Повышение качества образования путем комбинирования дисциплин: «робототехника» и «искусственный интеллект»

Федосин Сергей Алексеевич
профессор, к.т.н., заведующий кафедрой автоматизированных систем обработки информации и управления
Национальный исследовательский
Мордовский государственный университет им. Н. П. Огарева
ул. Большевистская, 68, г. Саранск, 430005, +7(8342)270256
fedosinsa@mrsu.ru

Плотникова Наталья Павловна
к.т.н., старший преподаватель кафедры автоматизированных систем обработки информации и управления,
Национальный исследовательский
Мордовский государственный университет им. Н. П. Огарева
ул. Большевистская, 68, г. Саранск, 430005, +7(8342)270256
linsierra@yandex.ru

Немчинова Елена Андреевна
студентка направления «Информатика и вычислительная техника»
Института электроники и светотехники
Национальный исследовательский
Мордовский государственный университет имени Н.П. Огарёва
ул. Б.Хмельницкого, 39, г. Саранск, 430005, (8342)478691
nemchinova.len@yandex.ru

Аннотация

В статье рассматривается способ организации учебной деятельности студента путем комбинирования двух дисциплин: «Робототехника» и «Искусственный интеллект». Описывается методология реализации проекта по созданию робота-гексапода, перемещением которого управляет искусственный интеллект на базе нейронной сети. Описывается построение робота, выбор архитектуры нейронной сети и алгоритма ее обучения, способ взаимодействия робота и нейронной сети. Говорится об эффективности применения методики комбинирования дисциплин.

The article discusses a method of organizing of student learning activities by combining two disciplines: Robotics and Artificial Intelligence. The methodology of a project to construct a hexapod robot, the movement of which is controlled by artificial intelligence based on a neural network are shown. The construction of the robot itself, the architecture of the neural network and the algorithm of its training, the way of the interaction of the robot and the neural network are described. It is talking about the effectiveness of applying the methodology of disciplines combination.

Ключевые слова

Комбинирование дисциплин, робототехника, нейронная сеть, робот-гексапод, Arduino Uno, Raspberry Pi

Combination of disciplines, robotics, neural network, robot hexapod, Arduino Uno, Raspberry Pi

Введение

Неотъемлемой частью изучения любой технической дисциплины является выполнение практических и лабораторных работ. Без практической основы нельзя говорить о глубоком погружении в выбранную область. Однако применение теоретических знаний часто требует не только хорошего ориентирования в изучаемом материале, но и компетентности студента в других областях. Чтобы студент понимал, где и каким образом применяются полученные теоретические материалы в реальной жизни, предлагается метод изучения дисциплин путем их комбинирования.

Рассмотрим такие дисциплины, как «Робототехника» и «Искусственный интеллект». На данный момент это две крупные, быстро развивающиеся сферы деятельности, которые все чаще удивляют мир новейшими технологиями и высокотехнологичными продуктами. В последнее время можно заметить тесное взаимодействие этих областей на примерах роботов, ориентирующихся в пространстве, автопилотов электрокаров и тому подобных проектов. Чтобы идти в ногу со временем, студентам можно предложить в качестве лабораторных работ выполнение проекта на стыке двух дисциплин.

Первоначально студент должен получить базовые теоретические знания по каждой из этих дисциплин. Далее следует получение практических навыков на основе теории внутри каждой дисциплины. Например, для искусственного интеллекта это будет написание алгоритмов работы и обучения искусственной нейронной сети, для робототехники – конструирование простейших роботов, снабжение их управляющим программным кодом. Следующим этапом станет практическая реализация взаимодействия между дисциплинами. В данной статье будет рассмотрен методология создания робота, управляемого искусственным интеллектом на базе нейронной сети.

Процесс моделирования и конструирования робота

В качестве модели рекомендуется использовать робота-гексапода, состоящего из туловища и шести конечностей, прикрепленных к нему (рис. 1). Он имеет простую конструкцию, его детали легко можно найти в интернет-магазинах или сделать самостоятельно. Внешне робот напоминает паука с шестью ножками [1].

