УДК 004.896 **DOI** 10.15622/ia.20.6.3

Е.Ю. Шмалько, Ю.А. Румянцев, Р.Р. Байназаров, К.Л. Ямшанов

ИДЕНТИФИКАЦИЯ НЕЙРОСЕТЕВОЙ МОДЕЛИ РОБОТА ДЛЯ РЕШЕНИЯ ЗАДАЧИ ОПТИМАЛЬНОГО УПРАВЛЕНИЯ

Шмалько Е.Ю., Румянцев Ю.А., Байназаров Р.Р., Ямшанов К.Л. **Идентификация нейросетевой модели робота для решения задачи оптимального управления.**

Аннотация. Для расчета оптимального управления требуется достоверная математическая модель объекта управления. В дальнейшем при реализации расчетных управлений на реальном объекте эта же модель может быть использована в навигации робота для прогнозирования его положения и корректировки показаний сенсоров, поэтому важно, чтобы модель достаточно адекватно отражала динамику объекта. Вывод модели часто требует значительного времени и иногда даже невозможен с использованием традиционных методов. Ввиду все большего разнообразия и чрезвычайно сложной природы объектов управления, включая разнообразие современных робототехнических систем, все большую актуальность приобретает задача идентификации, которая позволяет построить математическую модель объекта управления, имея входные и выходные данные о системе. Идентификация нелинейной системы представляет особый интерес, так как большинство реальных систем имеют нелинейную динамику. И если раньше идентификация модели системы заключалась в подборе оптимальных параметров для выбранной структуры, то появление современных методов машинного обучения открывает более широкие перспективы и позволяет автоматизировать сам процесс идентификации. В настоящей работе в качестве объекта управления рассматривается колесный робот с дифференциальным приводом в симуляционной среде Gazebo, которая на сегодняшний день является наиболее популярным программным пакетом при разработке и моделировании робототехнических систем. Математическая модель робота заранее неизвестна. Основная проблема заключается в том, что существующие математические модели не соответствуют реальной динамике робота в симуляторе. В работе рассматривается решение задачи идентификации математической модели объекта управления с помощью машинного обучения на основе нейронной сети. Представлен новый смешанный подход, основанный на использовании известных простых моделей объектов и идентификации неучтенных динамических свойств объекта с помощью нейронной сети на основе обучающей выборки. Для формирования обучающих данных был написан программный пакет, автоматизирующий процесс сбора с помощью двух ROS-узлов. Для обучения нейросети использовался фреймворк PyTorch и был создан программный пакет с открытым исходным кодом. Далее идентифицированная модель объекта используется для расчета оптимального управления. Результаты вычислительного эксперимента демонстрируют адекватность и работоспособность полученной модели. Представленный подход на основе комбинации известной математической модели и дополнительной идентифицированной нейросетевой модели позволяет использовать преимущества накопленного физико-математического аппарата и повысить его эффективность и точность за счет использования современных средств машинного обучения.

Ключевые слова: оптимальное управление, идентификация, нейронная сеть, Gazebo, дифференциальный робот.

1. Введение.

Расчет оптимального управления известными методами осуществляется на основе математической модели объекта в

пространсве состояний, представленной, как правило, в виде системы дифференциальных уравнений.

Традиционно динамические модели, описывающие интересующую систему, выводятся с использованием основных физико-математических законов [1]. В этом случае получение математической модели объекта управления - сложный процесс, требующий подробных специальных знаний об объекте управления. Разработка таких моделей может быть очень трудоемкой и, следовательно, дорогостоящей. Современное бурное развитие робототехники требует быстрых, универсальных и автоматических подходов к определению моделей объектов.

Современный альтернативный способ получения модели объекта - это ее идентификация, установление тождественности известному объекту. В задаче идентификации математическая модель объекта управления не известна полностью или частично, но исследователь имеет реальный объект управления или его симулятор [2]. Необходимо найти неизвестную многомерную функцию, описывающую динамику объекта. Пространство входных векторов этой функции есть пространство допустимых управлений для этого объекта.

Важным шагом в идентификации системы является определение типа используемой модели, поскольку в подавляющем большинстве методов реализуется так называемая параметрическая идентификация [3, 4]. Сначала выбирается определенная структура модели, которая считается подходящей для описания данного объекта. Далее проводится идентификационный эксперимент, в котором измеряются входные и выходные сигналы, а затем метод идентификации реализует настройку параметров модели в соответствии с некоторыми адаптивными законами, чтобы реакция модели на входной сигнал могла приблизительно соответствовать отклику реальной системы к тому же входному воздействию. Чаще всего используют идентификацию объектов с помощью линейных систем [5, 6], поскольку для них легко определить влияние различных входных сигналов на выход. Хотя линейные модели привлекательны по многим причинам, но они имеют существенные ограничения. В настоящее время возрос значительный интерес к методам идентификации нелинейных систем [7–9], с помощью методов машинного обучения на основе нейронных сетей [10–12] и методов символьной регрессии [13]. Они позволяют реализовывать не только параметрическую, но и структурную идентификацию систем [14].

Нейронные сети, являясь универсальным аппроксиматором, предоставляют мощный инструмент для идентификации нелинейных систем. В виду широко распространения и доступности программного обеспечения нейронные сети завоевали огромную популярность [15–18]. Были

разработаны и комбинированные подходы, основанные на эталонной модели [19, 20]. В работе [21] представлен пример нейросети с прямой связью для идентификации модели небольшого вертолета. Представлено сравнение офлайн и онлайн обучения. В статье [22] рассмотрен подход с использованием рекурентных нейросетей долгой краткосрочной памяти. Данный тип нейросети устойчив к изменяющемуся временному шагу и обычно используется для анализа аудио и видео. Существует не так много работ по идентификации сложных объектов управления. В работах [23, 24], представлен новый алгоритм SINDy (Sparse identification of nonlinear dynamical systems), позволяющий описывать динамику системы. Авторы продемонстрировали работоспособность алгоритма на широком круге задач, включая линейные и нелинейные осцилляторы и хаотическую систему Лоренца. Однако, данный алгоритм требует высокого качества входных данных. Известны также подходы к идентифицкации нейросетевых моделей [25, 26] мобильных роботов, но они еще не получили должного развития.

В статье представлена реализация смешанного подхода к идентификации объекта управления на основе применения нейросетей и расчета оптимального управления для полученной модели.

В качестве объекта управления рассматривается колесный мобильный робот. Разрабатывается программная реализация системы управления робота с использованием операционной системы ROS. Она предоставляет возможность работать со всеми аспектами системы управления, включая аппаратную абстракцию, низкоуровневое управление, передачу сообщений между процессами, и управление пакетами.

Разрабатываемые программные комплексы отрабатывались в имитационной среде-симуляторе Gazebo, которая является одним из популярнейших робототехнических симуляторов, благодаря своей совместимости с ROS. Gazebo - это 3D-симулятор, цель которого смоделировать робота так, чтобы дать близкую замену тому, как робот будет вести себя в реальной физической среде. Gazebo имеет достаточно достоверную симуляцию физики и различных физических явлений, учитывает влияние сил, а также имеет большое количество плагинов для симуляции работы датчиков, например лидаров или камер. Благодаря этим фактам, большинство разработчиков систем управления робототехнических систем по всему миру, используют данную связку ROS/Gazebo как стандарт для отработки разрабатываемых алгоритмов управления [27, 28].

В настоящей работе была использована готовая Gazebo модель ROSbot 2.0, интегрированная в ROS [29]. В качестве источника позиции,

использовался плагин, дающий истинные координаты робота. Модель робота в симуляторе представлена на рисунке 1.

