ВІСНИК ЛЬВІВ. УН-ТУ
Серія прикл. матем. інформ.
2009. Вип. 15. С. 291–312

VISNYK LVIV UNIV.
Ser. Appl. Math. Inform.
2009. Is. 15. P. 291–312

І Н Ф О Р М А Т И К А

УДК 681.142.36

ІНДУКТИВНІ МЕТОДИ МОДЕЛЮВАННЯ НАВІГАЦІЇ АГЕНТА

О. Годич, А. Накрийко, Ю. Щербина

Львівський національний університет імені Івана Франка
вул. Університетська, 1, м. Львів, 79000, e-mail: dais@franko.lviv.ua

Розглянуто два підходи до розв'язання задачі навігації агента. Головний елемент першого підходу – контролер, що ґрунтується на нейромережі, навченій на наборі експертних правил. Через значну чутливість до зміни розмірності простору станів такий підхід малопридатний для реального застосування. Використовуючи інший (індуктивний) підхід – навчання з підсиленням, в якому агент навчається лише в результаті взаємодії з навколишнім середовищем, можна отримати контролер, придатніший до практичного застосування. Подано аналіз параметрів алгоритму та їхнього впливу на результативність, швидкість і стабільність навчального процесу.

Ключові слова: навчання з підсиленням, експертні правила, нейронні мережі, автономна навігація агента.

1. ВСТУП

Робототехніка за останні десятиліття пройшла довгий шлях розвитку і з кожним роком набирає все більших обертів. Хоча до успішного розв'язання основних проблем робототехніки ще далеко, роботів застосовують у все ширшому колі задач. Одним з важливих і дуже цікавих розділів робототехніки є мобільні роботи. Вони ставлять перед дослідниками чимало складних і досі недостатньо добре розв'язаних проблем. Однією з таких проблем є задача навігації автономного мобільного агента (робота).

Ця задача може ставитися у різних формах, але найбільш загальною і складною є задача автономної навігації агента в умовах зовнішнього середовища, про яке наперед немає жодних даних. Тобто агенту не надають інформацію про розташування різноманітних перешкод (рухомих і нерухомих) в навколишньому середовищі. Агентом у цьому випадку є мобільний робот, який здатний рухатися самостійно. Ця задача є предметом досліджень багатьох галузей науки: теорії керування, теорії прийняття рішень, робототехніки, штучного інтелекту, машинного навчання тощо. За останні 20 років людство зробило суттєві кроки вперед у її вирішенні, проте повністю вирішеною проблему вважати ще зарано.

Ми розглянули два різних підходи до розв'язання цієї задачі і спробували оцінити переваги та недоліки кожного з них: підхід на підставі експертних правил і підхід на основі навчання з підсиленням.

2. ПІДХІД НА ОСНОВІ ЕКСПЕРТНИХ ПРАВИЛ

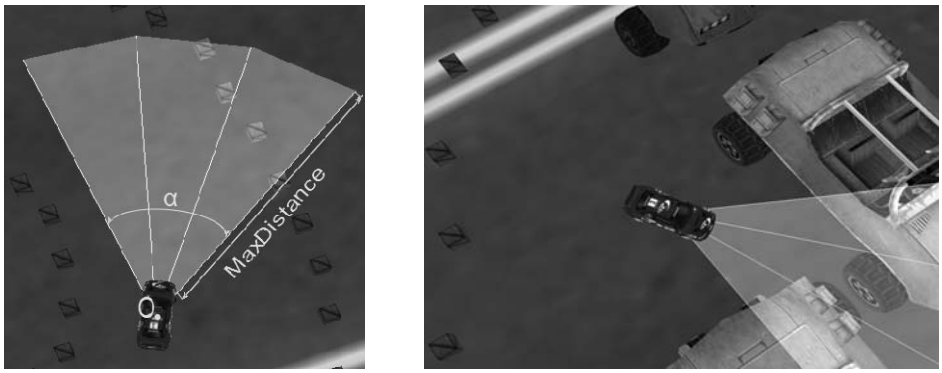
2.1. ФОРМУЛЮВАННЯ ЗАДАЧІ

Для першого підходу як агент обрали автомобіль з досить реалістичною фізикою руху, завданням якого був неперервний рух без конкретної кінцевої мети.

© Годич О.В., Накрийко А.В., Щербина Ю.М., 2009

Єдина вимога, яку ставили перед агентом, – мінімізувати зіткнення з перешкодами, заздалегідь оминаючи їх. Щоб якомога ліпше наблизити умови, в яких перебуває віртуальний автомобіль, до реальності, система керування сприймає лише об'єкти, які потрапляють у поле зору, обмежене певним кутом і дальністю огляду.

Агенту надають інформацію у відносних одиницях про те, наскільки близько є перешкода в межах кожного з трьох секторів "поля зору" (рис. 1, *а*). Керування автомобілем полягає у виборі значення прискорення з обмеженого діапазону та кута повороту керма, також у заданих рамках. Усі зазначені величини – дійсні числа.



а *б*

Рис. 1. Ілюстрації до першого підходу:
а) поле зору автомобіля; *б)* автомобіль зупинився і не може зробити розворот

2.2. ЕКСПЕРТНІ ПРАВИЛА ТА ШТУЧНА НЕЙРОННА МЕРЕЖА

Першим підходом до розв'язання задачі керування агентом було використання наперед заданого набору експертних правил, які становили пари еталонних вхідних і бажаних вихідних даних. Вхідними даними був вектор з відстаней до перешкод у трьох секторах поля зору, вихідними даними – вектор з еталонними прискоренням і кутом повороту. Оскільки усі можливі ситуації описати неможливо, то створили невелику вибірку з близько двадцяти еталонних навчальних пар, а для того, щоб узагальнити ці правила на усі можливі ситуації, використали нейромережу з прямим поширенням сигналу, яка завдяки здатності узагальнювати дані виконувала роль функції керування. Навчання відбувалося популярним алгоритмом зворотного поширення похибки. Завдяки узагальнюючій властивості нейромережі ми надіялися отримати контролер, здатний приймати "осмислені" рішення в ситуаціях, безпосередньо не заданих у системі правил, шляхом інтерполяції або екстраполяції "схожих" правил з навчальної вибірки.

Для перевірки ефективності рішення на основі експертних правил було створено віртуальне середовище, в якому рухався автомобіль. Керування відбувалося не через певні дискретні проміжки часу, а в неперервному режимі, як тільки ресурси комп'ютера давали змогу. Така особливість вносить певну залежність поведінки агента від потужності комп'ютера, на якому виконується симуляція, оскільки потужніший комп'ютер дає змогу більш оперативно керувати подальшим рухом агента.

2.3 РЕЗУЛЬТАТИ ПІДХОДУ НА ОСНОВІ ЕКСПЕРТНИХ ПРАВИЛ

Результати не можна охарактеризувати однозначно. З одного боку, система контролю на базі нейромережі досить адекватно керувала автомобілем з реалістичною фізикою, в більшості випадків успішно оминаючи перешкоди та забезпечуючи безперервний рух автомобіля в невідомому середовищі. Для підвищення рівня "природності" керування системі контролю надавали лише інформацію про об'єкти, які потрапляли в поле зору автомобіля з обмеженими дальністю та кутом огляду. Маючи лише ці обмежені дані, система керування забезпечувала завчасне обминання перешкоди з плавною зміною швидкості та напряму руху автомобіля. Варто зазначити, що важливу роль у досягненні зазначеної природності та адекватності керування відіграла вдало підібрана множина еталонних правил, що становила навчальну множину для нейромережі.

З іншого боку, найбільшою проблемою, з якою довелося зіткнутися, була наявність ситуацій, в яких автомобіль зупинявся і система контролю не могла вивести його з нерухомого стану. Одну з таких ситуацій зображено на рис. 1, *б*, в якій автомобіль заїхав у глухий кут і зупинився, не зумівши виконати розворот назад. Цю проблему, теоретично, можна було б розв'язати, ввівши у сприйняття агентом системи значення поточної швидкості та задавши додаткові експертні правила, які враховували б такі ситуації та давали змогу автомобілю виконувати розворот. Проте на практиці це досить проблематично, оскільки введення додаткової змінної вхідного вектора призводить до ускладнення поведінки всієї системи, підвищення рівня вимог до точності та повноти експертних правил.

