Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет інформаційно-аналітичних технологій та менеджменту
(повна назва)

Кафедра прикладної математики
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти другий (магістерський)

Генерація сферичних порожнинних структур в обмежених 3D областях
(тема)

Виконав:
здобувач _2_ року навчання, групи _ПМм-23-1 _
Рожко Д.В
(прізвище, ініціали)
Спеціальність
113 Прикладна математика
(код і повна назва спеціальності)
Тип програми _освітньо-професійна
(освітньо-професійна або освітньо-наукова)
Освітня програма
Прикладна математика
(повна назва освітньої програми)
Керівник проф. Яськов Г.М.
(посада, прізвище, ініціали)

Допускається до захисту

Зав. кафедри ПМ _____ Сидоров М.В.
(підпис) (прізвище, ініціали)

2025 р.

Харківський національний університет радіоелектроніки

Факультет _інформаційно-аналітичних технологій та менеджменту
Кафедра _прикладної математики
Рівень вищої освіти _другий (магістерський)
Спеціальність _113 Прикладна математика
(код і повна назва)
Тип програми _освітньо-професійна
(освітньо-професійна або освітньо-наукова)
Освітня програма _Прикладна математика
(повна назва)

ЗАТВЕРДЖУЮ:

Зав. кафедри ПМ _____
(підпис)
“ _25_ ” _листопада_ 2024 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

здобувачеві Рожку Денису Валерійовичу
(прізвище, ім’я, по батькові)

1. Тема роботи Генерація сферичних порожнинних структур в обмежених 3D областях

затверджена наказом по університету від 22 листопада 2024 р. № 1223 Ст

2. Термін подання здобувачем роботи до екзаменаційної комісії 6 січня 2025 р.

3. Вихідні дані до роботи математична модель задачі розміщення однакових куль у 3D контейнері обмеженому заданими площинами

4. Перелік питань, що потрібно опрацювати в роботі
1. Аналіз предметної області
2. Вибір і обґрунтування методу розв’язання
3. Програмна реалізація
4. Результати обчислювального експерименту
5. Аналіз можливих застосувань

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій

1. Актуальність теми роботи
2. Постановка задачі
3. Аналіз предметної області
4. Метод чисельного аналізу
5. Результати обчислювального експерименту

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Терміни виконання етапів роботи	Примітка
1	Підбір та вивчення технічної літератури за темою роботи	25 листопада – 1 грудня 2024 р.	виконано
2	Вибір та обґрунтування методу	2 – 8 грудня 2024 р.	виконано
3	Розробка алгоритму і програми	9 – 22 грудня 2023 р.	виконано
4	Проведення аналітичних досліджень та розрахунків	23 – 29 грудня 2024 р.	виконано
5	Робота над текстом пояснювальної записки	30 грудня 2024 р. – 9 січня 2025 р.	виконано
6	Представлення роботи на рецензію в ЕК	10 січня 2025 р.	виконано

Дата видачі завдання _25_ листопада 2024 р.

Здобувач _____
(підпис)

Керівник роботи _____ проф. Яськов Г.М
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ

Пояснювальна записка: 74 с., 3 табл., 10 рис., 1 дод., 21 джерело.

АДИТИВНЕ ВИРОБНИЦТВО, БАГАТОГРАННА ОБЛАСТЬ, ГЕОМЕТРИЧНА ОПТИМІЗАЦІЯ, КОНТЕЙНЕР, КУЛЯ, МЕТОД МУЛЬТИСТАРТУ, ОБЧИСЛЮВАЛЬНА ГЕОМЕТРІЯ, ПОРОЖНИННІ СТРУКТУРИ, РОЗМІЩЕННЯ КУЛЬ, ФУНКЦІЯ ЦІЛІ.

Об’єкт дослідження – задача розміщення куль у заданій тривимірній області.

Мета роботи – розробка ефективного методу розміщення куль для створення порожнинних структур із заданими геометричними параметрами.

