та документами. Нажаль, пошук алгоритму автоматичного розрахунку найкращого значення k – це все ще відкрита дослідницька проблема. Найкраще значення цього параметру залежить від конкретної задачі, корпусу, та його розміру. Значення k підбирають в залежності від цих факторів емпіричним способом.

Латентно-семантичний аналіз можна використати для пошуку плагіату в коді програми наступним чином. Оскільки ЛСА представляє кожен документ як невпорядковану сукупність термів, то для аналізу файлу з програмним кодом не потрібен синтаксичний аналізатор. Достатньо лише залишити в документі всі імена та ідентифікатори та прибрати все інше. Також прибираються коментарі (адже вони не мають відношення до безпосередньо логіки роботи програми), терми, які зустрічаються лише один раз, а також терми, які зустрічаються в усіх документах. Потім будується матриця термів-на-документи, виконується SVD декомпозиція та зменшення рангу матриці до k, щоб отримати матриці U_k, Σ_k та V_k. Далі отримується матриця документів-на-документи наступним чином:

$$F = (V_k\Sigma_k)(V_k\Sigma_k)^T. \quad (3)$$

Кожен елемент f_{ij} матриці F є коефіцієнтом подібності між документом та документом j. Пари документів, коефіцієнт подібності між якими вищий за певне встановлене значення, упорядковуються та видаються користувачеві. Звісно, цей метод сам по собі не дозволяє визначити які саме частини програмного коду були запозичені, а лише вказує на схожість між двома файлами. Тим не менш, цього може бути цілком достатньо, крім того, цей метод можна поєднати з іншими, наприклад з методом відбитків.

Список використаних джерел:

1. Mike Joy and Georgina Cosma. An approach to source-code plagiarism detection and investigation using Latent Semantic Analysis. IEEE Transactions on Computers, 2012.

2. R. Baeza-Yates and B. Ribeiro-Neto. Modern Information Retrieval. ACM Press / Addison-Wesley, 1999.

3. M.Berry, S.Dumais, G.O'Brien. Using linear algebra for intelligent information retrieval. Technical Report UT-CS-94-270, University of Tennessee Knoxville, TN, USA, 1994.

Стангріт Н.С.

студент,

Національний технічний університет України

«Київський політехнічний інститут імені Ігоря Сікорського»

ПОБУДОВА РОЗРЯДЖЕНОГО ВОКСЕЛЬНОГО ДЕРЕВА ОКТАНТІВ ДЛЯ ПОШУКУ ШЛЯХУ

Пошук шляху штучним інтелектом доволі актуальне питання. Це завдання виникає у таких галузях як розробка ігор, моделюванні деяких процесів. Слід уточнити, що ми говоримо про пошук шляху у віртуальному середовищі, а не реальному.

Більшість сучасних методів, які активно використовуються, розроблені для двох вимірного простору, або, у разі трьох вимірного простору, мають не чітку структуру, що ускладнює пошук шляху. Проте деякі агенти мають діяти у повноцінному трьох вимірному просторі. І хоча деякі моделі навігації дозволяють будувати себе в такому просторі, в такому процесі можуть виникнути проблеми, а утворена структура не є чіткою і збільшує час роботи з нею.

Найпопулярніші алгоритми використовують здобутки полігональної моделі, будуючи ніби трьох вимірну модель вільного від перешкод простору. Така модель називається навігаційна сітка.

На відміну від навігаційної сітки, яка використовує полігони, як засіб позначення вільного простору, ми будемо використовувати альтернативний шлях, вокселі.

Тож, сумуючи наведені вище доводи, можемо сказати, що актуальним питанням є розробка моделі навігації, яка буде спроможна працювати в повноцінному трьох вимірному середовищі, точніше буде навіть орієнтована на роботу тільки з ним.

Існують різні моделі, для вираження простору навігації, який використовується для пошуку шляхів. Пошук шляху по таким моделям поділений на декілька виді: глобальний пошук шляху, та локальний. Глобальний пошук дозволяє агенту рухатись з початкової позиції до кінцевої по оптимальному шляху. Цей оптимальний шлях зазвичай є найкоротшим. Глобальний пошук шляху вимагає інформацію про усі перешкоди на локації. Тоді як локальний пошук шляху дозволяю лише уникати перешкод, на шляху до цілі, і не завжди буває оптимальний.

В усіх моделях існують як плюси, так і мінуси. Можна побачити що для повноцінних випадків трьох вимірного простору наявно менше варіантів, і більшість з них має проблеми з швидкодією та витратами пам’яті. Також, у випадку об’ємної сітки навігації лише її варіант з об’ємними многогранниками дає можливість оперувати повним простором, на відміну від варіанту, де полігони можуть лежати в різних площиннах.

