен в результате измерений из текущего положения робота, это положение нам не известно. Для того чтобы вычислить вес частицы, необходимо простимулировать измерения из той точки где находится частица. Это делается с помощью построения луча в каждом направлении и вычислении расстояния до ближайшего объекта на карте. Таким образом, для каждой частицы мы получим вектор $z_t^* = \{z_t^{1*}, z_t^{2*}, \dots, z_t^{K*}\}$, который содержит ожидаемые измерения для этой частицы. Задача модели сенсоров в данном случае, посчитать вес частицы на основании того, насколько сильно z_t отличается от z_t^* , причем очень важно учесть, что даже самый качественный сенсор не выдает абсолютно точных измерений, т.е. необходимо учесть ошибки измерений.

Для упрощения модели примем также важное допущение о независимости сигналов сенсоров от соседних сенсоров. На практике это не всегда так, например, если один из сенсоров «видит» стену, существует немалая вероятность что соседние сенсоры также её «видят». Тем не менее, это допущение хорошо работает на практике и существенно упрощает вычисления, так как условие независимости позволяет нам рассматривать каждый дальномер отдельно и потом просто перемножить ре-

зультаты.

Рассмотрим три вида ошибок измерений: небольшие отклонения, отказ сенсора, и случайный шум.

Небольшие отклонения в показателях сенсора можно промоделировать с помощью нормального распределения вокруг ожидаемого значения:

$$p_{откл}(z_t^k | x_t, m) = \begin{cases} N(z_t^k, \delta_{откл}^2), & \text{если } 0 \leq z_t^k \leq z_{max} \\ 0, & \text{иначе.} \end{cases}$$

Стандартное отклонение $\delta_{откл}^2$ зависит от типа дальномера. Например, для лазерного дальномера эта величина будет намного меньше чем для ультразвукового. Также каждый дальномер характеризуется максимальным расстоянием измерения z_{max} .

Отказ сенсора может произойти если луч дальномера попадает на поверхность под очень острым углом, либо поверхность вообще не отражает сигнал. В таком случае дальномер обычно возвращает максимальное расстояние. Эта ситуация может быть смоделирована с помощью точечного распределения следующим образом:

$$p_{отк}(z_t^k | x_t, m) = \begin{cases} 1, & \text{если } z_t^k = z_{max} \\ 0, & \text{иначе.} \end{cases}$$

Случайный шум может возникать вследствие необъяснимых ошибок в работе дальномера, либо из-за наличия не смоделированных статических или динамических объектов среды. Например, в ультразвуковом дальномере такая ситуация проис-

ходит довольно часто из-за многократного «переотражения» сигнала. Данный шум будем моделировать с помощью равномерного распределения по всему пространству возможных сигналов:

$$p_{шум}(z_t^k | x_t, m) = \begin{cases} \frac{1}{z_{max}}, & \text{если } 0 \leq z_t^k < z_{max} \\ 0, & \text{иначе.} \end{cases}$$

В результате модель сенсоров робота можно представить в виде взвешенной суммы этих трех компонентов:

$$p = a_1 p_{откл} + a_2 p_{отк} + a_3 p_{шум}.$$

На практике параметры $\delta_{откл}^2$ и a_1 обычно определяются экспериментально и являются уникальными для каждого сенсора.

5. Результаты моделирования.

Работа предложенного алгоритма продемонстрирована на рис. 4, где показано, как робот определяет свое местонахождение в здании.

На рис. 4,а частицы распределены равномерно согласно распределению априорных вероятностей, показывающему наличие глобальной неопределенности в отношении положения робота. На рис. 4,б показано, как поступает первый ряд результатов измерений и частицы формируют кластеры в областях с высоким распределением апостериорных вероятностей состояний. На рис. 4,в показано, что поступило достаточное количество результатов измерений, чтобы все частицы собрались в одном месте.

Рисунок 4 – Результаты моделирования определения местоположения автономного робота в здании: а) робот генерирует множество предположений о своем местонахождении; б) формируются кластеры с наиболее высокими вероятностями; в) робот определил свое местоположение

Выводы

Полученные в работе результаты моделирования предложенного алгоритма локализации робота подтверждают адекватность модели и предложенного алгоритма, который ее реализует. Для реализации алгоритма на практике не требуется больших вычислительных мощностей, что позволяет его реализовать на стандартных вычислительных системах.

Предложенный алгоритм можно рекомендовать к использованию в автономных роботах.

Список использованных источников

1. Юревич Е. И. Основы робототехники / Е. И. Юревич. – СПб, 2010. – 252 с.
2. Thrun S. Probabilistic robotics / S. Thrun, W. Burgard, D. Fox – Seattle, 2000. – 492 p.
3. Гунченко Ю. О. Локалізація робота на

місцевості за допомогою фільтра частинок / Ю. О. Гунченко, А. В. Марченко, Л. В. Солодєєва // Тези доповідей X Міжнародної науково-практичної конференції “Військова освіта і наука: сьогодення та майбутнє”. — Київ, 2014. – С. 30.

4. Интернет портал Википедия // – Режим доступа: <https://ru.wikipedia.org/wiki/Python>. – 15.01.2015.

5. Swaroop C.H. A Byte of Python: пер. с англ. / С.Н. Swaroop – NY, 2013. – 159 p.

6. Интернет портал Википедия // – Режим доступа: <https://ru.wikipedia.org/wiki/PyQt>. – 19.01.2015.

Поступила в редакцию 16.02.2015

Рецензент: д.т.н., профессор Вычужа-нин В. В., Одесский национальный морской университет, г. Одесса.

А. В. Марченко, Ю. А. Гунченко, д.т.н., А. Ю. Козлов, к.т.н.

ЛОКАЛІЗАЦІЯ АВТОНОМНОГО РОБОТА ЗА ДОПОМОГОЮ МЕТОДУ ФІЛЬТРУ ЧАСТИНОК

У статті розглянута актуальна науково-практична задача – визначення місця розташування (локалізація) автономного робота на місцевості. Проведено аналіз методу фільтра частинок на основі фільтра Байєса, для практичної реалізації запропоновано алгоритм локалізації. Враховані особливості сенсорів-далекомірів, вплив на них шумів, відхилень і відмов. Моделювання алгоритму показало його адекватність і можливість застосування в робототехніці.

Ключові слова: робот, локалізація, фільтр частинок, далекомір, положення, ресемплінг (resampling).

A. V. Marchenko, Y. A. Gunchenko, DSc, A. E. Kozlov, PhD

LOCALIZATION OF AN AUTONOMOUS ROBOT USING THE METHOD OF PARTICLE FILTER

The article discusses current scientific and practical problem – location (localization) autonomous robot on the ground. The analysis method of particle filter based on Bayesian for the practical implementation of an algorithm for localization. Takes into account features of sensors, range finders, the impact on them of noise, deviations and failures. Modeling algorithm has shown its value and could be used in robotics.

Keywords: robot, localization, particle filter, range finder, a position resampling.