Методи дослідження – метод випадкового розміщення для початкового заповнення, метод мультистарту для глобальної оптимізації, метод локальної оптимізації, чисельне моделювання.

У даній роботі аналізується, проектується та програмно реалізується алгоритм розміщення порожнин у формі однакових куль у опуклому відомому тримірному контейнері, обмеженому площинами. Проведено системний аналіз задач розміщення куль, існуючих робот на дану тему та застосувань, де дана задача є актуальною. Проведено побудову формальної постановки задачі. Проаналізовано загальний алгоритм розв’язання задачі та кожен з його кроків окремо у контексті поставленої задачі. Сформовано та спроєктовано алгоритм вирішення задачі. Проаналізовані функціональні та нефункціональні вимоги до програмної реалізації. Реалізовано та протестовано програмну реалізацію.

Створена програмна реалізація алгоритму реалізована у загальному вигляді, тому може бути використана у будь-якій сфері, де задача розміщення куль у поставленому вигляді є актуальною. У першу чергу, це є важливим при оптимізації 3D моделі виробу в адитивному виробництві для економії матеріалу та часу виробництва.

ABSTRACT

Introductory note: 74 pages, 3 tables, 10 figures, 1 appendix, 21 sources.

ADDITIVE MANUFACTURING, BALL, CONTAINER, GEOMETRIC OPTIMISATION, MULTISTART METHOD, OBJECTIVE FUNCTION, POROUS STRUCTURES, SPHERE PLACEMENT, THREE-DIMENSIONAL DOMAIN.

The object of research is the problem of placing spheres in a given three-dimensional area.

The purpose is to develop an effective method of ball placement to create void structures with specified geometric parameters.

The research methods are random placement method for initial filling, multi-start method for global optimization, local optimization method, numerical modelling.

In this paper, designing and implementation of an algorithm for placing void structures in the form of identical spheres in a convex known three-dimensional container bounded by known planes are analyzed. A systematic analysis of the problems of placing hyperballs, existing works on this topic and areas where this problem is relevant is carried out. A formal formulation of the problem is constructed. The general algorithm for solving the problem and each of its steps is analyzed separately in the context of the problem. The algorithm for solving the problem is formed and designed. Functional and non-functional requirements for software implementation are analyzed. The software implementation of the algorithm is implemented and tested.

The created software implementation of the algorithm is applied in a general form, so it can be used in any field where the task of placing spheres stated is relevant. First of all, this is important when optimizing a 3D model of a product in additive manufacturing to save costs.

ЗМІСТ

ВСТУП

Актуальність теми. Актуальність теми полягає в необхідності оптимізації використання матеріалів і часу виробництва у процесах адитивного виробництва. Створення порожнинних структур у вигляді куль із заданими властивостями дозволяє знизити масу виробів, зменшити витрати матеріалів та необхідний час при адитивному виробництві. Тому необхідно розробити і автоматизувати алгоритм розміщення куль у заданому тримірному контейнері.

Мета і завдання кваліфікаційної роботи. Метою кваліфікаційної роботи є розробка методів для генерації порожнинних структур у формі куль в обмежених 3D областях та їх автоматизація.

Для досягнення поставленої мети під час виконання кваліфікаційної роботи необхідно виконати наступні завдання:

– провести огляд і аналіз сучасного стану задачі генерації порожнинних структур у тривимірних обмежених областях;

– дослідити існуючі підходи та алгоритми для генерації структур у формі куль в обмежених об’ємах, зокрема методи розміщення куль у 3D просторі;

– розробити модель розміщення порожнинних структур у формі куль у заданій області з урахуванням обмежень на мінімальну відстань між кулями та урахуванням зони заборони;

– реалізувати алгоритми генерації порожнинних структур і провести чисельні експерименти для тестових задач;

– оцінити ефективність розробленої методики та перевірити, наскільки генерація порожнин відповідає вимогам економії матеріалів і скорочення часу виробництва.

Об’єктом дослідження є процес генерації порожнинних структур у обмежених 3D областях.

