Министерство образования и науки Российской Федерации

Санкт-Петербургский политехнический университет   
петра великого

Институт металлургии, машиностроения и транспорта

Кафедра «Мехатроника и роботостроение» при ЦНИИ РТК

ОТЧЕТ

по научно-исследовательской работе

Применение глубоких нейронных сетей в задаче создания систем управления автономным мобильным роботом

(наименование темы работы)

Направление подготовки: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

|  |  |
| --- | --- |
| Выполнил студент гр. \_\_\_\_\_\_\_\_ | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ О.В. Литвинов |
|  |  |
| Руководитель | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ А.В. Бахшиев |
|  |  |
| Научный консультант | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ И.С. Фомин |

Санкт-Петербург 2018

Тут лист с заданием

Тут ничего нет, обычно задание печатается на двух сторонах, обе учитываются в общей нумерации. Содержание должно оказаться на странице 4

Содержание

[Введение 7](#_Toc534638137)

[1. Теоретическая часть 9](#_Toc534638138)

[1.1 Анализ подходов к созданию автономных робототехнических систем 9](#_Toc534638139)

[1.1.1 Эволюционные алгоритмы 9](#_Toc534638140)

[1.1.2 Алгоритмы на основе нечёткой логики 9](#_Toc534638141)

[1.1.3 Прямой метод / экспертный 9](#_Toc534638142)

[1.1.4 Ван шот 9](#_Toc534638143)

[1.1.5 Агентный подход 9](#_Toc534638144)

[1.1.6 Сравнение/вывод 9](#_Toc534638145)

[1.2 Анализ методов обучения с подкреплением. 9](#_Toc534638146)

[1.2.1 CEM 9](#_Toc534638147)

[1.2.2 DQN 9](#_Toc534638148)

[1.2.3 Duel\_DQN 9](#_Toc534638149)

[1.2.4 SARSA 9](#_Toc534638150)

[1.2.5 DDPG 9](#_Toc534638151)

[1.2.6 PRO 9](#_Toc534638152)

[1.2.7 TRPO 9](#_Toc534638153)

[1.2.8 Сравнение/вывод 9](#_Toc534638154)

[1.3 Формализация задачи 10](#_Toc534638155)

[1.4 Эволюционные алгоритмы 10](#_Toc534638156)

[2. Разработка дописать чего 13](#_Toc534638157)

[2.1 Например, параграфы про матан RL под выбранный класс задач 13](#_Toc534638158)

[2.2 Или разный матан для разных классов задач в разных параграфах. 13](#_Toc534638159)

[2.3 Выводы по разделу 13](#_Toc534638160)

[3. Реализация дописать чего 14](#_Toc534638161)

[3.1 Описание того, что мы разработали. 14](#_Toc534638162)

[3.2 Выводы по разделу 14](#_Toc534638163)

[4. Экспериментальное исследование 15](#_Toc534638164)

[4.1 План экспериментального исследования 15](#_Toc534638165)

[4.2 Название эксперимента 1 15](#_Toc534638166)

[4.3 Итп. 15](#_Toc534638167)

[4.4 Выводы по разделу 15](#_Toc534638168)

[5. Заключение 16](#_Toc534638169)

[Список использованных источников 17](#_Toc534638170)

# Введение

Несмотря на огромные достижения в области автоматизации, до сих пор существуют сферы человеческой деятельности, автоматизация которых либо крайне неэффективна, либо вообще не представляется возможной, когда, казалось бы, весьма несложные операции (с которыми легко справляется любой малоквалифицированный персонал) вообще не поддаются автоматизации. Заполнить многие из таких «свободных от автоматизации ниш» могли бы высокоадаптивные и автономные мобильные роботы (МР).

«Интеллектуализация» МР является важнейшим направлением развития МР. Под этим термином здесь понимается повышение уровня адаптивности МР к сложным быстроизменяющимся внешним условиям или повышение степени независимости (автономности) процесса функционирования МР от человека-оператора. С увеличением степени автономности МР упрощается процесс управления роботом, уменьшается отрицательное влияние человеческого фактора, возрастает общая эффективность от применения МР.

Основные трудности при этом состоят в создании алгоритмического обеспечения, позволяющего автоматически управлять движением роботов, используя информацию о его положении относительно инерциальной системы координат и препятствий местности.

