УДК 004.75
DOI: 10.15587/1729-4061.2020.213989

Побудова моделі мережевої взаємодії складових мультиагентної системи мобільних роботів

В. А. Дідук, В. Г. Гриценко, А. Д. Єрьоменко

Представлені результати роботи є першим етапом розробки повнофункціональної лабораторної системи дослідження алгоритмів машинного навчання. Актуальність роботи зумовлена відсутністю мережевих малогабаритних мобільних роботів та відповідного керуючого програмного забезпечення, що дозволило б проводити натурні експерименти в реальному часі. В роботі здійснено підбір мережевої технології передачі даних для керування мобільними роботами в реальному часі. На основі обраного протоколу передачі даних запропоновано повний стек технологій мережевої моделі мультиагентної системи мобільних роботів. Це дозволило побудувати мережеву модель системи візуалізації та дослідження алгоритмів машинного навчання. Відповідно до вимог мережевої моделі OSI щодо побудови подібних систем, модель включає в себе наступні рівні:

1) нижній рівень збору даних та виконавчих механізмів – мобільні роботи;

2) верхній рівень моделі – складається з серверу користувацького інтерфейсу та серверу підтримки бізнес-логіки.

Базуючись на побудованих діаграмі стеку протоколів та мережевій моделі здійснена програмно-апаратна реалізація отриманих результатів. У роботі використано JavaScript бібліотека React з технологією SPA (Single Page Application), технологію Virtual DOM (Document Object Model), що зберігається в оперативній пам'яті пристрою і синхронізується з реальним DOM. Це дозволило спростити процес керування клієнтами та зменшити мережевий трафік.

Модель надає можливість:

1) керувати прототипами роботів-клієнтів в реальному часі;

2) зменшити використання мережевого трафіку, в порівнянні з іншими технологіями передачі даних;

3) зменшити навантаження на центральні процесори роботів та серверів;

4) виконувати віртуальну симуляцію експерименту;

5) досліджувати виконання алгоритмів машинного навчання.

Ключові слова: мультиагентні системи, мобільні роботи, машинне навчання, мережева модель, WEB-інтерфейс, WebSocket.

1. Вступ

Пошук оптимальних шляхів вирішення завдань аналізу великих даних ставить високі вимоги до попередньої підготовки програміста, математика та інших фахівців суміжних професій. За відсутності у цих фахівців необхідних знань та досвіду, їм доводиться витрачати зайвий час на навчання, або взагалі відмовлятися від подальшої участі в дослідних чи конструкторських розробках.

Останнім часом досить широкого поширення набув напрямок машинного навчання, що уможливлює нівелювання вищеописаної проблематики. Користуючись теоретичними положеннями машинного навчання, програмісти мають змогу не писати громіздкі програмні продукти, що передбачатимуть всі можливі варіанти рішення. Натомість, можна використати один із загальноприйнятих алгоритмів самостійного знаходження оптимального розв'язку шляхом комплексного аналізу наявних даних. Отриманий результат аналізу дає можливість здійснювати прогнозування, робити висновки, приймати рішення тощо.

Розвиток мікропроцесорної техніки та сучасні досягнення робототехніки відкрили нову сферу використання алгоритмів машинного навчання. При сучасних темпах збільшення кількості роботів, актуальним є розробка алгоритмів аналізу даних та автономного прийняття рішення поведінки робота. Прикладами таких роботизованих систем є автомобілі, безпілотні дрони, роботи-кур'єри тощо. Реалізація замкненої системи в межах однієї технічної одиниці надає можливість мінімізувати кількість збурень і реалізація системи автономного керування роботом є не складною. При використанні значної кількості автономних мобільних роботів процес керування кожним окремо стає ресурсоємким і складно реалізовуваним [1]. В таких системах також є оптимальним використання алгоритмів машинного навчання.

Апробування алгоритмів машинного навчання зазвичай здійснюється на комп'ютері при аналізі значної кількості даних і результатом аналізу є інформація, представлена в текстовому вигляді. Нині, для використання на реальних роботах, особливо в мультиагентних системах, такий підхід не є оптимальним, оскільки не дає можливості візуалізувати результати рішення та провести натурний експеримент в динаміці. Існуючі моделі лабораторних роботів не мають можливості змінювати програму в процесі експерименту, або мають надмірну габаритність та високу вартість. Все це унеможливлює їх об'єднання в мультиагенті мережі для дослідження алгоритмів машинного навчання.

Таким чином, розробка систем дослідження мультиагентного управління колективними мобільними роботами, що можуть автономно функціонувати та приймати рішення є актуальною.

2. Аналіз літературних даних та постановка проблеми

Частка досліджень мультиагентних робототехнічних систем з кожним роком збільшується, про що свідчить значна кількість публікацій, зокрема в роботах [2–13]. Найменш дослідженими є можливості одночасного керування всіма складниками такої системи, чи їх автономна взаємодія. До таких завдань можна віднести керування груповою поведінкою: контроль за траєкторією слідування, контроль за зіткненнями роботів, тактичні маневри, розподіл завдань моніторингу відкритих чи закритих територій, розподіл завдань обробки даних та інше. На даний момент використання машинного навчання в мультиагентних системах мобільних роботів досліджено в роботах [2, 5, 6, 8–10, 12, 13] та інших. Так, в роботах [2, 4, 10] проведено лише аналіз:

1. Можливостей масштабування мультагентних систем мобільних роботів;
2. Граничної кількості агентів та розміру робочого поля;

3. Основних напрямків досліджень, які необхідні для розробки подібних систем без конкретних рішень щодо їх побудови.

В [3] увага приділяється розробці синхронних операторських систем управління одним роботом та синхронних мульти-операторських багатороботних систем. Недоліком роботи є відсутність можливості системи працювати без участі оператора, відсутність можливості збору даних роботи роботів та апробації алгоритмів машинного навчання і керування. В [5] приведено результати розробки системи двох роботів та апробації роботи алгоритму Dyna. Недоліком системи є відсутність зворотного зв'язку роботів з центральним комп'ютером, рух лише паралельно осям прямокутної системи координат. В роботах [6, 11] наведено результати апробації алгоритмів канонічного варіативного аналізу Ла-Римора та GraWoLF відповідно. В [8, 9, 13] надано результати самонавчання системи з обмеженої кількості роботів. Спільним недоліком робіт є: необхідність програмування кожного робота окремо і відсутність можливості збирати інформацію на сервер. Всі судження щодо результативності роботи алгоритму та системи в цілому здійснюються дослідниками опосередковано за результатами спостережень. Робота [7] носить переважно оглядовий характер, в якій аналізується гранична кількість агентів системи в межах одного робочого поля. В [12] описано значну кількість алгоритмів машинного навчання та опису моделей побудови фізичних систем мобільних роботів. Загальною рисою всіх існуючих робіт є використання закритих систем, що налаштовані на виконання попередньо запрограмованого одного алгоритму роботи агента (робота). При зміні завдань дослідження розробники змушені програмувати кожен робот окремо. Відповідно, затрачений час на зміну програми зростає пропорційно до кількості задіяних агентів в системі. Про ефективність алгоритму дослідники роблять висновки переважно за результатами візуального спостереження. До недоліків таких систем можна віднести:

1. Складність зміни програми діяльності кожного робота.