Предметом дослідження є методи і алгоритми генерації порожнинних структур у формі куль у обмежених 3D областях.

Методи дослідження. У роботі використовуються методи розміщення

куль у просторі, комбіновані підходи локальної і глобальної оптимізації, а також чисельні методи для проведення обчислювальних експериментів та аналізу ефективності запропонованих алгоритмів.

Публікації. Результати, отримані у роботі, було представлено на 3-му Міжнародному молодіжному форумі «НАВЧАННЯ І ВИКЛАДАННЯ: у світі після війни» (м. Харків, 08 листопада 2024 р.) [1].

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ТА ПОСТАНОВКА ЗАДАЧ ДОСЛІДЖЕННЯ

1.1 Типи задач розміщення куль різної вимірності

У задачах розміщення куль виділяють декілька типів задач, які відрізняються за низкою критеріїв. Одним із основних параметрів є розмірність задачі, яка визначає кількість ступенів свободи і просторових параметрів, необхідних для опису розташування куль. Такі задачі можуть бути одно-, дво-, тривимірними, а також багатовимірними. Зокрема, у випадку багатовимірних задач розміщення куль, що мають більше ніж три виміри, положення кожної кулі потребує визначення d координат її центру, де d є розмірністю простору. Це значно ускладнює процес моделювання та оптимізації, оскільки для такого завдання потрібно обробляти більшу кількість даних для кожної кулі.

Наступним важливим критерієм є подібність куль. Існують три основні варіанти: ідентичні, слабко неоднорідні та сильно неоднорідні кулі. Ідентичні об’єкти є конгруентними, тобто всі елементи мають однакову форму і розмір. Це дозволяє спрощувати моделювання та обчислення, оскільки до всіх об’єктів застосовуються однакові обмеження та властивості. У випадку слабко неоднорідних куль вони поділяються на кілька класів, кожен із яких складається з об’єктів однакової форми та розміру. Це дещо ускладнює задачу, проте дозволяє моделювати системи, де об’єкти незначно відрізняються між собою. Сильно неоднорідні кулі є найскладнішими для моделювання, оскільки кожен елемент набору має унікальні розміри та форму. У таких задачах розробка індивідуальних обмежень для кожного об’єкта є важливим аспектом побудови ефективної моделі.

Крім цього, задачі розміщення куль можна розрізняти за метою оптимізації. Залежно від цього критерію задачі поділяються на задачі мінімізації та максимізації функції цілі. Максимізація передбачає вибір такого розміщення куль, яке дозволяє досягти максимального значення функції цілі, наприклад, щільно-

сті заповнення або максимальної кількості розміщених об’єктів у заданому об’ємі. Це стосується випадків, коли потрібно розмістити якомога більше об’єктів у фіксованому просторі, зводячи до мінімуму невикористаний об'єм. Мінімізація ж спрямована на знаходження мінімального значення функції цілі, що може включати обсяг або площу незаповненої частини контейнера. Наприклад, задача може полягати у мінімізації витрат на транспортування чи зберігання, або в мінімізації площі для певної кількості куль у заданому просторі.

Додаткові обмеження також відіграють значну роль у класифікації задач розміщення куль. Одним із таких обмежень є заборонені зони — частини контейнера, в яких розміщення куль є неможливим або небажаним. Це може стосуватися областей, зайнятих іншим обладнанням, або частин контейнера, що мають спеціальний статус. Наприклад, у задачах промислового пакування певні області можуть бути зарезервовані для проходів або технічного обслуговування, що унеможливлює розміщення об'єктів у цих зонах. Іншим важливим обмеженням є мінімальна відстань між кулями, яка може бути необхідною для забезпечення безпечної взаємодії між об’єктами або для уникнення їхнього накладання. Таке обмеження часто використовують для систем, де об’єкти мають фізичний розмір і не можуть проникати один в одного, що забезпечує належну відстань для запобігання можливим перешкодам у функціонуванні системи.

