УДК 004.942

МЕТОДИКА ОБУЧЕНИЯ С ПОДКРЕПЛЕНИЕМ, ПРИМЕНЯЕМАЯ ДЛЯ МОДЕЛИРОВАНИЯ ПРОЦЕССОВ РОЕВОГО УПРАВЛЕНИЯ ОБЪЕКТАМИ

А.К. Дьячук, Е.В. Чекарева (Москва)

Введение

За последние годы нейронные сети прочно укоренились в повседневной жизни человека [1]. Несмотря на достаточно короткую историю, они уже внедрены в важнейшие аспекты нашей жизни, минимизируя участие оператора в монотонных работах на заводах и помогая там, где человеческий фактор может привести к фатальным последствиям.

Все чаще нейросети интегрируют в системы навигации, например, для управления беспилотными летательными аппаратами (БПЛА), оптимизации и планирования маршрута. Оптимальное планирование маршрута – сложная задача, обеспечивающая безопасность и точность передвижения аппаратов в заданных условиях, а применение методик глубокого обучения позволяет не только точно предсказывать траекторию за счет использования классических методов при разработке нейросетей, но и адаптировать систему к динамически меняющимся условиям карты и рельефа.

Роевое управление предполагает координацию действий нескольких агентов в сложной среде в режиме реального времени. Данная задача предполагает задействование больших вычислительных мощностей оборудования, учет неопределенностей среды и необходимость адаптивного масштабируемого поведения агентов.

Классические подходы к роевому управлению:

- алгоритмы «потенциальных полей»;
- Модель Бойда;
- централизованное управление с использованием классических методов, в частности алгоритма A*.

В отличие от классических подходов, обучение с подкреплением (Reinforcement Learning, RL) и глубокое обучение нейронных сетей позволяют агентам самостоятельно вырабатывать эффективную стратегию перемещения, исходя из собственного «опыта» и «вознаграждения». В условиях динамически меняющейся среды RL и нейросети стали незаменимым инструментом. Далее будут рассмотренных рассмотрены наиболее популярные алгоритмы [1].

Постановка задачи

Рассмотрим задачу моделирования автономной навигации беспилотных летательных аппаратов в двумерной среде с динамически меняющимся местоположением цели.

Основной целью исследования является разработка и обучение программного интеллектуального агента, способного управлять движением дронов таким образом, чтобы они самостоятельно достигали заданной цели, не имея предварительного знания о ее позиции и не используя заранее запрограммированный маршрут. Управление при этом должно осуществляться на основе методики обучения с подкреплением.

Среда обучения представляет собой двумерную плоскость фиксированного размера, в пределах которой объекты могут перемещаться. В каждом эпизоде стартовая точка каждого дрона и цель формируются случайным образом. Каждому аппарату соответствует отдельный обучающийся агент, принимающий решения на основе части информации о среде (наблюдения), а также (опционально) о других агентах в пределах

радиуса связи. Такие агенты действуют децентрализованно, не имея общей карты или координатора.

Постановка задачи следующая. Рассматривается рой из N дронов, где каждый i-й объект описывается набором параметров:

- положение дрона p_i – координаты в пространстве:

$$p_i = (x_i, y_i), \tag{1}$$

- движение дрона a_i – вектор скорости:

$$a_i = \left(v_x^i, v_y^i\right), \tag{2}$$

- наблюдение дрона o_i – информация, доступная для принятия решения, которая содержит собственные координаты объекта, координаты и расстояние до ближайшего соседа:

$$o_i = \left[x_i, y_i; x_g, y_g; d_{ij}\right], \tag{3}$$

где x_i, y_i – координаты дрона; x_g, y_g – координаты цели; d_{ij} – расстояние до ближайшего соседа; i – индекс рассматриваемого дрона; j – индекс соседнего дрона.

Физика процесса движения для каждого летательного аппарата остается такой же, как и в одноагентном случае, однако она применяется индивидуально к каждому дрону роя, то есть

$$p_i^{t+1} = p_i^t + \Delta t \cdot a_i^t, \tag{4}$$

где p_i^{t+1} – новая позиция дрона; p_i^t – текущая позиция дрона; a_i^t – действие дрона; Δt – шаг по времени.