2. Відсутність можливості накопичувати дані на сервері.

3. Відсутність інтерфейсу взаємодії користувача, який не є фахівцем з програмування контролерів, з агентами системи.

Проведений аналіз дає підстави припустити, що сьогодні є досить потужні математичні моделі побудови мультиагентних систем та взаємодії її складових. Проте не вистачає ефективного інструментального апарату дослідження їх роботи. Очевидно, оптимальною є розробка WEB-орієнтованої системи керування, в якій весь алгоритм обробки даних буде здійснюватися на сервері і керування кожним з агентів буде централізовано. Такий підхід надасть можливість:

– використовувати будь якого робота системи для відпрацювання довільних алгоритмів без зміни його програмного забезпечення;

– накопичувати зібрані дані на сервері для подальшого їх аналізу чи застосування як вихідні дані для наступних експериментів;

– збільшити кількість потенційних дослідників за рахунок розробки спрощеного інтерфейсу та дистанційного доступу до системи.

3. Мета та задачі дослідження

Метою роботи є побудова моделі мережевої взаємодії мультиагентної системи мобільних роботів, що можуть в реальному часі використовуватись дослідниками будь якого рівня підготовки для реалізації алгоритмів машинного навчання.

Для досягнення мети були поставлені такі завдання:

– обрати мережевий протокол передачі даних для керування віддаленими мікропроцесорними засобами у реальному часі;

– здійснити підбір стеку технологій та визначити роль і взаємозв'язок складових моделі мультиагентної системи мобільних роботів;

– апробувати запропоновані у дослідженні теоретичні положення, використовуючи сучасні засоби розробки програмного та апаратного забезпечення.

4. Вибір мережевого протоколу керування віддаленими мікропроцесорними засобами в реальному часі

Сучасний зв'язок в мережі Інтернет і пристроїв Internet of Things (IoT) в частинному випадку базується на протоколах HTTP (Hypertext Transfer Protocol), Websocket та MQTT. Якщо не потрібна реакція в реальному часі, зазвичай використовують HTTP, проте протокол не можна використати для організації клієнт-серверного з'єднання, необхідного для завдань дослідження.

Один із шляхів вирішення проблеми – застосування принципу Polling, відправки AJAX (Asynchronous JavaScript and XML) запитів через невеликі проміжки часу [14]. Як тільки сервер отримує запит, одразу надсилається відповідь, з'єднання розривається і сервер готовий обробити наступний запит (рис. 1).

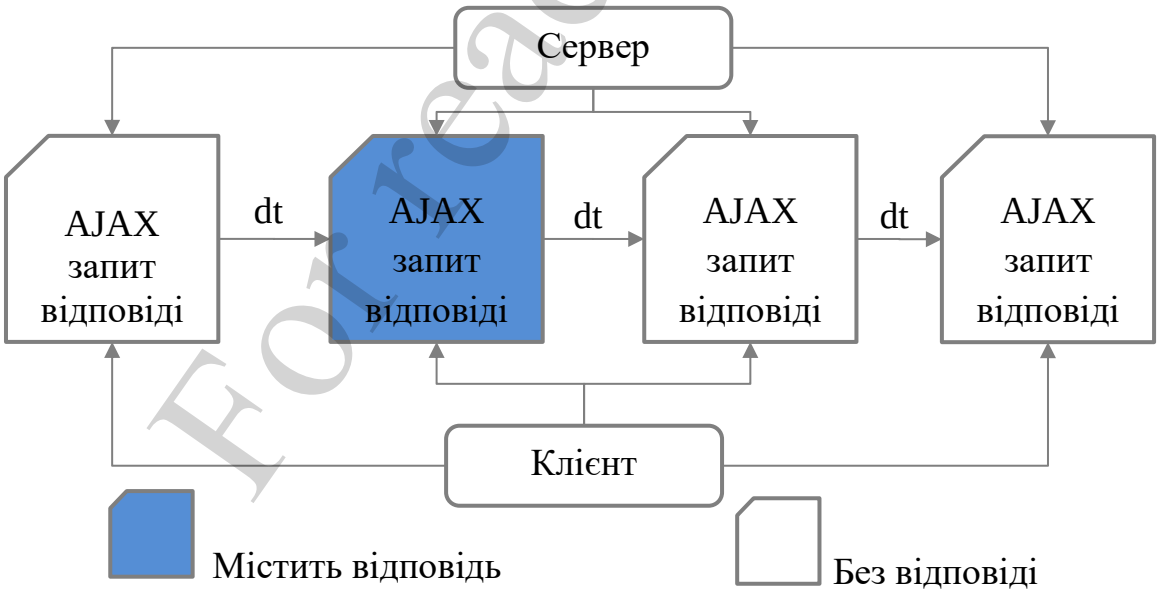

Рис. 1. Схема передачі даних методом Polling

Недоліками такого методу є перенасиченість даних (оскільки Header надсилається в усіх запитах, трафік високий) та детермінованість запитів.

Проблему детермінованості запитів частково вирішує Long Polling – метод при якому клієнт надсилає HTTP запит, проте сервер може затримати відповідь та надіслати клієнту у потрібний момент часу. Якщо відповідь не надіслана за певний проміжок часу, клієнт розриває з'єднання з сервером та створює новий запит (рис. 2).

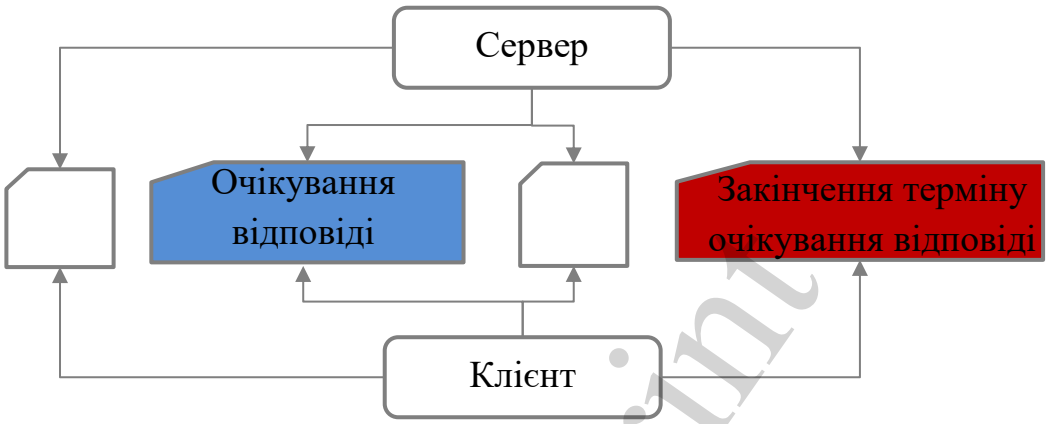

Рис. 2. Схема передачі даних методом Long Polling

При зменшенні дискретності запитів зростає мережевий трафік, що не припустимо для систем керування в реальному часі.