З урахуванням зазначених критеріїв існує кілька основних типів задач для розміщення куль у просторі. Задача розміщення рівних об'єктів, або IIPP (Identical Item Packing Problem), полягає у розміщенні максимальної кількості ідентичних куль у даному контейнері. Це класичний тип задачі, де основною метою є досягнення максимальної щільності заповнення. У задачі розміщення (Placement Problem) слабко неоднорідні скулі потрібно розмістити в контейнері з урахуванням оптимізації певної функції цілі, наприклад, мінімізації залишкової площі чи максимізації об’єму використаного простору. Задача про рюкзак (Knapsack Problem, KP) застосовується для сильно неоднорідних об’єктів, де обирається оптимальна підмножина об’єктів, що відповідає розмірам та обмеженням контейнера. Задача із змінними розмірами контейнера, або ODP (Open

Dimension Problem), передбачає розміщення усіх об'єктів у одному або декількох контейнерах, при цьому одна з характеристик контейнера є змінною, і мета полягає у знаходженні її мінімального значення.

Таким чином, класифікація задач розміщення куль за критеріями вимірності, подібності, функції цілі та обмежень дозволяє створювати моделі, що відповідають реальним потребам. Такий підхід забезпечує раціональне використання простору, врахування специфічних вимог безпеки між об'єктами та адаптацію моделі до унікальних особливостей кожного завдання.

1.2 Задача розміщення кругів

Проблема розміщення кругів у замкнутому просторі є складним завданням з багатими теоретичними та практичними застосуваннями, яке досліджувалось протягом багатьох десятиліть. Перша робота [2] була випущена у 1942 році. Далі дана тема поступово набувала розвитку і мала місце в усе більшій кількості публікацій таких як [3], у якій було більше детально розібраний метод вирішення або [4] та [5], де були розглянуті варіанти розміщення кіл з додатковими обмеженнями. Тим не менше, проблема розміщення кругів у контейнерах різних форм досі залишається актуальною Розміщення кругів у контейнерах таких форм, як круг, прямокутник або багатокутник, виникає в різноманітних сферах. Від геометрії до промислового застосування, ця задача вимагає пошуку ефективного способу розташування кругів у обмеженій площині, що дозволяє максимально використати доступний простір, забезпечуючи відсутність перекриття об’єктів. Класичними прикладами можуть бути пакування кабелів у кабель, розташування контейнерів з відходами у сховищі, моделювання біологічних частинок у клітині та інші.

Розглянемо детальніше декілька прикладів. Одним з найбільш поширених застосувань задачі розміщення кругів є упаковка проводів різного діаметру в єдиний кабель або трубопровід. У такій задачі необхідно розмістити круглі

проводи так, щоб використати весь доступний простір оболонки, мінімізуючи порожні зони. Це дозволяє зменшити загальний діаметр кабелю, що має вирішальне значення в авіаційній, автомобільній та космічній індустріях, де важлива мінімізація ваги та обсягу конструкцій. Припустимо, є кілька дротів різних діаметрів, які потрібно вкласти у кругову оболонку певного діаметра. У такому випадку застосовуються методи оптимізації, такі як метод найбільшого залишкового об'єму, що допомагає мінімізувати порожні зони шляхом найщільнішого розташування проводів. Це забезпечує мінімізацію діаметра оболонки, що дозволяє зменшити витрати на матеріали та підвищити ефективність передачі сигналу.

Іншим прикладом є оптимізація пакування круглих об’єктів, таких як пляшки або банки, у контейнер фіксованого розміру. Це завдання зустрічається в харчовій промисловості та логістиці, коли потрібно транспортувати або зберігати велику кількість круглих об'єктів. Для того щоб мінімізувати витрати на упаковку та транспортування, необхідно вмістити максимальну кількість таких об’єктів у контейнер, уникаючи пустот. Задача ускладнюється, якщо об'єкти мають різні діаметри, що вимагає розрахунків для мінімізації порожніх зон та забезпечення стійкості упаковки під час транспортування. Використання евристичного алгоритму на основі імітаційного відпалу дозволяє знайти оптимальну конфігурацію розміщення об'єктів із урахуванням обмежень контейнера та діаметра кожного об’єкта. Таким чином, правильне пакування зменшує обсяг порожнього простору, а отже, і витрати на упаковку і транспортування.