Варто зазначити, що ще однією причиною такої поведінки є особливість фізичної моделі автомобіля – для того, щоб виконати поворот автомобіль повинен рухатись (вперед або назад). Така вимога досить проблематична, якщо автомобіль зупинився і перешкоди не дають змоги рухатися вперед. Цю проблему можна вирішити, якщо як агент використати транспортний засіб з іншим принципом руху. Будь-який мобільний агент, здатний розвертатися на місці, міг би уникнути такої проблеми. Прикладом може слугувати система гусениць танка, яка дає змогу виконати поворот на будь-який кут, залишаючись на місці.

Важливою проблемою, що постає при зазначеному підході, є вибір структури нейромережі. Якщо використати недостатню кількість нейронів, то нейромережа не зможе достатньо вивчити набір правил і, відповідно, не зможе повністю використати експертні знання. З іншого боку, використовуючи надто значну кількість нейронів або ж перетренувавши нейромережу, існує загроза надто точного запам'ятовування правил без адекватного узагальнення їх на схожі ситуації. В такому випадку нейромережа буде точно виконувати правила в описаних ситуаціях, але незначна відмінність ситуації від еталонної призведе до неадекватної зміни керування.

Отже, при використанні експертних правил з'являється велика кількість практичних питань, на які немає чітких теоретичних відповідей, а все доводиться вирішувати внаслідок численних практичних експериментів. Саме тому було вирішено випробувати інший підхід, який би не вимагав трудомісткого та складного процесу вибору хорошої системи експертних правил – самонавчання на основі досвіду безпосередньої взаємодії з середовищем.

3. САМООРГАНІЗАЦІЙНИЙ ПІДХІД – НАВЧАННЯ З ПІДСИЛЕННЯМ

Оскільки використання попереднього підходу сильно залежить від якості системи експертних правил, які при найменшому ускладненні сприйняття агентом середовища призводять до значних ускладнень експертних правил, то було вирішено відійти від моделі навчання з учителем. Натомість використали самоорганізаційний підхід. Головна ідея такого підходу полягає в тому, щоб дати змогу агенту розробити власну "внутрішню систему правил" щодо поведінки в середовищі внаслідок безпосередньої взаємодії з ним. Взаємодіючи з середовищем, агент отримує певний досвід, якщо задати певний механізм оцінки агентом власних дій, то в результаті достатньої кількості "досвіду", можна надіятися, що агент розробить ефективну стратегію поведінки. Такий підхід назвали *навчанням з підсиленням (reinforcement learning)*. При використанні навчання з підсиленням відпадає потреба в розробленні системи експертних правил, хоча, натомість, виникає проблема вибору ефективної та точної системи оцінки дій агента. Проте для достатньо складних систем зазвичай значно легше визначити механізм оцінки дій, аніж розробити повну та якісну систему правил.

Навчання, в процесі якого агент розробляє систему правил внаслідок багаторазової взаємодії з середовищем, є індуктивним, оскільки корисна інформація "видобувається" з необробленої інформації про середовище (інформація про віддалі до перешкод, сигналізація зіткнень тощо). Дедуктивний процес навчання, використаний у попередньому підході, для розроблення внутрішньої системи правил використовував структуровану та оброблену інформацію про середовище та еталонне керування, подану у вигляді готових еталонних правил. Треба зазначити, що в такому випадку дедуктивний процес частково охоплював й індуктивний підхід, оскільки нейромережа свою внутрішню структуру вибудовувала також з необроблених (з погляду нейромережі) даних. Загалом експертний підхід – дедуктивний за своєю суттю.

Далі ми детальніше розглянемо індуктивний підхід і результати його практичного застосування.

3.1. ЗАГАЛЬНІ ПОНЯТТЯ НАВЧАННЯ З ПІДСИЛЕННЯМ

У парадигмі навчання з підсиленнями наявні дві сутності – середовище та агент, що діє у ньому. Агент здатний сприймати середовище та має набір допустимих дій, якими він впливає на середовище (в нашому випадку – рухається у ньому). Уся сукупність інформації, доступна агенту в кожен дискретний момент часу t, називається *станом* і позначається s_t. Відповідно, можна ввести поняття множини усіх можливих станів середовища $\mathsf{S}(s_t \in \mathsf{S})$. Варто зазначити, що множина можливих станів може бути дискретною і неперервною. В більшості реальних застосувань множина станів неперервна, що значно ускладнює навчання. Також визначимо множину *дій* агента, які допустимі в стані s_t – $\mathsf{A}(s_t)$. Множина допустимих дій в загальному випадку також неперервна.

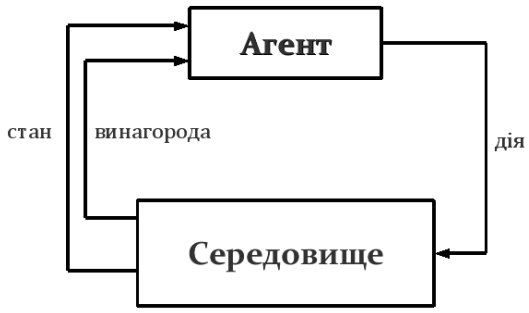


Рис. 2. Схема взаємодії агента та середовища

Схематично взаємодію агента та середовища показано на рис. 2. У дискретний момент часу t агент отримує *стан* середовища $s_t \in \mathsf{S}$, на основі цієї інформації він обирає та виконує певну *дію* $a_t \in \mathsf{A}(s_t)$, внаслідок чого отримує *миттєву винагороду* $r_t \in \mathsf{R}$. Мета агента – виробити таку *стратегію*, яка б кожній можливій ситуації s_t ставила у відповідність таку дію a_t, що дає в перспективі максимальну сумарну винагороду (*очікуваний прибуток* R_t). Цю стратегію прийнято позначати як функцію π. В найбільш загальному випадку, коли середовище, в якому повинен діяти агент, стохастичне, стратегія є відображенням $\pi : \mathsf{S} \times \mathsf{A} \to [0,1]$ простору станів і дій на ймовірність того, що дія a для стану s є оптимальною. В простіших системах, які детерміновані або стохастичність яких незначна, частіше використовують детерміновану стратегію $\pi : \mathsf{S} \to \mathsf{A}$, яка ставить у відповідність певному стану системи оптимальну дію в ньому.

Очікуваний прибуток R_t найчастіше визначають кількома способами, найпопулярніші з яких такі:

1) $R_t = \sum_{k=0}^{\infty} r_{t+k}$;

2) $R_t = \sum_{k=0}^{t_{\max}} r_{t+k}$;

3) $R_t = \sum_{k=0}^{\infty} \gamma^k r_{t+k}$.

Перше визначення очікуваного прибутку можна застосовувати в тих задачах, в яких дія агента може відбуватися як завгодно довго, можливо, навіть неперервно. Друге визначення підходить для задач, в яких дія агента відбувається епізодами зі скінченною кількістю ходів. Більш гнучким і зручним є третє визначення – зважена нескінченна сума миттєвих винагород. Параметр $\gamma \in [0,1]$ визначає коефіцієнт зважування миттєвих винагород. Його підбором можна гнучко регулювати важливість обраних дій залежно від їхньої віддаленості у часі. Якщо $\gamma = 0$, то агент “недалекоглядний” і буде дбати лише про максимізацію миттєвої винагороди тільки

на поточному кроці. При такому γ агент буде діяти жадібно і загалом така стратегія приведе лише до субоптимального очікуваного прибутку. В більшості задач часто буває вигідно на певному кроці виконати дію, яка не забезпечує максимальної винагороди на цьому кроці, зате дає змогу в майбутньому отримати більший сумарний прибуток. З наближенням γ до 1 агент надаватиме майбутнім винагородам все більшого значення, а отже, його поведінка стає “далекогляднішою”. В нашій задачі ми зупинимось на використанні саме зваженого очікуваного прибутку.

Існують різноманітні підходи до навчання агента. Одним з таких підходів є алгоритми, які ґрунтуються на побудові *функції корисності* $V(s)$, яка визначає максимальний очікуваний прибуток, який можна отримати зі стану s за умови строгого дотримання оптимальній стратегії і надалі. Серед цих методів можна виділити три категорії алгоритмів:

- методи динамічного програмування;
- методи Монте-Карло;
- методи часової різниці.

Ми розглянемо алгоритми, які належать до останньої категорії, оскільки перші дві не надто практичні для більшості реальних застосувань. Для детальнішого ознайомлення з цими методами можна звернутися до [2].