Задача каждого агента состоит в следующем:

- минимизировать расстояние до назначенной цели g_i;
- избегать слишком близкого расположения к другим дронам.

Указанные задачи формируются с помощью функции награды для каждого агента

$$r_i^t = -|p_i^t - g_i| + R_e, \tag{5}$$

где R_e – событийный компонент.

$$R_e = \begin{cases} +100, \text{если цель достигнута}, \\ -10, \text{если } \left|p_i - p_j\right| < d_{min} \text{ для любого } i \neq j, \\ 0, \text{во всех остальных случаях} \end{cases} \tag{6}$$

Функция награды обеспечивает не только приближение агентов к цели, но и предотвращает столкновения, давая отрицательную награду при чрезмерном сближении с одним из других объектов роя. Работу функции награды подробно описывает формула (6). Изначально агент не понимает, что столкновение – это плохо, при обучении он случайно сталкивается с другим агентом, за что получает штраф. Следовательно, нейронная сеть формирует будущую стратегию таким образом, чтобы избежать столкновений.

Так агент формирует стратегию поведения – приоритетным является избежание столкновения. Однако агенты «не видят» друг друга напрямую, достаточным является знание о расстоянии до ближайшего соседа (через сенсоры самого летательного аппарата или отправленные напрямую оператором).

В этом и заключается принцип самообучения. Агенты не знают как «правильно», но пронимают, какие действия «плохие», и стараются их избегать. Таким образом обучение направлено не только на достижение целей, но и на избежание столкновений с другими объектами роя, что играет основную роль в реальных системах групповой навигации.

Классические методы роевого управления

A. Метод потенциальных полей

Метод имитирует построение виртуального силового поля в среде передвижения БПЛА, то есть передвижение беспилотников происходит по силовым линиям поля. Математическая модель силового поля строится на основе полученных данных навигационной информации и отражает пространственные объекты или препятствия.

Каждому объекту приписывается виртуальный «электрический заряд» [2, 6]. При заданном начальном положении $r = (x, y)$ помещенный в цель P точечный отрицательный заряд создает притягивающее (attracting) потенциальное силовое поле:

$$U_{att} = U_{att}(r), \quad (7)$$

Препятствия, имеющие положительный точечный заряд, будет иметь отталкивающее (repelling) силовое поле:

$$U_{rep} = U_{rep}(r). \quad (8)$$

Тогда суперпозиция притягивающего и отталкивающего силовых полей образует потенциальное силовое поле U(r):

$$U(r) = U_{att}(r) + U_{rep}(r). \quad (9)$$

Вектор напряженности поля в точке $r(x, y)$ определяется определяется как антиградиент потенциала [3]:

$$E(r) = -\nabla U(r). \quad (10)$$

Такой метод хорошо работает при наличии статической карты, однако в динамических условиях он неэффективен. Кроме того, алгоритм имеет проблемы с обнаружением локальных минимумов, масштабируемостью и сильно зависит от ручной настройки коэффициентов, что делает его сложным в реализации.

B. Правила Бойда

Это один из первых и самых популярных подходов, который имитирует поведение стаи. Основан на трех основных поведенческих правилах:

- «выравнивание» – движение в одном направлении с соседями;
- «притяжение» – стремление к центру масс соседей;
- «избегание» – уклонение от столкновений с другими агентами.

Основные параметры, существенно влияющие на качество передвижения каждого агента, следующие:

- радиус или расстояние между агентами;
- угол обзора.

Для каждого беспилотника роя задаются начальные параметры и три составляющих вектора движения, положение каждого отдельноно беспилотника изменяется с учетом скорости и заданных параметров.

Путем сложения вектором определяется итоговый вектор скорости и вычисляется финальное положение агента [7].

Такой подход достаточно прост в реализации и основан на достоверных поведенческих принципах, однако плохо адаптирован к среде и условиям миссии, а также требует тонких настроек параметров.