Однако при наличии большого числа публикаций, в которых используются как классические, так и современные подходы к синтезу алгоритмов обработки информации и управления, задача остается не решенной в полной мере. Это связано с излишней идеализированностью ее постановки, не учитывающей принципиальную невозможность знания точной математической модели робота, из-за отсутствия учета возможностей и характеристик реальных датчиков и желания авторов решить плохо формализуемую проблему управления при наличии большого количества разнородной информации на основе одного, порой достаточно сложного алгоритма.

Учитывая успехи алгоритмов глубокого обучения в различных плохо формализуемых задачах, было решено проанализировать и сравнить качество синтеза алгоритмов управления автономными роботами на основе нейронных сетей с классическими не нейросетевыми алгоритмами.

Таким образом, актуальность задачи создания алгоритмов системы управления мобильных роботов на основе нейронных сетей, с одной стороны, определяется востребованностью автономных мобильных роботов, с другой - отсутствием эффективных алгоритмов, способных автономно решать плохо формализуемую задачу управления автономным роботом в различных средах.

# Теоретическая часть

## Анализ подходов к созданию автономных робототехнических систем

### Эволюционные алгоритмы

<https://elibrary.ru/item.asp?id=20291507&>

https://www.sciencedirect.com/science/article/pii/S1877050916308808

Эволюционные алгоритмы (ЭА) – это алгоритмы, созданные в качестве методов решения оптимизационных задач и основанные на принципах естественного отбора. ЭА моделируют базовые положения в теории биологической эволюции — процессы отбора (селекции), мутации и воспроизводства. Множество агентов, называемое популяцией, эволюционирует согласно правилам отбора в соответствии с целевой функцией. Таким образом, каждому агенту (индивидууму) популяции назначается значение его приспособленности (значение фитнесс-функции, являющейся частным случаем целевой функции) в окружающей среде. Размножение и мутация позволяют изменяться агентам и приспособляться к среде.

В робототехнике применение ЭА позволяет выполнять различные практические задачи, в частности задачи обследования местности и маневрирования между препятствиями. Для агентов (роботов) описывается алгоритм, позволяющий им обмениваться генетической информацией с целью адаптации к решению конкретной задачи. В качестве генетической информации выступают различные стратегии управления приводами и системами робота на основе конкретных входных значений с датчиков. Такие стратегии могут задаваться изначально как случайно, так и согласно какой-либо эвристике. Цель – научить робота взаимодействовать с неизвестной средой.

Преимущества подхода:

* Возможность использования в задачах, сложно или полностью не поддающихся анализу и формализации;
* Устойчивость в задачах с изменяющейся и/или частично неизвестной средой;
* Существование большого количества как естественных, так и созданных человеком вариаций генетических операций.

Недостатки подхода:

* Оценка функции приспособленности ([фитнесс-функции](https://ru.wikipedia.org/wiki/%D0%A4%D0%B8%D1%82%D0%BD%D0%B5%D1%81%D1%81-%D1%84%D1%83%D0%BD%D0%BA%D1%86%D0%B8%D1%8F)) для сложных проблем (задачи высокой размерности) требует больших вычислительных мощностей, что часто является фактором, ограничивающим использование алгоритмов искусственной эволюции;
* Кодирование генетической информации (иногда называется геномом или хромосомами) представляет собой также сложный процесс, пусть и менее сложный, чем формализация всей задачи;
* Симуляция на настоящих роботах требует большого объёма ресурсов, поэтому обычно синтез алгоритма производят в компьютерных симуляциях, а затем полученные алгоритмы встраиваются в настоящие роботы. Однако процесс создания самой симуляции также является трудоёмким.

### Алгоритмы на основе нечёткой логики

<https://elibrary.ru/item.asp?id=27289762>

http://www.dissercat.com/content/analiz-i-sintez-intellektualnykh-sistem-avtomaticheskogo-upravleniya-s-nechetkimi-regulyator

### Прямой метод / экспертный / формализация

Формализация – представление в виде формальной системы какой-либо содержательной информации, представленной на естественном языке, путём изъятия логических свойств элементов естественного языка, существенных отношений между этими элементами, а также определение принципов логической дедукции и критериев различия правильных способов рассуждения от неправильных.