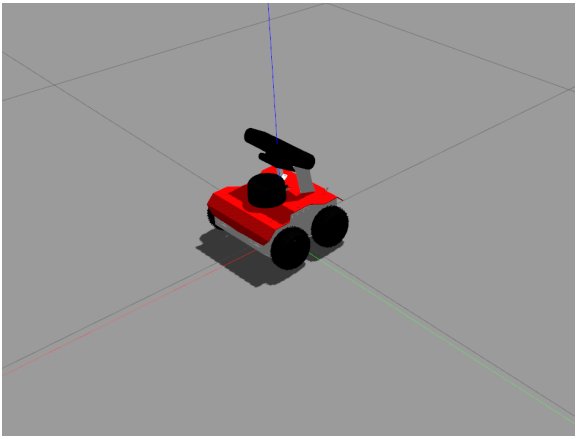


Рис. 1. ROSbot в симуляторе Gazebo

Представлен смешанный подход к идентификации, когда частично модель объекта известна, исходя из физических законов, а другая часть определяется нейронной сетью, обученной на основе специально сформированной обучающей выборки.

После реализации этапа идентификации, полученная модель с целью апробации была использована для численного решения задачи оптимального управления. Представленные результаты моделирования и симуляции в Gazebo демонстрируют адекватность полученной нейросетевой модели.

Получение рабочей адекватной модели реального объекта управления открывает возможность для разработки реализуемых алгоритмов оптимального управления, учитывающих тонкости динамики объекта.

2. Идентификация объекта управления. В работе используется виртуальный робот ROSbot 2.0, реализованный в среде для физической симуляции Gazebo. Робот представляет собой платформу на четырех неповорачивающихся вокруг вертикальной оси колесах. К каждому колесу присоединен электромотор. Используется дифференциальная схема управления: движение робота вперед и назад осуществляется путем подачи одинакового напряжения на все четыре электромотора; поворот робота вправо или влево осуществляется за счет подачи большего напряжения на электромоторы левых или правых колес соответственно.

Система координат, используемая для определения модели робота, показана на рисунке 2. Состояние робота в каждый момент времени определяется набором чисел: $\{x, y, \theta, v, \omega\}$, где x, y – координаты робота на плоскости, θ – угол между осью x и направлением робота, v – проекция скорости движения робота $\vec{v}$ на направление робота $\vec{h}$, ω – угловая скорость. Предполагается, что вектора $\vec{v}$ и $\vec{h}$ коллинеарны.

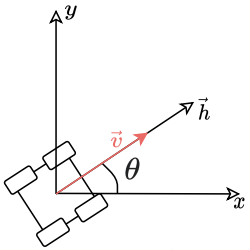


Рис. 2. Система координат робота. Вектор $\vec{h}$ показывает направление робота при движении вперед. Вектором $\vec{v}$ обозначена скорость движения робота.

Модель движения робота описывается известной системой дифференциальных уравнений [30]:

$$\begin{cases} \dot{x} = v\cos(\theta) \\ \dot{y} = v\sin(\theta) \\ \dot{\theta} = \omega \end{cases}. \tag{1}$$

Физически управление роботом в Gazebo реализовано с помощью двух сигналов: u^v – желаемая линейная скорость; u^ω – желаемая угловая скорость. Во многих работах для расчета оптимальных движений используется модель, в соответствии с которой сигналы управления напрямую передаются в систему $v = u^v$, $\omega = u^\omega$. В этом случае уравнения (1) преобразуются в следующие:

$$\begin{cases} \dot{x} = u^v\cos(\theta) \\ \dot{y} = u^v\sin(\theta) \\ \dot{\theta} = u^\omega \end{cases}. \tag{2}$$

Однако, на практике наблюдается совершенно иная картина. Во-первых, скорости не могут меняться бесконечно быстро. У системы всегда есть определенная динамика. Во-вторых, как правило, существуют некоторый регулятор на более низком уровне, который непосредственно управляет скоростью вращения колес (в симуляторе) или напряжением,

выдаваемым на моторы (на реальном роботе). Поэтому для адекватного описания модели движения робота, в систему (1) введены два дополнительных уравнения, описывающих инерционную часть системы.

$$\begin{cases} \dot{x} = v\cos(\theta) \\ \dot{y} = v\sin(\theta) \\ \dot{\theta} = \omega \\ \dot{v} = f^v(v, \omega, u^v, u^\omega) \\ \dot{\omega} = f^\omega(v, \omega, u^v, u^\omega) \end{cases}, \tag{3}$$

где f^v, f^ω - неизвестные функции, которые необходимо идентифицировать. Такие факторы, как работа неизвестного регулятора, трение колес о поверхность, инерция, неравномерное распределение массы робота не позволяют записать функции f^v, f^ω в явном виде.

Для численной реализации задачи идентификации запишем уравнение (3) в конечно-разностной форме при временном шаге Δt:

$$\begin{cases} x_{k+1} = x_k + \Delta t \cdot v_k cos(\theta_k) \\ y_{k+1} = y_k + \Delta t \cdot v_k sin(\theta_k) \\ \theta_{k+1} = \theta_k + \Delta t \cdot \omega_k, \\ v_{k+1} = F^v(v_k, \omega_k, u_k^v, u_k^\omega, \Delta t) \\ \omega_{k+1} = F^\omega(v_k, \omega_k, u_k^v, u_k^\omega, \Delta t) \end{cases}. \tag{4}$$

Для идентификации системы в работе было использовано приближение функций F^v, F^ω параметрическими функциями F_ϕ^v и F_ϕ^ω. Обучаемая функция с параметрами ϕ принимает на вход текущее состояние робота, вектор управления и временной шаг и выдает состояние робота в следующий момент времени:

$$\begin{cases} v_{k+1} = F_\phi^v(v_k, \omega_k, u_k^v, u_k^\omega, \Delta t) \\ \omega_{k+1} = F_\phi^\omega(v_k, \omega_k, u_k^v, u_k^\omega, \Delta t) \end{cases}. \tag{5}$$

Параметры ϕ подбираются таким образом, чтобы траектория движения объекта, описываемого системой уравнений (4) с моделью (5), как можно ближе повторяла реальную траекторию движения робота в Gazebo.

Ниже сравнивается модель (2) с моделью (3), чтобы понять какое преимущество дает учет динамики системы и неизвестного регулятора при решении задачи оптимального управления.

2.1. Формирование обучающей выборки. Был выбран принцип обучения с учителем, при котором нейросеть обучается с помощью набора

данных "вход - эталонный выход", а затем проверяется с помощью набора валидационных и тестовых данных, которые не попадали в обучающую выборку.

Для сбора данных использовался симулятор Gazebo и написан программный пакет для автоматизации процесса сбора [31] с помощью двух основных ROS-узлов (ROS-nodes).

Первый узел – генератор управления, публикующий сгенерированную по определенным правилам последовательность управления для робота. Длительность одного проезда ~ 50 с. Для каждого нового эксперимента случайным образом выбирались следующие параметры генератора управления: период изменения линейной скорости, период изменения угловой скорости, предельные значения и ускорения для линейной и угловой скоростей. Весь проезд разбивался на несколько периодов изменения скорости. В течение одного периода, робот ускорялся или замедляется с заданным ускорением, скорость не изменялась при достижении заданного предела. По завершении каждого периода знак ускорения менялся.

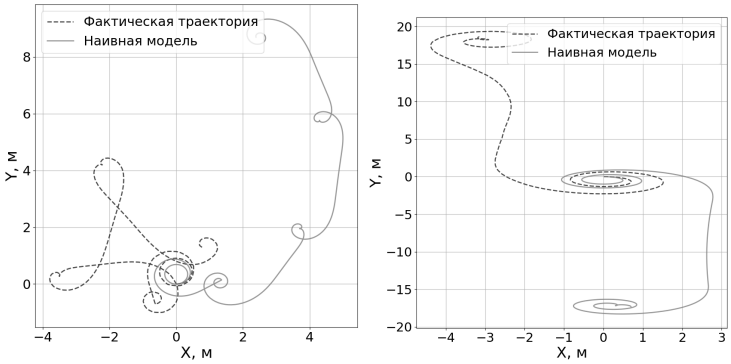
Второй ROS узел сохранял последовательность истинных координат и векторов состояния робота в течение движения, текущее управление и временную отметку. Временной шаг между отметками для каждого проезда генерировался случайно с равномерным распределением от 0.03 до 0.2 сек. Планируемый временной шаг для работы НС равнялся 0.1 сек. Варьирование временного шага использовалось для расширения диапазона работы НС. Часть данных, таких как примеры агрессивного вождения, собиралась с помощью телеуправления.

