ВЕСТНИК ВОЕННОГО ИННОВАЦИОННОГО ТЕХНОПОЛИСА «ЭРА», 2022, том 3, № 3, с. 306–313

## ЭЛЕКТРОННАЯ КОМПОНЕНТНАЯ БАЗА И ИНТЕЛЛЕКТУАЛЬНЫЕ СИСТЕМЫ УПРАВЛЕНИЯ

УДК 004.896

# РЕШЕНИЕ ЗАДАЧИ СЛЕДОВАНИЯ АГЕНТА ЗА ЛИДЕРОМ НА БАЗЕ МОДЕЛИ, ОСНОВАННОЙ НА ОБУЧЕНИИ С ПОДКРЕПЛЕНИЕМ

© 2022 г. Р. Б. Рыбка[1,*], В. А. Шеин[1], М. С. Скороходов[1], А. В. Грязнов[1], А. Г. Селиванов[1], А. Г. Сбоев [1,2]

[1] Национальный исследовательский центр "Курчатовский институт", Москва, Россия

[2] Национальный исследовательский ядерный университет "МИФИ", Москва, Россия

*E-mail: Rybkarb@gmail.com

Поступила в редакцию 15.03.2022 г.
После доработки 20.03.2022 г.
Принята к публикации 20.03.2022 г.

Особенностью построения моделей управления движением на базе обучения с подкреплением является потребность в быстро реагирующей среде, позволяющей эмулировать различные возникающие в процессе управления ситуации. Представлен подход к решению задачи движения агента, ведомого за целевым объектом лидером-ведущим, основанный на переносе модели, обученной в упрощенной модельной 2D-среде, в 3D-среду, похожую на реальный мир. В этой задаче глобальные координаты лидера-ведущего неизвестны, и требуется их вычисление в режиме реального времени в процессе управления. Перенос выполнен с помощью построения системы управления, учитывающей различия между средой обучения, и модели управления движением. Показано, что разработанная система управления движением на основе модели, обученной в упрощенной среде, позволяет решать задачу управления следованием с точностью 90%.

**DOI:** 10.56304/S2782375X2203007X

## ВВЕДЕНИЕ

Использование робототехнических комплексов в труднодоступных регионах актуализирует задачу разработки алгоритмов управления, в частности для случая движения агента за целевым объектом (ведущим) или движения в колонне, в условиях динамически изменяемого маршрута с учетом возникающих препятствий. При наличии возможных осложнений (динамические препятствия, потеря связи и т.д.) перспективным является создание управляющих нейросетевых моделей следования на базе технологий обучения с подкреплением (**RL-модели**) вместо применения классических алгоритмов TEB, DWA и др. [1, 2]. Эти алгоритмы требуют улучшения по ряду позиций: минимизации отклонения от траектории (**TEB**), учета динамических препятствий (**DWA**, **EBAND**), способности работы с большим количеством препятствий и опасных участков на местности с ограничениями по времени (все алгоритмы). RL-модели более адаптивны к изменяющимся условиям среды, однако требуют большого числа циклов обновления весов при получении новых наблюдений $S_t$ и наград $R_t$ (рис. 1). Это в значительной степени осложняет возможность обучения развиваемых моделей управления непосредственно в среде их использования. Использование упрощенных высокопроизводительных математических моделей реальных сред является распространенной практикой при построении RL-моделей. Данный подход продемонстрировал свою эффективность при решении таких задач, как манипуляции роботизированной рукой [3], управление движением четырехногого робота по различным типам местности [4], прыжки четы-

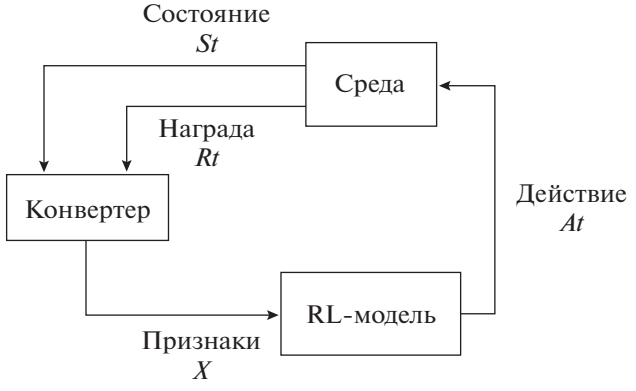


**Рис. 1.** Общая схема обучения с подкреплением.

реxного робота [5], гонки дронов по сложным трассам [6] и другие [7–11].

Таким образом, возникает необходимость переноса моделей из высокопроизводительных эмуляционных сред в среду реального применения. В данной работе рассмотрено решение задачи “следования за лидером” на базе RL-модели, полученной в эмуляционной 2D-среде, с переносом ее в 3D-среду, имеющую характерные признаки потенциальной среды использования модели. Среда для решения задачи “следования за лидером” описана в [12], а также представлена в открытом доступе [13].