При визначенні функції корисності $V(s)$ неявно розуміється, що стратегія, згідно з якою відбувається вибір оптимальної дії на поточному кроці та на всіх наступних кроках, фіксована. Отож, функція корисності визначається стосовно певної стратегії π.

Корисність стану s згідно зі стратегією π $V^{\pi}(s)$ – це очікуваний прибуток, який можна отримати, якщо почати зі стану s і діяти надалі строго згідно з певною стратегією π

$$V^{\pi}(s) = E_{\pi}\{R_t \mid s_t = s\} = E_{\pi}\{\sum_{k=0}^{\infty}\gamma^k r_{t+k} \mid s_t = s\}, \quad (1)$$

де E_{π} – математичне очікування в випадку, якщо агент діє згідно зі стратегією π. Варто зауважити, що *очікуваний прибуток термінального стану*, тобто стану, при переході в який діяльність агента припиняється і епізод вважається завершеним, завжди дорівнює нулю. Назвемо функцію V^{π} *функцією корисності станів для стратегії* π.

Часто використовують розширення функції $V^{\pi}(s)$, яке виявляється для деяких алгоритмів значно зручнішим та ефективнішим. Визначимо корисність виконання дії a в стані s згідно зі стратегією π $Q^{\pi}(s,a)$ як очікуваний прибуток, що можна отримати, виконавши у стані s дію a та надалі строго притримуючись тієї ж стратегії π

$$Q^{\pi}(s,a) = E_{\pi}\{R_t \mid s_t = s, a_t = a\} = E_{\pi}\{\sum_{k=0}^{\infty}\gamma^k r_{t+k} \mid s_t = s, a_t = a\}. \quad (2)$$

Назвемо функцію Q^{π} функцією корисності дій для стратегії π.

Функції корисності V^π та Q^π загалом точно визначити неможливо, особливо при неповних даних про модель динаміки системи, проте їх можна оцінити з досвіду. В найпростішому випадку це можна зробити так. Якщо агент притримується стратегії π і для кожного стану зберігає середнє значення прибутку, отриманого в усіх епізодах, що відбулися, то це значення згідно з теоремою про великі числа збіжиться до математичного очікування корисності для заданого стану $V^\pi(s)$ за умови, що середні значення будуть оновлюватися для кожного стану нескінченну кількість разів. Аналогічне твердження правильне і для функції Q^π, якщо зберігати середні значення для кожної пари (s,a). Такі методи оцінки функцій корисності покладено в основу методів Монте-Карло, про які ми згадували раніше. Якщо кількість станів і можливих дій значна, то вимога нескінченної кількості оновлень для кожного стану стає на заваді точному наближенню функцій корисності.

Функції корисності визначають частковий порядок на множині усіх стратегій. Стратегію π називатимемо *ліпшою*, ніж стратегія π', якщо її очікуваний прибуток більший або дорівнює прибутку стратегії π' для усіх станів $s \in \mathsf{S}$

$$\pi \geq \pi' \Leftrightarrow (\forall s \in \mathsf{S})\left\{V^\pi(s) \geq V^{\pi'}(s)\right\}. \qquad (3)$$

Завжди існує стратегія, яка ліпша або ж не гірша від усіх інших. Така стратегія називається *оптимальною*. Хоча оптимальних стратегій може бути декілька, всіх їх будемо позначати як π^*. Усі оптимальні стратегії мають ту саму функцію корисності станів, яка називається *оптимальною функцією корисності станів* V^*

$$V^* = \max_\pi V^\pi(s), \qquad \forall s \in \mathsf{S}. \qquad (4)$$

Оптимальні стратегії також мають ту саму *функцію корисності дій* Q^*

$$Q^* = \max_\pi Q^\pi(s,a), \qquad \forall s \in \mathsf{S}, \forall a \in \mathsf{A}(s). \qquad (5)$$

Завданням алгоритмів навчання, які ми розглядаємо, є максимально ефективне та точне наближення функції корисності до оптимальної, використовуючи обмежені дані про взаємодію з середовищем.

3.2. НАВЧАННЯ З ЧАСОВОЮ РІЗНИЦЕЮ (TD-МЕТОДИ)

Методи *навчання з часовою різницею (temporal-difference, TD-навчання)* – це поєднання ідей методів Монте-Карло та динамічного програмування. TD-методи можуть, як і методи Монте-Карло, навчатися лише з досвіду, не потребуючи знання динаміки моделі середовища (це необхідна умова для методів динамічного програмування). З іншого боку, так само як і в методах динамічного програмування, TD-методи оновлюють свої оцінки частково на підставі інших оцінок, не чекаючи закінчення епізоду (методи Монте-Карло змушені чекати закінчення епізоду).

Достатньо ефективні на практиці та прості TD-методи навчання – *Q-навчання* та *Sarsa*. Ці алгоритми використовують функцію корисності дій $Q^\pi(s,a)$. В основу їхньої роботи покладено ітераційне почергове наближення стратегії π та функції корисності станів $Q^\pi(s,a)$ до оптимальних: π^* та $Q^*(s,a)$.

Наразі розглянемо алгоритм Sarsa. Будемо розглядати випадок, коли діяльність агента відбувається епізодично, тобто гарантовано за скінченну кількість кроків агент переходить у термінальний стан. Суть алгоритму така.

Початково довільно ініціалізується функція $Q(s,a)$. На початку кожного епізоду агент (наприклад, випадково) розташовується у середовищі і йому подається початковий стан s_0. Після цього агент вибирає найліпшу дію a_0 для стану s_0 на підставі значень функції $Q(s_0,a)$. Далі для кожного кроку епізоду агент виконує обрану дію a_t і отримує від середовища відповідну миттєву винагороду r_t та наступний стан s_{t+1}. Знову обирається найліпша дія a_{t+1} для стану s_{t+1}. Визначивши усе це, агент оновлює свою функцію корисності згідно з таким правилом:

$$Q(s_t,a_t) \leftarrow Q(s_t,a_t) + \alpha\left[r_{t+1} + \gamma Q(s_{t+1},a_{t+1}) - Q(s_t,a_t)\right], \quad (6)$$

де $\alpha \in \mathsf{R}$ – крок навчання.

Цей процес відбувається доти, доки агент не досягне термінального стану. Після цього починається новий епізод і все починається спочатку. В таблиці наведено алгоритм Sarsa.

Алгоритм Sarsa

Ініціалізувати $Q(s,a)$ довільно
Для кожного епізоду повторювати:
  $t \leftarrow 0$
  Ініціалізувати s_t
  Обрати a_t для стану s_t згідно зі стратегією, що ґрунтується на Q (наприклад, ε-жадібної)
  Повторювати для кожного кроку епізоду:
    Виконати дію a_t, отримати r_t, s_{t+1}
    Обрати a_{t+1} для стану s_{t+1} згідно з обраною стратегією
    $Q(s_t,a_t) \leftarrow Q(s_t,a_t) + \alpha\left[r_t + \gamma Q(s_{t+1},a_{t+1}) - Q(s_t,a_t)\right]$
    $t \leftarrow t+1$
  доки s_t – не термінальний стан

Алгоритм Q-навчання дуже схожий до Sarsa і відрізняється лише у правилі оновлення значення функції корисності Q

$$Q(s_t,a_t) \leftarrow Q(s_t,a_t) + \alpha\left[r_t + \gamma \max_{a \in \mathsf{A}(s_t)} Q(s_{t+1},a) - Q(s_t,a_t)\right]. \quad (7)$$

Q-навчання добре розроблене теоретично, існують теореми, які доводять, що наближення Q-функції у границі при нескінченній кількості взаємодій з середовищем і нескінченній кількості оновлень кожної можливої пари (s_t,a_t) збігаються до оптимальної Q^*-функції.

Q-навчання дуже просте для розуміння та теоретично обґрунтоване, проте часто на практиці значно ліпше спрацьовує алгоритм Sarsa, який не має настільки хорошого теоретичного підґрунтя. Варто зазначити, що після достатньої кількості ітерацій навчання, коли значення Q-функції вже не будуть настільки випадково розподіленими, як на початку навчання, дія a_{t+1} на наступному кроці все частіше буде оптимальною. Отож, оновлення для Q- та Sarsa-навчання з часом будуть відбуватися однаково. У цьому разі Sarsa-навчання на практиці часто забезпечує швидшу збіжність Q-функції до оптимальної.

