Манипуляторы с управлением на основе машинного обучения с подкреплением

Али Сажи Маннаа [1], Андрей О. Зарубин [2]
ФГАОУ ВО Южный федеральный университет, г. Ростов-на-Дону – Таганрог
[1]ali88mannaa@gmail.com, [2]zarubin@sfedu.ru

***Аннотация*. Эта индустрия создана для ряда задач, в которых нет однозначного алгоритма решения. Соответственно, эти задачи не могут быть решены полностью автоматизированным образом. К таким задачам относится сварка стали с нечётким сварным швом, раскройка леса, сортировка, упаковка, конвейерные задачи с переменным окружением. Все эти задачи на данный момент решаются человеком. В результате применения алгоритмов можно достичь результатов вплоть до 70 % экономии на отладке умных манипуляторов и на задачах автоматизации для всей индустрии 4.0.**

Размер этого рынка, т. е. рынка роботов-манипуляторов составляет примерно 35 миллиардов долларов в мировом исчислении, и в частности в России, с невысоким уровнем автоматизации, 15 миллиардов рублей. Целевая аудитория – это часть производств, которые внедряют у себя автоматизацию, роботизацию и, соответственно, они занимаются обработкой деталей. Бизнес модель, по которой мы развиваемся, это создание аппаратного комплекса, чтобы производить апгрейд оборудования заказчиков.

Для обучения использован алгоритм DDPG. Само обучение производится в двух вариантах – с использованием HER (Hindsight Experience Replay) и без. Результаты сравниваются, и для работы берется лучшая модель.

Ключевые слова: алгоритм DDPG, машинное обучение с подкреплением, сварка стали, автоматизация, роботизация

I. Введение

Роботы стали неотъемлемой частью нашего современного мира, выполняя задачи, которые варьируются от простых бытовых дел до сложных промышленных операций. Эти интеллектуальные машины способны автономно взаимодействовать с окружающей средой и принимать решения на основе получаемой информации. Одной из ключевых технологий, позволяющих роботам учиться и адаптироваться к своей среде, является обучение с подкреплением.

Обучение с подкреплением – это отрасль искусственного интеллекта, которая фокусируется на обучении роботов или агентов принимать решения на основе проб и ошибок. Она черпает вдохновение из поведенческой психологии, где положительное и отрицательное подкрепление используются для формирования и изменения поведения. В контексте робототехники обучение с подкреплением включает предоставление обратной связи роботу на основе результатов его действий, стимулируя его учиться и улучшаться со временем.

Процесс обучения с подкреплением начинается с взаимодействия агента с окружающей средой. Агент совершает действия, и среда реагирует наградами или штрафами на основе результатов. Через повторные взаимодействия агент учится ассоциировать определенные действия с положительными наградами и другие с отрицательными штрафами. Этот процесс обучения осуществляется алгоритмами, которые оптимизируют стратегии принятия решений агента, стремясь максимизировать накопленные награды со временем.

Шаблон помогает отформатировать статью и придать единообразие всему изданию. Поля, ширина колонок, межстрочное расстояние, шрифты (соответствующие используемому языку), размер шрифтов заданы и, пожалуйста, не меняйте их. Все стили отображаются в коллекции экспресс-стилей.

Обучение с подкреплением революционизировало робототехнику, позволяя машинам приобретать новые навыки и адаптироваться к изменяющимся обстоятельствам без явного программирования. Оно успешно применяется в различных областях, включая автономные транспортные средства, робототехническое манипулирование и даже игровых роботов. Позволяя роботам учиться на опыте, обучение с подкреплением открывает возможности для более эффективных и адаптивных робототехнических систем.

В заключение, обучение с подкреплением играет важную роль в развитии интеллектуальных роботов. Оно дает им возможность учиться на своих взаимодействиях с окружающей средой и принимать обоснованные решения на основе прошлого опыта. По мере развития технологий мы можем ожидать дальнейшего развития методов обучения с подкреплением, что приведет к еще более способным и автономным роботам в будущем.