Модифікацією методу Long Polling є Streaming, при якому сервер не постійно підтримує нерозривно з'єднання і надсилає користувачеві інформацію невеликими порціями – пакетами (рис. 3).

Перевагою даного методу є відсутність затримок, пов'язаних з постійним створенням запитів. Header запиту відправляється лише при з'єднанні з сервером, що зменшує трафік передачі даних.

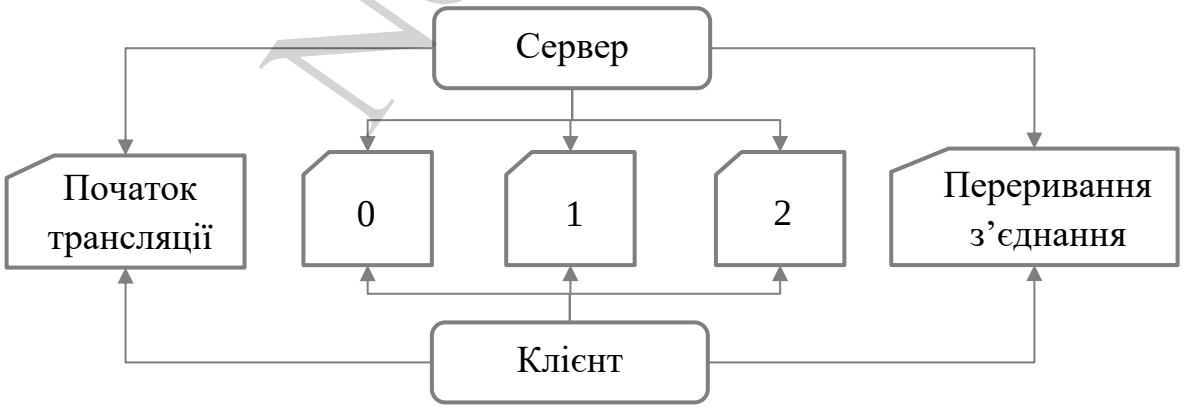

Рис. 3. Модель передачі даних методом Streaming

Недоліком методу є односторонній зв'язок, що не є придатним для моделювання алгоритмів машинного навчання.

Протокол передачі даних WebSocket поєднує переваги всіх вище названих методів, реалізовуючи асинхронну, клієнт-серверну “request-response” взаємодію. Сервер та клієнт за даного підходу є рівноправними учасниками обміну даних. Вся взаємодія між сервером та клієнтом відбувається за допомогою WebSocket API [14].

На рис. 4 наведено процес встановлення з’єднання, прийому, відправки повідомлень та розриву з’єднання між клієнтом та сервером.

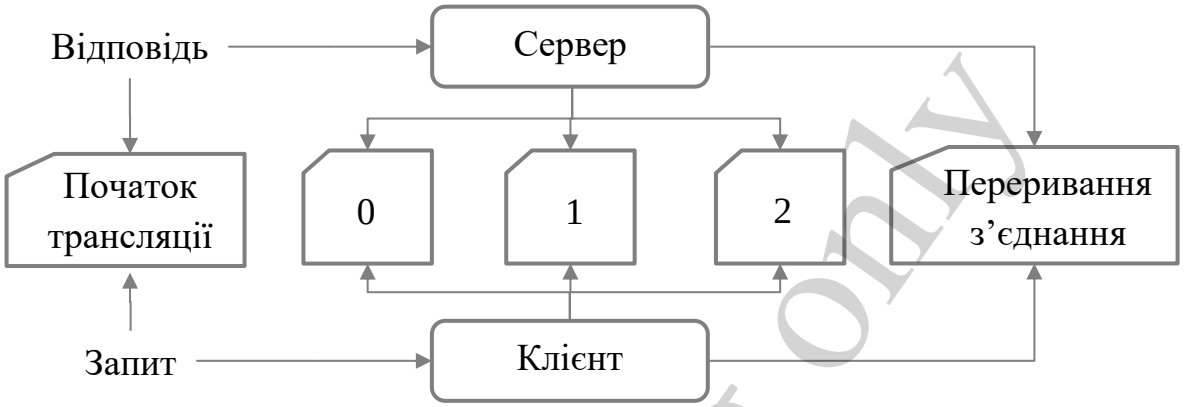

Рис. 4. Схема передачі даних протоколом WebSocket

Таким чином, найбільш доцільною є розробка моделі мультиагентної системи мобільних роботів з керуванням в реальному режимі часу на основі протоколу WebSocket. Також використання даного протоколу сприяє зменшенню використання Інтернет трафіку, та зменшенню завантаження центральних процесорів клієнта і сервера.

5. Підбір стеку технологій та побудова моделі мультиагентної системи мобільних роботів

Оскільки визначено, що для зручності та простоти доступу до підсистеми керування, вся система є WEB-орієнтованою, користувач взаємодіє з системою за допомогою WEB-додатка. В проекції на обрані технології, мережеву модель мультиагентної системи мобільних роботів для дослідження роботи алгоритмів штучного інтелекту представлено на рис. 5. Основні переваги вибору такого підходу – кросплатформність, віддалений доступ для всіх учасників процесу та інше.

Сучасним паттерном у індустрії WEB-розробки є розмежування back-end і front-end серверів. Такий підхід надає можливість зменшити навантаження з BLL (Business Logic Layer) – сервера, що обробляє запити користувачів, і відсилає команди роботам клієнтам.

UIL (User Interface Layer) – це усі ресурси, (медіа файли, зображення, JSX шаблони, CSS та JavaScript файли) які потрібні для побудови сторінки додатка, яку бачить користувач. Вони зберігаються на окремому front-end сервері, який надсилає їх клієнту при завантаженні додатка.

Для побудови UIL – WEB-додатка користувача, в роботі використана JavaScript бібліотека React (рис.6). Вона надала можливість використати SPA (Single Page Application) технологію що реалізує Client Side Render.

