УДК 004.896, 004.855.5 **Технические науки**

Немчинова Елена Андреевна, *студентка,*
Мордовский государственный университет им. Н.П. Огарева (г. Саранск)
E-mail: nemchinova.len@yandex.ru
Плотникова Наталья Павловна, *к.т.н., доцент кафедры АСОИУ*
Мордовский государственный университет им. Н.П. Огарева (г. Саранск),
E-mail: linsierra@yandex.ru

ИСКУССТВЕННЫЙ ИНТЕЛЛЕКТ НА БАЗЕ НЕЙРОННОЙ СЕТИ, РЕАЛИЗУЮЩИЙ ПЕРЕМЕЩЕНИЕ МОДЕЛИ СЛОЖНОГО ОБЪЕКТА В ТРЕХМЕРНОМ ПРОСТРАНСТВЕ

Аннотация: В статье рассматриваются основные понятия, необходимые для создания трехмерной модели сложного объекта. Производится выбор архитектуры нейронной сети для управления перемещением модели сложного объекта в трехмерном пространстве. Описываются алгоритмы обучения нейронной сети, а также основные формулы для расчета данных, требующихся в этих алгоритмах.

Ключевые слова: искусственная нейронная сеть, алгоритм обучения нейронной сети, трехмерная модель сложного объекта, перемещение в трехмерном пространстве.

Abstract: The article discusses the basic concepts of creating a three-dimensional model of a complex object. Architecture of the neural network for the complex object movement control is selected. The neural network learning algorithms are described, as well as the basic formulas of some important for the training algorithm values.

Keywords: artificial neural network, neural network learning algorithm, three-dimensional model of a complex object, moving in three-dimensional space.

Введение

В последнее время одним из популярных направлений применения искусственного интеллекта стало моделирование процессов осознанного перемещения объектов в пространстве. Освоение данного направления ставит перед исследователями задачу обучения нейронной сети, которая смогла бы выступать в роли мозга некоторого объекта и была способна анализировать окружающее его пространство. На текущем этапе развития решение поставленной задачи осуществляется в упрощенном варианте. Однако уже сейчас существует возможность управления моделями любых сложных объектов. Ярким примеров является разработка компанией Google DeepMind искусственного интеллекта, которому удалось научиться ходить, бегать, прыгать и преодолевать препятствия без каких-либо предварительных рекомендаций [5]. В свою очередь инженеры сети исследовательских лабораторий Disney Research создали алгоритм, который позволяет роботам адаптироваться к изменениям в конфигурациях и находить новые способы передвижения [6].

Целью данной работы является создание искусственной нейронной сети, управляющей перемещением модели сложного объекта в трехмерном пространстве. Основная задача модели — переместиться как можно дальше от своего первоначального положения.

Разработка модели сложного объекта

Модель объекта будет представлена совокупностью отрезков в трехмерном пространстве, концы которых соединены между собой [7]. В ней должны быть отражены основные части объекта, с мощью которых он перемещается. Соединения отрезков назовем «суставами» модели объекта. «Суставы» будут заданы в виде точек, имеющих свои координаты в трехмерном пространстве. Это позволит однозначно определить положение модели в каждый момент времени.

В модели будут представлены отрезки двух типов: подвижные, способные совершать поворот относительно точки крепления, и неподвижные, изменяющие свое положение в пространстве благодаря повороту подвижных отрезков или под действием внешних сил. Телом модели объекта будет являться набор жестко связанных отрезков, то есть неподвижных друг относительно друга. Тогда отрезки, способные менять свое положение относительно тела, будут представлять собой конечности модели. Рассмотрим модель некоторого объекта, изображенную на рисунке 1.

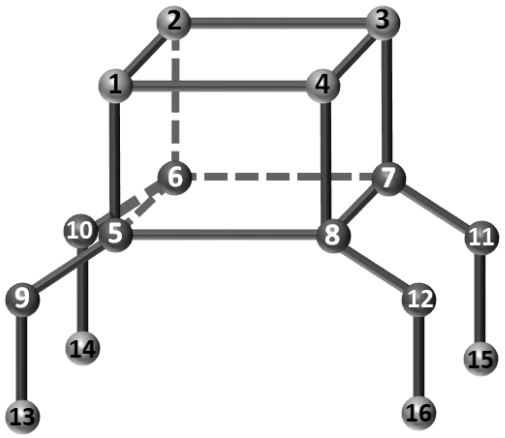

Рисунок 1 — Модель объекта

Исходя из рисунка 1, тело модели представлено параллелепипедом 1-2-3-4-5-6-7-8, объект имеет четыре ярко выраженные конечности (отрезки 8-12 и 12-16, 5-9 и 9-13, 7-11 и 11-15, 6-10 и 10-14), с помощью которых он будет перемещаться.