Приведены описание задачи “следования за лидером” и модель нейронной сети для ее решения, соответственно, представлено описание 3D-среды, имеющей характерные признаки среды использования нейросетевой RL-модели с выделением отличий 3D-среды от эмуляционной среды. Представлены архитектура системы управления и ее основные компоненты и результаты экспериментов по переносу обученной нейросетевой модели.

## МЕТОДЫ

Лидер является объектом, имеющим изменяемые характеристики, определяющие его движение, и двигается по некоторой заданной траектории, информация о которой ведомому недоступна. Движение происходит в среде, которая может включать в себя препятствия различной формы. Ведомый обладает набором сенсоров, позволяющих ему собирать информацию о среде, на основе которой возможно вести управление. Таким образом, задача следования за лидером формулируется следующим образом: в течение времени движения лидера необходимо выбирать значения (или изменения) управляемых характеристик объекта-ведомого на основании данных сенсоров таким образом, чтобы он повторял траекторию движения лидера с минимальным отклонением, а также соблюдая ограничения на минимальное и максимальное расстояние от лидера и избегая столкновений с объектами среды.

Симулируемая среда представляет собой двумерное евклидово пространство (2D-мир) с подвижными объектами (ведомым и ведущим), а также с препятствиями двух типов, в частности случайно расположенными препятствиями заданного размера и препятствиями, эмулирующими узкие места (например, мосты/броды/горные перевалы и т.д.). Управление подвижными объектами осуществляется установкой значений линейной скорости и скорости поворота для каждого из них в каждый момент времени $t$. Скорости ведущего устанавливаются на основе заданного ему маршрута следования и выбранных диапазонов ускорений, а скорости ведомого должны быть получены из модели управления следованием, обучаемой желаемому поведению непосредственно в этой среде.

В общем случае желаемое поведение ведомого должно удовлетворять нескольким условиям:

– двигаться по маршруту, который проходит ведущий, с минимальным отклонением;

– находиться на определенном расстоянии, не приближаясь и не удаляясь дальше заданных границ;

– регулировать скорость и направление движения в зависимости от изменения скорости и направления ведущего объекта;

– избегать столкновений с ведущим и прочими объектами.

Выполнение этих условий можно представить в виде награды, т.е. числа, которое характеризует правильность выполнения желаемого поведения ведомым. Общая награда $reward_t$ на момент времени $t$ вычисляется на основе следующих показателей:

• $d_{\min}$ – минимальное расстояние от ведомого до ведущего – дистанция, ближе которой ведомый не должен приближаться к ведущему;

• $d_{\max}$ – максимальное расстояние от ведомого до ведущего – дистанция, дальше которой ведомый не должен удаляться от ведущего;

• $dev$ – максимальное отклонение от маршрута – максимальное расстояние от ведомого до ближайшей точки маршрута ведущего;

• $d_{route}$ – текущее расстояние от ведомого до ближайшей точки маршрута;

• $d_{leader}$ – расстояние от ведомого до ведущего, вычисляемое как сумма расстояний между точками маршрута от ведущего до ближайшей к ведомому точке маршрута;

• $\varepsilon$ – расстояние, на котором объекты считаются находящимися в одной и той же точке.

В качестве базовой функции расчета награды реализована следующая:

• если $d_{leader} \le d_{\min}$, значит, ведомый находится слишком близко: $reward_t = reward_t - 5$;

• если $d_{route} \le \varepsilon_{pos}$, значит, ведомый находится на маршруте:

○ если $d_{\min} \le <= d_{leader} \le <= d_{\max}$, значит, ведомый держит нужную дистанцию от лидера на маршруте: $reward_t = reward_t + 1$;

○ если $d_{leader} > d_{\max}$, значит, ведомый отстал от ведущего: $reward_t = reward_t + 0.1$;

• если $d_{leader} > d_{\max}$, значит, ведомый отстал от ведущего: $reward_t = reward_t - 1$;

• если $d_{route} \le <= dev$, значит, ведомый находится в пределах допустимого расстояния от маршрута ведущего, но не на самом маршруте:

  ○ если $d_{\min} \le <= d_{leader} \le <= d_{\max}$, значит, ведомый держит нужную дистанцию от лидера на маршруте: $reward_t = reward_t + 0.5$;

• если $d_{route} > dev$, значит, ведомый сильно отклонился от маршрута $reward_t = reward_t - 1$;

• если границы объекта, который обозначает ведомого, пересеклись с границами объекта-ведущего или объекта-препятствия, $reward_t = reward_t - 10$ и симуляция мгновенно заканчивается (считается, что произошла авария).

Время в имитационной среде эмулируется в виде дискретной величины, измеряемой "кадрами" (*frame*). В среде реализован вариант, когда действие объекта длится в течение одного "шага" (*step*), которому соответствует переменное число "кадров", что эмулирует возможные задержки в передаче данных. Модель, определяющая поведение агента, получает состояние среды (наблюдение) как шаг, а ее задача – сгенерировать действие агента, которое будет применяться каждый кадр в течение одного шага.