3.3. СТРАТЕГІЇ ВИБОРУ ДІЙ

У розглянутих алгоритмах треба на кожному кроці робити вибір дії, яку повинен виконати агент. Це можна робити різними способами, проте для збіжності стратегії до оптимальної необхідною умовою є те, щоб кожна можлива дія була випробувана безліч разів за умови безмежної кількості ітерацій. Таку умову задовольняють так звані soft-стратегії, в яких ймовірність вибору кожної дії для будь-якого стану ненульова

$$\pi(s,a) > 0, \forall \pi, \forall s \in \mathsf{S}, a \in \mathsf{A}(s).$$

Найпростіший випадок soft-стратегії – ε-жадібна стратегія. В такій стратегії з ймовірністю $1-\varepsilon$ вибирається дія, що забезпечує максимум функції корисності, а з ймовірністю ε – випадкова допустима дія. Отож, будь-яка не жадібна дія виконується з ймовірністю $\frac{\varepsilon}{|\mathsf{A}(s)|}$, а "жадібна" – з ймовірністю $1-\varepsilon+\frac{\varepsilon}{|\mathsf{A}(s)|}$.

3.4. ТРУДНОЩІ У ВИПАДКУ НЕПЕРЕРВНОГО ПРОСТОРУ СТАНІВ

Якщо множина станів системи S дуже велика, як у випадку багатьох класичних ігор, або ж неперервна, що часто трапляється, наприклад, у робототехніці, то постає складне питання апроксимації функцій корисності. Існує багато варіантів вирішення цієї проблеми. Хоча ефективність кожного з методів сильно залежить від конкретної задачі, одним з найбільш універсальних методів є лінійні та нелінійні апроксиматори функцій. Лінійні апроксиматори, хоч і менш гнучкі та потужні, проте в більшості задач чудово справляються зі своїми завданнями. Водночас існують теоретичні гарантії збіжності до оптимальної стратегії для алгоритмів Sarsa та Q-навчання у випадку використання лінійних апроксиматорів функцій, якщо вони задовольняють певні умови ([3]).

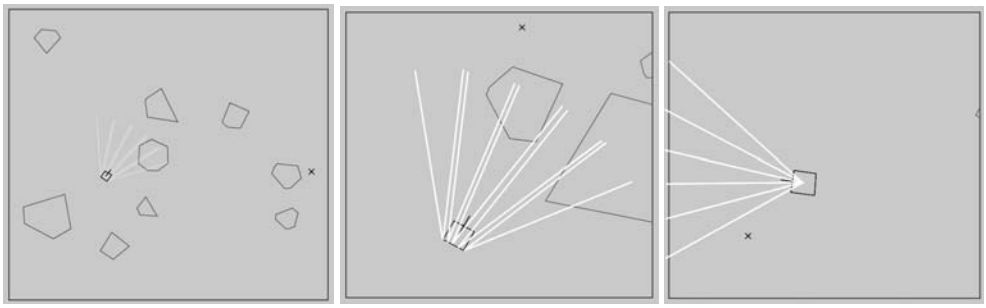
У випадку з нелінійними апроксиматорами, такими як штучні нейронні мережі, ситуація не настільки визначена. Загалом немає ніяких гарантій успішної збіжності до оптимальних стратегій. Для певних задач було доведено розбіжність розглянутих нами алгоритмів при використанні нейромережі як апроксиматора функції корисності (детальнішу інформацію можна знайти в [2]). Зрештою, нейромережі все ж таки успішно використовують у різноманітних алгоритмах навчання з підсиленнями. Здатність нейромереж до узагальнення дає змогу швидше навчати агента, оскільки, визначивши вартість певного стану, вона наближає і вартості близьких до нього станів. Отже, можна надіятись на швидшу збіжність стратегії до оптимальної.

У нашому випадку також була використана нейромережа як апроксиматор функцій і, як буде видно з результатів, досить успішно.

3.5. ФОРМУЛЮВАННЯ ЗАДАЧІ ДЛЯ ПІДХОДУ НА ОСНОВІ НАВЧАННЯ З ПІДСИЛЕННЯМ

Для того, щоб ефективніше використати навчання з підсиленням, ми внесли певні модифікації в формулювання задачі. Завдання агента було ускладнене – вимагалося, щоб агент не тільки оминав перешкоди, а й добирався до заданої цілі.

Щоправда, були деякі спрощення: час вважався дискретним, була спрощена фізика автомобіля. Тепер було можливо робити розворот на місці, що давало змогу уникнути певних проблем, з якими ми зіткнулися у попередньому підході. Також було дещо спрощено керування агентом: тепер були доступні лише п’ять дій.



а *б* *в*

Рис. 3. Ілюстрації до другого підходу:
а) віртуальне середовище; *б)* сприймання агентом середовища; *в)* невдалий варіант зони сприймання

Для випробування на практиці навчання з підсиленням в застосуванні до задачі керування автономним агентом було створено віртуальне середовище (див. рис. 3, *а*), в якому і відбувалися усі практичні експерименти. Середовище становить квадратну кімнату, в якій перебувають випадково генеровані та розташовані перешкоди – опуклі многокутники. Уся дія відбувається епізодами. Агент починає на початку кожного епізоду свій рух у випадковій позиції з випадковим початковим напрямом руху і повинен досягнути зазначеної точки, яка також генерується випадково. У цьому разі першочерговою вимогою є те, що агент повинен добратися до цілі без зіткнень з перешкодами та межами кімнати, другорядною – добирання до цілі повинно бути якомога швидшим.

Вхідні дані, які надавали агенту, були дещо розширені. Секторів огляду в агента стало 5, розміщувалися вони так, як показано на рис. 3, *б*. Таке розташування дає змогу сприймати об’єкти, які розташовані дуже близько збоку, що поліпшує поведінку агента. Також агенту надавали інформацію про відстань до цілі та кут між напрямом руху агента та напрямом на ціль.

Для навчання використовували алгоритм Sarsa. Як стратегію вибору дій вибрали ε-жадібну стратегію. Параметр γ прийняли таким, що дорівнює 0.99 для того, щоб зробити рух агента оптимальнішим у довгостроковій перспективі.

Агенту були доступні 5 можливих дій:

- рухатись прямо на фіксовану відстань;
- рухатись прямо на фіксовану відстань, одночасно повернувши наліво на фіксований кут повороту;

- рухатись прямо на фіксовану відстань, одночасно повернувши направо на фіксований кут повороту;
- залишитися на місці, повернувши наліво на фіксований кут повороту;
- залишитися на місці, повернувши направо на фіксований кут повороту.

Рух робота продовжується доти, доки не буде виконана одна з таких умов:

- досягнуто цілі;
- зіткнення з перешкодою або межею кімнати;
- перевищено ліміт кількості кроків (тайм-аут);

В усіх цих випадках епізод вважається закінченим. Залежно від того, яка саме умова спричинила закінчення епізоду, агенту надається певна винагорода. Після цього знову генерується випадковий епізод і все повторюється спочатку.

Нагадаємо, що для навчання з підсиленням треба визначити систему винагород, яка визначає яку винагороду отримує агент під час переходу з одного стану в інший. Ми вибрали таку систему винагород:

- при досягненні кінцевої цілі надавалася винагорода, яка дорівнює 1.00. Це є найбільшою миттєвою винагородою і повинно сприяти досягненню агентом кінцевої цілі;
- при зіткненні з перешкодою $-1+\frac{1}{2}e^{-2\frac{d_{crash}}{d_{size}}}$, де d_{crash} – відстань між місцем зіткнення з перешкодою та кінцевою ціллю агента; d_{size} – довжина сторони кімнати. Ця винагорода мінімальна, а отже, зіткнення – найменш бажана подія;
- при перевищенні ліміту дозволених кроків агенту видавали таку саму винагороду, як і у випадку зіткнення, проте більшу на 0.3. Це також небажана подія, проте безпечне пересування пріоритетніше, аніж зіткнення з перешкодою;
- в усіх інших випадках миттєва винагорода становила -0.01. Цим досягалося те, що агент намагався (неявно) мінімізувати кількість кроків до досягнення цілі.

Усі вхідні дані – дійсні числа, тому ми змушені були боротися з неперервним простором станів. Для цього як апроксиматор функції корисності ми використали нейромережу прямого поширення, яка на кожній ітерації навчалася одним проходом алгоритму зворотного поширення похибки. В усіх прошарках нейромережі як активаційну функцію використовували логістичну функцію ($f(x)=\frac{1}{1+e^{-x}}$), в останньому вихідному прошарку – одиничну ($f(x)=x$).