Задача синтеза алгоритма управления автономным роботом также требует формализации задачи, если она решается не методами искусственного интеллекта (ИИ). В задаче управления автономным мобильным роботом формализация заключается в составлении перечня всевозможных ситуаций, в которых может оказаться робот, и описание действий (и их порядка), которые необходимо в таких ситуациях принимать.

Преимущества:

* Полная интерпретируемость алгоритма, т.е. возможность объяснить действия робота в любой ситуации;
* Возможность оптимизации алгоритма управления;

Недостатки:

* Требование значительных ресурсов для анализа и формализации задачи, а также для составления и оптимизации алгоритма (зачастую уникального)

### Обучение с подкреплением

<http://apsheronsk.bozo.ru/Neural/Lec8.htm>

https://habr.com/post/308094/

Обучение с подкреплением — это класс методов машинного обучения, при котором происходит обучение агента (робота), который не имеет сведений о среде, но имеет возможность производить какие-либо действия в ней. Действия агента переводят среду в новое состояние и агент получает от среды некоторое вознаграждение или наказание в соответствии с функцией подкрепления. Функция подкрепления определяет цель в процессе обучения с подкреплением и является по сути соответствием между состояниями среды и числом, подкреплением, показывающим желательность, ценность состояния. В данном классе методов большое внимание уделяется поощрению/наказанию не только текущих действий, которые непосредственно привели к положительному/отрицательному результату, но и тех действий, которые предшествовали текущим. За продолжительный период действий робота в среде отвечает функция оценки. Функция оценки состояния - это итоговое подкрепление агента, которое предположительно может быть накоплено при последующих стартах из этого состояния. В то время как функция подкрепления определяет прямую, характерную желательность состояния среды, функция оценки показывает долгосрочную желательность состояний после принятия во внимание состояний, которые последуют за текущим, и подкреплений, соответствующих этим состояниям. Например, состояние может повлечь низкое непосредственное подкрепление, но иметь высокую оценку, потому как за ним регулярно следуют другие состояние, которые приносят высокие подкрепления. Единственная цель агента состоит в максимизации итогового подкрепления, которое тот получает в процессе длительной работы.

В робототехнике обучение с подкреплением активно применяется в задаче передвижения мобильного робота по лабиринту [<https://arxiv.org/abs/1608.05742>] и в задаче управления манипулятором [https://arxiv.org/abs/1504.00702, <https://arxiv.org/abs/1511.07111>]. Также существуют попытки обучать агента выполнять задачи разного плана [https://arxiv.org/pdf/1705.06366].

Преимущества:

Недостатки:

* Требования к вычислительным ресурсам в случаях многоразмерных задач;
* В процессе обучения не всегда находится оптимальная стратегия для получения максимальной долгосрочной награды

### Сравнение/вывод

## Анализ методов обучения с подкреплением.

### CEM

### DQN

[https://arxiv.org/abs/1312.5602]

Deep Q – Network (DQN) – улучшение классического алгоритма обучения с подкреплением Q-learning за счёт использования нейронных сетей.

Q-learning (в русскоязычной литературе иногда употребляется Q-обучение) – алгоритм, согласно которому агент на основе получаемого от среды вознаграждения формирует функцию полезности Q, что впоследствии дает ему возможность уже не случайно выбирать стратегию поведения, а учитывать опыт предыдущего взаимодействия со средой. Одно из преимуществ Q-обучения — то, что оно в состоянии сравнить ожидаемую полезность доступных действий, не формируя модели окружающей среды. Применяется для ситуаций, которые можно представить в виде Марковского процесса принятия решений (англ. Markov decision process (MDP)), то есть процесса, в котором вероятности переходов между состояниями не зависят от истории предыдущих переходов.