Інший цікавий приклад – проєктування супутникових модулів, де кожен компонент має круглу форму. У космічній інженерії кожен грам ваги та кожен сантиметр площі є критично важливими. При проєктуванні супутникових модулів інженери зіштовхуються із завданням розміщення круглих модулів, таких як антени, енергетичні блоки або сенсори, у обмеженому просторі супутника. У цьому випадку необхідно забезпечити максимальне використання об'єму при збереженні доступності кожного компонента. Використання математичних методів оптимізації дозволяє знайти конфігурацію з максимальною щільністю, що

забезпечує надійність роботи системи.

Ще одним важливим прикладом є розміщення контейнерів з радіоактивними відходами. Радіоактивні відходи, зокрема, зберігаються в круглих контейнерах, які потребують розміщення у захищеному сховищі. Важливо щільно розмістити контейнери у визначеній області, щоб мінімізувати площу захоронення. Окрім цього, кожен контейнер має знаходитися на певній безпечній відстані від інших, щоб забезпечити ізоляцію та мінімізувати ризик поширення радіації. Використання методів оптимізації розташування дозволяє не тільки зменшити площу сховища, а й знизити витрати на будівництво та забезпечити безпечну експлуатацію такого об'єкта.

Задачі розміщення кругів мають також своє застосування у біології, зокрема при моделюванні поведінки вірусних частинок у клітині. Частинки вірусів можна змоделювати як круглі об'єкти, розташовані у замкнутому просторі. Дослідники можуть вивчати взаємодії між частинками, розподіл у межах певної області та їхній вплив на інші клітини. Такий підхід дозволяє отримувати важливі дані про процеси розмноження вірусів, що стає актуальним для розробки вакцин.

Також приклади задач розміщення можна знайти у географічному моделюванні. Розміщення кіл тут може представляти природні або інфраструктурні об'єкти на карті. Завданням є розмістити об'єкти так, щоб вони займали мінімальну площу, зберігаючи певні пропорції та відстані. Це використовується для моделювання міст, транспортних систем або критичної інфраструктури, дозволяючи забезпечити раціональне використання площі, економію ресурсів і безпеку об’єктів.

Ці приклади демонструють, як важливою є задача оптимального розміщення кіл у замкнутому просторі для досягнення високої ефективності в різних галузях. Історія дослідження проблеми починається ще з часів Йоганна Кеплера, який вивчав щільність укладання гарматних ядр. Пізніше гексагональна структура була математично підтверджена як оптимальна. Сучасні дослідження зосереджуються на розробці алгоритмів, таких як імітаційний відпал, генетичні

алгоритми та локальний променевий пошук. Використання математичних моделей, алгоритмів оптимізації та обчислювальних методів дозволяє знаходити розв’язки для задач різної складності. Задачі оптимального розміщення кіл залишаються актуальними для наукових дисциплін та сприяють розвитку обчислювальної оптимізації та математичного моделювання.

1.3 Задача розміщення куль

Приклади задач розміщення куль у тривимірному просторі мають велике практичне значення в різних наукових та інженерних галузях. Така задача виникає у випадках, коли необхідно розмістити кульові об'єкти в обмеженому об'ємі, максимізуючи щільність їхнього розташування або мінімізуючи порожній простір між ними. У багатьох випадках кульові об'єкти є рівними (тобто мають однаковий діаметр), однак існують і задачі для розміщення нерівних (полідисперсних) куль, що створює додаткові ускладнення для обчислень та моделювання.