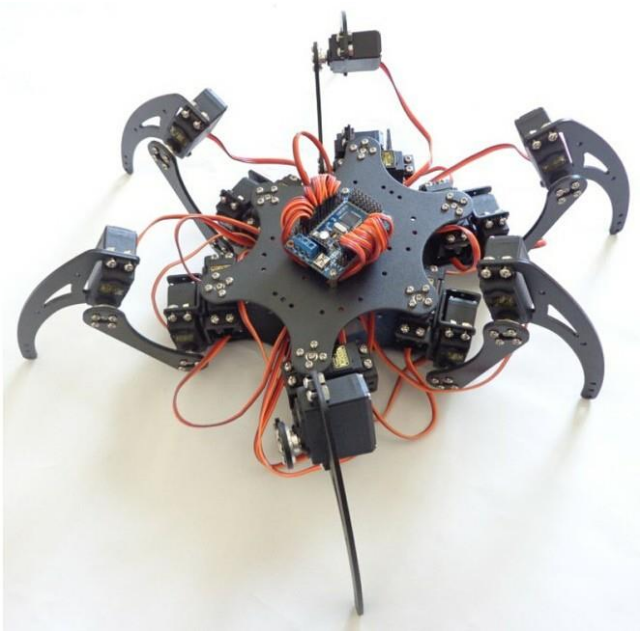

Рис. 1. Пример робота-гексапода

Тело конструируемого робота при этом состоит из двух параллельных пластин (верхней и нижней), на которых расположены отверстия и выступы для крепления платы и ног. Форма конечностей гексапода заимствована из биологии, поэтому для упрощения идентификации нужной части будет использоваться соответствующая терминология (рис. 2).

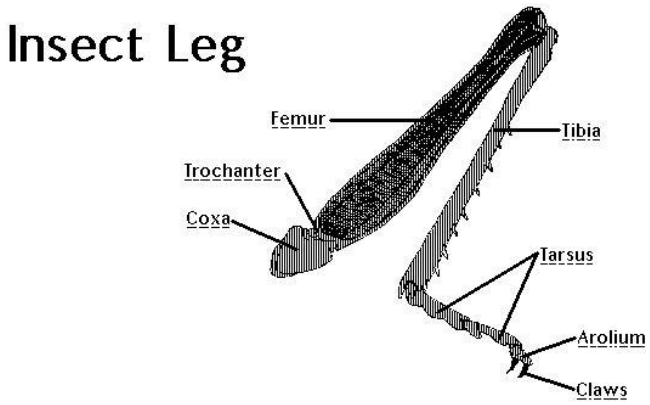

Рис. 2. Анатомия ноги насекомого

Конструкция конечности модели включает тазобедренный сустав (Coxa), бедро (Femur) и сустав, в котором оно совершает поворот (Trochanter), а также голень (Tibia) со своим суставом движения. Каждый сустав в модели представляется отдельным сервоприводом [2].

Конечности паука имеют три степени свободы для реализации возможности более адаптированного перемещения в трехмерном пространстве [3]. Движения совершаются в трех точках конечности. В суставе Coxa сервопривод поворачивает всю ногу таким образом, что любая ее точка перемещается по окружности, лежащей в плоскости, параллельной пластинам корпуса. Сустав Trochanter связан с бедром Femur и будет поднимать и опускать его. Вместе с ним, как единая система, перемещается и

часть Tibia. В момент работы сустава Trochanter в теории она считается неподвижной, жестко связанной с Femur. Однако на практике можно заметить, что действия могут совершаться параллельно, и Tibia будет поворачивается в соответствующем суставе вместе с работой остальных суставов конечности. Сустав голени Tibia вращает ее, перемещая вверх и вниз [4, 5].

Рекомендации по выбору комплектующих робота-гексапода

Материнская плата

Основным элементом является материнская плата. В настоящее время одной из самых популярных плат для подобных задач является Arduino. Будем использовать Arduino Uno. Она выполнена на базе процессора ATmega328p с тактовой частотой 16 МГц, обладает памятью 32 КБ и имеет 20 контролируемых контактов ввода и вывода для взаимодействия с внешним миром. Arduino является открытой платформой, состоящей из аппаратной и программной частей. Для программирования используется упрощенная версия языка C++. Разработку можно вести как в бесплатной среде Arduino IDE, так и с помощью произвольного инструментария C/C++. Программирование и общение платы с компьютером осуществляется через USB-кабель [6].