II. Обзор литературы

Обучение с подкреплением – это метод машинного обучения, который позволяет агенту (в данном случае роботу) учиться и адаптироваться к своей среде путем проб и ошибок. В процессе обучения агент взаимодействует со средой, выполняя определенные действия, и получает положительные или отрицательные вознаграждения в зависимости от результатов своих действий.

Процесс обучения с подкреплением состоит из нескольких основных компонентов. Во-первых, есть агент, который принимает решения и выполняет действия. Во-вторых, есть среда, в которой агент действует. Среда может быть физической (например, роботическая платформа) или виртуальной (например, симулятор). В-третьих, есть наблюдение, которое агент

получает от среды. Наблюдение может быть полным или частичным, в зависимости от того, какая информация доступна агенту. Наконец, есть вознаграждение, которое агент получает от среды в ответ на свои действия. Цель агента – максимизировать суммарное вознаграждение, которое он получает в процессе взаимодействия со средой [1].

Обучение с подкреплением широко применяется в робототехнике. Например, в области автономных транспортных средств агент (автомобиль) может учиться ездить безопасно и эффективно на основе вознаграждений, связанных с правильным выполнением задачи и избеганием аварий. В области робототехники манипуляции роботы могут использовать обучение с подкреплением для научиться выполнять сложные задачи, такие как сборка или перемещение предметов [3].

Обучение с подкреплением имеет огромный потенциал в развитии интеллектуальных роботов. С его помощью роботы могут учиться на основе своего опыта и принимать информированные решения на основе накопленных знаний. В будущем ожидается дальнейшее развитие методов обучения с подкреплением, что позволит создавать еще более способных и автономных роботов [2].

Единственное улучшение, которое можно было бы внести, – это предоставить более конкретные примеры того, как обучение с подкреплением используется в робототехнике, такие как конкретные задачи или приложения. Это помогло бы еще больше проиллюстрировать практичность и универсальность этого подхода [4].

III. Алгоритм обучения (DDPG, Unity, Python)

Одной из проблем политического представительства в робототехнике является поиск подходящего представления, которое может эффективно охватить сложность действий робота и окружающей среды, в которой он работает. Политика в основном является отображением состояний на действия, и она должна хорошо обобщаться на непредвиденные ситуации.

Еще одной проблемой является работа с высокоразмерными пространствами состояний и действий. Роботы часто работают в средах с большим числом возможных состояний и действий, что может затруднить обучение эффективной политике. Техники, такие как аппроксимация функций и глубокое обучение, применяются для решения этой проблемы.

Безопасность и надежность – это еще одна важная проблема в политическом представительстве для робототехники. Роботы должны уметь справляться с непредвиденными ситуациями и исправлять ошибки. Представление политики должно уметь учитывать ограничения безопасности и обрабатывать неопределенности в окружающей среде.

Наконец, возникает проблема исследования и использования в политическом представительстве. Роботы должны исследовать свою среду, чтобы открывать новые стратегии и учиться на своих ошибках, одновременно используя свои текущие знания для максимизации вознаграждений. Нахождение баланса между исследованием и использованием является ключевым для эффективного обучения.

В целом, политическое представительство в робототехнике сталкивается с проблемами, связанными с сложностью, размерностью, безопасностью и исследованием. Преодоление этих проблем является необходимым для разработки интеллектуальных и адаптивных роботов.