На рисунке 3 представлены примеры собираемых траекторий. Можно увидеть, как сильно расходятся движения простой модели и реального робота в симуляторе при одинаковых управляющих воздействиях.

Всего было собрано 238 траекторий, из низ 202 использовалось для обучения, 18 для валидации и 18 для тестирования нейросети.

Для того, чтобы нейросеть после обучения с учителем могла предсказать следующее состояние системы, набор данных для обучения должен содержать как можно больше всевозможных состояний системы [11]. Поэтому, диапазон собраных значений для линейной скорости принимался равным от -1.5 до 1.5 м/с, а угловой скорости — от -2.5 до 2.5 рад/с. Собранные данные имеют нормальное распределение (см. рисунок 4). По диагонали показаны частоты различных значений величин v, ω, u_v, u_ω. Вне диагонали показаны совместные распределения разных величин.

Для простой модели (2) также сохраняется последовательность координат и векторов состояния в течение движения, для того, чтобы в дальнейшем оценить эффективность работы нейросетевой модели. Сбор



(a) (b)

Рис. 3. Пример траекторий собраных с помощью генератора управления

данных о поведении простой модели происходит одновременно со сбором данных о роботе.

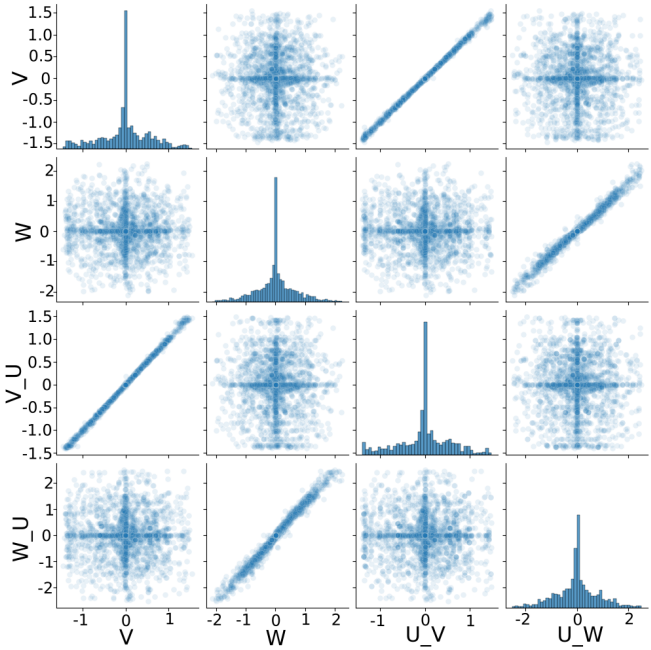


Рис. 4. Распределение данных

2.2. Обучение нейронной сети. В качестве обучаемой модели (5) была выбрана искуственная нейронная сеть с архитектурой "многослойный персептрон". Конкретные параметры модели описаны ниже.

Программное обеспечение. Для обучения нейросети использовался фреймворк PyTorch и был создан программный пакет с открытым исходным кодом для обучения нейросетевой модели [32]. Обучение реализовано с помощью трех основных классов, написанных на языке python.

Класс модели робота RosbotModel - данный класс содержит нейросеть, методы для расчета функции потерь, метрик, функцию перехода в следующее состояние. Для нейросети можно задать необходимые параметры: количество слоев, нейронов, тип модели и т.д. Можно сказать, что RosbotModel предоставляет интерфейс для взаимодействия с нейросетевой моделью, а также методы для оценки ее эффективности.

Класс RosbotDataset представляет собой структуру, содержащую набор данных об одном собранном проезде. Каждый объект класса содержит информацию о залогированном проезде и предоставляет интерфейс для взаимодействия с данными.

Класс Trainer содержит методы для непосредственного обучения нейронной сети. Данные методы взаимодействуют с объектами RosbotModel и массивами из RosbotDataset.

Работа выполнялась с использованием инфраструктуры Центра коллективного пользования «Высокопроизводительные вычисления и большие данные» (ЦКП «Информатика») ФИЦ ИУ РАН (г. Москва).

Метрика качества модели. Для оценки качества модели оценивалась точность предсказания траектории движения моделью на проездах, которые не были включены в обучающую выборку. Нейросеть получает на вход начальное состояние системы и последовательность управления для тестового проезда. Как и на этапе обучения, предсказывается траектория движения робота. Для сравнения предсказанной и действительной траекторий была использована метрика ATE – средний модуль поступательного смещения (absolute translation error):

$$ATE = \frac{1}{N}\sum_{t=0}^{N}\sqrt{(x_t - \hat{x}_t)^2 + (y_t - \hat{y}_t)^2},$$

где N - количество точек в траектории; x_t, y_t – координаты точки фактической траектории; $\hat{x}_t$, $\hat{y}_t$ – координаты точки предсказанной траектории.

Для оценки качества предсказанного угла была использована метрика MAE – средний модуль отклонения (mean absolute error).

$$MAE = \frac{1}{N}\sum_{t=0}^{N}|yaw_t - \hat{yaw}_t|,$$

где yaw_t – фактическая величина угла рысканья; $\hat{yaw}_t$ – предсказанное значение угла рысканья.

Многошаговая функция потерь. При обучении нейросети минимизируется отклонение предсказанных состояний $\{\mathbf{x}_k\}_{k=n}^{n+N}$ от истинных $\{\hat{\mathbf{x}}_k\}_{k=n}^{n+N}$ на отрезке траектории $k \in [n, n+N]$

$$\mathcal{L} = \sum_{k=n}^{n+N}(\mathbf{x}_k - \hat{\mathbf{x}}_k)^2. \tag{6}$$

Начало отрезка n выбирается случайно. Длина отрезка N является внешним параметром. Предсказание состояний $\{\mathbf{x}_k\}_{k=n}^{n+N}$ делается последовательно, начиная с известного начального состояния $\hat{\mathbf{x}}_n$. При этом в нейросеть на каждом шаге, начиная со второго, подается предыдущее предсказание:

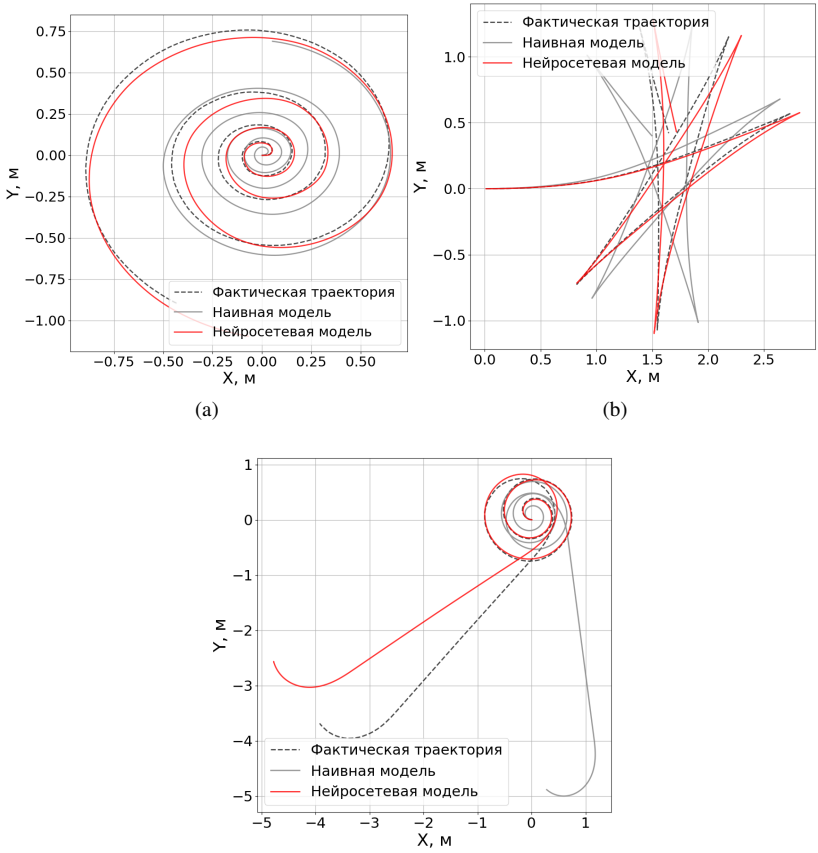
$$\begin{cases}\mathbf{x}_{k=n+1} = F_\phi(\hat{\mathbf{x}}_n, \Delta t)\\ \mathbf{x}_{k>n+1} = F_\phi(\mathbf{x}_{k-1}, \Delta t)\end{cases}. \tag{7}$$

