**На первой странице должен быть титульник, но я его перенёс в отдельный файл. В введении добавил строку о задачи курсовой работы.**

Оглавление

[Введение 3](#_Toc486223250)

[1. Нейронные сети 3](#_Toc486223251)

[2. Самообучение с подкреплением 6](#_Toc486223252)

[2.1 Q-learning 7](#_Toc486223253)

[2.2 Глубинный q-learning 8](#_Toc486223254)

[3. Испытания 9](#_Toc486223255)

[Вывод 13](#_Toc486223256)

[Список литературы 14](#_Toc486223257)

# Введение

С увеличением возможностей современных роботов увеличивается и сложность выполняемых ими задач. Многие задачи сложно формулируются в виде формул и алгоритмов. То, что человек понимает на подсознательном уровне, сложно объяснить машине. Изначально, человек рождается с минимальными знаниями об окружающей среде. В процессе познания окружающего мира, человек совершает действия, в ответ на которые окружающее его пространство генерирует поощрения или наказания. Так маленькому ребенку родители запрещают рисовать в комнате, но поощряют сладким, когда он съел весь ужин. В ХХ веке ученые, занимающиеся кибернетикой, сформулировали математическую модель обучения, взяв за основу модель обучения людей. Этот способ назвали обучение с подкреплением. В современном мире, эти модели способны обучаться делать то, что сложно сформулировать человеку. Например, ходьбе, игре в компьютерные игры, схватыванию предметов различной формы. В этой курсовой работе будет рассмотрен метод Q-learning, варианты его архитектуры, способы его применения, пример разработки и испытаний собственной системы управления роботом в задаче движения по траектории с обходом препятствий[1],[2].

Целью данной курсовой работы является исследование применения метода машинного обучения q-learning в задаче движения мобильного робота по траектории с объездом препятствия.

В рамках курсовой работы требуется рассмотреть и разработать программное обеспечение, реализующее глубинного обучения (deep q-learning) робота Kuka youBOT в среде симулирования V-REP задаче следованию робота в заданную точку с обходом препятствий.

# 1. Нейронные сети

В 50-х годах прошлого века ученые предложили модель искусственного нейрона вместе с моделью сети на основе искусственных нейронах. Модель сети была реализована в виде программы, она могла оперировать только с логическими нулями и единицами. Развитие нейронных сетей показало, что для расширения области их применения необходимо, чтобы нейрон мог работать не только с бинарными, но и с непрерывными (аналоговыми) сигналами. Такое обобщение модели нейрона было сделано Уидроу и Хоффом, которые предложили в качестве функции срабатывания нейрона использовать нелинейные функции.

Модель искусственного нейрона была списана с модели биологического нейрона (рисунок 1.1). Нейрон состоит из клеточного тела с ответвляющимися от него дендритами (получателями сигнала) и аксона - длинного отростка нейрона. Приспособленного для проведения возбуждения и информации от тела нейрона к нейрону или от нейрона к исполнительному органу.  Место передачи сигнала между двумя нейронами называется синапсом. Мозг человека состоит из 100 миллионов таких клеток, организованных в сеть. Клетки способны обрабатывать сигналы динамически во времени и образовывать новые связи.



Рисунок 1.1. Модель биологического нейрона

Модель искусственного нейрона представлена на рисунке 1.2.

Модель искусственного нейрона состоит из множества входов и одного выхода. Нейрон вычисляет взвешенную сумму входных данных[4]:

Далее значение поступает на функцию активации. Основные виды функции активации – это ступенчатая функция, сигнум функция, функция сигмоида, линейная функция и частично линейная функция. Значение выхода функции активации поступает на вход другим нейронам.