Алгоритм Q-learning пытается получить значение функции Q в явном виде для всех возможных пар «состояние среды - действие», чтобы потом максимизировать ожидаемый результат, найдя оптимальную стратегию. Однако даже в самых простых задачах робототехники число состояний среды очень велико, а если его умножить на количество возможных действий, то получится таблица, обрабатывать которую на борту робота будет затруднительно. Поэтому в алгоритме DQN таблица Q заменена функцией, которую приближает (аппроксимирует) свёрточная нейронная сеть. Помимо этого у DQN есть ещё несколько усовершенствований по сравнению с классическим алгоритмом Q-learning. Во-первых, практика показывает, что обучаться непосредственно на состояниях, происходящих друг за другом, — плохая идея, так как такие состояния слишком похожи друг на друга, сильно коррелируют, причём со временем их распределение, естественно, сдвигается в зависимости от действий (или, например, положения) робота, но остается локализованным. Это мешает эффективному обучению, ведь в обычной постановке задачи обучения предполагается, что тренировочные данные независимы, а распределение данных со временем не меняется. Поэтому по мере обучения DQN сначала накапливает некоторый опыт, сохраняя свои действия и их результаты на протяжении какого-то времени, а потом выбирает из этого опыта случайную выборку (mini-batch) отдельных примеров для обучения, взятых в случайном порядке. Во-вторых, важную роль для успеха сыграло то, что при обучении DQN сеть, которая отвечала за целевую функцию, была отделена от сети, которая собственно обучается. Сделано это по причине того, что на практике нейронные сети довольно быстро заходят в локальные экстремумы и начинают очень глубоко исследовать глобально не необходимые части пространства поиска, бесцельно тратя ресурсы и фактически не обучаясь. Избежать этого можно сделав так, чтобы сеть не сразу использовала обновленную версию в целевой функции, а обучалась достаточно долгое время по старым образцам, прежде чем использовать обновлённую целевую функцию. В-третьих, на практике обычно применяется архитектура, в которой возможное действие агента не подается на вход, а просто у сети столько выходов, сколько возможных действий, и, получая на входе состояние, сеть пытается предсказать результаты полезности, ценности каждого действия (после чего, естественно, выбирает максимальный). Это важное улучшение, потому что оно позволяет получить ответы для сразу всех действий за один проход по сети, что ускоряет происходящее в разы, а сеть от этого сильно сложнее не становится, ведь основная часть ее «логики» остается прежней и используется заново.

Преимущества:

* Простой относительно других алгоритмов глубокого обучения с подкреплением, в следствие чего обучается быстрее более сложных алгоритмов;
* Благодаря усовершенствованиям справляется с проблемой локальных минимумов;

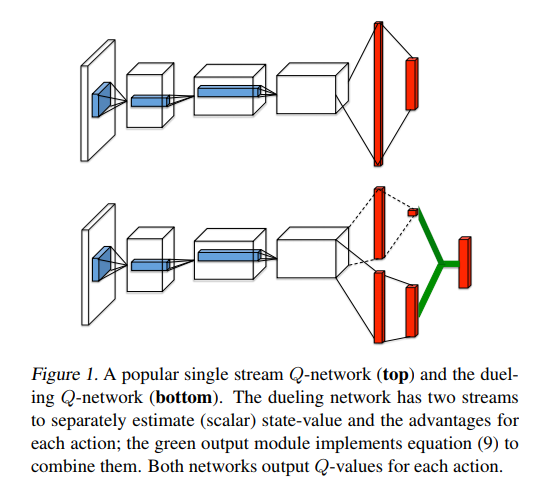
Недостатки:

### Duel DQN

Duel DQN является улучшением алгоритма DQN за счёт разделения Q-сети на два канала, один из которых вычисляет оценку позиции V, которая является функцией только состояния среды, а другой — зависящую от действия функцию преимущества (advantage function) A. На последнем этапе они просто складываются в прежнюю функцию Q. Таким образом, одна часть сети обучается оценивать позицию как таковую, а другая — предсказывать, насколько полезны будут разные наши действия в этой позиции. Такое небольшое изменение в архитектуре сети приближает смысл оценки к естественному, что часто существенно улучшает результаты [Human-Level Control through Deep Reinforcement Learning / V. Mnih et al. / / Nature,

2015, vol. 518, no. 7540. - P. 529-533. …… Playing Atari With Deep Reinforcement Learning / V. Mnih et al. / / NIPS Deep Learning

Workshop/2013.].