Сервоприводы

Так как корпус гексапода металлический, то вся конструкция получается достаточно тяжелой, поэтому необходимо, чтобы сервоприводы смогли выдержать создаваемую нагрузку. Подходящим сервоприводом может быть TowerPro MG996R. Механизм сервопривода выполнен из металла. Диапазон вращения равен 180 градусам, что достаточно для поставленной задачи. Для шестиногого гексапода необходимо 18 сервоприводов для обеспечения работы суставов каждой из конечностей [6].

Для удобства подключения сервоприводов и их корректной работы на материнскую плату обычно устанавливается дополнительная плата расширения. Для данного случая можно использовать Multiservo Shield, подходящий для многосуставных роботов. Он дает возможность управлять 18 сервоприводами. На плате расширения установлен отдельный микроконтроллер ATmega48, все силы которого направлены на то, чтобы аккуратно и точно в нужное время передавать управляющие сигналы на подключенные сервоприводы. Это позволяет избежать подергиваний приводов в произвольные моменты времени, как это происходит при использовании стандартной библиотеки Servo. В дополнение к 18 выводам для сервоприводов, управляемым выделенным микроконтроллером, на плату также вынесены 6 выводов Arduino напрямую. Таким образом, возможное количество сервоприводов в вашем устройстве может достигать 24 штук. Шилд имеет разъемы для своего собственного питания. Если питание не подведено, то он работает от самой платы Arduino [6]. Пример подключение сервоприводов к Multiservo Shield представлен на рисунке 3.

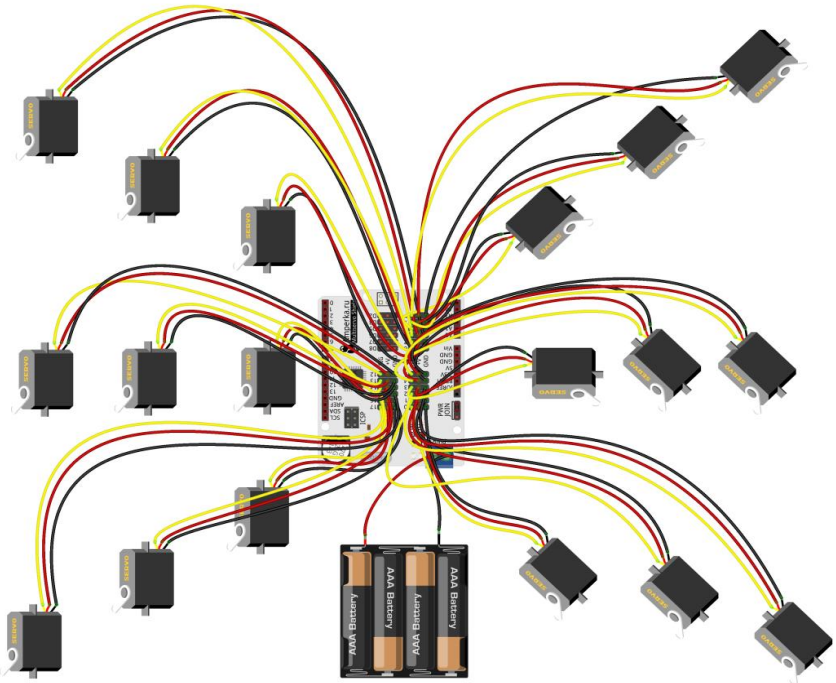

Рис. 3. Пример подключения сервоприводов гексапода к Multiservo Shield

Методология разработки архитектуры нейронной сети

После того, как робот был сконструирован, необходимо реализовать управляющую систему – мозг робота. В качестве такой системы будет выступать искусственная нейронная сеть. Перед ней стоит задача обучиться таким образом, чтобы гексапод из исходного состояния определил направления движения и прошел как можно дальше в этом направлении от своей исходной позиции. Нейронная сеть должна в каждый момент времени отправлять управляющие сигналы роботу, “говоря” ему, куда и в каком направлении он должен повернуть конкретную часть конкретной конечности.

Для поставленной задачи подходит искусственная нейронная сеть, имеющая архитектуру многослойного персептрона [7]. Количество скрытых слоев сети можно выбрать равны двум, что является простым, но эффективным вариантом для подобных задач (рис. 4).