При обучении методом градиентного спуска, ошибка одного предсказанного состояния $\mathbf{x}_k$ влияет на работу нейросети на всех предыдущих шагах, так как градиент, при обратном распространении ошибки, передается дальше через переменную $\mathbf{x}_{k-1}$, которая является выходом нейросети на предыдущем шаге.

При тестах выяснено, что многошаговая функция потерь (при $N > 1$) приводит к модели, которая значительно лучше предсказывает реальную траекторию робота, согласно метрикам, введенным выше, чем при $N = 1$. Это объясняется тем, что наличие шума в данных и конечный размер обучающей выборки приводят к переобучению модели. Многошаговая функция потерь значиетльно снижает дисперсию градиента при оптимизации методом градиентного спуска, и, тем самым, борется с переобучением. Аналогичные результаты с многошаговой функцией потерь описаны в [17].

Параметры модели. Параметры модели, такие как количество слоев, число нейронов в слоях, а также функция активации, подбирались экспериментально путем сравнения метрики ATE на валидационной выборке. Число слоев персептрона выбиралось в диапазоне между 2

и 10. Число нейронов на всех скрытых слоях делалось одинаковым и выбиралось между 16 и 1024. Были протестированы функции активании ReLU, LeakyReLU и ELU.



(a) (b)

(c)

Рис. 5. Примеры предсказанных траекторий

В виду затратности перебора всех возможных комбинаций по времени, была выбрана следующая тактика исследования параметров. Зафиксировав число нейронов на значении 128 для каждой функции активации, были проверены разные количества слоев. Для всех функций активации оказалось, что 4 слоя всегда дают меньшее значение метрики.

Этот параметр был зафикцирован. Далее были проведены эксперименты с разным количеством нейронов и разными функциями активации.

В результате сравнения, наиболее точной показала себя нейросеть с 4 слоями, с 128 нейронами на всех скрытых слоях и функцией активации ELU. Результаты обучения нейросетевой модели представлены на рисунке 5. Среднее значение ATE по всем тестовым траекториям составило 0.66 м. Для простой модели среднее значение ATE – 2.66 м.

3. Решение задачи оптимального управления. Для верификации возможности использования идентифицированной модели для управления роботом решим задачу оптимального управления.

Задана математическая модель колесного робота (4), куда входит функция, реализуемая нейросетью, определяющая значения скоростей робота в зависимости от текущих значений скорости и текущих управляющих воздействий.

Управление имеет ограничения

$$-1 \leqslant u_i \leqslant 1,\ i = 1, 2.$$

Задано начальное и целевое конечное состояние:

$$\mathbf{x}^0 = [0\ 0\ 0\ 0\ 0]^T,$$

$$\mathbf{x}^f = [x^f\ y^f\ \theta^f\ 0\ 0]^T.$$

В рассматриваемой задаче заданы фазовые ограничения в виде статических препятствий круглой формы.

Функционал качества управления включает в себя время достижения целевой точки, минимальное расстояние от траектории робота до целевой точки, длину траектории и штраф за наезд на препятствие:

$$J = a_0 t_f +$$

$$a_1 \sqrt{(x^f - x(t_f))^2 + (y^f - y(t_f))^2 + (\theta^f - \theta(t_f))^2} +$$

$$a_2 \sum_{k=1}^{N-1} \sqrt{(x_k - x_{k-1})^2 + (y_k - y_{k-1})^2} +$$

$$a_3 \sum_{i=1}^{S} \sum_{j=1}^{K} \vartheta(r_i^2 - (\bar{x}_i - x(j\Delta t))^2 - (\bar{y}_i - y(j\Delta t))^2)\Delta t \to \min, \qquad (8)$$

где t_f - ограниченное время завершения процесса управления

$$t_f = \begin{cases} t, \text{ если } t < t^+ \text{ и } \left\|\mathbf{x}(t) - \mathbf{x}^f\right\| \leqslant \varepsilon \\ t^+ \text{ – иначе} \end{cases},$$

ε и t^+ - заданные положительные величины, $N = t_f/dt$ - количество шагов по времени до достижения целевой точки, dt - шаг по времени равный 50 мс, $(\bar{x}_i, \bar{y}_i)$ - координаты центров заданных фазовых ограничений, r_i - радиусы заданных фазовых ограничений, $i = 1, \ldots, S$, S - количество ограничений, $K = \left\lfloor \frac{t^+}{\Delta t} \right\rfloor$, a_0, a_1, a_2, a_3 - весовые коэффициенты, $\vartheta(a)$ - ступенчатая функция Хевисайда.

Задача оптимального управления заключается в нахождении такой последовательности управляющих воздействий u_k^v, u_k^ω, которые минимизируют функцию J. Алгоритм должен выдавать управляющие сигналы u_k^v, u_k^ω для $k = (1..N = t_f/dt)$, где $t_f <= t^+$ - время достижения роботом целевой точки, а $dt = 50$ мс. Для $t^+ = 10$ с $N = 200$, что дает размерность пространства, в котором надо найти минимум функции стоимости J, равное 400. Для численного решения задачи оптимального управления применим прямой общий подход на основе кусочно-полиномиальной аппроксимации функции управления во времени. В связи с большой размерностью пространства поиска в настоящей работе будем искать управление в кусочно-линейном виде, т.е. искать управляющие воздействия с временным шагом равном 1 сек, а в промежутках выдавать управление по линейному закону. Причем, если оптимизационный алгоритм выдает управление, выходящее за рамки допустимых управлений, то линейная функция ограничивается допустимым пределом.

Для поиска минимума функции (8) применяем оптимизационный метод роя частиц PSO [33, 34]. За счет фазовых ограничений функционал (8) имеет невыпуклую форму, а значит градиентные методы могут не найти глобальный минимум. Метод PSO в противоположность градиентным методам не требует дифференцируемости функции, а также способен широко исследовать пространство поиска и находит глобальный минимум с большой вероятностью [35].

3.1. Метод PSO для расчета оптимального управления. Формально, определим функцию стоимости $f : R^n \to R$, глобальный минимум которой следует найти. Функция принимает на вход вектор из n вещественных чисел и возвращает вещественное число. Градиент f неизвестен и не требуется. Задача найти такой вектор $\mathbf{a}$, для которого $f(\mathbf{a}) < f(\mathbf{b})$ для любого $\mathbf{b}$ из пространства поиска, т.е. найти глобальный минимум.

Классический метод PSO итеративно подбирает аргументы многомерной функции стоимости, стремясь найти ее глобальный минимум. Метод использует следующую терминологию. Существует популяция или рой частиц. S – количество частиц в рое. Каждая частица имеет положение $\mathbf{x}_i \in R^n$ и скорость $\mathbf{v}_i \in R^n$. Частицы итеративно перемещаются в пространстве поиска решения в соответствии с уравнением движения. Для каждой частицы хранится ее лучшее положение за все итерации: $\mathbf{p}_i \in R^n$. Также хранится вектор $\mathbf{g} \in R^n$ – лучшее положение среди всех частиц роя за все итерации. На каждой итерации алгоритма скорость частицы меняется так, чтобы ее положение $\mathbf{x}_i$ стремилось одновременно к $\mathbf{p}_i$ и к $\mathbf{g}$. На движение частицы оказывают влияние еще и случайные величины. Критерием останова вычислений может быть либо достижение заданного значения функции стоимости, либо выполнение заданного количества итераций.