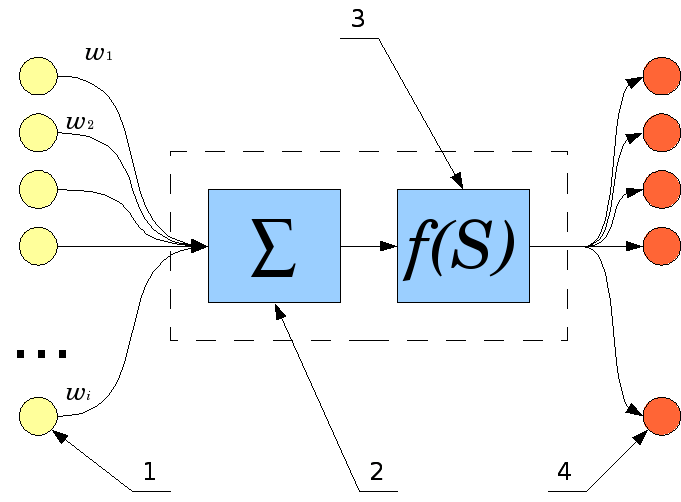


Рисунок 1.2. Строение искусственного нейрона

1 – входные сигналы, 2- сумматор, 3 – функция активации, 4 – выходной сигнал нейрона

Нейроны организованы в сеть, являющуюся взвешенным графом. Две наиболее распространенных архитектур нейронной сети – сеть прямого распространения и рекуррентные сети. Все нейроны в сети организованы в слои. Вход одного слоя «подключен» к входу другого слоя. То есть каждый нейрон одного слоя связан с каждым нейроном другого слоя.

Для обучения нейронной сети используется метод градиентного спуска. Он используется для нахождения локального минимума функции. Он последовательно, начиная с первого значения итеративно, спускается в отрицательном направлении градиента для достижения глобального минимума. Такая функция называется функцией потери. Принято, что функция потери *F(w)* с параметром *w* на каждой итерации обновляет веса связей в направлении, котором находится минимум функции *F*. Обновление весов происходит по формуле[4]:

Где скорость обучения. Она определяет с размер шага взятого для нахождения минимума функции.

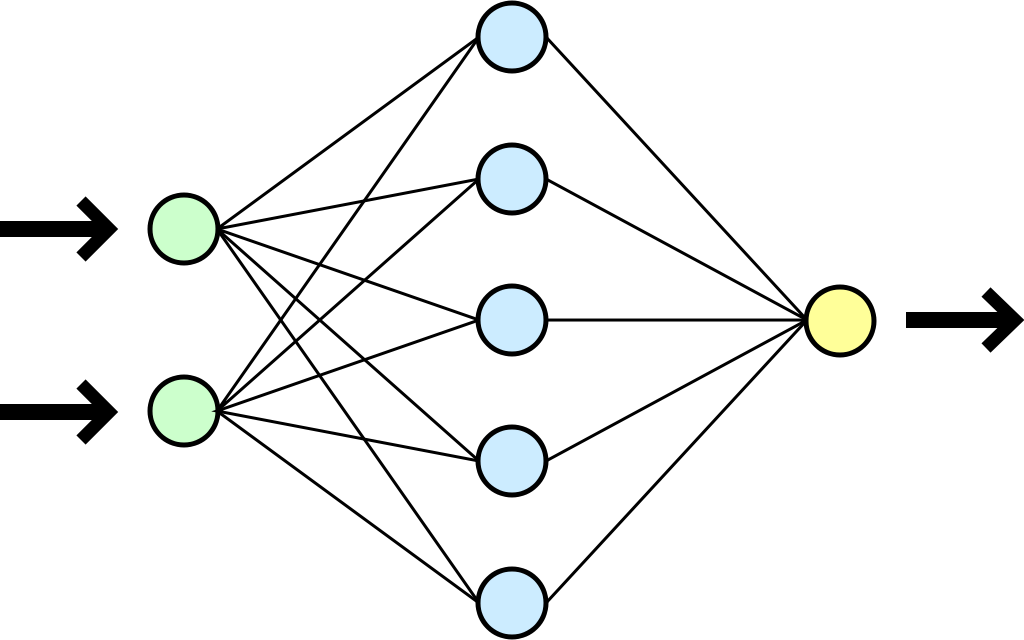


Рисунок 1.3. Иллюстрация взаимосвязей между слоями нейронной сети

# 2. Самообучение с подкреплением

Самообучение с подкреплением – область машинного обучения, в которой агент ищет оптимальное поведение путем взаимодействия с окружающей средой. Окружающей средой называют всё, что не является агентом.