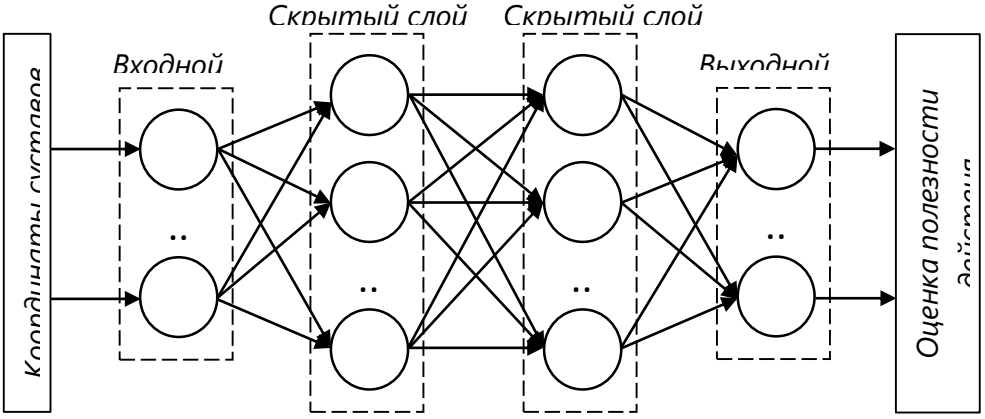

Рис. 4. Пример архитектуры искусственной нейронной сети

Существует несколько подходов для однозначного описания состояние робота в каждый момент времени. Это описание подается на вход нейронной сети для анализа и предсказания следующего действия. Один из способов – помещение робота в некоторую систему координат. В этом случае гексапод описывается координатами точек его суставов в трехмерном пространстве. Каждой точке соответствует три значения – координаты по оси абсцисс, ординат и аппликат [8]. Данный способ может иметь некоторые погрешности в рассчитываемых значениях, что связано с влиянием на робота сторонних сил, но он не требует подключения дополнительных устройств считывания и контроля положения робота. Положение системы координат может быть выбрано произвольно. Удобнее всего считать, что робот начинает свое движение от начала координат. После совершения каждого движения, поворота части конечности на заданный угол, координаты гексапода пересчитываются и характеризуют уже обновленное положение паука в пространстве. Таким образом, на вход нейронной сети будут поданы координаты точек суставов робота, однозначно характеризующие его положение в пространстве в каждый момент времени [9, 10].

Рекомендуемый способ управления роботом с помощью нейронной сети состоит в следующем. В каждый момент времени она на основе опыта предыдущих действий делает предположение о том, какое элементарное действие нужно совершить на текущем шаге. Под элементарным действием предполагается поворот части конечности на некоторый угол. Однако нейронная сеть не может повернуть сервопривод на произвольный угол. Для решения данной проблемы вводится понятие состояния части конечности. Диапазон ее поворота разбивается на несколько равных углов, ограниченных лучами. Положение части конечности или, в случае сустава Coxa, проекции конечности при совпадении с одним из этих лучей и будет называться состоянием. В этом случае поворот будет рассматриваться как переход из одного состояния в другое, а нейронная сеть будет давать оценку выгодности этого перехода. После совершения очередного действия проверяется устойчивость модели в новом состоянии, при необходимости изменяется ее положение с учетом падения. Когда вычислено итоговое положение модели в пространстве, данные могут быть переданы на вход нейронной сети для генерации следующего действия. Таким образом, результатом работы нейронной сети является оценка полезности совершения моделью каждого допустимого действия. На нейроны выходного слоя поступает информация, соответствующая элементарному действию частей конечностей.

Методические рекомендации по выбор алгоритма обучения искусственной нейронной сети

Для обучения искусственной нейронной сети потребуется комбинация двух алгоритмов. В качестве первого можно использовать алгоритм обратного распространения ошибки, RProp или RMS Propagation, в качестве второго — алгоритмом Q-Learning [11].

Алгоритм обратного распространения ошибки

Этот алгоритм представляет собой метод вычисления градиента, который используется при обновлении весовых коэффициентов многослойного персептрона. Основная идея этого метода состоит в распространении сигналов ошибки от выходов сети к её входам, в направлении, обратном прямому распространению сигналов в обычном режиме работы [12].

Resilient Back Propagation

RProp (Resilient Back Propagation) является эвристическим алгоритм обучения, в котором для корректировки весов и смещений сети используются знаки значений градиентов текущей и предыдущей эпох [13].