Псевдокод PSO описан в алгоритме 1. Значения $\mathbf{b}_{lo}$ и $\mathbf{b}_{up}$ ограничивают пространство поиска. Параметры ω, ϕ_p, ϕ_g и α устанавливаются пользователем и существенно влияют на скорость сходимости алгоритма.

Для расчета оптимального управления с помощью метода PSO, требуется определить функцию стоимости и пространство поиска решения. Пространство поиска решения было задано как пространство последовательностей управляющих сигналов u_k^v, u_k^ω, заданных с шагом по времени в 1 сек. Каждая такая последовательность - это одна частица из роя. В функцию стоимости траектории (8) управление напрямую не входит, поэтому для определенной частицы надо сначала получить траекторию. В коде это делается с помощью функции **propagate control to states**. На вход она принимает последовательность управляющих сигналов u_k^v, u_k^ω, заданных с шагом по времени в 1 сек. Далее шаг уменьшается до 50 мс с помощью линейной интерполяции. После чего для расчета зависимости скорости от времени начальное состояние робота и вся последовательность управления подается на вход модели робота (наивной (2) и нейросетевой (5)). И, наконец, зависимость скорости от времени преобразуется в траекторию в соответствии с (4). Полная блок-схема алгоритма приведена на рисунке 6.

В работе использовались следующие параметры алгоритма PSO: $\omega = 1, \phi_p = 0.95, \phi_g = 0.05, \alpha = 1.0$. Алгоритм стабильно сходится к минимальным значениями функции стоимости за несколько сотен итераций. Результаты расчета в виде зависимости сигналов управления от времени приведены на рисунке 7, а соответствующая траектория движения - на рисунке 8.

Algorithm 1 PSO

Задать лучшее известное положение всех частиц:
$\mathbf{g} \sim U(\mathbf{b}_{lo}, \mathbf{b}_{up})$
for частица i=1, ...,S **do**
  Задать начальное положение частицы:
  $\mathbf{x}_i \sim U(\mathbf{b}_{lo}, \mathbf{b}_{up})$
  Лучшее известное положение этой частицы:
  $\mathbf{p}_i \leftarrow \mathbf{x}_i$
  Уточнить лучшее известное положение среди всех частиц:
  if $f(\mathbf{p}_i) < f(\mathbf{g})$ **then**
    $\mathbf{g} \leftarrow \mathbf{p}_i$
  end if
  Задать скорость частицы:
  $\mathbf{v}_i \leftarrow U(-|\mathbf{b}_{up} - \mathbf{b}_{lo}|, |\mathbf{b}_{up} - \mathbf{b}_{lo}|)$
end for
while критерий останова не достигнут **do**
  for частица i=1, ...,S **do**
    for координата d=1, ...,n **do**
      Сэмплировать случайные числа:
      $r_p, r_g \sim U(0, 1)$
      Обновить скорость частицы:
      $\mathbf{v}_i^d \leftarrow \omega \cdot \mathbf{v}_i^d + \phi_p r_p \cdot (\mathbf{p}_i^d - \mathbf{x}_i^d) + \phi_g r_g \cdot (\mathbf{g}^d - \mathbf{x}_i^d)$
      Где ϕ_p, ϕ_g – постоянные коэффициенты.
    end for
    Обновить положение частицы:
    $\mathbf{x}_i \leftarrow \mathbf{x}_i + \alpha \cdot \mathbf{v}_i$
    Где α – постоянный коэффициент.
    Уточнить лучшее известное положение частицы:
    if $f(\mathbf{x}_i) < f(\mathbf{p}_i)$ **then**
      $\mathbf{p}_i \leftarrow \mathbf{x}_i$
    end if
    Уточнить лучшее известное положение среди всех частиц:
    if $f(\mathbf{p}_i) < f(\mathbf{g})$ **then**
      $\mathbf{g} \leftarrow \mathbf{p}_i$
    end if
  end for
end while

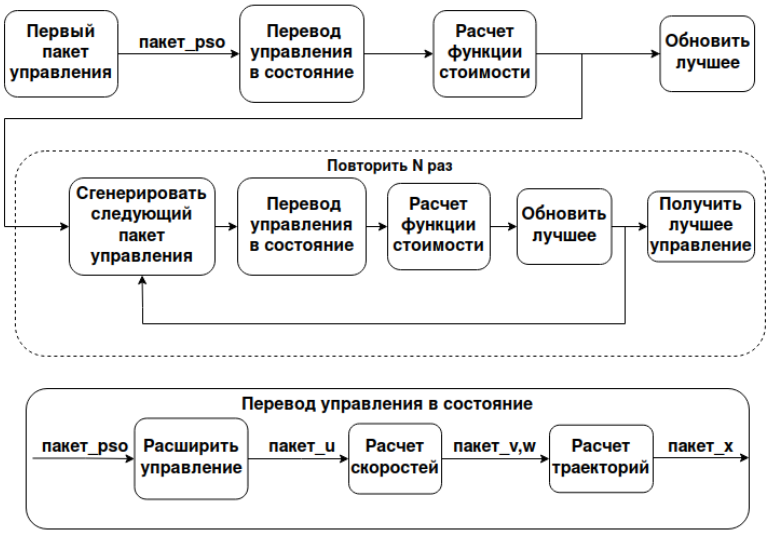


Рис. 6. Блок-схема алгоритма PSO для расчета оптимального управления

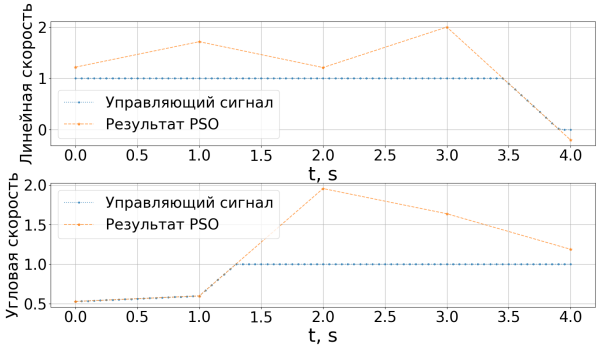


Рис. 7. Результаты расчета оптимального управления

3.2. Верификация полученного управления в симуляторе. Для того, чтобы убедиться в том, что нейросеть может использоваться не только для моделирования, но и для планирования движения реального робота, необходимо проверить, будет ли робот двигаться по той же траектории, что и модель при планировании. В общем случае это может оказаться не так, даже если модель показывает хорошие результаты предсказаний на тестовых проездах. Для этого найденная оптимальная последовательность

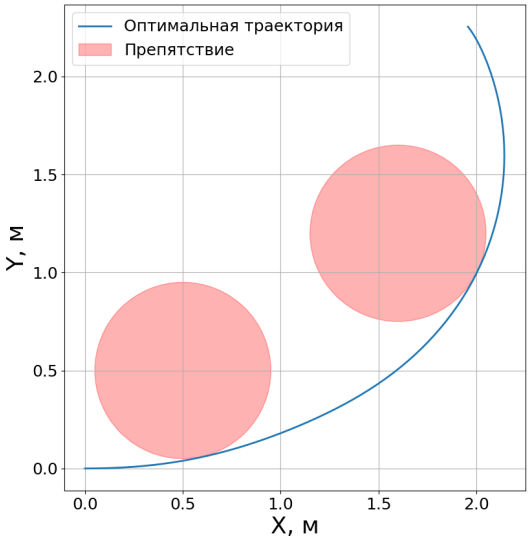


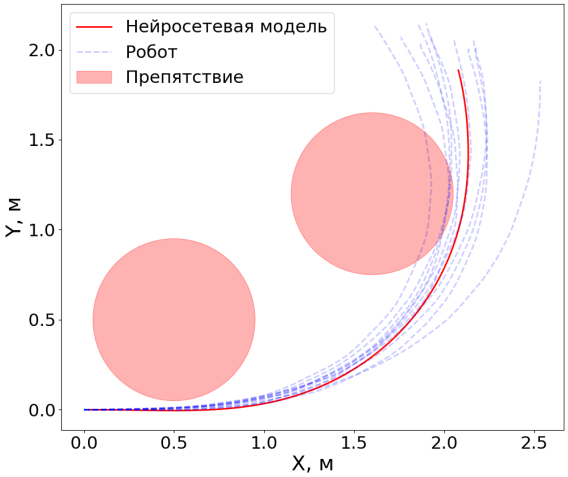
Рис. 8. Результаты расчета оптимальной траектории