C. Централизованное планирование маршрута

Данный метод используется только при наличии полной информации. Существует множество методов, которые можно использовать при централизованном планировании маршрута, например, $A*$ – алгоритм поиска на графе $G\ (V, E)$, который

совмещает в себе преимущества поиска в ширину и оценочной функции, что повышает эффективность процесса поиска за счет используемой эвристики [8]. Функция перехода считается по формуле (11) и включает две основные функции: эвристическая функция h(x) и функция оценки расстояния от начальной до текущей точки g(x):

$$F(x) = h(x) + g(x), \quad (11)$$

В свою очередь, эвристические функции могут выбираться, исходя из условий решаемой задачи:

- Манххэттенское расстояние (сетка с 4 направлениями движения);
- Евклидово расстояние (сетка с 8 направлениями движения);
- расстояние Чебышева (сетка с 8 направлениями движения);
- оптимистичные эвристики;
- абстрактные эвристики.

Такой метод оптимален при наличии полной информации о ситуации, что в реальных условиях практически невозможно, к тому же требует больших вычислительных мощностей, большого количества времени при генерации маршрутов для каждого агента, а также плохо масштабируется по числу агентов [8-10].

Таким образом, можно сделать вывод о том, что в современных условиях, когда набор данных ограничен, а среда является динамичной, классические методы неоспоримо проигрывают методам глубокого обучения. Поэтому в данной работе предлагается использовать обучение с подкреплением, позволяющее дрону самостоятельно адаптироваться к среде исходя из собственного «опыта».

Обучение с подкреплением

Обучение с подкреплением (*Reinforcement Learning*, RL) – это раздел машинного обучения, в котором интеллектуальный агент изучает и выбирает оптимальную стратегию в заданной среде. В рамках RL агент, взаимодействуя с окружающей средой, получает числовые оценки эффективности своих действий в виде награды и модифицирует свою стратегию поведения с целью максимизации суммарного вознаграждения.

В отличии от классических методов данный алгоритм не требует начальных знаний о модели среды, поскольку обучается по методу «проб и ошибок», а каждый агент фокусируется на принятии решений, которые максимизируют «награду» в долгосрочной перспективе.

Основные термины в RL:

- *Агент* – объект, принимающий решение;
- *Среда* – область, в которой агент принимает решения.
- *Вознаграждение* – обратная связь, которую получает агент после выполнения действия. Оно может быть как положительным, так и отрицательным и помогает определить дальнейшую тактику агента.
- *Политика* – стратегия, по которой агент определяет следующее действие в каждом состоянии.

В основе данного метода лежит непрерывная обратная связь между средой и агентом, как показано на рис. 1. Цикл начинается с того, что среда предоставляет агенту текущее состояние (показатели с датчиков и положение других агентов роя), затем агент анализирует полученную информацию и выбирает дальнейшую стратегию движения к цели и взаимодействия с другими агентами. Среда реагирует на действие агента, переходит в новое состояние и одновременно выдает агенту «поощрение» или «наказание». С каждой итерацией агенты набираются опыта и улучшают свою стратегию поведения [11].

Такую модель поведения можно описать через концепцию Марковского процесса принятия решений, она обеспечивает математическую основу для анализа

последовательности состояний, действий и вознаграждений. Во избежание локальных оптимумов агент вынужден делать неочевидные действия, однако существуют стратегии, позволяющее этого избежать.

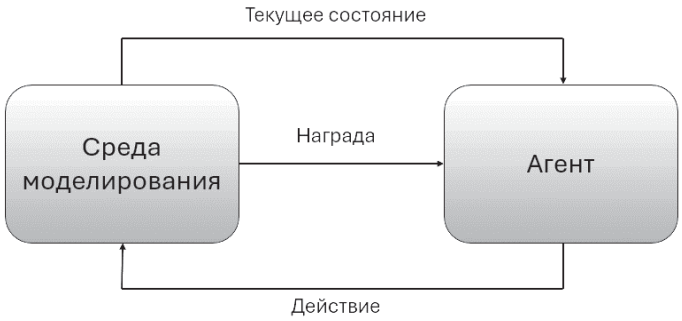

Рис. 1. Схема принципа работы RL

В рамках данной работы каждая итерация начинается с того, что положения дрона и целевой точки задаются случайным образом, дрон может передвигаться в любом направлении, а задача нейронной сети – обучить агента доходить до цели как можно быстрее.

Для этого в программной среде *Matlab*® создается среда обучения, на вход которой агент получает:

- текущее положение: координаты *x* и *y*;
- положение цели: координаты x_g и y_g.