Модель самообучения формально можно записать таким способом: во время *t* модель и окружающая среда находятся в состоянии *s ∈ S.* Состояние описывает актуальную информацию о текущей ситуации. Агент выбирает действие  *a ∈ A*. Значения *s* и *a* могут быть или дискретными или непрерывными. В следующий дискрет времени *t+1* агент получает вознаграждение *r ∈ R* и переходит в следующее состояние. Выбор действия в зависимости от состояния осуществляет функция стратегии π. Стратегия может быть детерминистической – когда для определенного состояния выбирается одинаковое действие, или вероятностное – действие выбирается с какой-то вероятностью.

В стратегическом плане агент стремиться максимизировать предположительный выигрыш, который рассчитывается, как сумма выигрышей[4]:

*Rt = rt+1 + rt+2 + … + rT*

Этот расчет подходит для моделей с конечным числом состояний. В обычных ситуациях *T* стремится к бесконечности. Тогда предыдущее выражение стремится к бесконечности. Для учета данного факта вводится скорость уменьшения γ[4]:

*Rt = rt+1 + γ rt+1  + +* … =



Рисунок 2.1. Модель обучения с подкреплением

## 2.1 Q-learning

Основой q-learning является *Q(s, a)* функция. Вводится понятие опыта, которое описывается как *(s, a, r, s’)*, в которой *s* – состояния агента на текущей итерации, *a* – совершенное действие, *r* – выигрыш, *s’* – новое состояние агента. Обновление Q-функции выполняется следующим образом[3]:

*Q(st, at) = Q(st, at) + α (r + γmaxa Q(st+1, at) – Q(st, at))*

Алгоритм Q-learning представлен в листинге 2.1.

Листинг 2.1. Алгоритм Q-learning

|  |
| --- |
| 1.Инициализация стратегии π1(a|s);  2.Инициализация состояния среды s1;  3.**цикл** t = 1,...,T,.. выполнять  4. агент выбирает действие согласно стратегии at ∼ πt(a|st);  5. среда генерирует вознаграждение rt+1 ∼ p(r|at, st) и новое состояние st+1 ∼ p(s|at, st);  6. Q(st, at) := Q(st, at) + αt(rt+1 + γ maxa Q(st+1, a ) − Q(st, at));  7. если состояние st – терминальное тогда игра останавливается;  8.**конец** условия конец цикла |

## 2.2 Глубинный q-learning

Существует множество способов аппроксимации Q-функции. Наиболее популярным способом является аппроксимация нейронной сетью. Впервые такой метод аппроксимации был использован для обучения нейронной сети игре на Atari-2600. В качестве нейронной сети была использована сверточная нейронная сеть. Опыты показали, что разработанную архитектуру НС можно применять для всех игр. В среднем, нейронная сеть играла лучше, чем люди в целом. Также с помощью глубинного обучения обучали роботов брать предметы разной формы. В перспективе применение глубинного Q-learning в задачах робототехники, финансах, системах управления и медицине.

В связи с тем, что аппроксимация происходит нейронной сетью, то был изменен алгоритм Q-learning (листинг 2.2)[4].

