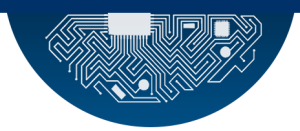
СПбГЭТУ «ЛЭТИ»
ПЕРВЫЙ ЭЛЕКТРОТЕХНИЧЕСКИЙ



А. Ю. Филатов

Simultaneous Localization And Mapping

Фрагмент конспекта

СПбГЭТУ «ЛЭТИ», 2022 г.

1 EKF-SLAM

1.1 Что такое simultaneous localization and mapping(SLAM)?

Метод SLAM (simultaneous localization and mapping) – метод одновременной локализации и построения карты. Используется в мобильных автономных средствах для построения или обновления карты неизвестной местности, одновременно отслеживая местоположение агента(робота) в ней.

Ранее роботы имели проблему с локализацией и определением своего местоположения в той или иной среде.

Эта проблема решается с помощью технологии SLAM, которая одновременно решает проблему локализации и картирования. Это очень полезно для внутренних платформ, таких как беспилотные летательные аппараты и платформы дополненной реальности, а также для наружных применений, таких как автономные транспортные средства.

Суть метода заключается в следующем: с помощью датчиков, в качестве которых могут использоваться одометры, акустические датчики или лазерные дальномеры, производится вычисление расстояния от агента (в качестве агента могут использоваться дроны, роботы, автомобили и т.д.) до препятствия, находящегося в поле зрения датчиков, с последующим построение карты местности в 2D или 3D формате (картирование).

Об используемых алгоритмах

EKF-Slam[1] основан на расширенном фильтре Калмана[2], шаги которого состоят из:

- Движения робота
- Обнаружение новых опорных точек(Landmarks)
- Определение какие опорные точки встречались ранее
- Коррекция местоположения при помощи EKF

Главная идея алгоритма EKF-SLAM

Основная идея алгоритма заключается в том, что после каждого получения данных об окружающей обстановке, происходит:

- Прогноз состояния и ковариационной матрицы ошибок с использованием модели движения робота;

- Определение ошибки прогноза и ковариационной матрицы по наблюдениям;
- Коррекция оценок состояния и ковариационной матрицы;
- Расширение вектора состояния и ковариационной матрицы.

Ориентиры, обнаруженные датчиками аппарата на каждом шаге, включают в себя ориентиры, уже существующие на карте, а также новые ориентиры. Существовавшие ориентиры были использованы для оценок состояния на приведенной выше последовательности действий. Новые ориентиры добавляются в вектор состояния системы через процесс инициализации.

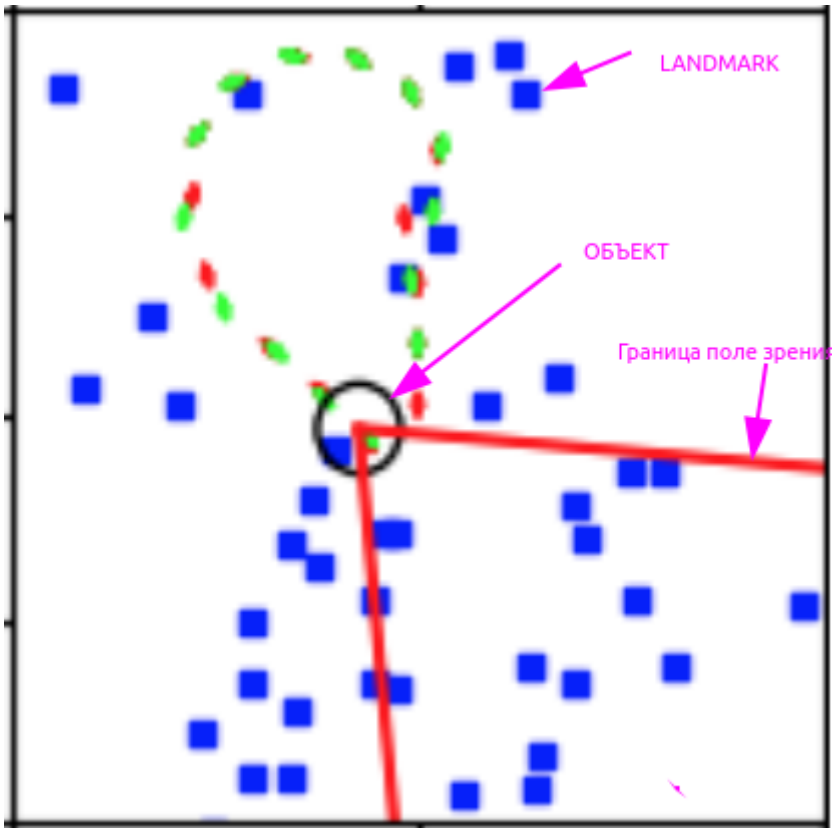


рис. 1

Описание элементов на рис. 1:

- Объект - робот, который движется в пространстве;
- Landmark – опорная точка, по которой робот ориентируется в пространстве;
- Граница поле зрения – это граница, внутри которой находятся опорные точки. Если опорная точка встречалась ранее, то ее

позиция использования для корректировки положения робота. Если нет, то она добавляется во множество опорных точек, которые уже видел робот

Псевдокод для EKF-SLAM

```
initialize map()
time = 0
robot = Robot() # Instance of robot
while (execution() == true) do

	control = acquire control signal()
	move robot(robot, control)

	for each sensor in robot->list of sensors
		raw = sensor->acquire raw data()

		for each observation in sensor->feasible observations()

			measurement = find known feature(raw, observation)
			update map(robot, sensor, landmark, observation, measurement)
		end

	measurement = detect new feature(raw)

	landmark = init new landmark(robot, sensor, measurement)
	create new observation(sensor, landmark)
end
time ++
end
```

Список источников

1. Введение в SLAM и EKF-SLAM [электронный ресурс] URL: https://www.iri.upc.edu/people/jsola/JoanSola/objectes/curs_SLAM/SLAM2D/SLAM%20course.pdf (Дата обращения: 02.05.2022)

2. Подробное описание работы фильтра Калмана в EKF-SLAM [электронный ресурс] URL: http://vestnikprib.ru/articles/1034/1034.pdf (Дата обращения: 02.05.2022)

2 FAST-SLAM

2.1 Что такое Fast-SLAM?