### SARSA

### DDPG

[Николенко книга]

[https://medium.freecodecamp.org/an-introduction-to-policy-gradients-with-cartpole-and-doom-495b5ef2207f]

[https://towardsdatascience.com/introduction-to-various-reinforcement-learning-algorithms-i-q-learning-sarsa-dqn-ddpg-72a5e0cb6287]

[https://www.cs.ubc.ca/~gberseth/blog/demystifying-the-many-deep-reinforcement-learning-algorithms.html]

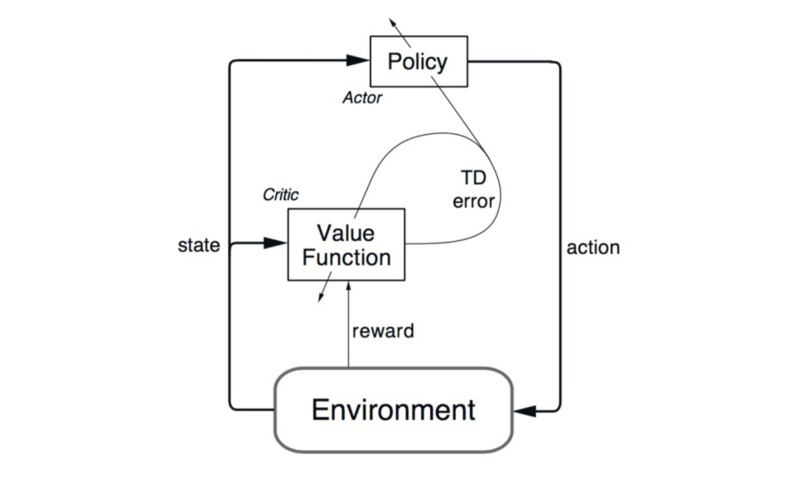
В алгоритмах Q-learning, DQN, Duel DQN и SARSA действия выбираются согласно максимальному Q-значению. В таких алгоритмах (они называются value-based, то есть основанных на значениях функции Q) стратегия представляет собой ни что иное, как выбор действия с максимальным значением Q. Проблема value-based подхода заключается в необходимости вычисления оценки награды для каждого возможного действия в каждом возможном состоянии. Особенно остро эта проблема встаёт, когда дело касается бесконечного пространства действий (обычно это физические задачи, например, классическая задача балансировки обратного маятника. Также это задачи управления вращением колёс, движением манипулятора). Например, DQN создавался для решения проблемы с достаточно высокоразмерным пространством состояний (пиксели на экране в играх Atari), но при этом очень малоразмерным дискретным пространством действий (до двух десятков комбинаций кнопок). Но задачи физического контроля имеют непрерывные и высокоразмерные пространства действий. DQN не может быть непосредственно применён к непрерывным пространствам даже при грубой дискретизации пространства действий, так как размер выходного слоя нейронной сети Q-network растёт экспоненциально с ростом степеней свободы агента.

В алгоритмах градиентного спуска по стратегиям (policy gradient (PG) methods) вместо обучения функции Q, возвращающей ожидаемую ценность каждого действия в каждом состоянии, напрямую обучается стратегия, которая состоянию ставит в соответствие действие. В данном семействе алгоритмов центральную роль занимает стратегия, согласно которой агент совершает действие в зависимости от текущего состояния и от параметров самой стратегии (обычно это веса нейронной сети). Задача обучения с подкреплением заключается в максимизации целевой функции. Вознаграждение, которое напрямую влияет на значение целевой функции, зависит от стратегии, которая в свою очередь зависит от своих параметров, следовательно, сама целевая функция зависит от параметров стратегии. PG, который является на данный момент центральным методом обучения с подкреплением в задачах робототехники, базируется на идее оптимизации целевой функции методом градиентного спуска по стратегиям вместо построения моделей окружающей среды или оценки функции Q.

Такой подход работает не в каждой задаче, так как, используя разумные ресурсы, можно посчитать градиент только за ограниченное количество шагов и с ограниченной точностью. Однако благодаря policy gradient theorem производную можно вычислять для более продолжительных промежутков, например для целой траектории. На наборах траекторий и суммарной награды по итогу прохождения соответствующих траекторий обучаются параметры стратегии.