Листинг 2.2. Алгоритм глубинный q-learning

|  |
| --- |
| 1. Инициализация памяти D размером N  2. Инициализация весов нейронной сети  3. Инициализация стратегии π1(a|s);  4. Инициализировать количество циклов K, после которых происходит обучение нейронной сети  5. Инициализация состояния среды s1;  6. **цикл** t = 1,...,T,.. выполнять  7. агент выбирает действие согласно стратегии at ∼ πt(a|st);  8. среда генерирует вознаграждение rt+1 ∼ p(r|at, st) и новое состояние st+1 ∼ p(s|at, st);  9. Q(st, at) := Q(st, at) + αt(rt+1 + γ maxa Q(st+1, a ) − Q(st, at));  10. если состояние st – терминальное тогда игра останавливается;  11. Положить в N (st, at)  12 **Если** t % K равно 0, то обучить нейронную сеть на наборе из N  13.**конец** условия конец цикла |

# 3. Испытания

Для испытаний алгоритма был разработан программный комплекс, состоящий из симулятора V-REP и программы на Python. Симулятор V-REP был использован из-за того, что он достаточно прост в освоении, имеет большой набор моделей, в том числе роботов и датчиков, имеет API для удаленного управления симуляцией и роботом. Python был выбран из-за того, что данный язык позволяет быстро проектировать различные системы, обладает большой стандартной библиотекой, для него можно найти множество реализаций алгоритмов машинного обучения, включая нейронные сети. Его недостатком является его скорость.

Архитектура нейронной сети представлена на рисунке 3.1. Для аппроксимации значений ЛИДАРа его зону действия разбивают на 18 секторов, в каждом секторе находят точку с минимальным расстоянием до робота, составляется вектор из 18 таких значений (рисунок 3.2). На вход нейронной сети подается 20 значений. Первые 18 входов нейронной сети являются значением вектора, описанного выше. Остальные 2 входа заполняются значением дальности до целевой точки и значением пеленга на эту точку.

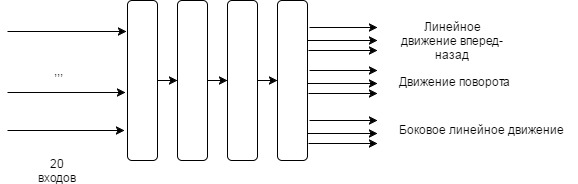


Рисунок 3.1 Архитектура нейронной сети

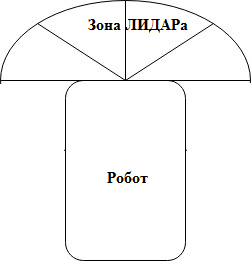


Рисунок 3.2. Пример деление зоны ЛИДАРа на сектора

На выходе нейронной сети получаются 3 Q функции. Первые три выхода – это значения Q функции линейного движения, следующие три – это значения Q функции поворота, последние три – значения Q функции бокового движения. Благодаря такому подходу, за один такт робот может совершить три разных движения.

Изначально нейронную сеть тренировали в сцене с большим количеством препятствий (рисунок 3.3). Вознаграждение учитывало расстояние до цели, пеленг на неё и близость к стенам. Первые версии нейронных сетей имели разные архитектуры, но на выходе всегда были нейроны с линейной функцией активации. Скрытые слои имели разные функции активации типа сигмоида и relu. На практике данная архитектура показала не очень впечатляющие результаты: после 100000 итерации обучения робот ехал прямиком на стены и препятствия, не определял направление до препятствия, его действия носили одинаковый характер: он или кружился на месте или ехал в одну сторону. После анализа поведения робота, по прошествии 100000 итераций было решено изменить стратегию обучения робота и поменять архитектуру НС.