Приращение весовых коэффициентов вычисляется на каждом шаге следующим образом:

$$\Delta_{ij} = \begin{cases} \min\left(\Delta_{ij}(t-1)\cdot\eta^{+}, \Delta_{max}\right), \text{если } \frac{\partial E}{\partial w_{ij}}(t-1)\cdot\frac{\partial E}{\partial w_{ij}}(t) > 0 \\ \min\left(\Delta_{ij}(t-1)\cdot\eta^{-}, \Delta_{min}\right), \text{если } \frac{\partial E}{\partial w_{ij}}(t-1)\cdot\frac{\partial E}{\partial w_{ij}}(t) < 0 \end{cases}, \quad (1)$$

где t — значение веса на текущем шаге;

E — значение веса на предыдущем шаге;

$\frac{\partial E}{\partial w_{ij}}$ — значение скорости обучения;

$sign$ – значение градиента на текущем шаге.

RMS Propagation

RMS Propagation (Root Mean Square Propagation) базируется на классическом алгоритме обратного распространения ошибки с тем отличием, что часто обновляемые веса корректируются меньше остальных [14,15].

Расчет значений корректировки весовых коэффициентов искусственной нейронной сети производится по следующей формуле:

$$W_t = W_{t-1} - \frac{\eta}{\sqrt{E[g^2]_t - E[g]_t^2 + \varepsilon}} g_t, \quad (2)$$

где W_t — значение веса на текущем шаге;

W_{t-1} — значение веса на предыдущем шаге;

η — значение скорости обучения;

g_t – значение градиента на текущем шаге;

$E[g]_t$ — скользящее среднее значения градиента на текущем шаге;

$E[g^2]_t$ — скользящее среднее значение квадрата градиента на текущем шаге,

ε — константа, задающая значение погрешности. Данная формула представляет собой модификацию стандартного алгоритма RMS Propagation, заключающуюся в применении значения корректировки квадрата скользящего среднего градиента совместно с бегущим средним квадрата градиента.

Вычисление значений градиентов производится по методу обратного распространения ошибки (Backpropagation).

Q-Learning

Данный метод применяется в искусственном интеллекте при агентном подходе, когда нельзя четко определить желаемый результат. Он основан на введении функции Q, отражающей полезность каждого возможного действия агента для его

текущего состояния. Значения функции формируются в зависимости от вознаграждения. Благодаря этим значениям агент не случайно выбирает стратегию поведения, а учитывает опыт предыдущих действий [9].

Перед началом обучения формируется множество возможных состояний S агента, а также набор допустимых для него действий A. Выполнение некоторого действия $a \in A$ приводит к переходу агента из одного состояние в другое. В результате выполнения действия агенту начисляется вознаграждение. Задачей агента является максимизация суммарной награды.

Процесс обучения представляет собой итерационное уточнение значения функции Q. На каждом шаге рассчитывается значение оценки полезности совершения каждого допустимого действия для текущего состояния агента путем запуска нейронной сети. Затем производится корректировка оценки полезности последнего действия.

После всех расчетов полученные значения сохраняются в тренировочный набор.

Совмещенный алгоритм

С помощью алгоритма Q-Learning производится формирование желаемых выходных данных для искусственной нейронной сети. Полученные данные помещаются в тренировочный набор, после чего с помощью алгоритма RMS Propagation происходит обучение сети. На выходе нейронной сети формируется оценка целесообразности поворота отрезков модели.

Обучение продолжается в течение нескольких эпох до тех пор, пока значение ошибки не станет меньше некоторой допустимой погрешности.

Способ внедрения НС в управление процессом перемещения робота

Так как нейронная сеть не сможет эффективно работать на плате Arduino Uno, то необходимо разделить программную часть между двумя устройствами. На плате Arduino Uno можно оставить ту часть, которая управляет сервоприводами, то есть посылает сигнал, какому сервоприводу и на какой угол повернуться. Все расчеты целесообразно перенести на отдельную машину. Для этих целей хорошо подходит Raspberry Pi из-за простоты взаимодействия с платой Arduino, небольших размеров и простоты настройки. На этом микроконтроллере размещается нейронная сеть, которая будет отправлять команды на Arduino. Также на Raspberry Pi рассчитывается обновленное положение робота в пространстве. Контроллер отправляет Arduino Uno сообщение о движении конечности робота, в котором содержится указание номера сервопривода и угол поворота. Arduino останется лишь отправить сигнал. Взаимодействие микроконтроллеров реализуется простой последовательной связью по USB-кабелю. Разработка под Raspberry Pi ведется на Python3.