Используемая в работе модель построена на основе архитектуры многослойного персептрона. В качестве входных признаков использован вектор из семи значений от 0 до 1, которые соответствуют относительному расстоянию до ближайшей точки из истории перемещений лидера, лежащей в одном из семи секторов перед ведомым роботом. Наблюдая за перемещением лидера, история его позиций запоминается ведомым через определенные промежутки времени. Для запоминаемых точек определяется их принадлежность к одному из семи секторов в полукруге перед роботом. После этого для каждого сектора выбирается ближайшая точка и оценивается расстояние до нее. Это расстояние делится на максимально допустимое расстояние до лидера (4 м). Полученный вектор из семи значений является входом для нейросетевой модели, описывающей текущее состояние среды для решаемой задачи следования за лидером (далее *X*). Обучение модели проводили на базе алгоритма PPO [14] с использованием разработанной симуляционной 2D-среды. PPO – это алгоритм обучения с подкреплением, который не моделирует полностью окружающий мир и его динамику (*model-free*), проводит обновление весов, опираясь только на историю переходов с использованием последнего варианта модели (*on-policy*), использует архитектуру актор-критик, включающую в себя модель поведения (согласно которой на основе наблюдений выбирается действие агента) и модель-критик (предсказывающую ожидаемую награду на основе наблюдений). Метод является более стабильным и эффективным в плане использования накопленных данных, чем стандартный алгоритм policy gradient [15], при этом более прост в реализации, чем *trpo* (*trust region policy optimization*) [16].

В ходе обучения веса модели модифицируются для максимизации целевой функции

$$L^{CLIP}(\theta) = \\ = \mathbb{E}_t\left[\min\left(r_t(\theta) \times A_t, clip\left(r_t(\theta), 1-\epsilon, 1+\epsilon\right) \times A_t\right)\right], \quad (1)$$

где $r_t(\theta) = \pi_\theta(a_t|s_t)/\pi_{\theta_{old}}(a_t|s_t)$ – отношение между логарифмами вероятностей выбранных действий $a_t$ для наблюдений $s_t$ для новой $\pi_\theta$ с обновленными весами и предыдущей $\pi_{\theta_{old}}$, которая была использована при сборе истории; $\theta$ – веса policy-модели, $A_t$ – преимущество, $\epsilon$ – гиперпараметр, используемый для ограничения целевой функции, позволяющий избегать слишком больших изменений весов модели.

Для оценки качества полученной модели в симуляционной среде было сгенерировано 100 тестовых маршрутов. Результирующая модель полностью прошла 42 маршрута из 100.

В качестве среды для применения обученной модели использовали 3D-мир на базе симулятора Gazebo [17], позволяющего более детально проводить физическое моделирование, представленный на рис. 2, 3.

3D-мир содержит сложный ландшафт, различные по форме и типу статические препятствия. Для решения задачи следования в 3D-мир добавлены два агента-робота: ведущий (лидер) и ведомый, габариты которых сопоставимы с их аналогами в 2D-мире. Ведомый робот оснащен различными сенсорами, которые используются для выработки управляющих сигналов. В табл. 1 представлено сопоставление характеристик эмуляционной среды (2D-мир), в которой проводилось обучение RL-модели, со средой ее использования (3D-мир).

Указанные различия требуют реализации соответствующего функционала в системе управления при осуществлении переноса обученной в эмуляционной среде RL-модели в среду использования.

*Система управления движением на базе RL-модели*. Основной сложностью переноса RL-модели является отсутствие эталонных координат ведущего в 3D-мире, что требует построения точек маршрута относительно ведомого. Схема взаимодействия основных компонентов системы управления и среды представлена на рис. 4.

Назначения основных компонентов системы и входные/выходные данные описаны далее.

Детекцию осуществляли на основе анализа изображения с камеры ведомого в симуляционном мире Gazebo. Обработку изображения вы-