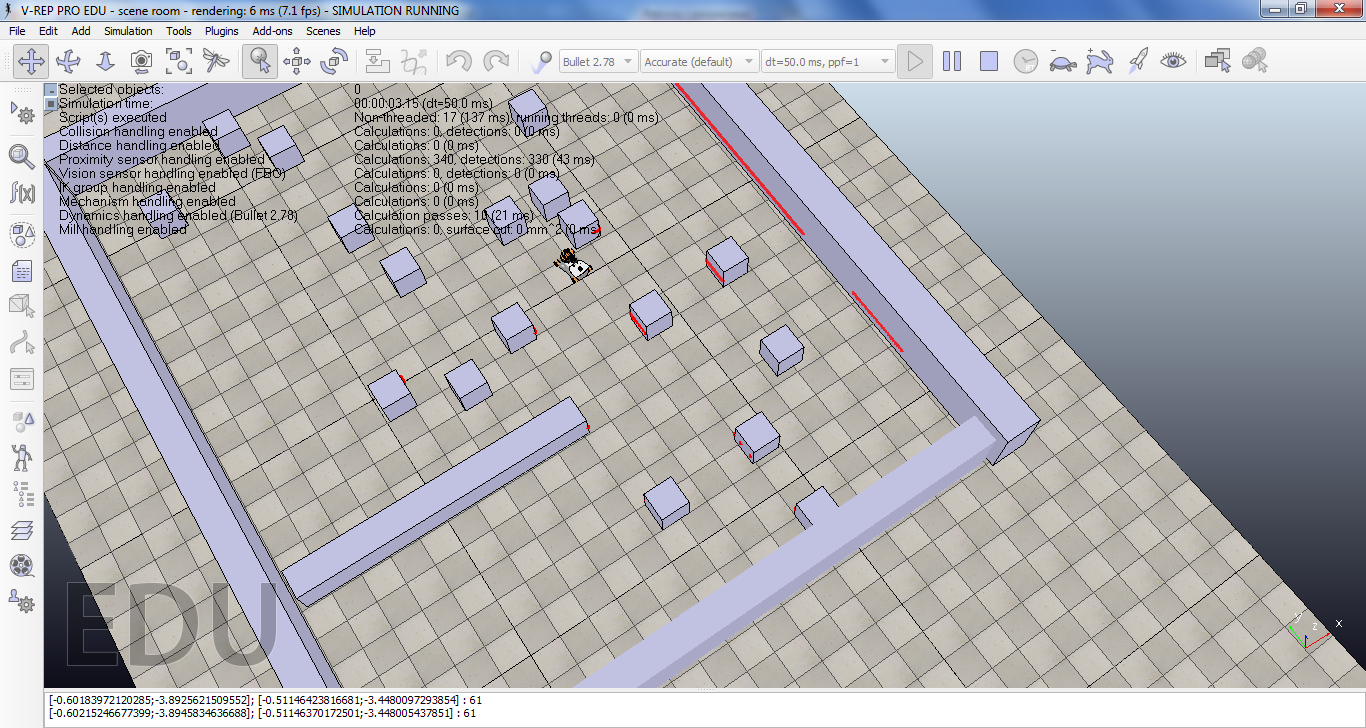


Рисунок 3.3. Фрагмент моделирование робота KUKA youBOT во V-REP

Новая архитектура имела следующий вид: она состояла из 4х слоев, 1-й – входной, 2-й – линейный с 18 нейронами, 3-й – relu с 15 нейронами, 4-й – линейный с 9 выходными нейронами. Правила учитывали только время, затраченное на движение к точке и столкновение с препятствием. Такое вознаграждение должно побудить робота находить наикратчайшую траекторию до точки с учетом обхода препятствий. Все препятсвтия со сцены были убраны, оставить только четыре стены (рисунок 3.4 и рисунок 3.5). На данный момент обучение составляет порядка десятков тысяч итерации, что не дает право судить о правильности выбора архитектуры. Но, впоследствии, после успешного обучения нейронной сети в простой сцене, без препятствий следует осуществить обучения на сцене с препятствиями. После чего можно утверждать, что нейронная сеть обучилась движению по траектории с обходом препятствий.