Также для модели задается информация о том, в каких «суставах» и каким образом будут двигаться отрезки ее конечностей. Например, отрезок 5-9 будет совершать поворот в «суставе», обозначенном точкой 5, а отрезок 6-10 поворачивается в точке 6 и так далее для каждого подвижного отрезка. На множество допустимых углов поворота отрезков накладываются ограничения. Они представляют собой сектор шара с центром в точке, являющейся «суставом» поворота для конкретного отрезка. Сектора задаются при

проектировании модели относительно ее тела для корректного их применения в любом положении объекта относительно системы координат.

Перемещение модели в пространстве будет неупругим. Данный способ характеризуется фиксацией отрезка конечности в некотором положении, в которое он пришел после совершения очередного действия. Поворот отдельного отрезка приводит к его переходу в новое состояние и ожиданию дальнейшего действия или реакции окружающей среды.

Особенности применения нейронной сети

Управление действиями модели будет производиться с помощью нейронной сети. В каждый момент времени она на основе опыта предыдущих действий будет делать предположение о том, какое элементарное действие нужно совершить на текущем шаге. Под элементарным действием предполагается поворот отрезка на некоторый угол. Однако нейронная сеть не может повернуть отрезок на произвольный угол. Для решения данной проблемы вводится понятие состояния отрезка. Сектор шара, ограничивающий движения отрезка, разбивается на некоторое количество более мелких секторов. Тогда состоянием отрезка будет являться такое его положение, при котором его конец, не прикрепленный к текущему для данного момента времени «суставу» поворота, находится в центре основания малого шарового сегмента. В этом случае поворот отрезка 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.

Архитектура искусственной нейронной сети

Искусственная нейронная сеть будет представлять собой многослойный

персептрон. Количество скрытых слоев сети будет равно двум (рисунок 2). На входы нейронная сеть будет получать координаты точек, соответствующих «суставам» модели в трехмерном пространстве. Каждая точка описывается тремя координатами — значениями по осям абсцисс, ординат и аппликат [2].

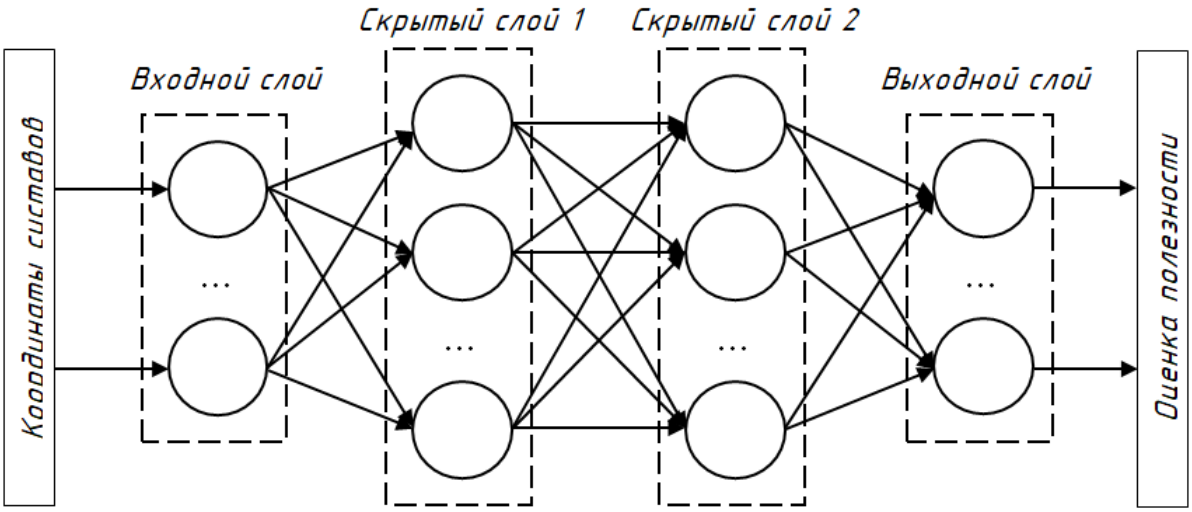

Рисунок 2 – Архитектура искусственной нейронной сети

Результатом работы нейронной сети является оценка полезности совершения моделью каждого допустимого действия [3]. На нейроны выходного слоя поступает информация, соответствующая элементарному действию частей подвижных конечностей.

Алгоритм обучения искусственной нейронной сети

Для обучения искусственной нейронной сети можно использовать алгоритм RMS Propagation совместно с алгоритмом Q-Learning [4].

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

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

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

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) [1].

Q-Learning 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.

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 по следующей формуле:

$$Q(State_t, Action_{t-1}) = Reward_t + \gamma \cdot max_{Action_t} Q(State_t, Action_t), \quad (2)$$

где *Reward* — вознаграждение, полученное агентом за предыдущее действие;

γ — коэффициент, отражающий степень доверия новому действию ($0 \leq \gamma \leq 1$), он показывает, что для агента имеет большую ценность: сиюминутные награды или будущие. Чем ближе значение коэффициента к единице, тем быстрее он забывает предыдущие действия.