Таким образом, вектор состояния среды для агента определяется четырьмя основными параметрами:

$$O_t = [x_t, y_t, x_g, y_g], \tag{12}$$

где O_t – наблюдение (observation) в момент времени *t*, векторная величина; x_t, y_t – координаты БПЛА в момент наблюдения; x_g, y_g – координаты целевой точки.

В ответ на это наблюдение агент должен указать, в какую сторону и с какой скоростью будет двигаться – вектор значений скорости дрона по оси *x* и по оси *y* $\overrightarrow{(V_x, V_y)}$.

Для поощрения дрона вводится функция «награды». Чем ближе дрон к цели, тем больше назначается награда, если дрон достигает цели, то он получает максимальную награду. Таким образом, агент учится передвигаться в среде обучения. Такой обучающий алгоритм называется *Policy Gradient* (градиент стратегии). Его суть заключается в том, что агент принимает решения при помощи нейросети, а затем встроенные функции среды *Matlab*® корректируют нейросеть на основе полученных наград.

Со временем нейронная сеть начинает все чаще выбирать корректные, оптимальные решения, которые ведут к успеху.

В данной работе нейронная сеть реализует стратегию агента, который в зависимости от текущего состояния среды формирует управляющее воздействие в виде вектора скорости (v_x, v_y).

Таблица 1. Параметры архитектуры нейронной сети

Слой	Размерность	Функция активации	Назначение
Входной слой	4	–	Принимает на вход вектор
Первый скрытый обучающий слой	64	–	Первичная обработка признаков
Первый слой активации	64	ReLU	Функция активации вносит нелинейность
Второй скрытый обучающий слой	64	-	Усиливает и уточняет признаки, нейросеть обучается более детально
Второй слой активации	64	ReLU	Функция активации вносит нелинейность
Выходной слой	2	Linear	Генерирует управляющее воздействие

На вход нейронной сети подается 4 параметра, текущие координаты и координаты цели $[x_t, y_t, x_g, y_g]$, по этим параметрам формируется размерность входного слоя.

Выходной слой содержит два нейрона и формирует вектор действия агента $\overrightarrow{(V_x, V_y)}$. Первый скрытый обучающий слой: выполняет полносвязное преобразование, нейросеть извлекает первичные абстрактные признаки. Первый слой активации применяет следующую функцию Rectified Linear Unit:

$$ReLU(x) = \max(0, x). \quad (13)$$

Он обнуляет отрицательные значения, оставляя только положительные, что приводит к нелинейности. Это позволяет нейронной сети обучаться более эффективно.

Второй скрытый слой получает на вход активированные значения. Этот слой усиливает и уточняет признаки, что позволяет нейронной сети различать более сложные и скрытые признаки, более эффективно работать в нестандартных ситуациях.

Второй слой активации еще больше усиливает способность нейронной сети к обучению, вносит вариативность в обучение, усиливает ее, что позволяет нейросети обобщать поведение в разных ситуациях.

Параметры нейронной сети обновляются градиентным методом, а непосредственно обучение нейронной сети происходит с использованием инструментов пакета *Reinforcement Learning Toolbox* в *Matlab*®. Простота архитектуры позволяет системе функционировать в условиях ограниченных вычислительных ресурсов, например непосредственно на борту дрона.

Тренировка агента проводится итеративно в течение нескольких сотен эпизодов, каждый из которых представляет собой отдельную симуляцию: агент стартует в случайной позиции, и цель также размещается случайным образом на карте. В ходе тренировки агент обучается на последовательности "состояние — действие — награда", постепенно улучшая свою стратегию.

Представленный подход позволяет агенту обучиться эффективной стратегии навигации в статической, но не известной заранее среде, используя только сигналы награды – без предопределённой карты, маршрутов или правил навигации, что делает

Reinforcement Learning особенно подходящим для задач динамического роевого управления.

Таким образом, в рамках данной работы реализован базовый агент автономного управления на основе обучения с подкреплением с применением Policy Gradient-алгоритма. Его структура обеспечивает непрерывное управление на основе нейросети, а принцип обучения позволяет адаптироваться к разным условиям, при этом не требуя ручной настройки правил или моделей среды. Такой подход легко масштабируется.