Для апроксимації функції $Q(s,a)$ використовували 5 окремих нейромереж, кожна з яких відповідала за свою дію. Цей підхід згідно з [1] ефективніший, ніж використання однієї нейромережі з кількістю виходів, яка дорівнює кількості можливих дій. В такому випадку для роботи нейромережі достатньо меншої кількості ітерацій навчання, а також меншої кількості нейронів.

На вхід нейромережі подавали спеціально закодовані 7 відомих параметрів стану: 5 значень відстаней для кожного з секторів поля зору, а також відстань до цілі та кут. Спосіб кодування взяли з [1]. Експерименти засвідчили, що таке кодування

вхідних даних суттєво полегшує задачу навчання нейромережі і робить його ефективнішим та якіснішим.

Усі вагові коефіцієнти нейромережі були початково ініціалізовані нулями, що завдяки одиничній функції у вихідному прошарку початково давало значення нуль для будь-якого входу. Це досить важливий момент. У літературі ([2]) рекомендують використовувати таке початкове наближення функції корисності $Q(s,a)$, яке б було оптимістичним стосовно реальної функції вартості. В такому випадку агент, вибравши певну дію і отримавши винагороду, яка є нижчою, ніж початкове оптимістичне наближення, наступного разу в тій самій ситуації буде схилятися до вибору іншої дії, пробуючи кожного разу іншу дію. Таке початкове випробування усіх можливих дій забезпечує швидше та точніше наближення реальної функції корисності.

Агент на перших стадіях навчання діє дуже неефективно. Оскільки ймовірність добратися до цілі, минаючи перешкоди, в умовах випадково генерованого середовища дуже незначна, то початково постійно буде отримуватися негативне значення винагороди. Тому нульова ініціалізація нейромережі становить оптимістичні сподівання, і, відповідно, є корисною евристикою.

3.6. РЕЗУЛЬТАТИ

Практичні експерименти засвідчили, що зазвичай агенту достатньо 1.2 мільйонів кроків навчання для того, щоб розробити ефективну власну стратегію, після чого вона практично не змінюється. Ефективність контролера оцінювали у відсотках успішних добирань до цілі за останні 5000 епізодів, щоб знівелювати вплив фактично випадкових дій на початку навчання.

Початково значення ε у ε-жадібній стратегії було фіксованим на позначці 0.1, що становило розумний компроміс між випадковістю поведінки та можливістю дослідження “нерозвіданих” дій. Вершини секторів області сприймання починалися у центральній точці робота. Таке поєднання налаштувань давало змогу агенту досягати успішності в $60\% - 65\%$ випадків. Робот поводив себе цілком адекватно, намагаючись оминути перешкоди, проте часто зачіпаючи боковою стороною краї перешкоди. Це зумовлено обмеженістю поля зору (рис. 3, *в*), оскільки сектори захоплювали лише ту частину простору, яка була безпосередньо перед роботом, ніяк не даючи знати про перешкоди, що є близько ліворуч та праворуч.

Для того, щоб дати більше інформації про навколишні перешкоди, поле зору робота було дещо видозмінене (див. рис. 3, *б*). Вершини секторів були зміщені в задню частину і рівномірно розподілені по всій задній стороні. Отже, у робота тепер були два сектори, які надавали йому інформацію про дуже близькі перешкоди, які розташовувалися ліворуч і праворуч від нього. Ці вдосконалення допомогли підвищити успішність агента до позначки $80\% - 82\%$ в більшості тестових запусків. Аналіз нової стратегії засвідчив, що поведінка робота справді стала значно безпечнішою та ефективнішою. У цьому разі можна було спостерігати як робот оминає перешкоду, об’їжджаючи її “впритул”, і час від часу виконує випадкові повороти в бік перешкоди. Це було зумовлено тим, що значення ймовірності вибору випадкової дії ε вибрали досить великим – 0.1. Тобто в середньому в одному випадку з десяти агент виконував випадковий вибір дії, що призводило до зіткнень при близькості до перешкоди.

Зменшення ε до 0.01 призвело до значного погіршення навчального процесу, оскільки алгоритму Sarsa не надавали достатньої можливості досліджувати нові стратегії, натомість поведінка агента збігалася до субоптимальної стратегії.

У результаті цих спостережень ми вирішили використати змінний показник ε, який лінійно спадав від 0.2 до 0.01 протягом 1 мільйона кроків навчання, після чого фіксувався на мінімальній позначці. Таке вдосконалення повинно було б допомогти агенту на початку дослідження детальніше досліджувати можливі невипробувані дії, зменшуючи ймовірність вибору неоптимальних дій тоді, коли в результаті багатьох ітерацій вивчена функція цінності давала хороше уявлення про оптимальні дії і необхідність додаткового дослідження зменшувалася.

Експеримент засвідчив, що таке вдосконалення насправді дає ліпші результати. З усіма вищезазначеними вдосконаленнями в 8 випадках із 10 успішність агента становила 87% – 92% успішних епізодів. В інших 2 випадках з 10, на жаль, агент виробляв стратегію, яка дуже часто просто зациклювала робота на місці, оскільки добратися агент до цілі не міг, ефективнішим варіантом було залишатися на місці і отримати винагороду все ж вищу, ніж при зіткненні.

Для того, щоб сформувати певне уявлення про еволюцію поведінки агента, розглянемо (рис. 4) рух агента в межах тієї самої конфігурації перешкод і початкових умов на різних стадіях навчання. На цих ілюстраціях символом + позначено рух робота, + в квадраті – фінальна позиція робота, + в крузі – початкова позиція робота, x – ціль.

Як бачимо, без будь-якого навчання рух агента зовсім випадковий. Після 100 тис. ітерацій агент намагається рухатися в напрямі до цілі, ігноруючи будь-які перешкоди. Після 500 тис. ітерацій агент будь-якою ціною намагається уникнути зіткнення, але перевищує ліміт дозволених кроків і не добирається до цілі. Після 1.2 млн ітерацій, як видно з рис. 4, *г*, агент рухається якомога оптимальнішою траєкторією в напрямі до цілі, досить віртуозно оминаючи перешкоди.

3.7. ВПЛИВ СТРУКТУРИ ТА ПАРАМЕТРІВ НЕЙРОМЕРЕЖІ ТА ε-ФУНКЦІЇ НА НАВЧАННЯ

Початково ми проводили усі експерименти з параметрами нейромережі та алгоритму навчання, описаними вище. У всіх експериментах ми використовували нейромережу, яка складалася з трьох прошарків, в першому розташовувалися 13 нейронів, в другому – 8, в останньому вихідному – один нейрон, який мав одиничну активаційну функцію і забезпечував лінійну комбінацію виходів попереднього прошарку.

Згодом ми вирішили дослідити вплив на ефективність і стабільність результатів навчання структури нейромережі та η-функції кроку навчання, яка застосовується при навчанні нейромережі за допомогою зворотного поширення похибки. Також було очевидним, що суттєвий вплив має вигляд ε-функції у ε-жадібній стратегії вибору дій, яку ми використовували.

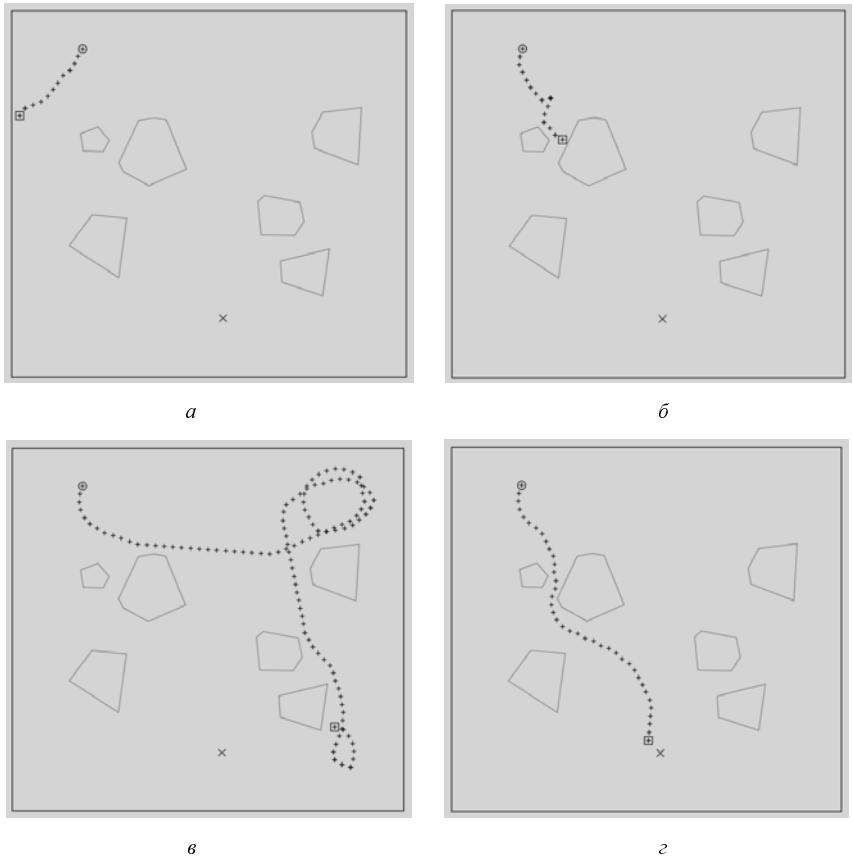
Для тестування обрали такі функції, яким ми для зручності дали умовні назви:

- η-функція кроку навчання нейромережі:
  - *eta-linear* – лінійно-спадна від 0.3 до 0.01 за 600000 ітерацій;
  - *eta-exp* – експоненційно-спадна від 0.4 до 0.01 за 600000 ітерацій (рис. 5, *а*);

- *eta-spikes* – пилкоподібна з чотирма “зубцями”, яка спадає від 0.4 до 0.01 за 600000 ітерацій (рис. 5, *б*).

- ε -функція:
  - *eps-linear* – лінійно-спадна від 0.3 до 0.0 за 300000 ітерацій;
  - *eps-spikes* – пилкоподібна з чотирма “зубцями”, яка спадає від 0.3 до 0.01 за 300000 ітерацій (аналогічна до *eta-spikes*).

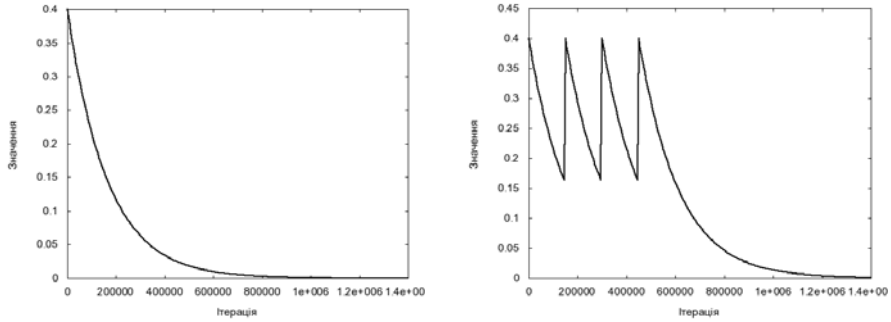
Ми провели тести з усіма можливими комбінаціями вищезазначених функцій для двох структур нейромережі: 10-6-1 та 7-4-1. Далі розглянемо ті, які, на нашу думку, заслуговують уваги. Для кожної розглянутої трійки параметрів (структура нейромережі, *eta* - та ε -функції) зображено три графіки, на яких показані кількості епізодів у відсотках від останніх 5000, які закінчились, відповідно, успішним досягненням цілі, зіткненням з перешкодою та перевищенням ліміту дозволених кроків у епізоді. Кожна крива на рисунку відповідає одному з трьох тестових запусків.



а *б*

в *г*

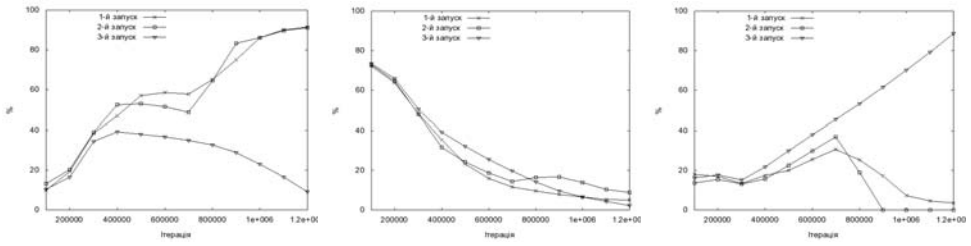
Рис. 4. Приклади траєкторій в умовах одного і того ж середовища:
а) без навчання; *б)* після 100 тис. ітерацій; *в)* після 500 тис. ітерацій; *г)* після 1.2 млн. ітерацій



а *б*

Рис. 5. Функції: *а)* експоненційна; *б)* пилкоподібна

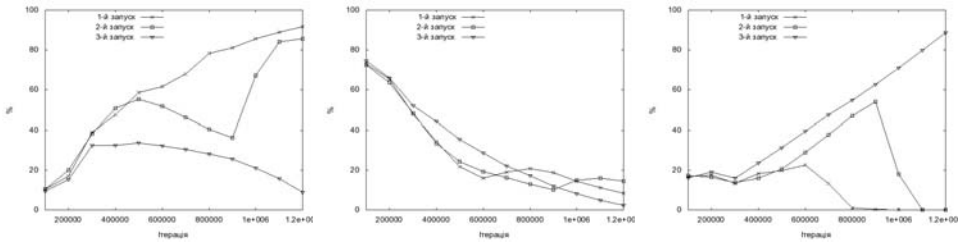
- Нейромережа: 10-6-1, η-функція: *eta-spikes*, ε-функція: *eps-linear* (рис. 6). Цей випадок аналогічний до усіх інших, які ми тут не будемо розглядати. Як бачимо, два з трьох тестових запусків досягають високої успішності, проте третій показує, що така комбінація параметрів не забезпечує належної стабільності результатів, тому що в третьому тестовому запуску агент "зривається" на стратегію, яка віддає перевагу тайм-ауту. Така ситуація притаманна більшості досліджених нами комбінацій параметрів.



а *б* *в*

Рис. 6. Нейромережа: 10-6-1, eta-spikes, eps-linear: *а)* успіх; *б)* зіткнення; *в)* тайм-аут

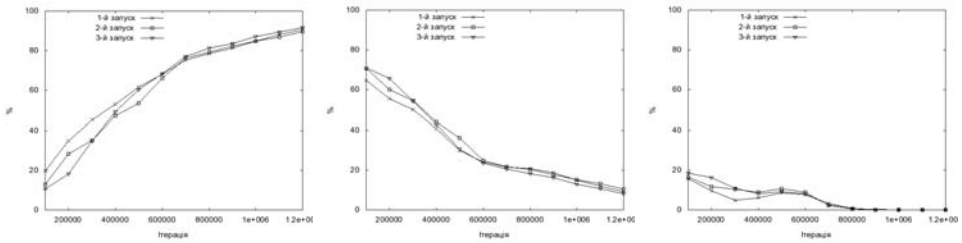
- Нейромережа: 7-4-1, η-функція: *eta-spikes*, ε-функція: *eps-linear* (рис. 7). Як бачимо, аналогічна поведінка, як і у попередньому випадку, незалежна від конфігурації нейромережі.



а *б* *в*

Рис. 7. Нейромережа: 7-4-1, eta-spikes, eps-linear: *а)* успіх; *б)* зіткнення; *в)* тайм-аут

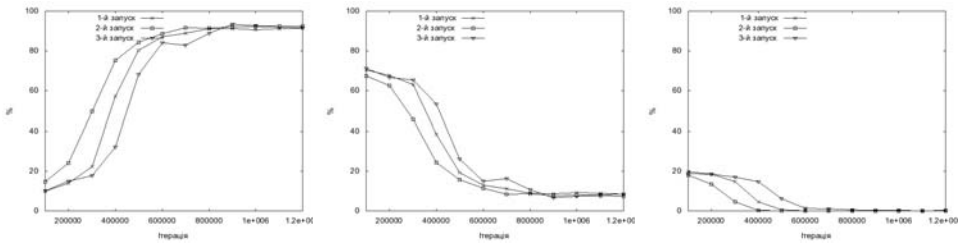
- Нейромережа: 10-6-1, η-функція: *eta-spikes*, ε-функція: *eps-spikes* (рис. 8). Ця комбінація параметрів дуже добра, забезпечує стабільне успішне навчання і порівняно високу швидкість збіжності. Варто зазначити, наскільки схожі графіки усіх трьох тестових запусків, що ще більше підтверджує хороші характеристики стабільності цієї комбінації.