После всех расчетов полученные значения сохраняются в тренировочный набор.

Вознаграждение за совершенное действие учитывает следующие параметры:

1) расстояние, пройденное моделью от ее начального положения:

$$AllDist_t = \sqrt{(CurPosX_t - StartPosX)^2 + (CurPosY_t - StartPosY)^2}, \quad (3)$$

где $CurPosX_t$, $CurPosY_t$ — значение координаты центра тяжести модели по оси абсцисс и ординат, соответственно, в текущий момент времени;

StartPosX, *StartPosY* — значение координаты центра тяжести модели по оси абсцисс и ординат, соответственно, в начальный момент времени;

2) расстояние, пройденное моделью за последнюю единицу времени:

$$PrevStepDist_t = \sqrt{(CurPosX_t - CurPosX_{t-1})^2 + (CurPosY_t - CurPosY_{t-1})^2}; \quad (4)$$

3) штраф за приближение центра тяжести модели к «земле»:

$$CenterOfGravity_t = -\frac{k_1}{CenterOfGravityZ_t}, \quad (5)$$

где $CenterOfGravityZ_t$ — координата центра тяжести модели по оси аппликат;

k_1 — коэффициент, показывающий силу влияния данного параметра на общее значение вознаграждения;

4) штраф за падение модели:

$$Falling_t = -\frac{k_2}{MinDistToSupportPoint_t}, \quad (6)$$

где $MinDistToSupportPoin_t$ — минимальное значение среди расстояний от центра тяжести модели до точек опоры, рассчитанное для проекций точек на плоскость, образуемую осями абсцисс и ординат;

k_2 — коэффициент, показывающий силу влияния данного параметра на

общее значение вознаграждения;

5) штраф за приближение верхней точки («головы») модели к «земле»:

$$Head_t = -\frac{k_3}{MinHeadZ_t}, \quad (7)$$

где $MinHeadZ_t$ — минимальное значение среди координат точек «головы» модели по оси аппликат;

k_3 — коэффициент, показывающий силу влияния данного параметра на общее значение вознаграждения.

Формула для расчета вознаграждения представляет собой комбинацию выше указанных параметров и может быть записана в следующем виде:

$$Reward_t = \alpha_1 * AllDist_t + \alpha_2 \cdot PrevStepDist_t + \alpha_3 \cdot CenterOfGravity_t + \\ + \alpha_4 \cdot Falling_t + \alpha_5 \cdot Head_t \ , \quad (8)$$

где $\alpha_i \ (i = \overline{1,5})$ — коэффициент, принимающий значения 0 или 1 и показывающий, будет ли учитываться текущий параметр при расчете вознаграждения.

Совмещенный алгоритм

С помощью алгоритма Q-Learning производится формирование желаемых выходных данных для искусственной нейронной сети (значения рассчитываются по формулам (2) – (7)). 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 [8]. Ошибка выхода нейронной сети вычисляется по следующей формуле:

$$\frac{1}{2}\left[Reward + \gamma \cdot max_{Action_t} Q(State_t, Action_t) - Q(State_{t-1}, Action_{t-1})\right]^2. \quad (9)$$

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

Заключение

Проведение исследований в данной области упростит процесс моделирования перемещения сложных объектов в трехмерном пространстве. Результаты исследований могут быть применены в робототехнике и позволят конструировать роботов, перемещающихся с помощью конечностей, работой которых будет управлять искусственный интеллект. Рассмотрение и использование различных архитектур нейронных сетей и алгоритмов их обучения помогут выбрать наиболее эффективные методы построения искусственного интеллекта на базе нейронной сети.

Библиографический список:

1. Короткий С. Нейронные сети: алгоритм обратного распространения; статья — 15 с.
2. Оссовский С. Нейронные сети для обработки информации / Пер. с польского И. Д. Рудинского. — М.: Финансы и статистика, 2002. — 344с.: ил.
3. Хайкин Саймон Нейронные сети: полный курс. — 2-е изд., перераб. и доп. — М.: Издательский дом «Вильямс», 2006 — 1104 с.: ил.
4. DEMYSTIFYING DEEP REINFORCEMENT LEARNING [Электронный ресурс]. — Режим доступа: http://neuro.cs.ut.ee/demystifyingdeep-reinforcement-learning/.— Загл. с экрана.
5. Google's DeepMind AI Just Taught Itself To Walk [Электронный ресурс]. — Режим доступа: https://www.youtube.com/watch?v=gn4nRCC9TwQ. — Загл. с экрана.
6. 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.
7. Learn to Walk (genetic algorithm & Neural Network) [Электронный ресурс]. — Режим доступа: https://www.youtube.com/watch?v=h-89xjWpV4U. — Загл. с экрана.
8. Neural Networks for Machine Learning [Электронный ресурс]. — Режим доступа: http://www.cs.toronto.edu/~tijmen/csc321/slides/lecture_slides_lec6.pdf .