управляющих сигналов была подана на робот в симуляторе Gazebo. Было сделано 10 проездов, в каждом из которых на робот в Gazebo подавалась одинаковое управление. Примеры полученных траекторий проездов в симуляторе представлены на рисунке 9-(а). Траектории проездов оказываются разными вследствии того, что Gazebo моделирует реальную систему со своими случайными возмущениями, а в данном случае по сути реализовано разомкнутое управление без обратной связи.

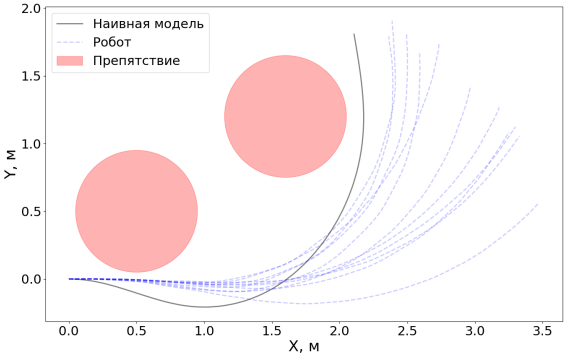
Для сравнения полученной нейросетевой модели управления с наивной моделью (2) задача оптимального управления была также решена и для наивной модели. Полученное управление было подано на робота в Gazebo, и также были собраны траектории по 10 проездам (рисунок 9-(б)).

Отклонение реальной траектории от запланированной было оценено с помощью метрики ATE. ATE для наивной модели составила 0.34 ± 0.16, для нейросетевой модели 0.11 ± 0.04. Это говорит о том, что нейросетевая модель лучше описывает изменение состояния робота.

4. Заключение. В настоящей работе представлен подход на основе нейронной сети для идентификации модели объекта управления. Используется смешанный подход, когда модель объекта частично известна. На основе представленного подхода была создана модель колесного робота Rosbot 2.0 в симуляционной среде Gazebo с помощью искусствен-



(a)



(b)

Рис. 9. Пример проездов робота в симуляторе Gazebo по запланированной траектории. Проезды робота различаются между собой из-за недетерминированного поведения симулятора Gazebo. Отображенные препятствия учитывались при планировании движения

ной нейросети. Эксперименты показали, что нейросеть предсказывает движение колесного робота и может автоматически учитывать такие факторы, как инерция и сила трения. Показано, что с помощью нейросетевой модели можно планировать оптимальную траекторию движения робота. Ошибка нейросетевой модели как при моделировании движения, так и при

планировании траектории оказалась незначительно больше, чем разброс траекторий вследствие недетерминированного поведения самой системы.

Основными сильными сторонами представленного подхода является его практическая реализуемость и универсальность применения. В настоящей работе все данные для обучения нейросети формируются с имитационного симулятора Gazebo, который максимально близко к реальным воспроизводит физику роботов и внешней среды, учитывая возможные помехи и неопределенности. Использование симулятора Gazebo позволило автоматизировать процесс сбора данных для обучения нейронной сети. Кроме того, в настоящей работе мы протестировали полученную нейросетевую модель на примере решения прикладных задач, таких как задача оптимального управления.

Представленную модель можно использовать как для оптимального планирования маршрута в известных условиях, как представлено в статье, так и при разработке других алгоритмов, основанных на предсказании поведения модели. Представленный метод идентификации открыт для использования и редактирования, исходный код находится в репозитории Github [31, 32]. В качестве дальнейшей работы полученную модель планируется использовать для численных расчетов в различных задачах оптимального управления с фазовыми ограничениями, в том числе с динамическими фазовыми ограничениями, когда в задаче имеется группа роботов, а также в задаче навигации для дополнительного уточнения положения робота по модели в условиях автономного движения, а также при реализации алгоритмов планирования пути на основе предсказания поведения модели.

Литература

1. Зенкевич С.Л., Назарова А.В. Система управления мобильного колесного робота // Вестник МГТУ им. Н.Э. Баумана. Сер. “Приборостроение”. 2006. № 3, С.31–51.
2. Gazebo simulation environment tutorial. URL: https://www.gazebosim.org/tutorials?tut=ros_overview (дата обращения: 26.11.2021).
3. Ljung L. System Identification: Theory for the User (second ed.) // Upper Saddle River, New Jersey: Prentice-Hall, 1999.
4. Dastangoo P., Ramirez-Serrano A. Non-linear Parameter Identification for Humanoid Robot Components // The 7th International Conference of Control, Dynamic Systems, and Robotics. 2020.
5. Алексеев А.А. , Кораблев Ю.А., Шестопалов М.Ю. Идентификация и диагностика систем: учеб. для студ. высш. учеб. заведений // М.: Издательский центр «Академия», 2009.
6. Cox P., Toth R. Linear parameter-varying subspace identification: A unified framework // Automatica. 2021. 123. 109296.
7. Sjöberg J., Zhang Q., Ljung L., Benveniste A., Delyon B., Glorennec P., Hjalmarsson H., Juditsky A. Nonlinear black-box modeling in system identification: a unified overview // Automatica. 1995. vol. 31(12). pp. 1691—1724.

8. Nelles O. Classical Polynomial Approaches // Nonlinear System Identification. Springer, Berlin, Heidelberg, 2001.
9. Fakhrizadeh Esfahani A., Dreesen P., Tiels K., Noël J.-P., Schoukens J. Parameter reduction in nonlinear state-space identification of hysteresis // Mechanical Systems and Signal Processing. 2017. 104.
10. Liu G. P. Nonlinear identification and control: a neural network approach // Springer Science & Business Media. 2012.
11. Werbos P. J. (n.d.). Neural networks for control and system identification // Proceedings of the 28th IEEE Conference on Decision and Control. 1989.
12. Fu Z. J. et al. Nonlinear systems identification and control via dynamic multitime scales neural networks //IEEE transactions on neural networks and learning systems. 2013. vol. 24. no. 11. pp. 1814–1823.
13. Dang,T.P., Diveev, A.I., Kazaryan, D.E., Sofronova, E.A. Identification Control Synthesis By The Network Operator Method // Proceedings 2015 IEEE 10th Conference on Industrial Electronics and Applications (ICIEA). 2015. pp. 1559–1564.
14. Дивеев А.И., Софронова Е.А., Шмалько Е.Ю. Метод идентификационного синтеза управления и его применение к мобильному роботу // Информационные и математические технологии в науке и управлении. 2016. № 2. С. 53-61.
15. Gautam P. System identification of nonlinear Inverted Pendulum using artificial neural network // 2016 International Conference on Recent Advances and Innovations in Engineering (ICRAIE). 2016. pp. 1–5.
16. Zheng D. D., Xie W. F., Luo C. Robust identification for singularly perturbed nonlinear systems using multi-time-scale dynamic neural network //2017 IEEE 56th Annual Conference on Decision and Control (CDC). 2017. pp. 6487–6492.
17. Mohajerin N., Waslander S. L. Multistep prediction of dynamic systems with recurrent neural networks //IEEE transactions on neural networks and learning systems. 2019. vol. 30. no. 11. pp. 3370–3383.
18. Khodabandehlou H., Fadali M. S. Nonlinear System Identification using Neural Networks and Trajectory-Based Optimization //arXiv preprint arXiv:1804.10346. 2018.
19. Williams G. et al. Aggressive driving with model predictive path integral control //2016 IEEE International Conference on Robotics and Automation (ICRA). 2016. pp. 1433–1440.
20. Williams G. et al. Autonomous racing with autorally vehicles and differential games //arXiv preprint arXiv:1707.04540. 2017.
21. Samal, M. K., Anavatti, S., Garratt, M. Neural Network Based System Identification for Autonomous Flight of an Eagle Helicopter // IFAC Proceedings. 2008. vol. 41(2). pp. 7421—7426. 2008.
22. Yu Wang. A new concept using LSTM Neural Networks for dynamic system identification // American Control Conference (ACC). 2017.
23. Brunton S.L., Proctor J.L., Kutz J.N. Discovering governing equations from data: Sparse identification of nonlinear dynamical systems. 2015.
24. Kaiser E., Kutz J. N., Brunton S. L. Sparse identification of nonlinear dynamics for model predictive control in the low-data limit //Proceedings of the Royal Society A. 2018. vol. 474. no. 2219. P. 20180335.
25. Jian'an X., Mingjun Z., Jian Z. Kinematic model identification of autonomous mobile robot using dynamical recurrent neural networks //IEEE International Conference Mechatronics and Automation. 2005. vol. 3. pp. 1447–1450.
26. Roy T., Barai R.K. and Dey R. Identification of Differentially Driven Wheeled Mobile Robot using Neural Networks // International Journal of Electrical, Electronics and Computer Engineering. 2013. vol. 2(2). pp.38–45.