Моделирование навигации агентов роевого интеллекта на основе обучения с подкреплением

Одной из ключевых задач при управлении беспилотным летательным аппаратом является навигация по неизвестной среде и поиск кратчайшего маршрута до заданной цели. В традиционном подходе к решению этой задачи широко применяются общепринятые и популярные алгоритмы планирования [8]. Все данные методы предполагают наличие карты среды, матрицы стоимостей перемещений и заранее определённой цели, причём траектория обычно рассчитывается один раз, не учитывая динамические изменения среды.

Однако в условиях динамической среды — например, при поиске пострадавших после катастрофы или в неизвестной местности без карты — этот подход демонстрирует слабую гибкость. Более того, при управлении роем дронов становится трудно централизованно рассчитывать оптимальные маршруты для всех агентов. В таких случаях эффективной альтернативой является использование методов обучения с подкреплением, позволяющих агенту самостоятельно вырабатывать стратегию перемещений, основанную на опыте взаимодействия со средой.

В рамках реализации, рассмотренной в данной работе, агент обучается направлять дрона к цели как можно быстрее, используя только текущее положение и координаты цели. При этом отсутствует любая информация о карте, маршрутах или модель поведения. Агент не знает, как далеко находится цель, и должен обучиться двигаться интуитивно — минимизируя путь через обобщённую стратегию действий. Такой способ можно назвать скрытым или неявным планированием: маршрут не строится явно, но поведение агента формируется так, что он достигает цели за минимальное количество шагов.

На первом этапе обучения агент ещё не имеет достаточного опыта, его движения хаотичны: он перемещается случайным образом, чаще всего даже не попадая в заданную область. Однако по мере накопления опыта и корректировки стратегии с помощью функции награды поведение значительно улучшается. Агент начинает выполнять более "разумные" действия: направляется к цели по прямой линии, избегает ненужных разворотов, сокращает путь. После достаточного количества эпизодов обучения поведение агента визуально становится аналогичным действиям, просчитанным классическим поисковым алгоритмом, но при этом не требует явного задания маршрута.

Рассмотрим ситуацию, когда дронам необходимо отыскать человека в лесном массиве. Большой участок леса делится на сектора между группами дронов. В заданной среде – карта размером 15×15 условных единиц – осуществляется поиск человека, находящегося в пределах карты в случайно заданной начальной точке (рис. 2). Три дрона, обученные каждый своим RL-агентом, стартуют из разных участков карты, а их главная цель – совместными действиями сократить неизученные территории, избегая столкновений, избыточных маршрутов и повторного сканирования территории. Таким образом, они гарантированно найдут человека, если он есть в этом секторе, и передать спасательным службам его точное местоположение.

Отсутствие заранее известной информации о местоположении цели делает метод RL максимально подходящим вариантом для обучений программного обеспечения. Важно отметить, что RL-агент принимает решения в дискретных временных шагах, ориентируясь только на количественную оценку полученной награды, без доступа к информации о полной карте или более глобальной стратегии. Тем не менее итоговая модель поведения модифицируется нейросетью, которая обучается таким образом, чтобы в ответ на вход (координаты) давать такие выходные параметры (вектор скорости), которые минимизируют путь до цели.

Исходя из специфики полетного задания и полетных условий, рой может выбирать различные траектории движения (Рис. 2):

- движение по орбите или круговой облет/патрулирование: дроны вращаются по широкой круговой траектории вокруг условного центра;
- движение по сужающейся спирали или маневр схождения в центр: дроны летят по спирали, приближаясь к центральной точке, может использоваться для фокусировки на цель;
- прочесывание местности по зигзагу: дроны движутся вперед, совершая резкие повороты влево-вправо, часто используется для поиска или сканирования определенной территории;
- круговой маневр в построении "клин": вся группа дронов движется по круговым траекториям (по орбите) вокруг центра, организованное круговое движение в конкретном боевом или тактическом порядке;
- основная траектория полета группы: во время полета дроны меняют построение для эффективности или выполнения тактических задач.