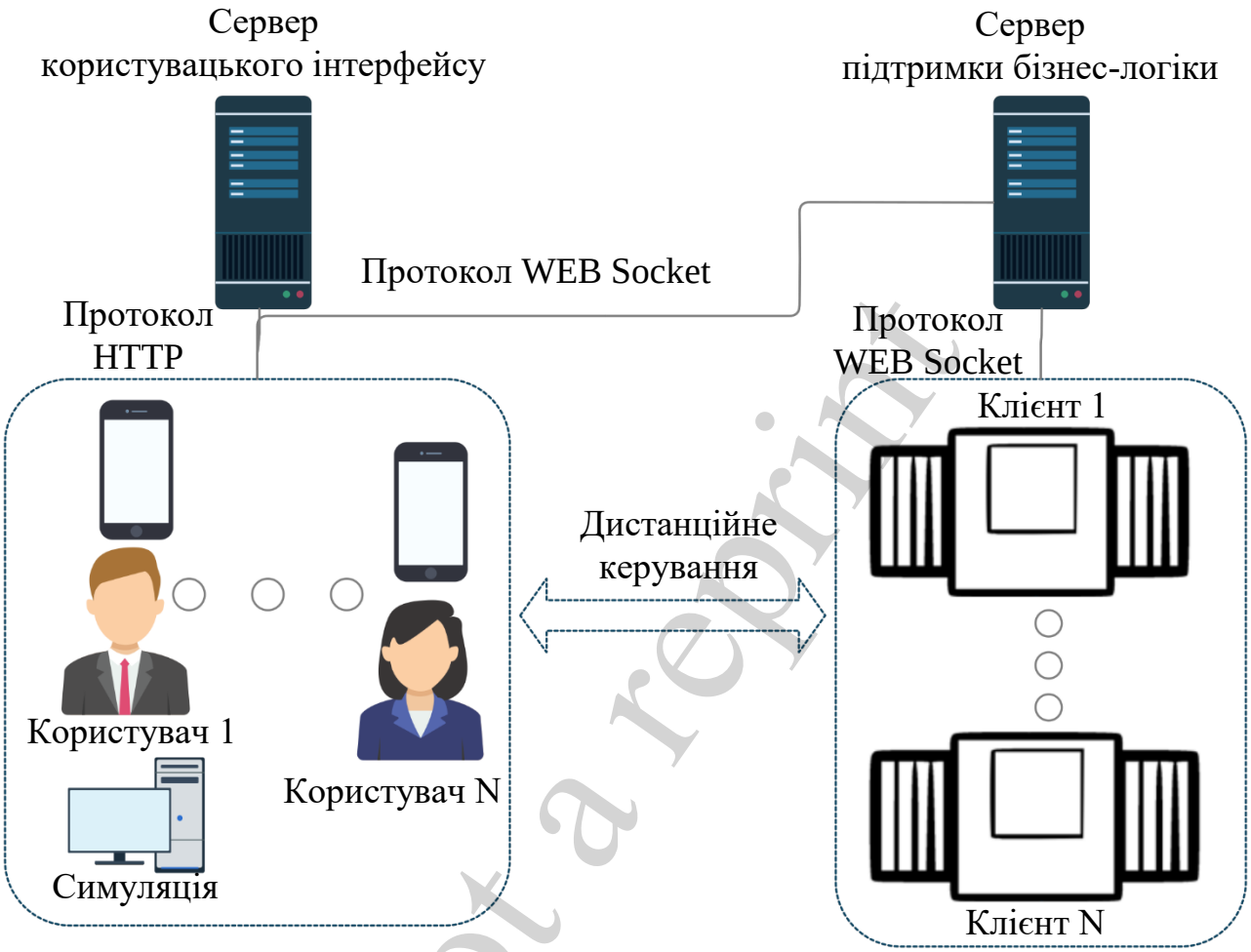

Рис. 5. Модель мережевої взаємодії складових мультиагентної системи мобільних роботів

BLL побудований на основі Node JS та фреймворка Express. При розробці програмної частини моделі, з React використано додатковий плагін, який надає можливість використовувати TypeScript. Недоліком JavaScript є відсутність строгої типізації, що погіршує як стабільність роботи, так і безпеку додатків. TypeScript додає можливість строгої типізації змінних.

Для зручності зберігання даних та побудови React компонентів додатку користувача використана бібліотека Redux, за допомогою якої дані, що надходять з сервера через API, створюють події (actions), взаємодіють та комбінуються з даними вже збереженими у додатку, і потрапляють у глобальне сховище (store), де можуть бути використані у будь якій частині додатка.

Для реалізації SPA парадигми в роботі використано JavaScript-бібліотеку React. Під час першого завантаження сторінки React відправляє клієнту лише

один початково не заповнений index.html шаблон. Разом з ним користувач отримує JavaScript файли, які реалізують JSX шаблони, тобто шаблони, які містять React компоненти, на основі яких будується Virtual DOM (Document Object Model), що є ідеалізованим станом інтерфейсу додатка користувача. Virtual DOM зберігається в оперативній пам'яті пристрою і синхронізується з реальним DOM методами бібліотеки React.