27. Lavrenov R., Magid E., Matsuno F., Svinin M., Suthakorn J. Development and Implementation of Spline-based Path Planning Algorithm in ROS/Gazebo Environment // SPIIRAS Proceedings. 2019. Vol. 18. pp. 57-84.
28. Zhang B., Liu P. Control and benchmarking of a 7-DOF robotic arm using Gazebo and ROS // PeerJ Computer Science. 2021. Vol. 7.
29. URL: https://github.com/husarion/rosbot_description (дата обращения: 26.11.2021).
30. Šuster P., Jadlovská A. Tracking Trajectory of the Mobile Robot Khepera II Using Approaches of Artificial Intelligence // Acta Electrotechnica et Informatica. vol. 11.
31. Программный пакет для сбора данных для обучения сети в симуляторе Gazebo URL: https://github.com/urock/rosbot (дата обращения: 26.11.2021).
32. Программный пакет для обучения нейросетевой модели URL: https://github.com/FastSense/robot_model_training (дата обращения: 26.11.2021).
33. Kennedy J., Eberhart R. Particle swarm optimization //Proceedings of ICNN'95-international conference on neural networks. IEEE, 1995. vol. 4. pp. 1942–1948.
34. Diveev A., Shmalko E. Evolutionary Computation for Synthesis of Control System for Group of Robots and Optimum Choice of Trajectories for their Movement // Proceedings of the OPTIMA-2017 Conference, Petrovac, Montenegro. 2017. pp. 158–165.
35. Дивеев А.И., Константинов С.В. Исследование практической сходимости эволюционных алгоритмов оптимального программного управления колесным роботом // Известия РАН Теория и системы управления. 2018. № 4. том 57. С. 80–106.

Шмалько Елизавета Юрьевна — канд. техн. наук, старший научный сотрудник, отдел управления робототехническими устройствами, ФИЦ ИУ РАН. Область научных интересов: современные вычислительные методы в управлении техническими устройствами, методы символьной регрессии и эволюционные вычисления с приложениями для идентификации, оптимизации и синтеза систем управления. Число научных публикаций — 100. e.shmalko@gmail.com; Вавилова, 44/2, 119333, Москва, Россия; р.т.: +7(964)636-6669.

Румянцев Юрий Андреевич — аспирант, ФИЦ ИУ РАН. Область научных интересов: методы машинного обучения с приложениями для идентификации, оптимизации и синтеза систем управления, энергоэффективные бортовые вычислительные системы. Число научных публикаций — 0. urock@fastsense.tech; Вавилова, 14/2, 125124, Москва, Россия; р.т.: +7(926)590-9868.

Байназаров Руслан Рысбекович — инженер машинного обучения, ООО Фаст Сенс Студия. Область научных интересов: машинное обучение и наука о данных. Число научных публикаций — 0. ruslan@dtlabs.tech; Ямского поля 3-я, 2/8, 125124, Москва, Россия; р.т.: +7(926)590-9868.

Ямшанов Константин Леонидович — студент магистратуры, Новосибирский государственный технический университет (НГТУ). Область научных интересов: алгоритмы оптимизации, синтез систем управления. Число научных публикаций — 0. kostya.yam@dtlabs.tech; пр-т К. Маркса, 20, 630073, Новосибирск, Россия; р.т.: +7(926)590-9868.

DOI 10.15622/ia.20.6.3

E. SHMALKO, Yu. RUMYANTSEV, R. BAINAZAROV, K. YAMSHANOV

IDENTIFICATION OF NEURAL NETWORK MODEL OF ROBOT TO SOLVE THE OPTIMAL CONTROL PROBLEM

Shmalko E., Rumyantsev Yu., Bainazarov R., Yamshanov K. **Identification of neural network model of robot to solve the optimal control problem.**

Abstract. To calculate the optimal control, a satisfactory mathematical model of the control object is required. Further, when implementing the calculated controls on a real object, the same model can be used in robot navigation to predict its position and correct sensor data, therefore, it is important that the model adequately reflects the dynamics of the object. Model derivation is often time-consuming and sometimes even impossible using traditional methods. In view of the increasing diversity and extremely complex nature of control objects, including the variety of modern robotic systems, the identification problem is becoming increasingly important, which allows you to build a mathematical model of the control object, having input and output data about the system. The identification of a nonlinear system is of particular interest, since most real systems have nonlinear dynamics. And if earlier the identification of the system model consisted in the selection of the optimal parameters for the selected structure, then the emergence of modern machine learning methods opens up broader prospects and allows you to automate the identification process itself. In this paper, a wheeled robot with a differential drive in the Gazebo simulation environment, which is currently the most popular software package for the development and simulation of robotic systems, is considered as a control object. The mathematical model of the robot is unknown in advance. The main problem is that the existing mathematical models do not correspond to the real dynamics of the robot in the simulator. The paper considers the solution to the problem of identifying a mathematical model of a control object using machine learning technique of the neural networks. A new mixed approach is proposed. It is based on the use of well-known simple models of the object and identification of unaccounted dynamic properties of the object using a neural network based on a training sample. To generate training data, a software package was written that automates the collection process using two ROS nodes. To train the neural network, the PyTorch framework was used and an open source software package was created. Further, the identified object model is used to calculate the optimal control. The results of the computational experiment demonstrate the adequacy and performance of the resulting model. The presented approach based on a combination of a well-known mathematical model and an additional identified neural network model allows using the advantages of the accumulated physical apparatus and increasing its efficiency and accuracy through the use of modern machine learning tools.

Keywords: optimal control, identification, neural network, Gazebo, differential robot.

Shmalko Elizaveta — Ph.D., Senoir researcher, Department of robotic control, FRC CSC RAS. Research interests: modern computational methods in control of technical systems, symbolic regression methods and evolutionary computation with applications for identification, optimization and synthesis of control systems. The number of publications — 100. e.shmalko@gmail.com; 44/2, Vavilova, 119333, Moscow, Russia; office phone: +7(964)636-6669.

Rumyantsev Yuri — Ph.D., Senoir researcher, Department of robotic control, FRC CSC RAS. Research interests: modern computational methods in control of technical systems, symbolic regression methods and evolutionary computation with applications for identification, optimization

and synthesis of control systems. The number of publications — 100. e.shmalko@gmail.com; 44/2, Vavilova, 119333, Moscow, Russia; office phone: +7(964)636-6669.