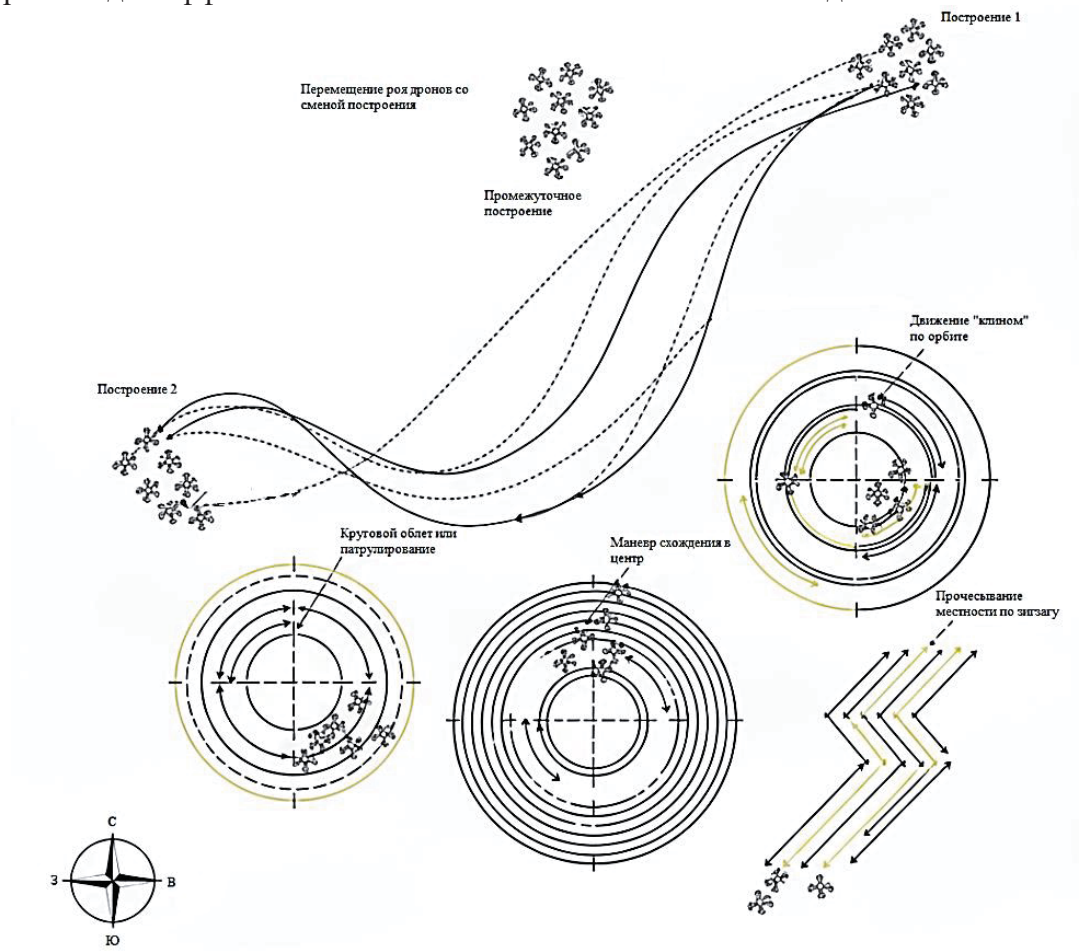

Рис. 2. Модель траекторий движения роя

Таким образом, решение задачи навигации и планирования маршрута реализуется неявно, система обучается в процессе взаимодействия агента с обучающей средой. Именно поэтому такие стратегии особенно полезны для роевых систем: каждый дрон может использовать один и тот же агент или собственную копию модели и двигаться к своей цели автономно, без оценивания маршрута с учётом всего окружения.

Описанное поведение доказывает, что обучение с подкреплением может быть использовано не только для задач распознавания образов или позиционных игр, но и в задачах с реальными физическими ограничениями, например, навигация дронов в среде с препятствиями. При этом агент не нуждается в полной информации о среде, функционирует в реальном времени и может быть масштабирован без существенного изменения архитектуры.

Анализ результатов

Для оценки эффективности обучения агента на основе алгоритма Policy Gradient были проведены серии модельных экспериментов. Целью являлось изучение динамики процесса обучения, а также анализ поведения дронов после завершения обучения в различных условиях:

- достижение целей из разных стартовых точек;
- избегание столкновений с другими дронами;
- согласованные и безопасные действия в общей обучающей среде;
- умение применить полученные навыки в незнакомых начальных условиях.