Заключение

В статье были представлены методические указания по реализации проекта, целью которого является конструирование робота, перемещением которого управляет искусственная нейронная сеть. В результате описанных действий студент не просто выполняет абстрактные задания, предлагаемые в лабораторных работах, а получает готовый цельный продукт. При этом он погружается и углубляется сразу в две дисциплины, укрепляя практикой свои теоретические знания. Он наглядно представляет область применения изученного материала в реальной жизни, поэтому

сможет в дальнейшем использовать накопленный опыт не только в аналогичных проектах. Также студент в процессе выполнения описанной работы выстраивает систему ассоциаций, которая способствует улучшению мыслительного процесса и может дополняться новыми знаниями указанных дисциплин и веяниями других наук.

Данный проект может быть рассредоточен на некоторое количество лабораторных работ или представлен в виде курсового проекта. Он имеет в своей структуре множество глобальных параметров, легко поддающихся корректировке. К ним можно отнести изменение конструкции робота, изменение архитектуры искусственной нейронной сети, выбор алгоритма ее обучения, значений, способа учета перемещения робота и даже цели работы нейронной сети.

Таким образом, симбиоз двух дисциплин при обучении позволит лучше раскрыть каждую из них и поможет приблизить студента к профессиональной деятельности.

Литература

1. Вукобратович М. Шагающие роботы и антропоморфные механизмы. — М.: Мир, 1976. — 541 с.
2. Гаврилов А. В. Архитектура гибридной системы управления мобильного робота //Мехатроника, автоматизация, управление. – 2004. – №. 8. – С. 30-37.
3. DEMYSTIFYING DEEP REINFORCEMENT LEARNING [Электронный ресурс]. — Режим доступа: http://neuro.cs.ut.ee/demystifyingdeep-reinforcementlearning/.— Загл. с экрана.
4. Федосин С. А., Плотникова Н. П., Немчинова Е. А., Макарова Н. В. Особенности обучения построению моделей перемещения сложных объектов, обладающих искусственным интеллектом на базе нейронной сети. Образовательные технологии и общество. 2018. Т. 21. №3. С. 290-297.
5. Ha S., Kim J., Yamane K. Automated Deep Reinforcement Learning Environment for Hardware of a Modular Legged Robot //2018 15th International Conference on Ubiquitous Robots (UR). – IEEE, 2018. – С. 348-354.
6. Энциклопедия знаний Амперки [Электронный ресурс]. — Режим доступа: http://wiki.amperka.ru/.
7. Оссовский С. Нейронные сети для обработки информации / Пер. с польского И. Д. Рудинского. — М.: Финансы и статистика, 2002. — 344с.: ил.
8. Федосин С. А., Немчинова Е. А., Плотникова Н. П. Искусственный интеллект на базе нейронной сети, реализующий перемещение модели сложного объекта в пространстве. Наукоемкие технологии. 2018. Т. 19. №7. С. 23-29.
9. Neural Networks for Machine Learning [Электронный ресурс]. — Режим доступа: http://www.cs.toronto.edu/~tijmen/csc321/slides/lecture_slides_lec6.pdf .
10. S. Levine, C. Finn, T. Darrell, and P. Abbeel. End-to-end training of deep visuomotor policies //The Journal of Machine Learning Research. – 2016. – Т. 17. – №. 1. – С. 1334-1373.
11. Duan Y. Benchmarking deep reinforcement learning for continuous control / Y. Duan, X. Chen, R. Houthooft, J. Schulman,P. Abbeel. //International Conference on Machine Learning. – 2016. – С. 1329-1338.
12. Короткий С. Нейронные сети: алгоритм обратного распространения; статья — 15 с.
13. D. Silver, G. Lever, N. Heess, T. Degris, D. Wierstra, and M. Riedmiller. Deterministic policy gradient algorithms //ICML. – 2014.
14. Google's DeepMind AI Just Taught Itself To Walk [Электронный ресурс]. — Режим доступа: https://www.youtube.com/watch?v=gn4nRCC9TwQ. — Загл. С экрана.

15. Learn to Walk (genetic algorithm & Neural Network) [Электронный ресурс]. — Режим доступа: https://www.youtube.com/watch?v=h-89xjWpV4U. — Загл. с экрана.