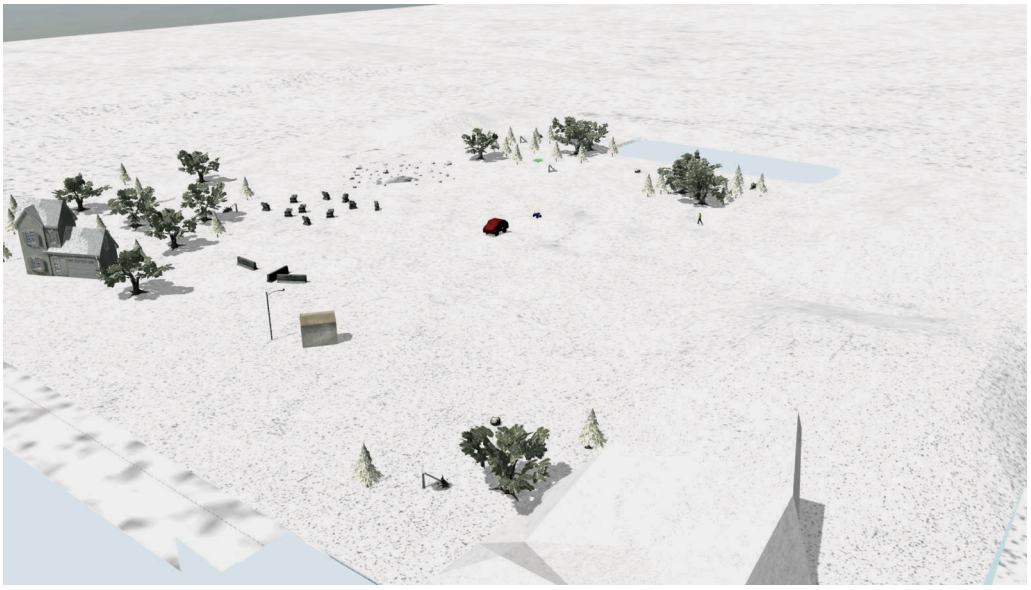

**Рис. 2.** Пример симуляционного 3D-мира в Gazebo.

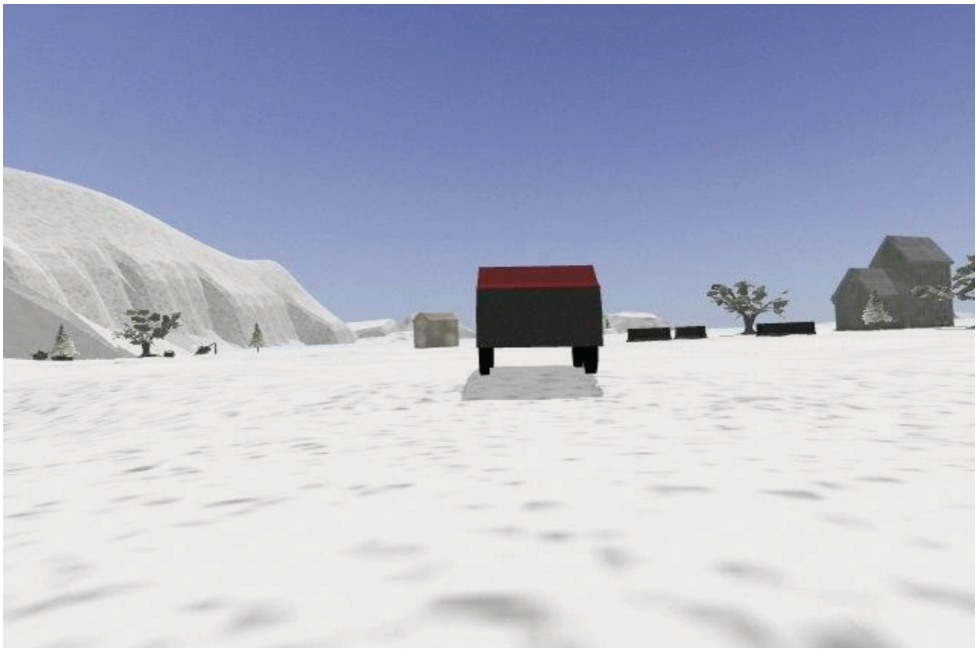

**Рис. 3.** Пример симуляционного 3D-мира с камеры ведомого.

полняли с использованием сверточной нейросетевой модели Single Shot Multibox Detection (**SSD**) [18]. Задача модели – определить набор границ для детектируемых на изображении объектов (*bounding box*) и оценить наличие экземпляров классов объектов в этих границах. Для обучения SSD собрали набор данных, состоящий из изображений с камеры ведомого с разрешением 640 × 480 в формате “JPEG”, который содержит целевой объект интереса – лидера. Каждое изображение размечали в соответствии с информацией о границах объекта в виде прямоугольника и классе целевого объекта. Всего собрали и разметили 246 изображений. Для обучения SSD использовали 80% от общего числа данных. Точность детекции составила 71% по метрике mAP.

Основой для вычисления координат ведущего являются:

– ориентация агента-ведомого в 3D-мире ($\psi_{robot}$);

**Таблица 1.** Сопоставление эмуляционной среды обучения и среды применения

| Признаки | 2D-мир | 3D-мир |
|---|---|---|
| Число измерений пространства | 2D | 3D |
| Шаг симуляции | По тактам (шагам) | По времени |
| Наблюдения | Точки маршрута | Данные с сенсоров |
| Физические свойства среды | Кинематическая | Динамическая |

– положение камеры ведомого ($\psi_{camera}$);

– $\theta$ – угол между направлением камеры ведомого и положением лидера на изображении с камеры по горизонтальной оси;

– $l$ – расстояние от ведомого до лидера.