а *б* *в*

Рис. 8. Нейромережа: 10-6-1, eta-spikes, eps-spikes: *а)* успіх; *б)* зіткнення; *в)* тайм-аут

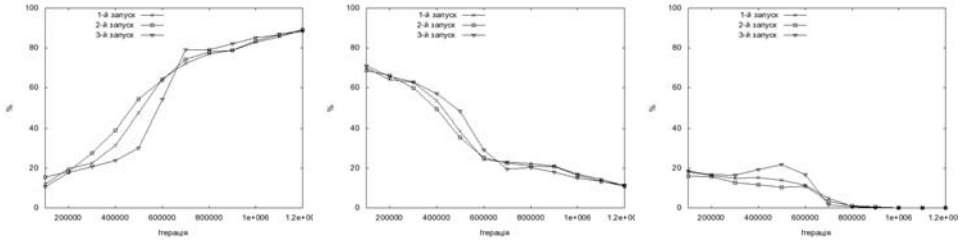
- Нейромережа: 10-6-1, η-функція: *eta-exp*, ε-функція: *eps-spikes* (рис. 9). Аналогічно до попередньої ситуації хороша стабільність та висока ефективність навченого агента, щоправда на початкових етапах навчання можна спостерігати досить значні відмінності у графіках. Щоправда після того, як вибрані η- та ε-функції досягають своїх мінімальних значень, ситуація з часом вирівнюється.



а *б* *в*

Рис. 9. Нейромережа: 10-6-1, eta-exp, eps-spikes: *а)* успіх; *б)* зіткнення; *в)* тайм-аут

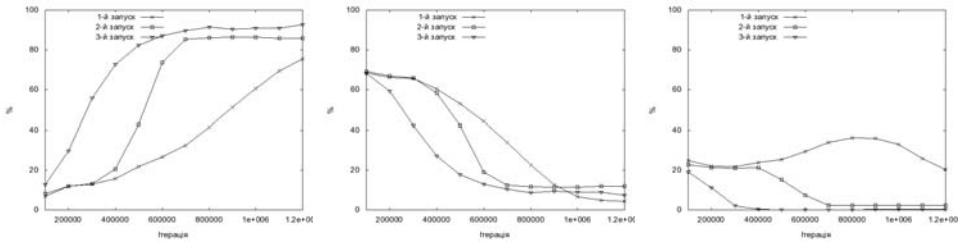
- Нейромережа: 7-4-1, η-функція: *eta-spikes*, ε-функція: *eps-spikes* (рис. 10). Аналогічно як і у випадку з більшою структурою нейромережі хороша стабільність та висока ефективність агента.



а *б* *в*

Рис. 10. Нейромережа: 7-4-1, eta-spikes, eps-spikes: *а)* успіх; *б)* зіткнення; *в)* тайм-аут

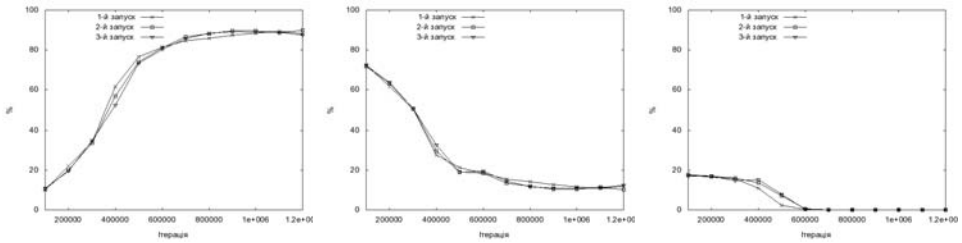
- Нейромережа: 7-4-1, η-функція: *eta-exp*, ε-функція: *eps-spikes* (рис. 11). Тут ситуація дещо відрізняється від схожої комбінації для нейромережі структури 10-6-1. Ми можемо спостерігати значні розбіжності в ефективності агента. Якщо продовжити навчання далі, то графіки ефективності агента зблизяться.



а *б* *в*

Рис. 11. Нейромережа: 7-4-1, eta-exp, eps-spikes: *а)* успіх; *б)* зіткнення; *в)* тайм-аут

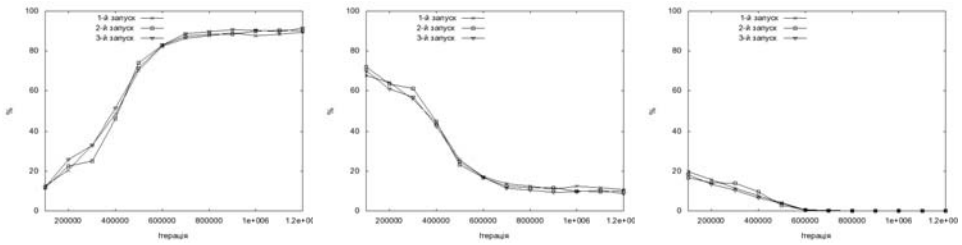
- Нейромережа: 7-4-1, η-функція: *eta-linear*, ε-функція: *eps-linear* (рис. 12). Ця комбінація виявилася дуже хорошою для нейромережі структури 7-4-1, хоча аналогічна комбінація функцій для більшої нейромережі поводила себе нестабільно. Очевидно, відіграло роль те, що лінійна η-функція у випадку більшого розміру нейромережі призводить до її самозбудження і робить навчання невдалим. Також помітно, наскільки близькими є графіки успішності агента, а отже, і стабільність цієї комбінації функцій.



а *б* *в*

Рис. 12. Нейромережа: 7-4-1, eta-linear, eps-linear: *а)* успіх; *б)* зіткнення; *в)* тайм-аут

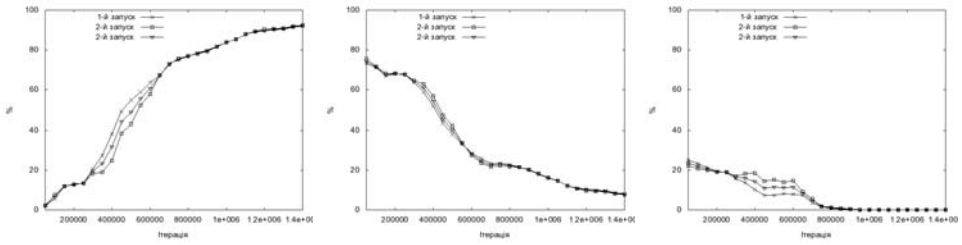
- Нейромережа: 7-4-1, η-функція: *eta-linear*, ε-функція: *eps-spikes* (рис. 13). Практично ідентична до попередньої ситуація. З цієї та попередньої комбінації видно, що в цих випадках ε-функція стратегії не має жодного впливу на навчання, що ще раз підкреслює виняткову важливість правильної настройки нейромережі та її параметрів.



а *б* *в*

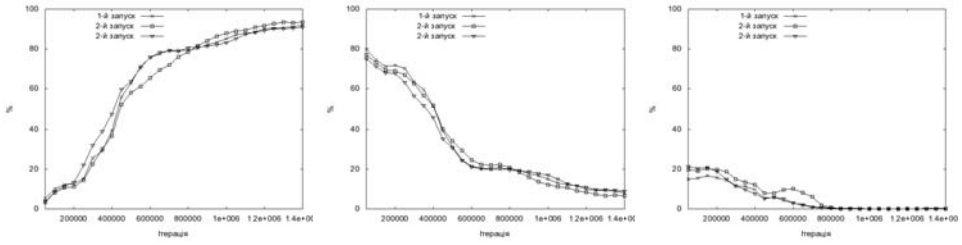
Рис. 13. Нейромережа: 7-4-1, eta-linear, eps-spikes: *а)* успіх; *б)* зіткнення; *в)* тайм-аут

Отримавши такі результати, ми спробували перевірити, наскільки впливає на успішність агента розмір нейромережі. Вибравши для тестів найбільш стабільну (згідно з попередніми тестами) комбінацію функцій: *eta-spikes* та *eps-spikes*, ми запустили тести для нейромереж таких конфігурацій: 3-2-1, 5-3-1 та 7-4-1. Отримані результати (рис. 14, 15, 16) дають підстави зробити цікавий висновок: для успішного навчання агента в сформульованій задачі зовсім не потрібна велика нейромережа, з завданням справляється нейромережа структури 10-6-1 та невелика структури 3-2-1.