Тож, якщо провести дослідження по цій моделі, і змінити деякі її принципи, такі як: розташування вузлів; обробка не вільного, а зайнятого простору; обробка пустих просторів не в режимі графу, можна очікувати отримувати результати, які будуть ліпшими за стандартну модель.

Таким чином, наша мета – пошвидшення пошуку шляху в моделі навігації, зменшення її розміру, можливість динамічної зміни.

Щоб реалізувати заявлені задачі необхідно визначити в якому виді буде представлена модель. Так як ми будемо використовувати Воксельну сітку, за основу можна використати дерево октанів, чи k-мірне дерево.

Основна ідея, це ділення вузла дерева на 8 потомків, розташованих, як октанти в декартовій системі координат. Так як нас хвилює використання пам’яті, слід одразу взяти модифіковану версію такого дерева – Розряджене дерево октанів, яка не створює потомків, коли немає необхідності і такі потомки не несуть корисної інформації. [1].

K-мірне дерево – структура даних, яка служить для організації точок в k-мірному просторі. Являє собою варіант бінарного дерева пошуку.

Така структура дозволяє швидку шукати сусідів елементу, або сам елемент по його позиції. Так пошук йде в вузол, центр якого знаходиться найближче до позиції, по якій ми шукаємо. Складність такого пошуку буде наступним:

$$O\left(k * n^{1\frac{1}{k}}\right), \quad (1)$$

де $k -$ розмірність простору, $n -$ кількість вузлів

Для того, щоб можна було розташовувати модель в просторі під будь-яким кутом, та зменшити витрату пам'яті на зберігання координат, необхідно використовувати матрицю трансформації. Вона дозволить зберігати усі координати цілочисенними.

Одна з основних відмінностей досліджуваної моделі в тому, що сама вона містить інформацію лише про перешкоди, а не вільний простір. Тож методи побудови графу, які використовуються в інших моделях тут не допоможуть.

Для реалізація запропонованих алгоритму та моделі доцільно використовувати спеціальне програмне забезпечення, яке повинно відповідати наступним вимогам.

Отже, маємо V, який представляє собою куб, де ми знаємо дві вершини – VA, VB, що розташовані на протилежних вершинах довільної діагоналі цього кубу. Центр V буде наступним:

$$V^C = \frac{V^A + V^B}{2} \quad (2)$$

А розмір:

$$V^S = |V^A - V^B| \quad (3)$$

V може містити, або не містити, 8 менших V. Менші V будемо записувати як V[1.8]. Індекси цих вершин строго відповідають їх розташуванню, як показано на малюнку 1.

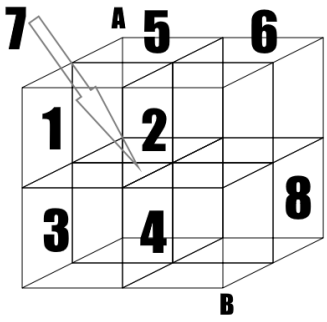

Рис. 1. Розбиття V на менші V

Також маємо множину C, яка включає в себе всі коллайдери довільної форми. Умова, яка необхідна, щоб поділити V на менші V наступна:

$$V \subset \forall C \quad (4)$$

Тож, якщо V перетинається з будь-яким елементом множини C, і:

$$|V^S| > S_{min} \quad (5)$$

до де S_{min} – мінімальний розмір будь-якого вокселя, то воксель розбиватися на 8 менших вокселів.