Значения $\theta$ и $\varphi$ вычисляли с использованием формулы (1).

$$f(k, R, \gamma) = \arctan\left(\frac{2 * k - R}{R} * \tan\left(\frac{\gamma}{2}\right)\right). \quad (2)$$

Здесь $k$ – точка на изображении в пикселях, $R$ – разрешение изображения с камеры ведущего, $\gamma$ – угол обзора камеры ведущего (град).

Для определения $\theta$ использовали следующие аргументы функции (2): $k$ – точка в центре прямоугольной области ведущего по горизонтальной оси, $R = 640$, $\gamma = 80$.

Для определения расстояния до ведущего необходимо осуществить фильтрацию всех точек, полученных с лидара, и оставить только те точки, которые относятся к ведущему. Выделенный массив точек использовали для определения расстояния до ведущего. Проводили расчет расстояния $l$ до всех отфильтрованных точек лидара, представленных в виде значений по трем координатам: $x$ – глубина, $y$ – ширина, $z$ – высота:

$$l = \sqrt{x^2 + y^2 + z^2}. \quad (3)$$

По минимальному $l$ выделяли соответствующие ему координаты точки лидара, используемые для обновления координат лидера (формулы (5), (6)). Для этого с использованием формулы (4) определяли $\theta_{leader}$ – ориентация ведущего в пространстве:

$$\theta_{leader} = \theta + \psi_{robot} + \psi_{camera}. \quad (4)$$

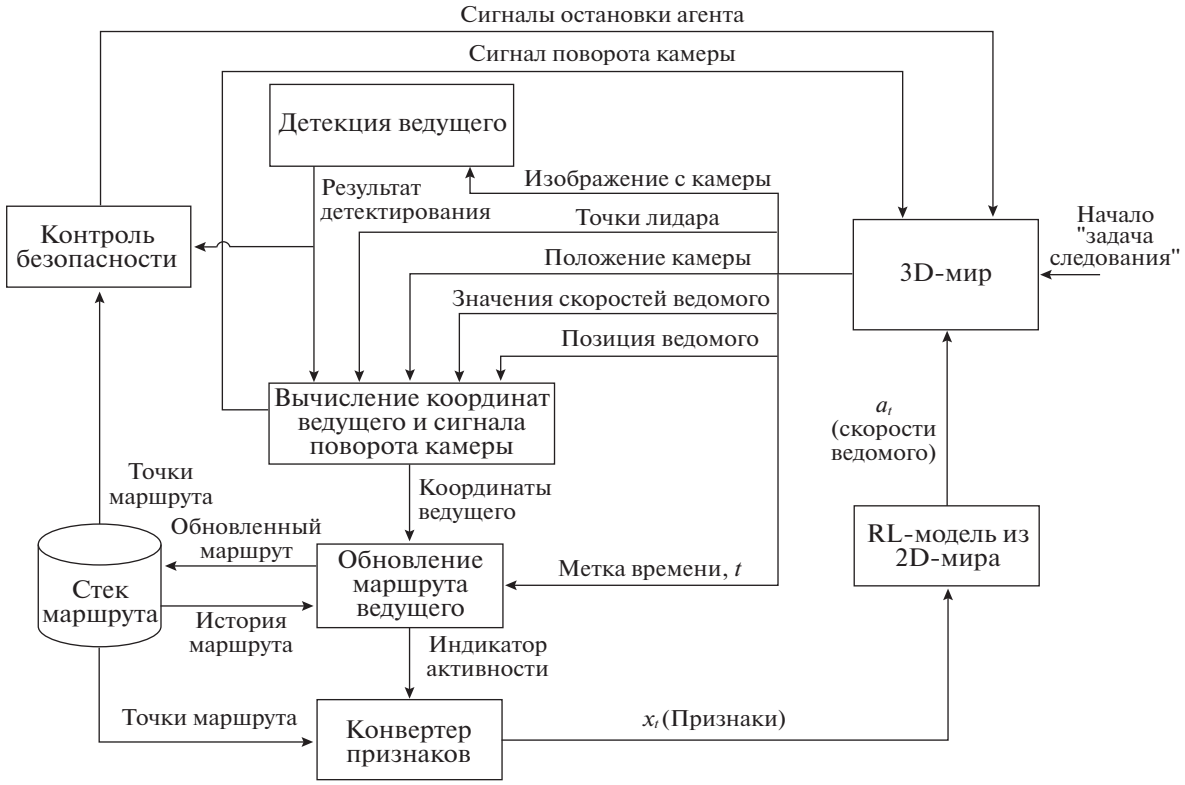


**Рис. 4.** Архитектура системы для решения задачи следования за лидером на базе RL-модели.

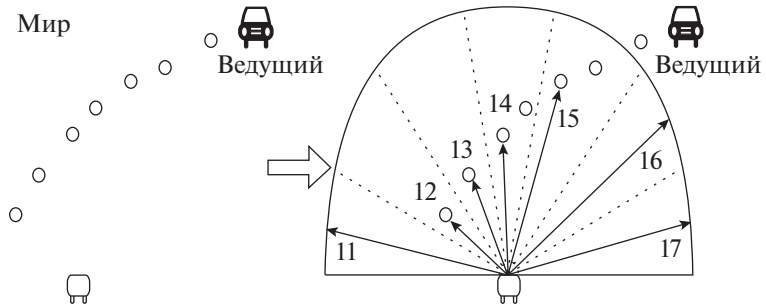


**Рис. 5.** Схематичное представление сопоставления маршрута лидера и входных признаков RL-модели.

Координаты лидера относительно ведомого рассчитывали как

$$x = x_0 + l * \cos(\theta_{leader}), \quad (5)$$

$$y = y_0 + l * \sin(\theta_{leader}), \quad (6)$$

где $x_0$, $y_0$ – координаты ведомого, имеющие значения (0,0).

С помощью значения ($x$, $y$) на каждой итерации получения состояния среды для решаемой задачи формировали историю маршрута ведущего.

Реализована функция поворота камеры ведомого при поворотах ведущего, обеспечивающая прерывание получения координаты лидера при резких поворотах. Желаемое положение камеры ведомого $\psi_{camera}(t+1)$ рассчитывали на основе формулы (7), в результате чего формируется управляющий сигнал поворота камеры:

$$\psi_{camera}(t+1) = \psi_{camera}(t) + \theta, \quad (7)$$

где $\psi_{camera}(t)$ – положение камеры в текущий момент времени.

Координаты $x$ и $y$ с текущего момента времени $t$ сохраняются в “стек маршрута”.

На основе изложенного выше на каждом временном шаге $t$ история маршрута ведущего ($S$) дополняется рассчитанными координатами ($x$, $y$) ведущего относительно ведомого. Однако в момент времени $t + 1$ ведомый находится уже в другом положении относительно положения ведущего в момент времени $t$. Поэтому для формирования актуальных значений признаков на основе истории маршрута и текущего положения ведомого необходимо скорректировать все точки в истории маршрута ведущего на некоторую переменную $\Delta C$ ($\Delta C_x$, $\Delta C_y$), определяющую изменение положения ведомого с шага $t$ до шага $t + 1$ (т.е. за время $\Delta t$).

Расчеты перемещений по каждой координате проводили с использованием формул

$$\Delta C_x = \Delta V_x * \Delta t, \quad (8)$$

$$\Delta C_y = \Delta V_y * \Delta t, \quad (9)$$

где $\Delta V_x$ – средняя продольная скорость робота, которая рассчитывается как $\Delta V_x = \frac{V_x(t) + V_x(t+1)}{2}$, $\Delta V_y$ – поперечная скорость робота, рассчитываемая как $\Delta V_y = \frac{V_y(t) + V_y(t+1)}{2}$.

На каждом шаге обновления координат все координаты маршрута изменяются на $\Delta C$. Пусть $S(t) = \{[x(t), y(t)]\}$, тогда в момент времени $t + 1$ история дополняется новыми координатами ведущего, а предыдущие значения координат изменяются на $\Delta C$:

$$S(t+1) = \{[x(t) - \Delta C_x(t+1), y(t) - \Delta C_y(t+1)], [x(t+1), y(t+1)]\}. \quad (10)$$

Точка удаляется из истории в случае ее нахождения от ведомого в радиусе 0.9 м.

Задача конвертора – осуществить преобразование обновленной истории точек маршрута ведущего в признаки для RL-модели. Схематическое представление этого преобразования представлено на рис. 5.

В процессе работы в каждый момент времени $t$ на конвертер поступает обновленная история точек маршрута ведущего. Конвертер осуществляет проверку нахождения точки в одном из семи секторов в диапазоне от 0° до 180°. Если в секторе отсутствуют точки из истории, тогда выбирается максимальное расстояние до границы радара (25 м), как, например, l1, l6, l7, показанные на рис. 5. Если в зоне присутствует точка из истории маршрута, для данного сектора в данный момент времени записывается расстояние до данной точки, как, например, l2, l3. В случае, когда в сектор попадает несколько точек из истории маршрута ведущего, выбирается точка, ближайшая к роботу и аналогичным образом для данного сектора записывается расстояние до ближайшей точки в секторе, например l4, l5. Далее записанные расстояния для каждого сектора нормируются на выбранную максимальную дальность (25 м). В ре-