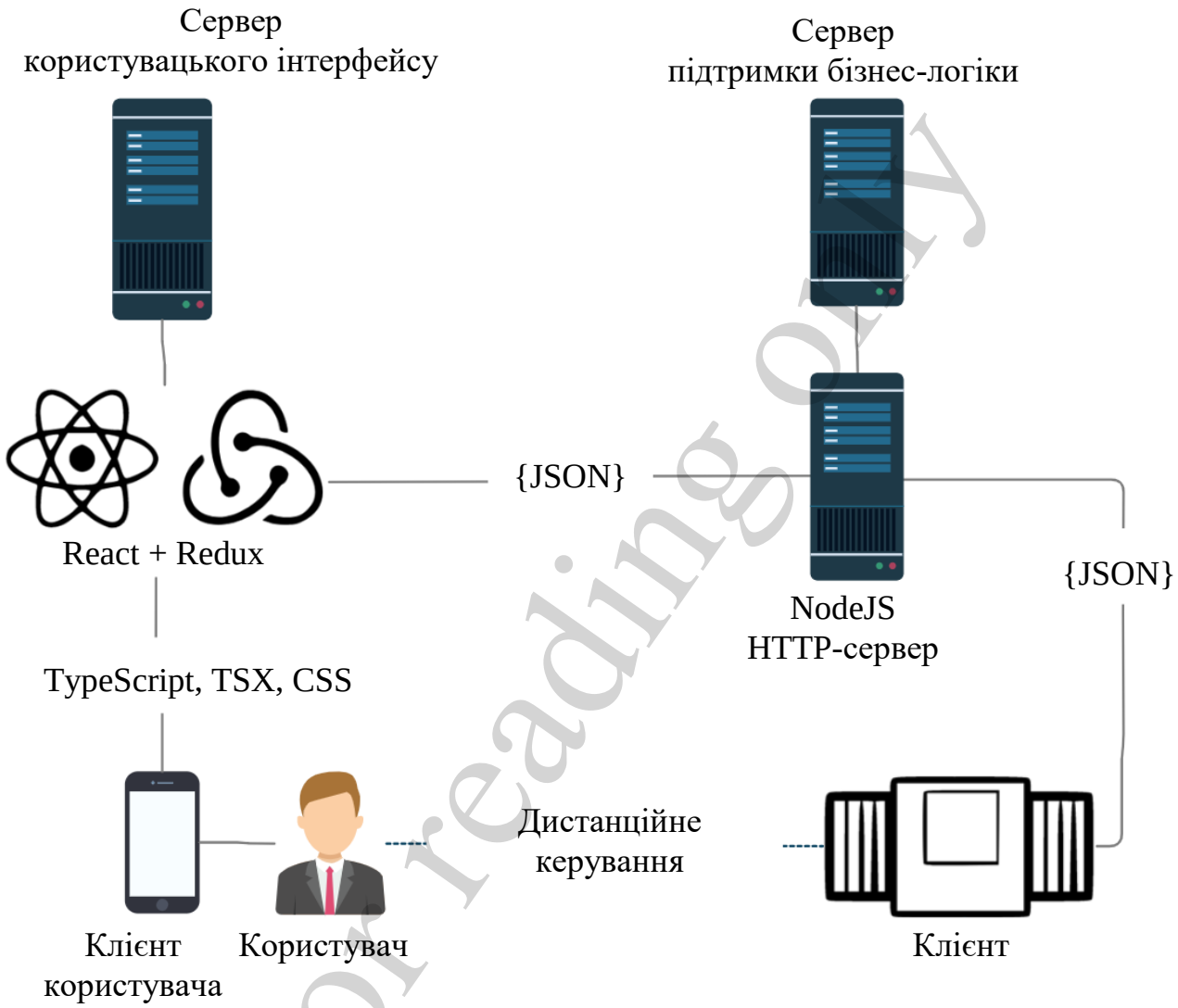

Рис. 6. Діаграма використаних технологій

При надходженні нової інформації з back-end сервера змінюється стан компонентів у Virtual DOM. Для цього використано WebSocket, хоча можна використати RestAPI та AJAX. Бібліотека React порівнює програмні гілки в VDOM та DOM. У випадку, коли була зафіксована невідповідність, відбувається перебудова сторінки (rerender) (рис. 7).

В роботі використано останню версію React, що надало доступ до хуків (hooks) – функцій, які надають можливість використовувати локальний стан компоненти (state) та інші можливості React без використання класів [15].

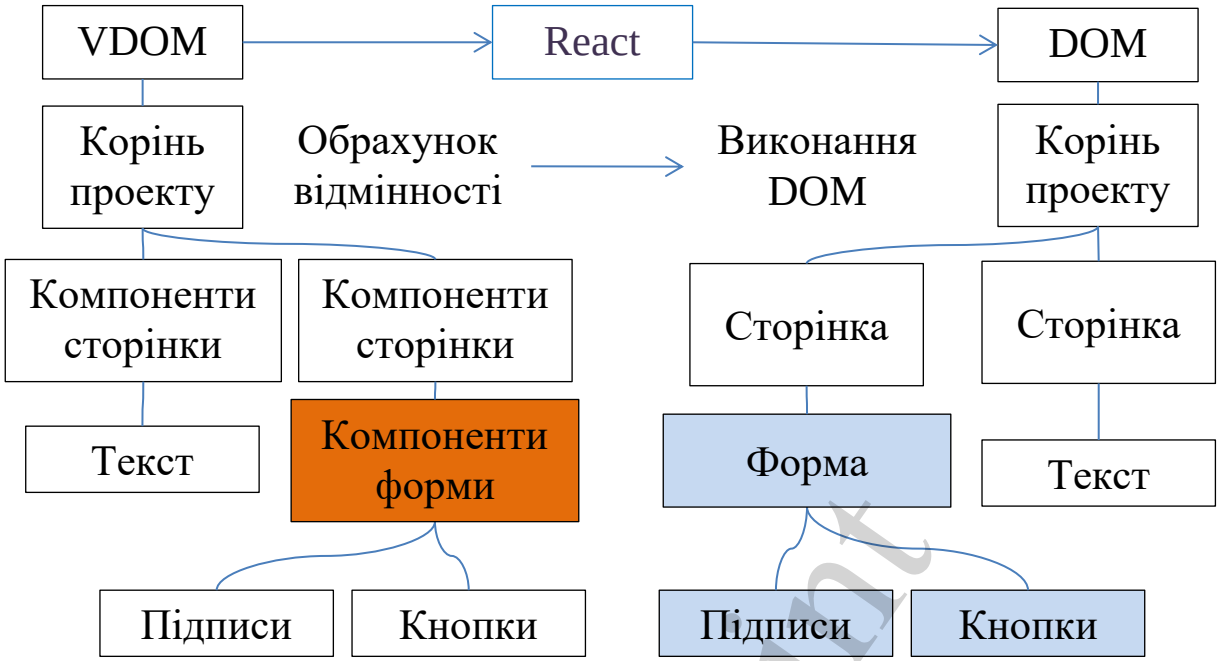

Рис. 7. Схема побудови DOM на основі VDOM

Під час використання функціонального стилю, React компоненти декларуються як функції. В програмній реалізації моделей використано Arrow Function Declaration метод при створенні функцій. Усі функції є анонімними і присвоюються константі, що потрібна для їх виклику. Для уникнення збоїв керування значною кількістю агентів, потрібно дотримуватися строгої типізації даних та зберігати їх сигнатуру. При декларуванні компоненти повинні бути вказані чіткі типи вхідних аргументів та результат роботи функції. Усі функціональні компоненти React повертають тип FC (Functional Component), що є JSX розміткою (рис. 8), на основі якої будується VDOM.

6. Апробація моделі мультиагентної системи мобільних роботів для дослідження алгоритмів машинного навчання

Для перевірки результатів теоретико-практичної розробки в роботі реалізовано можливість програмної симуляції роботи мультиагентної системи мобільних роботів. Симуляція поведінки роботів була реалізована за допомогою технології canvas, що є елементом HTML5. Canvas надає можливість включати у веб сторінку елементи растрової 2D графіки, а також побудову апаратно-прискореної 3D графіки на основі WebGL.

Додаток складається з двох секцій: Control та Simulation, на рис. 8 відповідно ліва та права частина.