Baynazarov Ruslan — Engineer of machine learning, Fast Sense Studio. Research interests: data science and machine learning. The number of publications — 0. ruslan@dtlabs.tech; 2/8, Yamskogo field 3-rd Str., 125124, Moscow, Russia; office phone: +7(926)590-9868.

Yamshanov Konstantin — Graduate student, Novosibirsk State Technical University (NSTU). Research interests: optimization algorithms, synthesis of control systems. The number of publications — 0. kostya.yam@dtlabs.tech; 20, K.Marx Str., 630073, Novosibirsk, Russia; office phone: +7(926)590-9868.

References

1. Zenkevich S.L., Nazarova A.V. Sistema upravleniya mobil'nogo kolesnogo robota [Control system of a mobile wheeled robot] // Vestnik MGTU im. N.E. Baumana. Ser. "Priborostroyeniye". 2006. № 3, pp.31-51. (In Russ.)
2. Gazebo simulation environment tutorial. URL: https://www.gazebosim.org/tutorials?tut=ros_overview (accessed: 26.11.2021).
3. Ljung L. System Identification: Theory for the User (second ed.) // Upper Saddle River, New Jersey: Prentice-Hall, 1999.
4. Dastangoo P., Ramirez-Serrano A. Non-linear Parameter Identification for Humanoid Robot Components // The 7th International Conference of Control, Dynamic Systems, and Robotics. 2020.
5. Alekseev A.A., Korablev Yu.A., Shestopalov M.Yu. Identifikatsiya i diagnostika sistem: ucheb. dlya stud. vyssh. ucheb. zavedeniy [Identification and diagnostics of systems: textbook for students]. - Publishing Center "Academy", Moscow, 2009. (In Russ.)
6. Cox P., Toth R. Linear parameter-varying subspace identification: A unified framework // Automatica. 2021. 123. 109296.
7. Sjöberg J., Zhang Q., Ljung L., Benveniste A., Delyon B., Glorennec P., Hjalmarsson H., Judit-sky A. Nonlinear black-box modeling in system identification: a unified overview // Automati-ca. 1995. vol. 31(12). pp. 1691—1724.
8. Nelles O. Classical Polynomial Approaches // Nonlinear System Identification. Springer, Berlin, Heidelberg, 2001.
9. Fakhrizadeh Esfahani A., Dreesen P., Tiels K., Noël J.-P., Schoukens J. Parameter reduction in nonlinear state-space identification of hysteresis // Mechanical Systems and Signal Processing. 2017. 104.
10. Liu G. P. Nonlinear identification and control: a neural network approach // Springer Science & Business Media. 2012.
11. Werbos P. J. (n.d.). Neural networks for control and system identification // Proceedings of the 28th IEEE Conference on Decision and Control. 1989.
12. Fu Z. J. et al. Nonlinear systems identification and control via dynamic multitime scales neural networks //IEEE transactions on neural networks and learning systems. 2013. vol. 24. no. 11. pp. 1814–1823.
13. Dang,T.P., Diveev, A.I., Kazaryan, D.E., Sofronova, E.A. Identification Control Synthesis By The Network Operator Method // Proceedings 2015 IEEE 10th Conference on Industrial Elec-tronics and Applications (ICIEA). 2015. pp. 1559–1564.
14. Diveev A.I., Sofronova E.A., Shmalko E.Yu. Metod identifikacionnogo sinteza upravleniya i ego primenenie k mobil'nomu robotu [Method of identification synthesis of control and its application to a mobile robot] // Information and Mathematical Technologies in

Science and Management [Informatsionnyye i matematicheskiye tekhnologii v nauke i upravlenii]. 2016. vol. 2. pp. 53–61. (In Russ.)
15. Gautam P. System identification of nonlinear Inverted Pendulum using artificial neural network // 2016 International Conference on Recent Advances and Innovations in Engineering (ICRAIE). 2016. pp. 1–5.
16. Zheng D. D., Xie W. F., Luo C. Robust identification for singularly perturbed nonlinear systems using multi-time-scale dynamic neural network // IEEE 56th Annual Conference on Decision and Control (CDC). 2017. pp. 6487–6492.
17. Mohajerin N., Waslander S. L. Multistep prediction of dynamic systems with recurrent neural networks //IEEE transactions on neural networks and learning systems. 2019. vol. 30. no. 11. pp. 3370–3383.
18. Khodabandehlou H., Fadali M. S. Nonlinear System Identification using Neural Networks and Trajectory-Based Optimization //arXiv preprint arXiv:1804.10346. 2018.
19. Williams G. et al. Aggressive driving with model predictive path integral control // IEEE International Conference on Robotics and Automation (ICRA). 2016. pp. 1433–1440.
20. Williams G. et al. Autonomous racing with autorally vehicles and differential games //arXiv preprint arXiv:1707.04540. 2017.
21. Samal, M. K., Anavatti, S., Garratt, M. Neural Network Based System Identification for Au-tonomous Flight of an Eagle Helicopter // IFAC Proceedings. 2008. vol. 41(2). pp. 7421—7426. 2008.
22. Yu Wang. A new concept using LSTM Neural Networks for dynamic system identification // American Control Conference (ACC). 2017.
23. Brunton S.L., Proctor J.L., Kutz J.N. Discovering governing equations from data: Sparse identi-fication of nonlinear dynamical systems. 2015.
24. Kaiser E., Kutz J. N., Brunton S. L. Sparse identification of nonlinear dynamics for model predictive control in the low-data limit //Proceedings of the Royal Society A. 2018. vol. 474. no. 2219. P. 20180335.
25. Jian'an X., Mingjun Z., Jian Z. Kinematic model identification of autonomous mobile robot using dynamical recurrent neural networks // IEEE International Conference Mechatronics and Automation. 2005. vol. 3. pp. 1447–1450.
26. Roy T., Barai R.K. and Dey R. Identification of Differentially Driven Wheeled Mobile Robot using Neural Networks // International Journal of Electrical, Electronics and Computer Engineering. 2013. vol. 2(2). pp.38–45.
27. Lavrenov R., Magid E., Matsuno F., Svinin M., Suthakorn J. Development and Implementation of Spline-based Path Planning Algorithm in ROS/Gazebo Environment // SPIIRAS Pro-ceedings. 2019. Vol. 18. pp. 57-84.
28. Zhang B., Liu P. Control and benchmarking of a 7-DOF robotic arm using Gazebo and ROS // PeerJ Computer Science. 2021. vol. 7.
29. URL: https://github.com/husarion/rosbot_description (accessed: 26.11.2021).
30. Šuster P., Jadlovská A. Tracking Trajectory of the Mobile Robot Khepera II Using Approaches of Artificial Intelligence // Acta Electrotechnica et Informatica. vol. 11.
31. Data acquisition software package for network training in Gazebo simulator URL: https://github.com/urock/rosbot (accessed: 26.11.2021).
32. Software package for training a neural network model URL: https://github.com/FastSense/robot_model_training (accessed: 26.11.2021).
33. Kennedy J., Eberhart R. Particle swarm optimization //Proceedings of ICNN'95-international conference on neural networks. IEEE, 1995. – vol. 4. pp. 1942–1948.
34. Diveev A., Shmalko E. Evolutionary Computation for Synthesis of Control System for Group of Robots and Optimum Choice of Trajectories for their Movement // Proceedings of the OPTIMA-2017 Conference, Petrovac, Montenegro. 2017. pp. 158–165.

35. Diveev, A.I., Konstantinov S.V. Issledovanie prakticheskoj skhodimosti evolyucionnyh algoritmov optimal'nogo programmnogo upravleniya kolesnym robotom [Study of the Practical Convergence of Evolutionary Algorithms for the Optimal Program Control of a Wheeled Robot]. Journal of Computer and Systems Sciences International. 2018. vol. 57. no. 4. pp. 80–106. (In Russ.)