Об используемых алгоритмах

Fast Slam алгоритм основан на использовании фильтра частиц, шаги которого состоят из:

- выделение следующего набора частиц, используя предложенное распределение;
- вычисление весов важности частиц;
- resampling - избавление от наиболее не достоверных гипотез(частиц).

Представление частицы

- набор взвешенных выборок(образцов);

$$\mathcal{X} = \left\{ \left\langle x^{[i]}, w^{[i]} \right\rangle \right\}_{i=1,\ldots,N}$$

Рис. 1.

- каждая выборка - одна гипотеза состояния объекта;
- для SLAM на основе признаков - использование информации об ориентирах.

$$x = (x_{1:t}, l_{1,x}, l_{1,y}, \ldots, l_{M,x}, l_{M,y})^T$$

pose **landmarks**

Рис. 2.

Главная идея алгоритма

Набор частиц используется для моделирования пути робота, каждый образец является гипотезой пути. Для каждого образца мы можем рассчитать индивидуальную карту ориентиров.

Каждый ориентир представляет собой 2x2 EKF-фильтр каждая частица должна поддерживать M(количество ориентиров) отдельных EKF

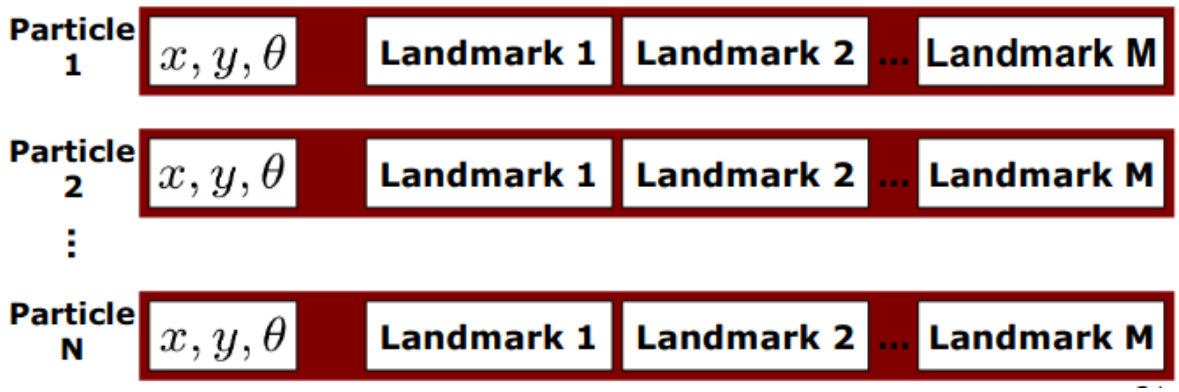


Рис. 3

Ключевые шаги FastSLAM 1.0

- найти новую позицию робота, используя данные о движении робота на данном шаге;

$$x_t^{[k]} \sim p(x_t \mid x_{t-1}^{[k]}, u_t)$$

Рис. 4.

- рассчитать вес частиц - предсказать их новые состояния;

Вычисление веса частиц

ожидаемое наблюдение

$$w^{[k]} = |2\pi Q|^{-\frac{1}{2}} \exp\left\{-\frac{1}{2}(z_t - \hat{z}^{[k]})^T Q^{-1} (z_t - \hat{z}^{[k]})\right\}$$

измеренная ковариация

Рис. 5

- обновить достоверность состояния частиц, используя информацию о наблюдаемых ориентирах(по правилу обновления EKF);
- провести resampling, чтобы избавиться от недостоверных гипотез.

Псевдокод

```
true_state = initialize state()
true_landmarks = initialize landmarks(N_landmarks)
particles = initialize particles(N_particles)

time_step = 0

while (execution() == true) do
     time_step += time_delta

     move = get robot movement(time_step)
     true_state, landmarks, noise_move = get new observation from state(true_state, move, true_landmarks)

     for each particle in particles do
          particle = get particle state from movement(particle, noise_move)
     end for

     for each landmark in landmarks do
          if landmark never seen before then
               add new landmark to particles info(particles, landmark)
          else
               update particles info (particles, landmark)
```

```
        end for

        particles = resampling particles (particles)

        estimate_state = calculate estimated robot state
(particles)

    end while
```

Сложность алгоритма:

- обновление частиц робота на основе контроля O(N);
- включить наблюдения в фильтр Калмана O(N log M);
- провести resampling O(N log M).

Таким образом, сложность – O(N log M), где N - количество частиц, M - количество ориентиров на карте.

Список источников

1. FastSLAM – Feature-based SLAM with Particle Filters [электронный ресурс] URL: http://ais.informatik.uni-freiburg.de/teaching/ws12/mapping/pdf/slam10-fastslam.pdf (Дата обращения: 02.05.2022)
2. Montemerlo M. et al. FastSLAM: A factored solution to the simultaneous localization and mapping problem //Aaai/iaai. – 2002. – Т. 593598.
3. FastSLAM: Landmark-Based Mapping [электронный ресурс] https://www.cs.utexas.edu/~kuipers/slides/L17-FastSLAM.pdf (Дата обращения: 02.05.2022)