В качестве основной архитектуры использовался RL-агент, построенный на алгоритме Policy Gradient с простой нейронной сетью. Каждый агент обучался с использованием индивидуальной функции награды, заключающейся в минимизации расстояния до цели, бонусе за её достижение и штрафе за опасное сближение с другим дроном.

Результаты, достигнутые в ходе серии из 20-ти испытаний:

- в 95% случаев дроны успешно находили цель, избегая столкновений;
- в среднем цель была обнаружена за 40 шагов;
- все агенты в среднем за симуляцию покрывают более 80% карты.

Поиск прекращается, когда хотя бы один аппарат достигает цели с приемлемой погрешностью – 1 условная единица. Отсутствие связи между дронами позволяет им самостоятельно сканировать местность, что в некоторых случаях приводит к невыполнению задачи поиска человека из-за попадания цели в "узкую зону", что, однако, легко решается увеличением количества объектов в рое.

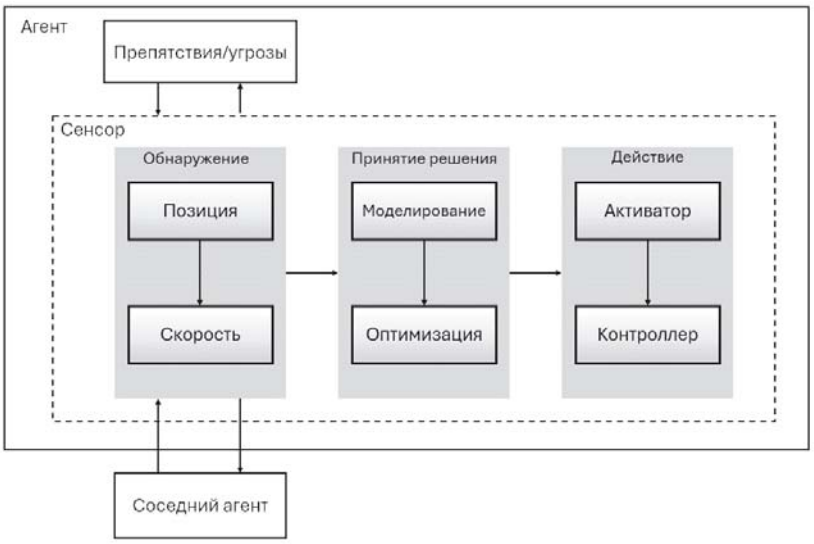

Рис. 3. Метод самоорганизации и принятия решений

Дроны, управляемые их собственными агентами, не взаимодействуют друг с другом напрямую, но они обмениваются информацией косвенно через окружающую среду, поэтому каждый беспилотник принимает решение о корректировке направления движения, основываясь на своей территории и наблюдениях, полученных из окружающей среды. В этой реализации функция вознаграждения предусматривает наказание за приближение к соседу (см. рис. 3). Несмотря на отсутствие явного взаимодействия, агенты учатся принимать решения, принимая во внимание действия других агентов в окружающей среде. Поиск прекращается, когда какой-либо беспилотник достигает цели с допустимой погрешностью не менее одной условной единицы.

После обучения поведение агентов можно охарактеризовать как планомерное и безопасное. Несмотря на независимое формирование стратегии поведения каждого БПЛА, итоговое движение роя выглядит согласованным и организованным, а система штрафов существенно повлияла при корректировке траектории, что в перспективе позволит добавить большее количество объектов в рой.

Таким образом, реализованная архитектура RL-агентов успешно обучается координированному перемещению и индивидуальному достижению целей.

Заключение

В данной работе рассмотрена задача управления БПЛА в условиях частичной неопределённости среды и отсутствия заранее заданной карты движения. Основной акцент делался на разработке подхода на базе метода обучения с подкреплением (Reinforcement Learning), способного заменить классические статические алгоритмы построения маршрутов в задачах реального времени.

На первом этапе была реализована модель движения одного дрона в двумерной среде, в которой целевая точка случайно задаётся в начале каждого эпизода.