- Роборуки в Unity и реальном мире ничего не знают друг о друге. В реальном мире центр роборуки смещён, а также смещены нулевые положения серв. Считаем ниже, что сервы могут двигаться в диапазоне [-$angle_{max}$, $angle_{max}$]. На рис. 1 представлены

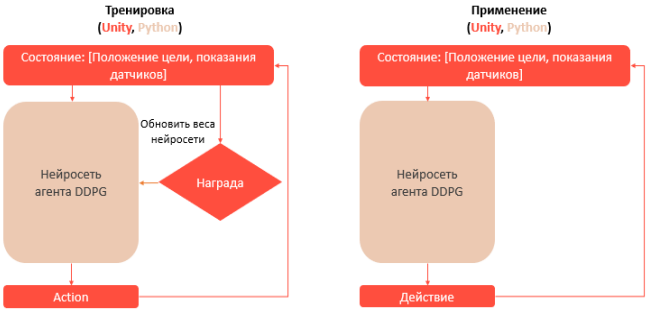

Рис. 1.

- $\mathbf{R}_t = [x_t, y_t, z_t]$ - цель
- $\mathbf{R}_{end} = [x_{end}, y_{end}, z_{end}]$ – точка захвата роборуки
- $\mathbf{A} = \{a_1, a_2, a_3\}$ – углы сочленений роборуки
- Цель = $\mathbf{R}_t$, Показания датчиков = {$\mathbf{R}_{end}$ - $\mathbf{R}_t$, $\mathbf{R}_{end}$, **A**}
- Действие = **A**

1. Привязать систему координат к роборуке (*f1*)

- Вычисленные координаты конца роборуки по углам поворотов сочленений в реальном мире должны соответствовать координатам конца роборуки в Unity. Чтобы рассчитать координаты конца роборуки надо как-то зафиксировать систему координат. Сделаем это так: начало координат поместим в центр основания роборуки, за ось oz примем вектор, перпендикулярный основанию и проходящий через центр основания. За ось ox выберем ось, проходящую через центр основания вдоль угла поворота сервы основания $a_1 = 0$. Ось oy определим векторным произведением **oy** = [**oz** x **ox**]. В Unity надо оси задать точно также. Зная расположение осей мы вычисляем по заданным $A = \{a_1, a_2, a_3\}$ координаты конца роборуки по формуле $\mathbf{R}_{end} = f1(a_1, a_2, a_3)$ и проверяем себя по значениям, которые выводит Unity.

2. Откалибровать углы поворота серв (*f2*)

- В формуле $\mathbf{R}_{end} = f1(a_1, a_2, a_3)$ важно в реальном мире проверить соответствие начальных положений углов серв с Unity. Удобно в Unity за 0 принять вертикальную ориентацию всех сочленений (вдоль oz, например). В реальном мире выставляем A={0,0,0} и смотрим, насколько отклоняются сервы от вертикального положения. Если отклоняются, то при подаче на вход нейросети надо выполнить преобразование углов поворота: $A_{for\ NN} = f2(A)$ – прибавляем к углу, выдаваемому мотором сервы измеренный транспортиром угол отклонения от вертикальной

оси с учётом знака, когда мы вручную выставили 0.

3. Привести координаты к единому масштабу

- На вход нейросети надо подавать нормированные значения координат. Делим $\mathbf{R}_{end}$ на общую длину сочленений роборуки (как при тренировке в Unity, так и в реальном мире) $\mathbf{R}_{end} = \mathbf{R}_{end}/scale$.

4. Преобразовать распознаваемые координаты цели ($\mathbf{R}_t$) в систему координат роборуки и отмасштабировать (*f3*)

- В реальном мире координаты цели мы получаем с помощью распознавания изображения с камеры. У камеры есть своя система координат и масштаб. Поэтому оси ox и oy должны быть наклеены на корпус роборуки, чтобы камера распознала направление и масштаб (размер меток может быть равен, например, 0.1 общей длины сочленений). Две оси нужны, чтобы если рука закрывала собой одну ось, камера могла видеть вторую.
- **R**t from camera – координаты цели, которые распознала камера
- (x1, y1), (x2, y2) – координаты двух точек на корпусе роборуки, которые распознала камера. По этим точкам находим координаты центра роборуки [(x1+x2)/2, (y1+y2)/2], направление оси ox внутренней системы координат роборуки [x2-x1, y2-y1], ось oy через векторное произведение и масштаб. $\mathbf{R}_t$ = *f3* ($\mathbf{R}_t$ from camera) – то, что надо подавать на вход нейросети.