Для отримання доступу до сторінки керування, користувач повинен обрати username та під’єднатися до віртуального клієнта – “Cube”, заповнивши поля форми у спливаючому вікні (рис. 9).

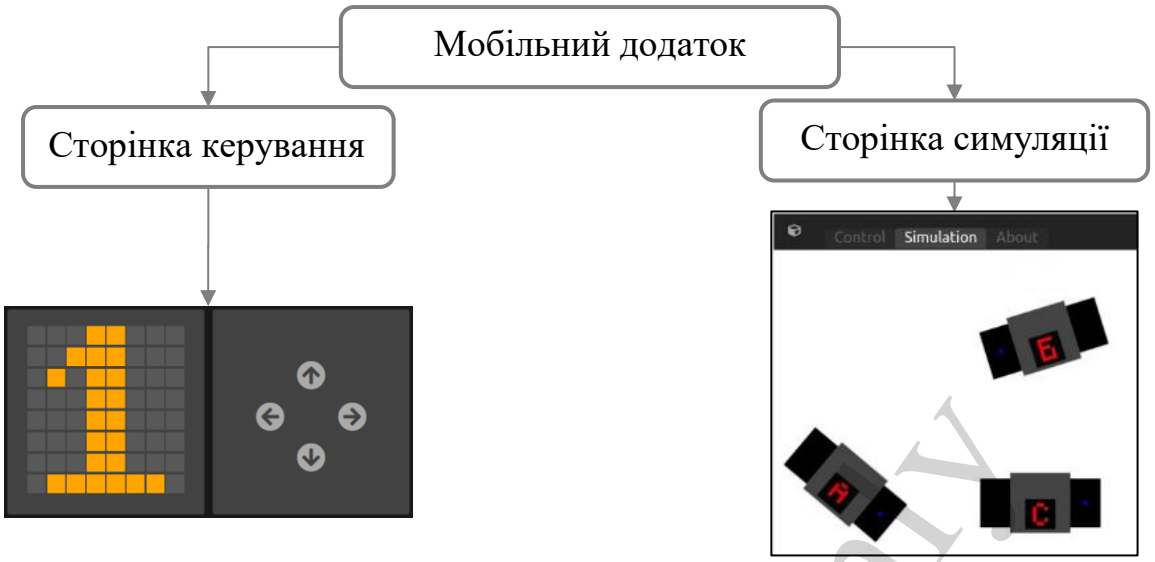

Рис. 8. Сторінки керування та симуляції мультиагентною системою

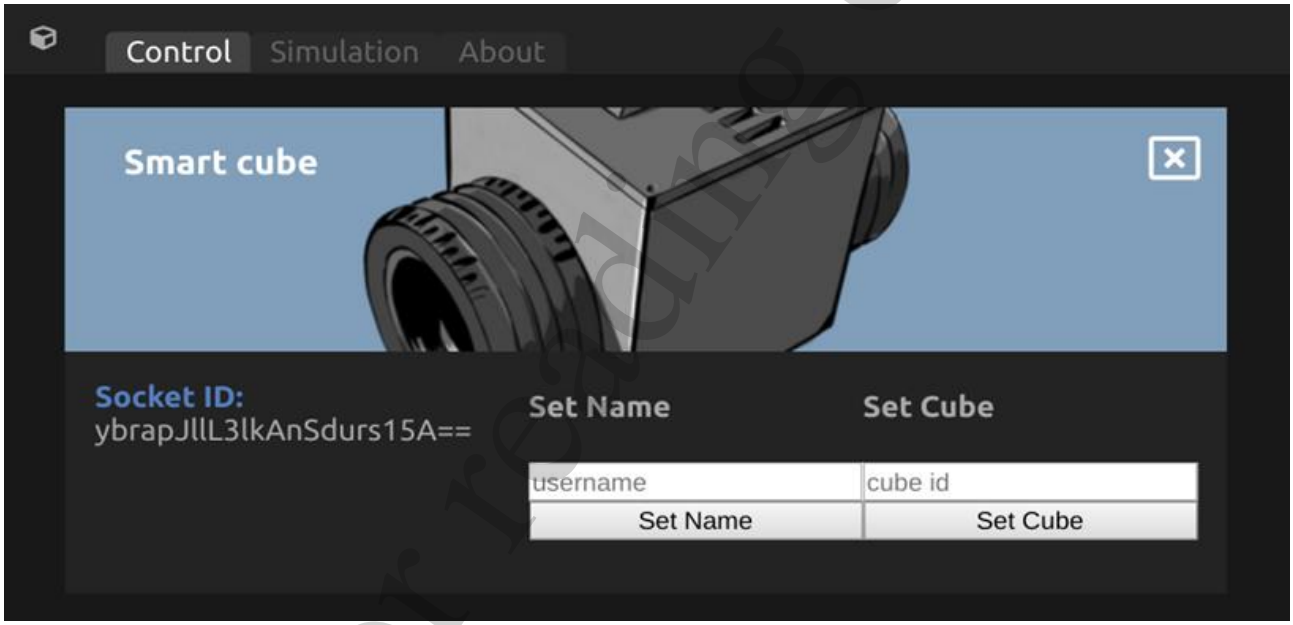

Рис. 9. Інтерфейс форми під'єднання до робота "Cube"

Форми заповнення імені користувача та id проходять валідацію на стороні сервера, та в разі виникнення помилки, користувачеві надходить повідомлення з відповідним текстом.

Заповнивши поле "Set cube" ідентифікаційним номером робота, не з'єднаного з іншим користувачем, user встановлює зв'язок з роботом і отримує доступ до секції керування.

Користувач може взаємодіяти апаратним забезпеченням робота, зокрема світлодіодною матрицею робота, що використана для генерації індивідуальної мітки робота "Cube" та пультом керування переміщення окремим роботом (рис. 8).

Вкладка “Simulation” – містить віртуальні моделі роботів “Cube”, та надає можливість здійснювати як віртуальну симуляцію роботи системи, так і відображати реальну поведінку системи мобільних роботів (рис. 10).