Агент — управляемый искусственным интеллектом, построенным на алгоритме Policy Gradient — учился на основе значений награды приближаться к целевой точке с минимальными затратами количества шагов. В ходе обучения использовалась компактная архитектура нейросети, способная обобщать опыт и демонстрировать стратегию поведения, приближенную к классическим планировщикам, при отсутствии в ней заранее запрограммированных алгоритмов формирования маршрутов. Проведённые эксперименты показали, что агент способен эффективно и стабильно достигать цели, демонстрируя корректное поведение при различных начальных условиях. Поведение RL-агента после обучения отличалось высокой точностью, снижением количества шагов и устойчивостью даже при масштабировании области или начальных координат.

На втором этапе была расширена постановка задачи на мультиагентную систему – рой БПЛА. Для этого была предложена математическая формализация среды для нескольких агентов, включая индивидуальное управление, децентрализованное принятие решений и элементы кооперативного поведения. Применена система штрафов за сближение дронов и поощрение за достижение цели, что дополнительно мотивировало агентов развивать согласованное поведение в рое. Реализация методики с использованием трёх агентов подтвердила возможность масштабирования подхода и сохранения обучаемости в условиях их совместного взаимодействия.

Полученные результаты демонстрируют перспективность применения методов глубокого обучения с подкреплением для организации автономного координированного управления роем БПЛА.

Литература

1. **P. Makkar.** Reinforcement Learning: A Comprehensive Overview, International Journal of Innovative Research in Computer Science and Technology (IJIRCST), Vol. 12, Issue 2. P. 119-125, 2024. DOI: 10.55524/ijircst.2024.12.2.21.
2. **Филимонов А.Б.** Конструктивные аспекты метода потенциальных полей в мобильной робототехнике. / А. Б. Филимонов, Н. Б. Филимонов // Автометрия. Optoelectronics, instrumentation and data processing: научный журнал. 2021. Том 57, N 4. С. 45–53. DOI: 10.15372/AUT20210406.
3. **Филимонов А. Б., Филимонов Н. Б.** Constructive Aspects of the Method of Potential Fields in Mobile Robotics // Optoelectronics, Instrumentation and Data Processing. Новосибирск: СО РАН, 2021. Vol. 57, №4. С. 371-377. DOI: 10.3103/S8756699021040063.
4. **Чжу Юйцин**. Формирование управления полетом группы беспилотных летательных аппаратов на основе алгоритма многоагентной модели роения. Информатика, телекоммуникации и управление. 2022. Vol. 15. № 4. P. 22-36.
5. **Захаров Н.С., and Харисов А.Р.** Численное моделирование алгоритмов роевого управления для автономных групп беспилотных летательных аппаратов. Вестник науки. 2025. Vol. 3. № 6 (87), P. 1935-1945.
6. **Y. Peng, Y. Zhang, Y. Wang, and C. Wang.** Multi-agent deep reinforcement learning-based cooperative search strategy for UAV swarm Sensors. 2022. Vol. 22. № 7. P. 2604, Apr. DOI:10.1016/j.dcan.2023.06.003.
7. **Y. Yang, X. Xiong, and Y. Yan.** UAV formation trajectory planning algorithms: A review, Drones. 2023. Vol. 7, №1, P. 62, Jan. DOI:10.3390/drones7010062.
8. **A. Diyachuk.** Comparative analysis of optimal path planning algorithms using deep learning technique. 2024 International Conference on Cyber-Physical Social Intelligence (ICCSI), Doha, Qatar. 2024. DOI:10.1109/iccsi62669.2024.10799336.
9. **J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov.** Proximal policy optimization algorithms, arXiv preprint arXiv:1707.06347, Jul. 2017. DOI:10.48550/arXiv.1707.06347.
10. **R. J. Williams.** Simple statistical gradient-following algorithms for connectionist reinforcement learning, Machine Learning. 1992. Vol. 8. P. 229-256,
11. **N. Ayanian and V. Kumar.** Decentralized feedback controllers for multiagent teams in environments with obstacles, IEEE Transactions on Robotics. 2010. Vol. 26. №5. P. 878-887. DOI:10.1109/TRO.2010.2062070.
12. **M. Tan.** Multi-agent reinforcement learning: Independent vs. cooperative agents, in Proc. 10th Int. Conf. Machine Learning (ICML), Amherst, MA, USA, Jul. 1993, P. 330-337. DOI:10.1016/B978-1-55860-307-3.50049-6.