Стратегия бывает двух видов: детерминированная и стохастическая. Детерминированная состоянию ставит в соответствие действие, которое необходимо в этом состоянии выполнить. Детерминированные стратегии используются в детерминированных средах или в средах, где случайные события слабо влияют на поведение агента (например, движение исправного робота в лабиринте при стабильных погодных условиях и при однородном освещении). Стохастические стратегии состоянию среды ставят в соответствие распределение над действиями. Это означает, что разные действия могут быть выбраны для одного и того же состояния. Такие стратегии используются в средах, где доля случайных или неопределённых событий значительна, то есть почти в любой в естественной среде. Детерминированные стратегии проще описываются и обучаются, поэтому алгоритмы deterministic policy gradient (DPG) более популярны, нежели stochastic policy gradient (SPG).

На основе идей, принёсших успех DQN, был создан алгоритм deep deterministic policy gradient (DDPG) [https://arxiv.org/abs/1509.02971], применимый для непрерывного пространства действий. Также важной чертой DDPG является применение архитектуры “actor-critic” (устоявшегося русского названия не существует, но приблизительный перевод – “действуеющее лицо-критик”), которая является по сути объединением идей Q-learning и policy gradient (рисунок).

В данном алгоритме обучаются две нейронные сети: сеть-actor обучает параметры стратегии, чтобы поданному на вход состоянию ставилось в соответствие лучшее действие, и сеть-critic, обучающаяся оцениванию стратегии по ошибке, которая высчитывается по тому же правилу, что и при обновлении Q-функции в Q-learning. Но при этом функция оценки используется только для оптимизации параметров стратегии, но сама при этом при выборе действия не используется.

Преимущества алгоритма DDPG:

* Алгоритмам, основанным на оптимизации функции оценки, присущи большие колебания во время обучения. Эти колебания связаны с тем, что выбор действия может резко измениться при сколь угодно малых изменениях оценки действий. В алгоритмах, основанных на оптимизации стратегии, ищется градиент по стратегии, поэтому процесс обучения проходит более гладко;
* Возможность рассмотрения проблемы исследования (problem of exploration) независимо от алгоритма DDPG;
* Возможность работать с высокоразмерными непрерывными пространствами действий

Недостатки алгоритма DDPG:

* Вычисление градиента по стратегии приводит к возможной сходимости в локальном максимуме вместо глобального. Такая проблема отсутствует в value-based подходе, где постоянно ищется лучшее из возможных действий;
* В случае отсутствия добавленной извне формализации исследования среды, алгоритм не будет исследовать среду вообще.

### PRO

### TRPO

### Сравнение/вывод

## Формализация задачи

# Разработка дописать чего

Тут все сложно, т.к. я не представляю, что у нас сделано. :\

## Например, параграфы про матан RL под выбранный класс задач

## Или разный матан для разных классов задач в разных параграфах.

Тут везде матан, формулы. НО только те, что мы используем дальше в исследованиях. Все что просто обзорное улетает в главу 1.

## Выводы по разделу

# Реализация дописать чего

## Описание того, что мы разработали.

В принципе этого раздела может и не быть, если нам он не нужен, а дальше мы будем сразу же исследовать на готовых решениях к примеру. Либо тут может быть что-то вспомогательное, например какие-то скрипты которые нам помогают и **которые можно использовать повторно**.

## Выводы по разделу

# Экспериментальное исследование

## План экспериментального исследования

Тут перечень экспериментов, какие процессы мы хотим изучить.

## Название эксперимента 1

Содержание эксперимента 1, графики и выводы из них. Целью может быть например исследование зависимости скорости обучения от сложности задачи, или от сложности среды (и то и другое должно как-то количественно описываться).

## Итп.

## Выводы по разделу

# Заключение

По пунктам что именно было проделано в работе, чтобы, не читая работу можно было все понять отсюда.

Направление дальнейших исследований:

- раз

- два

- три

- итп

Содержит:

* Выводы по работе
* Рекомендации по дальнейшим исследованиями (даже в случае, если автор сам не планирует дальше работать над этой темой!)

# Список использованных источников

* Источники должны быть не старше 5 лет от года написания НИР.
* Не менее 15 источников.
* Не менее 50% ссылок на зарубежную литературу.
* Ссылки на каждый источник должны быть в тексте (в корректном месте).