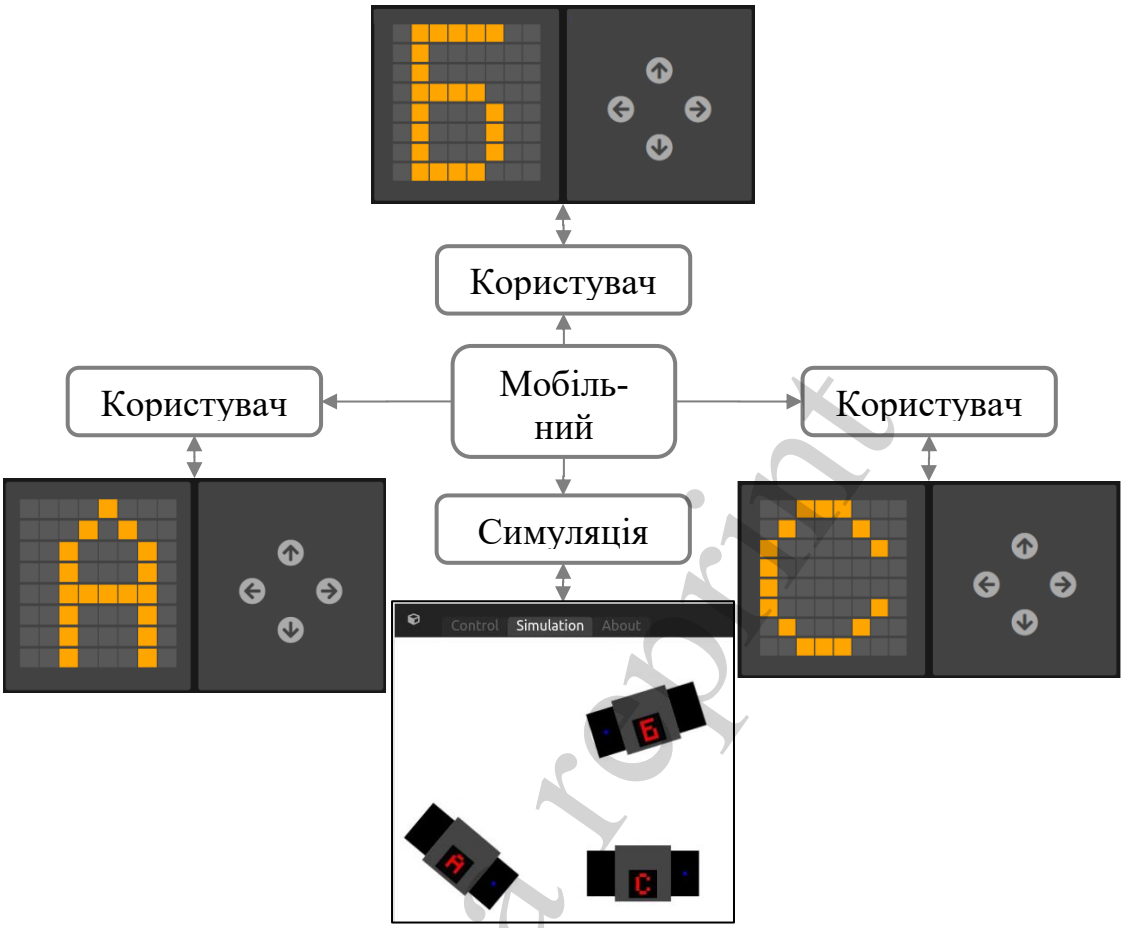

Рис. 10. Діаграма зв’язку моделей в симуляції та користувачів системи

Для оптимізації роботи додатка використано технологію lazy loading. Оскільки додаток користувача складається з двох великих секцій: джойстика управління, та вікна симуляції, – одночасно в RAM завантажується тільки одна частина VDOM дерева з потрібною секцією (рис. 11).

На момент розробки системи створено лише дві експериментальні апаратні одиниці роботів “Cube”. Роботи – це двоколісна платформа з центральною віссю та нульовим кліренсом. За рахунок такого рішення, роботи мають малі розміри центрального блоку – 50x50x50мм. В кришку робота вмонтовано світлодіодну матрицю для генерації графічної мітки, яка надасть можливість розрізняти кожного окремого клієнта на робочому полі. Для тестування інтерфейсу керування, та навантаження на back-end сервер була створена симуляція поведінки системи. Дана симуляція віртуальне середовище в якому знаходяться моделі роботів “Cube”. Моделі можуть бути як цілком віртуальними, так і відповідати командам, що надходять експериментальним зразкам приєднаним до системи.

Разом з командами двигунам, сервер відправляє оновлені дані про положення точки обертання, кута повороту, та де саме знаходиться точка обертання

робота. Таким чином, теоретичні результати дослідження реалізовані на віртуальній моделі та частково з використання апаратної реалізації, що надало можливість перевірити коректність розробки теоретичних положень та окреслило перспективи подальшого масштабування системи.

7. Обговорення результатів побудови моделі мережевої взаємодії складових мультиагентної системи мобільних роботів

Основна задача сучасних інженерів розробників, що працюють з алгоритмами штучного інтелекту, – апробація наявних та розробка нових алгоритмів. При цьому, значною проблемою є можливість перевірки результатів їх роботи на фізичних моделях. Типовим рішенням є розробка і апробація алгоритмів під конкретну задачу. Такий підхід звужує коло потенційних дослідників, оскільки не кожна дослідна установа може дозволити купівлю вартісних роботів чи приймати участь в комерційних розробках. Натомість, багато дослідних університетів мають в своєму штаті математиків, що займаються теоретичними розробками. Часом подібні роботи не отримують відповідного відгуку в науковому середовищі через відсутність апробованих результатів на фізичних зразках. Пропонована модель є універсальним інструментарієм, що дозволить мінімізувати витрати на апаратне забезпечення та за рахунок спрощеного WEB-орієнтованого інтерфейсу користувача забезпечити доступність для дослідників всіх рівнів підготовки. На даний момент в рамках роботи реалізовано два дослідні фізичні зразки – роботів “Cube”, апаратна реалізація яких описана в [16]. WEB-інтерфейс містить сторінку ручного керування (рис. 8–10), яку в подальшому буде доповнено функціоналом конструктора команд для побудови алгоритму програми автоматизованого управління діями роботів під час експерименту. Відповідно до побудованих моделей (рис. 5, 6), програмну складову апаратної частини роботів пропонується реалізувати у вигляді клієнтів, що виконують адресні команди від сервера. За такого підходу не має необхідності при зміні завдань, змінювати прошивку кожного робота окремо. Все керування здійснюється централізовано, з сервера. Система керування забезпечила можливість перевірки роботи системи віртуального моделювання та керування роботами в реальному часі. При цьому, для перевірки граничних можливостей системи відносно часу керування, сервер підтримки бізнес-логіки був розташований в Сінгапурі, а сервер користувацького інтерфейсу – в США. При цьому два користувачі також були територіально віддалені один від одного та від фізичних моделей роботів на відстань близько 30км. При цьому користувачі та роботи працювали в мережах різних провайдерів. Виміри, проведені під час експерименту, показали затримку між сигналом керування та реакцією робота рівною приблизно 0.9–1.2 с. При підключенні користувачів та клієнтів до одної мережі з використанням локального сервера, латентність системи зменшилася до 6 мс.