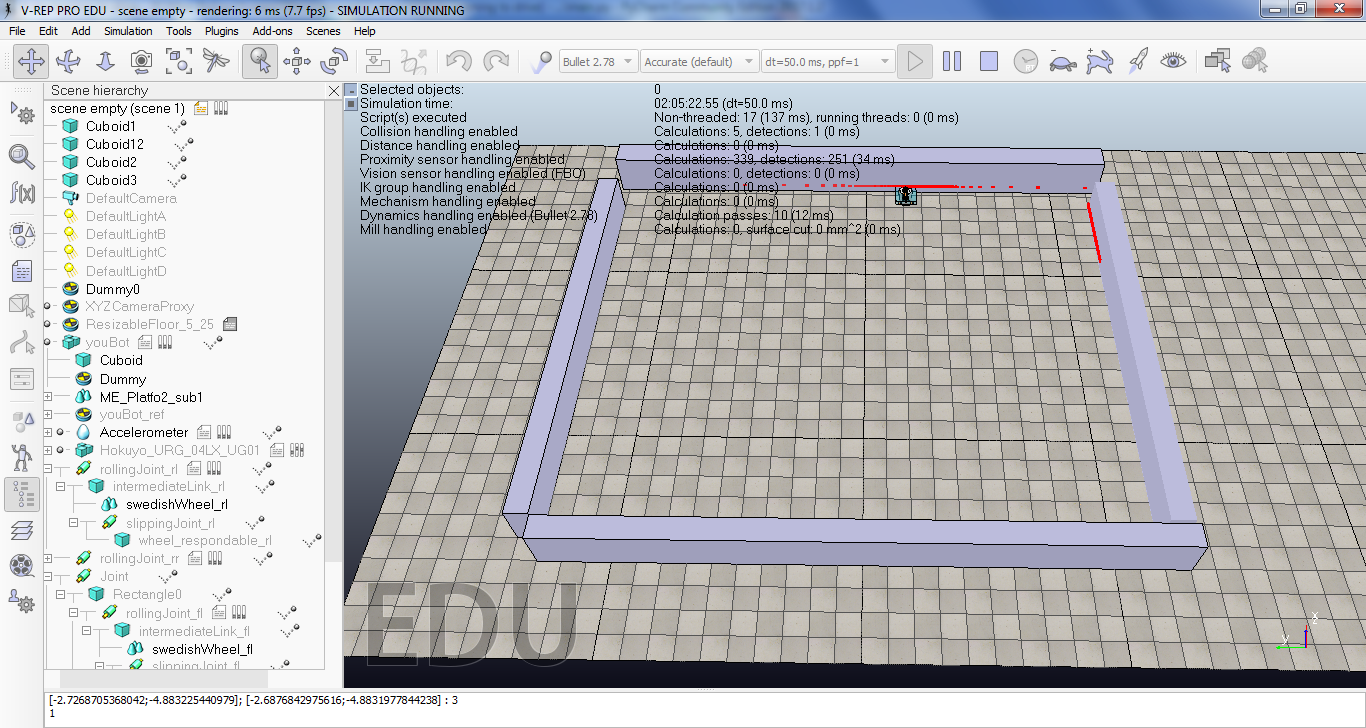


Рисунок 3.4. Фрагмент моделирование робота KUKA youBOT во V-REP

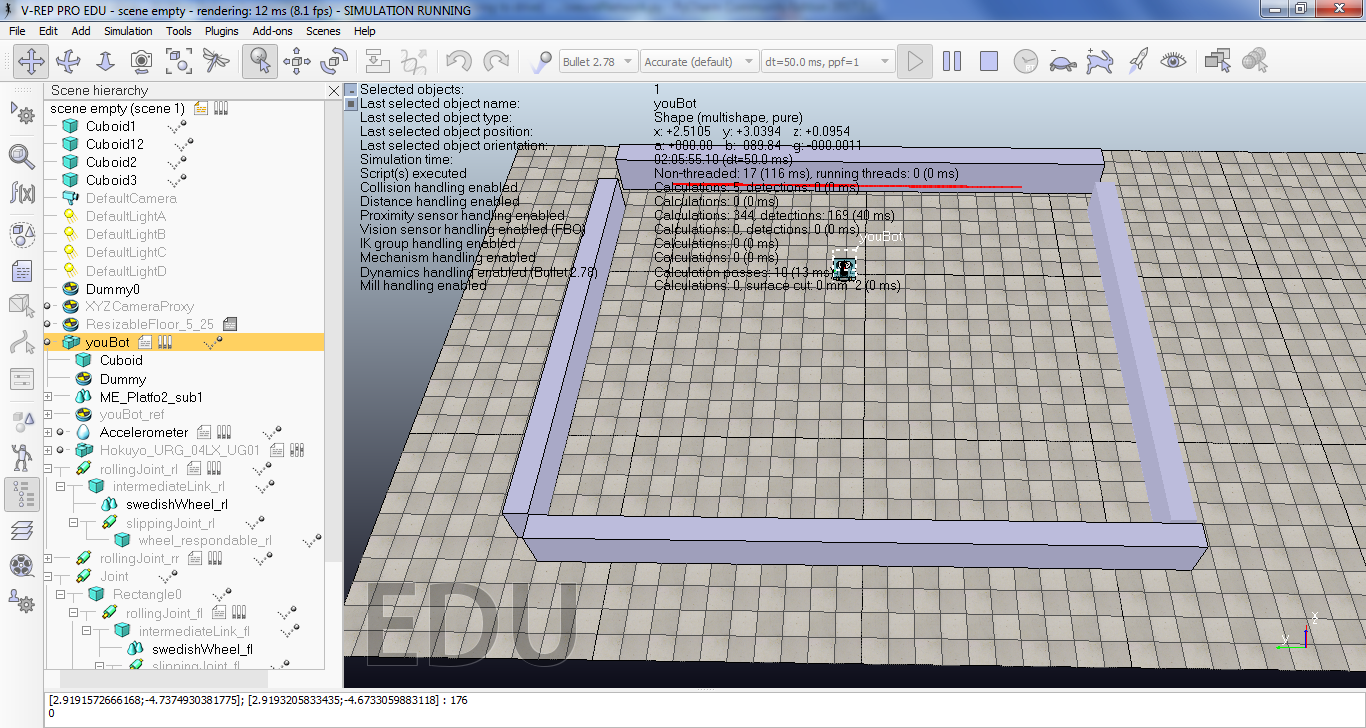


Рисунок 3.5. Фрагмент моделирование робота KUKA youBOT во V-REP

# Вывод

Обучение с подкреплением достаточно перспективная область машинного обучения. В тех задачах, где цель трудно сформулировать с помощью формул, решение можно отыскать с помощью использование Q-learning, а если для аппроксимации использовать нейронные сети, то можно ожидать хорошего результата обучения.

В ходе исследования алгоритма были выявлены следующие проблемы глубинного Q-обучения:

1. Для обучения требуется большое количество времени
2. Время обучения зависит от архитектуры нейронной сети
3. Для ускорения обучения нужно проводить обучение на высокопроизводительных компьютерах
4. Лучше проводить параллельное обучение нескольких типов архитектур нейронных сетей, так как шанс того, что именно архитектура данного типа лучше решит задачу, достаточно мал

На данный момент текущая модель обучается успешнее предыдущих моделей. Но для полной уверенности стоит подождать до 1000000 итерации.

# Список литературы

1. Bruno Siciliano “Handbook of robotics”/ Bruno Siciliano, Oussama Khatib (Eds.) / Springer – 1628p
2. Roland Illah R., SIEGWART NOURBAKHSH «Introduction to Autonomous Mobile Robots» [Электронный ресурс] / Roland Illah R., SIEGWART NOURBAKHSH MIT Press – Режим доступа:  <http://home.deib.polimi.it/gini/robot/docs/siegwart.pdf>
3. Richard S. Sutton and Andrew G. Barto «Reinforcement Learning» [Электронный ресурс] / Москва БИНОМ. Лаборатория знаний 2014 – Режим доступа:  http://files.pilotlz.ru/pdf/cC2500-9-ch.pdf
4. Stephen JAMES «3D Simulated Robot Manipulation Using Deep Reinforcement Learning» [Электронный ресурс] / – Режим доступа: https://arxiv.org/pdf/1609.03759.pdf