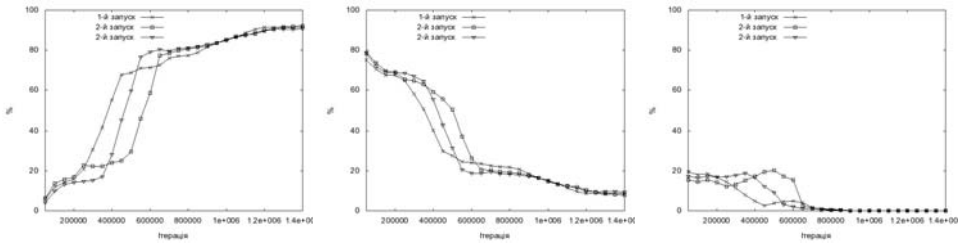
а *б* *в*

Рис. 14. Нейромережа: 3-2-1: *а)* успіх; *б)* зіткнення; *в)* тайм-аут



а *б* *в*

Рис. 15. Нейромережа: 5-3-1: *а)* успіх; *б)* зіткнення; *в)* тайм-аут



а *б* *в*

Рис. 16. Нейромережа: 7-4-1: *а)* успіх; *б)* зіткнення; *в)* тайм-аут

4. ВИСНОВКИ

Використовуючи парадигму навчання з підсиленням, ми створили контролер навігації робота, який, ґрунтуючись на обмеженій інформації про навколишній світ, забезпечує добирання робота до цілі, уникаючи зіткнення з перешкодами. Як засвідчили практичні експерименти, розроблений контролер досить ефективний і забезпечує успішне добирання до цілі в абсолютній більшості випадків (у середньому 87% – 92%).

Якщо порівнювати використаний підхід з попереднім підходом на базі системи експертних правил, які ми використали раніше, можна зауважити, що навчання з підсиленням забезпечує значно більшу гнучкість, даючи змогу змінювати структуру

стану системи, без жодних інших ускладнень. У випадку з системою експертних правил така зміна одразу приводить до труднощів, оскільки треба розробляти абсолютно нову систему правил, що є надзвичайно трудомістким і складним процесом. На жаль, досить важко кількісно порівняти два зазначені підходи, оскільки, по-перше, завдання агента в цих двох задачах були дещо різні: просто обминання перешкод у випадку з системою правил та добирання до цілі з обминанням перешкод у випадку навчання з підсиленням. По-друге, фізика моделей автомобіля та робота суттєво відрізняється, що унеможливлює об'єктивне порівняння. Суб'єктивно поведінка самостійно навченого робота виглядає значно раціональнішою, природнішою та більш адекватною порівняно з поведінкою автомобіля у схожих ситуаціях (наприклад, при під'їзді до близько розташованої перешкоди).

Варто зазначити, що в обох підходах є позитивні та негативні сторони. Для підходу на базі експертних правил характерним є швидкий процес навчання, проте сама розробка правил досить громіздка. Для навчання з підсиленням навпаки: немає затрат часу та сил на розробку системи правил, оскільки мобільний агент сам розробляє свої внутрішні правила, але процес навчання триває досить довго. Наприклад, виконання 2 млн. навчальних кроків на середньостатистичному комп'ютері триває понад годину. Неоднозначним для навчання з підсиленням є його залежність від евристичних параметрів: зміна параметра ε у ε-жадібній стратегії вибору дій з константного значення 0.1 на лінійно-спадну послідовність від 0.2 до 0.01 призвело до поліпшення результатів на 5% – 10%. Сильну залежність ефективності та стабільності навчання з підсиленням від вибору ε-функції стратегії вибору дій та η-функції кроку навчання нейромережі добре видно з експериментів описаних вище.

Можливо, саме така залежність навчання з підсиленням від евристичних параметрів є поясненням того, що розроблена нами система контролю за своїми показниками поступається схожій системі, розробленій G.A. Rummery ([1]), для якої характерними були показники рівня успішності робота в межах 96% – 99%. Мабуть, якіснішим підбором різноманітних евристичних параметрів навчання можна було б досягнути ефективнішої поведінки агента.

Зручність і легкість внесення змін у структуру середовища, його стану чи фізики моделі без зміни навчального процесу, а також хороші показники ефективності навченого агента – безсумнівні переваги навчання з підсиленням. У випадку з проблемою навігації, незначна зміна сприймання роботом навколишнього середовища так, що поле зору мінімально захоплювало територію зліва та справа від нього, без будь-яких інших вдосконалень дає поліпшення результатів на 15% – 17%.

Особливої уваги заслуговує використання штучної нейромережі як апроксиматора функції. Хороші властивості апроксимації та узагальнення нейромережі – головна складова успішного застосування навчання з підсиленням для складних проблем з неперервним простором станів, як у випадку з навігацією агента. Нейромережа показала себе чудовим інструментом, який доповнює парадигму навчання з підсиленням. Незважаючи на те, що в літературі досі не було подано будь-яких теоретичних доведень та умов збіжності алгоритмів навчання з підсиленням при використанні нелінійних апроксиматорів функцій, вкотре було показано практичну застосовність штучних нейромереж як функцій корисності для алгоритмів навчання з підсилення у складних задачах контролю.

Щоправда, нейромережа вносить додатковий рівень складності у розроблені алгоритми, вимагаючи обережного та точного підбору параметрів навчання самої нейромережі, а також вибору оптимальної структури, кількості нейронів та їхніх активаційних функцій.

Загалом індуктивні методи навчання дуже перспективний напрям роботи, оскільки дають змогу створювати ефективні контролери для задач, з якими дедуктивні методи не справляються через складність та неоднозначність сформульованого завдання.

ЛІТЕРАТУРА

1. *Rummery, G. A.* 1995. Problem Solving with Reinforcement Learning. Ph.D. thesis. Cambridge University Engineering Department.
2. *Sutton, R. S., Barto, A. G.* 2002. Reinforcement Learning: An Introduction. A Bradford Book, The MIT Press, Cambridge, Massachusetts, London, England.
3. *Coulom, R.* 2002. Reinforcement Learning Using Neural Networks, with Applications to Motor Control. Ph.D. thesis. Institut National Polytechnique de Grenoble.
4. *Kaelbling L., Littman M., Moore A.* 1996. Reinforcement Learning: A Survey. Journal of Artificial Intelligence Research 4, 1996, 237-285
5. *Philippe Kunzle*. Vehicle Control with Neural Networks. http://www.gamedev.net/reference/articles/article1988.asp

ИНДУКТИВНЫЕ МЕТОДЫ МОДЕЛИРОВАНИЯ НАВИГАЦИИ АГЕНТА

О. Годыч, А. Накрыйко, Ю. Щербина

Львовский национальный университет имени Ивана Франко,
ул. Университетская, 1, г. Львов, 79000, e-mail: dais@franko.lviv.ua

Рассмотрено два подхода к решению задачи навигации агента. Главной особенностью первого подхода является использование контроллера базирующегося на нейронной сети, обученной на наборе экспертных правил. В результате чрезмерной чувствительности к изменениям размерности пространства состояний, данный подход малоприменим в реальных условиях.

Использование индуктивного подхода базирующегося на обучении с усилением, в котором агент обучается только под средством взаимодействия с окружающей средой, можно получить контроллер, который более приемлем для практического использования.

Детально проанализированы параметры алгоритма и их влияние на результативность, скорость и стабильность процесса обучения.

Ключевые слова: обучение с усилением, экспертные системы, нейронные сети, автономная навигация агента.

INDUCTIVE METHODS FOR MODELLING AGENT NAVIGATION

O. Hodych, A. Nakryiko, Y. Shcherbyna

Ivan Franko National University of Lviv,
Universytetska str, 1, Lviv, 79000, e-mail: dais@franko.lviv.ua

This article discusses and compares two approaches for solving autonomous agent navigation problem. The first approach utilizes feed-forward neural network trained using a set of expert rules describing rules of navigation under different conditions. Due to an extreme sensitivity to the change of state space dimensionality, this approach is not considered very practical for real navigation tasks.

The use of an inductive approach, which is based on reinforcement learning, where agents self-organize as a result of interaction with the surrounding environment, yields a much better result. The obtained results suggest the possibility of using such agents in real-life applications.

The article presents a detailed discussion of parameter initialization, and its influence on the speed and stability of the learning process.

Key words: reinforcement learning, expert systems, neural networks, autonomous agent navigation.

Стаття надійшла до редколегії 07.09.2009
Прийнята до друку 04.11.2009