5. Последовательность отладки багов

Работаем во внутренней системе координат роборуки (камера не нужна):

- проверяем, чтобы подаваемые углы поворота серв приводили к одинаковым наклонам в юнити и реальном мире (правильная калибровка нуля отсчёта углов)
- проверяем, чтобы по заданным углам поворота координаты конца роборуки в юнити и рассчитанные по формуле (пункты 1–3) были одинаковыми
- для натренированного алгоритма задаём в юнити цель, вычисляем её нормированные координаты, подаём эти координаты в качестве цели в алгоритм, как будто распознали камерой (при этом надо умножить их на общую длину сочленений реальной руки). Проверяем совпадение наклонов сочленений в юнити и реальном мире.

Работаем с внешней системой координат камеры:

- ставим цель, распознаём её координаты, выполняем преобразование п.4 и проверяем (для начала достаточно примерно), чтобы они соответствовали системе координат роборуки (масштаб и ориентация, которые мы нанести на корпус!)
- комбинируем распознавание координат и DDPG алгоритм. Результат показан на рис. 2.

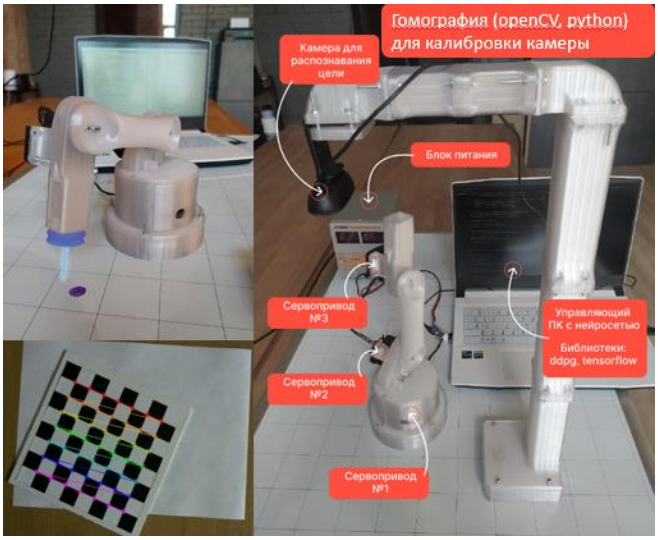

Рис. 2.

IV. Заключение

Мы используем алгоритм обучение с подкреплением. Это когда создаётся виртуальный двойник оборудования, которое мы хотим обучить. Мы полностью воссоздаём эту задачу в виртуальной среде, а дальше создаём модель машинного обучения, в которой присутствуют агент, среда и награда. Таким образом, когда наш двойник успешно выполняет задачу, он получает награду. Получает подкреплением и тренирует свой собственный алгоритм и такое обучение происходит по кругу до достижения определённой точности.

В результате, мы достаём этот алгоритм из виртуальной среды и применяем к реальному оборудованию и он успешно выполняет задачу.

Список литературы

[1] Sutton R.S., Barto A.G. Reinforcement Learning: An Introduction; MIT Press: Cambridge, MA, USA, 1998.

[2] Pastor P., Kalakrishnan M., Chitta S., Theodorou E., Schaal S. Skill Learning and Task Outcome Prediction for Manipulation. In Proceedings of the International Conference on Robotics and Automation (ICRA), Shanghai, China, 9–13 May 2011; pp. 3828–3834.

[3] Rosenstein M.T., Barto A.G., van Emmerik R.E.A. Learning at the level of synergies for a robot weightlifter. Robot. Auton. Syst. 2006, 54, 706–717.

[4] Kormushev P., Calinon S., Caldwell D.G. Robot Motor Skill Coordination with EM-Based Reinforcement Learning. In Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Taipei, Taiwan, 18–22 October 2010; pp. 3232–3237.