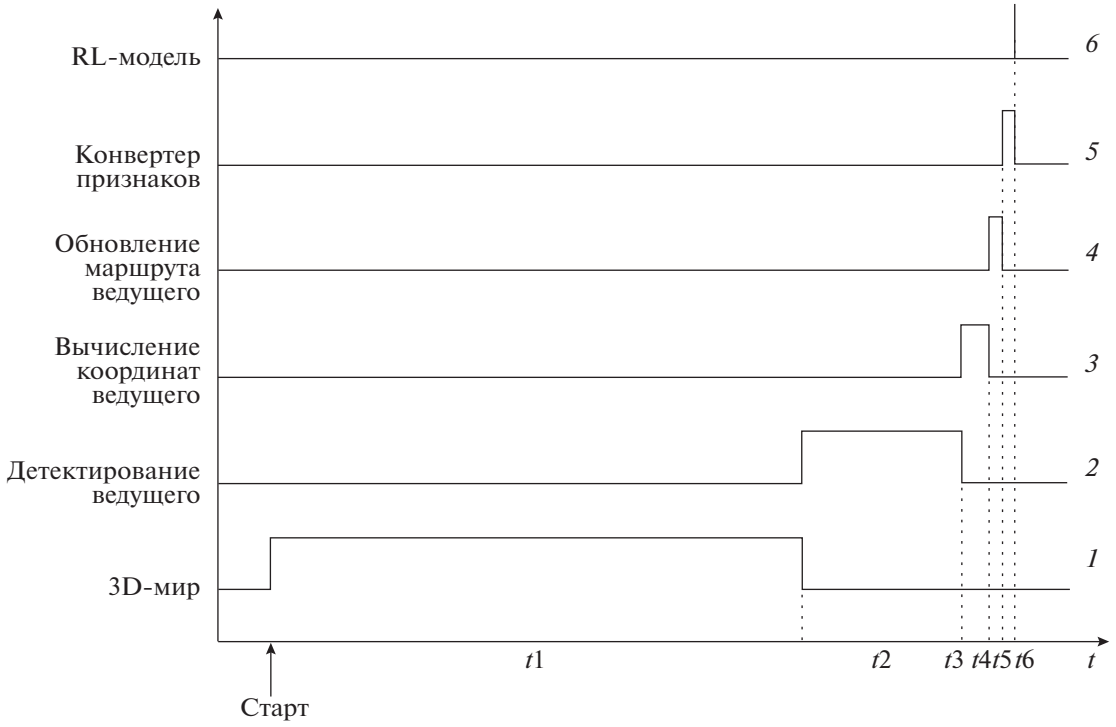


**Рис. 6.** Диаграмма времени работ компонентов системы 3D-мир – $t1$ (*1*), детектирование ведущего – $t2$ (*2*), вычисление координат ведущего – $t3$ (*3*), обновление маршрута ведущего – $t4$ (*4*), конвертер признаков – $t5$ (*5*), RL-модель – $t6$ (6).

зультате данные истории маршрута записываются в семь значений от 0 до 1, которые являются входными признаками RL-модели.

С целью предотвращения нештатных ситуаций, таких как потеря видимости ведущего, отсутствие истории точек маршрута, близкое нахождение к ведущему, в контроллере реализован модуль безопасности.

В случае если на изображении с камеры ведомого отсутствует ведущий, то идет отправка соответствующего сообщения, которая вызывает остановку ведущего. При этом если в истории маршрута ведущего присутствуют точки, то ведомый продолжает свое движение. По прибытии в последнюю точку ведомый осуществляет остановку и продолжает отправлять информацию о потере ведущего. В случае, когда ведомый снова видит ведущего, происходит отправка сообщения с информацией о том, что он найден и можно продолжить движение. При близком нахождении к ведущему осуществляется принудительная остановка ведомого.

## РЕЗУЛЬТАТЫ И ИХ ОБСУЖДЕНИЕ

Применение разработанной системы требует корректировки подбора ряда настроек, связанных в первую очередь со временем выполнения действия, зависящим от скорости выработки сигналов разработанной системой управления.

На рис. 6 представлена характерная диаграмма времен работы компонентов системы управления, RL-модели и 3D-мира.

Запуск системы проводили на оборудовании следующей конфигурации.

Детектирование ведущего проводится на сервере с видеокартой Nvidia 1660 Super, процессором Intel Xeon E5-1620 v3 3.50 GHz, оперативной памятью 16 Gb.

RL-модель запускается на сервере с видеокартой Nvidia GT 1030, процессором Intel Core i5-7400 3.00 GHz, оперативной памятью 16 Gb.

Запуск 3D-мира, вычисление координат ведущего и конвертацию признаков проводили на компьютере без видеокарты с процессором Intel Core i7-3770 3.40 GHz и оперативной памятью 8 Gb.