Відео-трансляцію руху роботів для користувачів було на першому етапі реалізовано засобами Zoom Cloud Meetings. Отримані результати цілком задовольняють лабораторним потребам та є придатними до використання при подальшій розробці системи дослідження алгоритмів машинного навчання. При переході до побудови реальних систем чи збільшенні кількості клієнтів понад 200 модель потрібно модифікувати. Необхідно перейти до багаторівневої архітек-

тури з використанням додаткових протоколів, таких як прогресивні реалізації HTTP та MQTT. Також варто передбачити місце в моделі для додаткового локального сервера. До його функцій відноситься завантаження підпрограми керування клієнтами, збір даних та відправка статистики на віддалений сервер.

Подальший розвиток даного дослідження полягає в наступному:

1) розробка удосконалених мобільних роботів;

2) збільшення функціоналу роботів та реалізації повноцінного зворотного зв'язку їх з сервером для передачі даних з датчиків;

3) реалізація системи розпізнавання образів для аналізу робочого поля та визначення координат кожного з роботів;

4) розробка понятійного користувацького інтерфейсу з простим конфігуратором поведінки робота;

5) удосконалення мережевої моделі роботів для можливості керування значною кількістю роботів з мінімальною затримкою в часі.

8. Висновки

1. За результатами пошуку й апробації існуючих технологій передачі даних з'ясовано, що для початкового етапу експерименту доцільною є розробка моделі мультиагентної системи мобільних роботів на основі протоколу WebSocket. Серед наявних технологій дана модель має найбільшу швидкодію, найменше завантажує центральні процесори клієнта та сервера, мінімізує витрати Інтернет трафіку.

2. Визначено складники моделі мультиагентної системи мобільних роботів, їх роль і взаємозв'язок. Базуючись на обраному протоколі WebSocket та структурі побудованої моделі, визначено базовий стек технологій для програмної реалізації результатів роботи. Представлені структури (рис. 5, 6) відповідають еталонній мережевій моделі OSI (The Open Systems Interconnection model) та включають в себе розподіл завдань на рівень користувацького інтерфейсу та рівень підтримки бізнес-логіки. Це надало можливість керувати прототипами роботів-клієнтів у реальному часі, моделювати та досліджувати виконання алгоритмів машинного навчання.

3. Здійснено апаратно-програмну реалізацію розроблених теоретичних положень. В роботі використано JavaScript бібліотека React з технологією SPA (Single Page Application) для реалізації Client Side Render. Для оптимізації швидкодії роботи системи використано технологію Virtual DOM (Document Object Model), що зберігається в оперативній пам'яті пристрою і синхронізується з реальним DOM.

Отримані результати дослідження придатні для побудови лабораторних моделей і надають можливість забезпечити відповідність навчальним та дослідницьким цілям.

Література

1. Степанов, П. П. (2019). Применение алгоритмов группового управления и машинного обучения на примере игры "Battlecode". Кибернетика и программирование, 1, 75–82. doi: https://doi.org/10.25136/2306-4196.2019.1.23527

2. Yang, E., Gu, D. (2004). Multiagent Reinforcement Learning for Multi-Robot Systems: A Survey. URL: https://www.researchgate.net/profile/Dongbing_Gu/

publication/2948830_Multiagent_Reinforcement_Learning_for_Multi-Robot_Systems_A_Survey/links/53f5ac820cf2fceacc6f4f1a.pdf

3. Elhajj, I. H., Goradia, A., Xi, N., Kit, C. M., Liu, Y. H., Fukuda, T. (2003). Design and analysis of internet-based tele-coordinated multi-robot systems. Autonomous Robots, 15, 237–254. doi: http://doi.org/10.1023/A:1026266703684

4. Cao, Y. U., Fukunaga, A. S., Kahng, A. B. (1997). Cooperative mobile robotics: antecedents and directions. Autonomous Robots, 4, 7–27. doi: http://doi.org/10.1023/A:1008855018923

5. Van der Zwaan, S., Moreira, J. A. A., Lima, P. U. (2000). Cooperative learning and planning for multiple robots. Proceedings of the 2000 IEEE International Symposium on Intelligent Control. Held Jointly with the 8th IEEE Mediterranean Conference on Control and Automation (Cat. No.00CH37147). doi: https://doi.org/10.1109/isic.2000.882949

6. Asada, M., Uchibe, E., Hosoda, K. (1999). Cooperative behavior acquisition for mobile robots in dynamically changing real worlds via vision-based reinforcement learning and development. Artificial Intelligence, 110 (2), 275–292. doi: https://doi.org/10.1016/s0004-3702(99)00026-0

7. Touzet, C. F. (2000). Robot awareness in cooperative mobile robot learning. Autonomous Robots, 8, 87–97. doi: https://doi.org/10.1023/A:1008945119734

8. Mataric, M. J. (1997). Reinforcement learning in the multi-robot domain. Autonomous Robots, 4, 73–83. doi: https://doi.org/10.1023/A:1008819414322

9. Michaud, F., Mataríc, M. J. (1998). Learning from history for behavior-based mobile robots in non-stationary conditions. Machine Learning, 31, 141–167. doi: https://doi.org/10.1023/A:1007496725428

10. Fernandez, F., Parker, L. E. (2001). Learning in large cooperative multi-robot domains. International Journal of Robotics and Automation, 16 (4), 217–226.

11. Bowling, M., Veloso, M. (2003). Simultaneous adversarial multi-robot learning. IJCAI'03: Proceedings of the 18th international joint conference on Artificial intelligence, 699–704. URL: http://www.cs.cmu.edu/~mmv/papers/03ijcai-grawolf.pdf

12. Liu, J., Wu, J. (2001). Multiagent Robotic Systems. CRC Press, 328. doi: https://doi.org/10.1201/9781315220406

13. Matarić, M. J. (2001). Learning in behavior-based multi-robot systems: policies, models, and other agents. Cognitive Systems Research, 2 (1), 81–93. doi: https://doi.org/10.1016/s1389-0417(01)00017-1

14. Srinivasan, L., Scharnagl, J., Schilling, K. (2013). Analysis of WebSockets as the New Age Protocol for Remote Robot Tele-operation. IFAC Proceedings Volumes, 46 (29), 83–88. doi: https://doi.org/10.3182/20131111-3-kr-2043.00032

15. Introducing Hooks (2020). React. URL: https://en.reactjs.org/docs/hooks-intro.html

16. Дідук, В. А., Савченко, Б. С. (2020). Робототехнічна система з віддаленим керуванням. Всеукраїнська науково-практична Інтернет-конференція "Автоматизація та комп'ютерно-інтегровані технології у виробництві та освіті: стан, досягнення, перспективи розвитку". Черкаси, 46–49. URL: https://conference.ikto.net/pub/akit_2020_16-22march.pdf