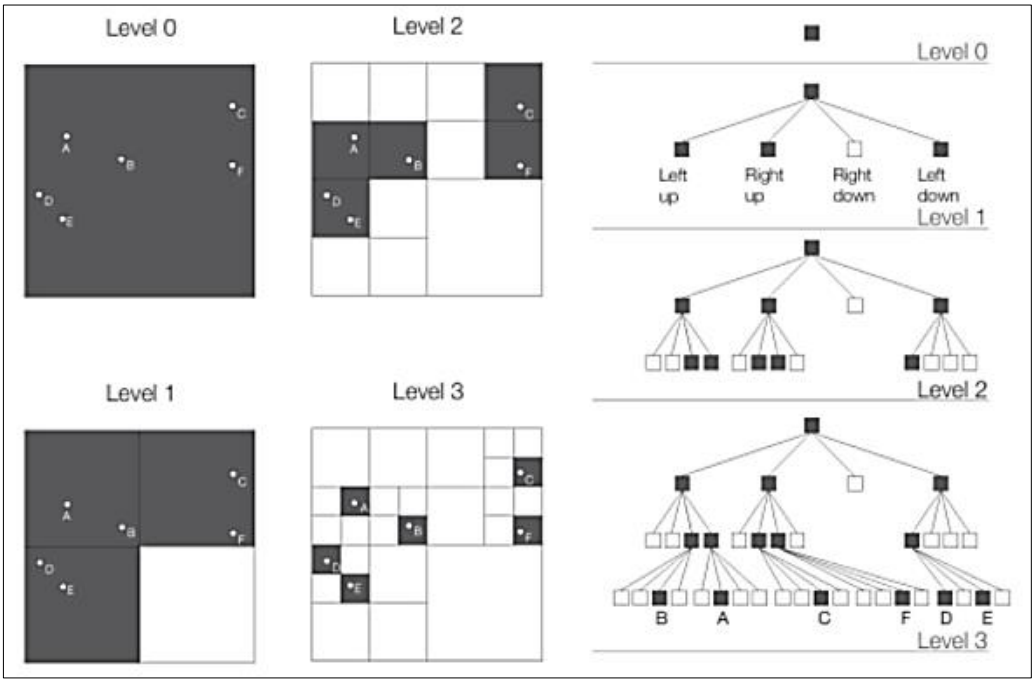

Рис. 2. Приклад розбиття на двовимірному просторі

Таким чином розділення кожного вокселя виконується рекурсивно, доки воксель не буде перетинатися з будь-яким коллайдером, чи його розмір не буде менше за деякий заданий мінімальний розмір [2].

Для побудови графу необхідні найменші вокселі, тобто такі, у яких V[1.8] = ∅. Якщо воксель не моє перетину з жодним коллайдером, то в залежності від обраної моделі він може не прийматися до уваги. Кожна сторона вокселя обробляється в залежності від обраної моделі побудови. На виході обробки маємо множини N[1.8], які зберігають сторону S, та утворені нею вузли S[1.5](Кожна сторона утворює п’ять вузлів). Основним параметром моделі є вектор гравітації, чи його відсутність, та максимальний кут нахилу площини до нього.

Якщо довжина вектору гравітації G не дорівнює нулю, або максимальний кут нахили до гравітації $\alpha_{max} > 0$, виконується тест сторін:

$$acrcos\left(\frac{NG}{|N||G|}\right) > \alpha_{max}, \quad (7)$$

де N – нормаль деякої сторони.

Усі вокселі сортуються по розміру, від найбільшого, до найменших, – B[1..∞]. Починаючи з найбільших вокселів знаходяться дублікати вершин, до уваги беруться протилежні сторони вокселів, такі дублікати видаляються.

Коли усі дублікати вершин, що знаходяться в одно розмірних вокселя були видалені, починається видалення вузлів, незважаючи на направлення сторони та розмір вокселя. Усі вершини тестуються на дублікати, та зайві видаляються.

Такий дворазовий прохід дозволяє очистити вузли від зайвих дублікатів на першому проході, який більш швидкий, ніж повний пошук дублікатів по всіх вузлах. Та видалити вузли, розташовані в середині перешкод.

Отже, розроблено модель навігації, яка, на відміну від інших, відповідає таким вимогам: при побудові надається інформація про перешкоди, які можуть бути виражені у виді простих геометричних об'єктів та полігональної сітки; модель зберігається для її повторного використання; при використанні, подаючи початкову точку шляху і кінцеву, маємо отримувати повний шлях; можна використовувати як локальний пошук шляху, так і глобальний, або їх комбінацію

Список використаних джерел:

1. Geometric Modeling Using Octree Encoding, DONALD MEAGHER, Rensselaer Polytechnic Institute, Troy, New York 12181, June 19, 1981.
2. P. Hubbard, «Interactive collision detection,» Proceedings of IEEE Symposium on Research Frontiers in Virtual Reality '93, 1993.

Старцев О.Р.
студент,
Национальный технический университет Украины
«Киевский политехнический институт имени Игоря Сикорского»

РАЗВИТИЕ ДОРОЖНО-ТРАНСПОРТНЫХ СЕТЕЙ В ОБОЗРИМОМ БУДУЩЕМ

Стремительное развитие автомобильного транспорта и его удешевление привело к огромному скачку количества этих самых автомобилей на улицах наших городов и стран. Увеличение числа автотранспорта на дорогах приводит к созданию заторов [1, с. 20], что негативно сказывается на времени передвижения человека или груза из точки А в точку Б.

К сожалению, расширение дорог за счет увеличения автомобильных полос приводит к уменьшению заторов лишь на малый период времени. Дальше происходит очередной рост количество автомобилей и появляются новые заторы. И катализатором увеличения количества автомобилей в данном случае служит именно расширение дорог.

Условный пользователь общественного транспорта (который в наших странах является далеким от комфортного) замечает, что его коллега, который передвигается по городу на личном автомобиле, гораздо меньше тратит времени на передвижение от дома до работы и обратно. Все это благодаря многополосным дорогам и относительно невысокой цене на автомобиль. Наш условный пользователь отправляется в банк, где получает кредитные средства на покупку недорого автомобиля. И вот, в нашем мире появился еще один водитель, а на дороге появилось транспортное средство, которое заняло