В табл. 2 представлены значения времен работы компонентов системы на выбранном оборудовании. Самым продолжительным по времени является шаг применения действия в 3D-мире. В нем выполняется заданное с RL-модели действие на протяжении 0.5 с. Далее происходит детектирование ведущего, что занимает в среднем 0.15 с. Обработка изображения на сервере занимает 0.04 с. Оставшееся время занимает передача ин-

**Таблица 2.** Значения времени работы компонентов системы

| Компонент | Переменная | Время работы, с |
|---|---|---|
| Gazebo (действие) | $t1$ | 0.5 |
| Детектирование ведущего | $t2$ | ~0.15 |
| Вычисление координат ведущего | $t3$ | ~0.02 |
| Обновление координат ведущего | $t4$ | ~0.005 |
| Конвертер признаков | $t5$ | ~0.0005 |
| RL-модель | $t6$ | $\sim 5.7 \times 10^{-6}$ |
| Суммарное время | | ~0.7 |

формации. Далее происходят вычисление координат ведущего, обновление истории маршрута ведущего и формирование вектора признаков для RL-модели, что в сумме занимает ~0.025 с. Цикл начинается заново после получения результата с RL-модели.

Анализ эффективности системы оценивали по формуле расчета награды, используемой при обучении RL-модели (описано выше)

На 100 тестовых запусков с разной длительностью маршрута ведущего (от 50 до 150 м) задача следования за лидером в 90 случаях выполнена, суммарная награда составила 3 тыс. очков из возможных 6 тыс., что объясняется непостоянным нахождением в заданной дальности от ведущего без потери видимости. В оставшихся 10 случаях решение задачи было прервано блоком безопасности из-за потери ведомым видимости следования ведущего.

## ЗАКЛЮЧЕНИЕ

Предложен подход к решению задачи следования за лидером на базе модели нейронной сети, обученной по принципам обучения с подкреплением в упрощенном симуляционном 2D-мире. На его основе создана система управления движением ведомого робота в условиях, близких к реальным, в 3D-мире с моделированием физики движения, связанной с ландшафтом. Разработанная система предусматривает отсутствие координат ведущего относительно ведомого и их расчет с использованием сенсоров ведомого: лидара и изображения с камеры. В результате апробации созданной системы 90% тестовых маршрутов пройдены успешно, тогда как в 10% случаев произошла принудительная остановка из-за потери видимости ведущего. Таким образом, реализованный подход может использоваться для решения задачи “следования за лидером”.

Работа выполнена при поддержке НИЦ “Курчатовский институт” (приказ № 2754 от 28.10.2021).

## СПИСОК ЛИТЕРАТУРЫ

1. *Rösmann C., Feiten W., Wösch T. et al.* // ROBOTIK. Proc. 7th German Conference on Robotics, Germany, Munich, 2012. P. 74.
2. *Fox D., Burgard W., Thrun S.* // IEEE Robotics Automation Magazine. 1997. V. 4 (1). P. 23.
3. *Akkaya I., Andrychowicz M., Chociej M. et al.* // International Conference on Computer, Control, Electrical, and Electronics Engineering (ICCCEEE). 2020.
4. *Lee J., Hwangbo J., Wellhausen L. et al.* // Science robotics 5.47 (2020): eabc5986.
5. *Bellegarda G., Nguyen Q.* // “Robust quadruped jumping via deep reinforcement learning.” arXiv preprint arXiv:2011.07089 (2020).
6. *Song Y., Steinweg M., Kaufmann E., Scaramuzza D.* // IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). 2021. P. 1205.
7. *Azar A.T., Koubaa A., Ali Mohamed et al.* // Electronics 2021. V. 10 (9). P. 999.
8. *Yue L., Yang R., Zhang Y. et al.* // “Deep Reinforcement Learning for UAV Intelligent Mission Planning.” Complexity 2022.
9. *Liu C., Van Kampen E.J.* // AIAA SCITECH 2022 Forum. 2022. P. 0793.
10. *Salvato E., Fenu G., Medvet E., Pellegrino F.A.* // IEEE Access 2021. V. 9. P. 153171.
11. *Xie J., Zhou R., Liu Y. et al.* // Appl. Sci. 2021. V. 11 (2). P. 546.
12. *Шеин В.А., Сбоев А.Г., Рыбка Р.Б.* // ЛАПЛАЗ-2022. Сборник научных трудов. P. 121.
13. A continuous environment for reinforcement learning of the task of the following the leader. https://github.com/sag111/ContiniousEnvironment_-Follower_Leader
14. *Schulman J., Wolski F., Dhariwal P. et al.* // arXiv preprint arXiv:1707.06347v2. 2017.
15. *Sutton R.S., McAllester D., Singh S., Mansour Y.* // Advances in neural information processing systems. 1999. V. 12.
16. *Schulman J., Levine S., Moritz P. et al.* // CoRR, abs/1502.05477. 2015.
17. *Koenig N., Howard A.* // IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) V. 3. 2004. P. 2149.
18. *Liu W., Anguelov D., Erhan D. et al.* // Computer Vision–ECCV 2016: 14th European Conference. 2016. V. 1 (14) P. 21.