Один з перших і найвідоміших прикладів розміщення куль використовується для моделювання рідин. У 1959 році Бернал у [6] уперше запропонував модель випадкового щільного розміщення рівних куль для симуляції рідинних станів. Згідно з цією моделлю, розташування частинок у рідині можна наближено описати як випадкове та щільне розміщення рівних куль. Дослідження показали, що властивості таких матеріалів значною мірою залежать від щільності й конфігурації частинок. Ця модель надала можливість вивчати різні стани матеріалів, включно з рідинами, кристалами та гранульованими речовинами.

Розміщення рівних куль у тривимірних контейнерах часто використовується для моделювання поведінки пористих матеріалів, зокрема наноструктурованих матриць, які заповнюються атомами або молекулами як це описано у [7]. У хімічній інженерії цей підхід дає змогу описувати канали в матеріалах з високою пористістю, заповнених речовиною у вигляді молекул, які набувають

конфігурації, аналогічної щільно упакованим кулям. Ефективне розміщення куль у контейнері дозволяє моделювати структуру та властивості таких матеріалів, що є особливо важливим для передбачення процесів транспортування в газорідких середовищах або вивчення речовин на рівні критичної щільності.

Ще одним значним прикладом є каталітичні реактори та теплообмінники, де кульові частинки виконують роль каталітичного середовища для хімічних реакцій, що описано у [8]. Частинки повинні бути розташовані таким чином, щоб забезпечувати оптимальну площу для реакцій, а також максимізувати щільність розміщення для збільшення ефективності процесу. Задачі такого типу також актуальні для ядерних реакторів, де щільне розміщення паливних елементів дозволяє підвищити енерговидобуток. Завдяки моделям розміщення рівних куль ([9, 10]) можна досягти стабільного розташування частинок, що забезпечує належну циркуляцію рідин чи газів та сприяє підвищенню ефективності реактора.

У будівництві та інженерії використовуються моделі розміщення куль для дослідження структурних властивостей бетону та інших будівельних матеріалів. Оскільки бетон має численні пори, він піддається корозії, особливо під впливом хлоридів, які можуть проникати через пори та спричиняти руйнування арматури. Дослідники вивчають заповнення пор в бетоні наночастинками для підвищення його міцності та стійкості. Використання дрібних і великих частинок у порах бетону дозволяє блокувати доступ шкідливих елементів і підвищує загальну міцність матеріалу. У цьому випадку задача розміщення куль різних розмірів дозволяє знайти оптимальне співвідношення для максимального заповнення пор, як це описано у [11, 12].

Розміщення нерівних куль у тривимірному просторі має важливі застосування у медицині, особливо у плануванні радіохірургії пухлин, як це описано у [13, 14]. Задачі оптимального розміщення куль різного розміру дозволяють створювати точні моделі, які допомагають спрямовувати радіохвилі на пухлину з мінімальним впливом на здорові тканини. У цьому контексті кулі виконують роль зон опромінення, які потрібно розташувати таким чином, щоб досягти

найвищого рівня покриття пухлини. Розв'язання таких задач оптимізації вимагає використання складних методів, зокрема методів глобальної оптимізації та комбінації квадратичного програмування з лінійними обмеженнями.

Ще одним практичним застосуванням задачі розміщення куль є моделювання процесів у гірничорудній, нафтовій та торф'яній промисловості. У цих галузях об'єкти дослідження представлені у вигляді колоїдних систем, які можна апроксимувати як розміщення куль різного розміру. Ефективне розміщення куль дозволяє зменшити порожнечі між частинками, збільшуючи щільність заповнення та оптимізуючи процеси видобутку чи обробки матеріалів. Цей підхід використовується при моделюванні процесів, таких як дроблення, флотація та збагачення руд.

Задачі розміщення куль також зустрічаються у моделюванні гранульованих матеріалів та сипучих середовищ, як у [15]. Вивчення щільного випадкового розміщення куль дозволяє створювати моделі поведінки сипучих матеріалів під впливом зовнішніх факторів, таких як вібрація або зміна температури. Ці задачі є важливими для таких галузей, як харчова промисловість, фармацевтика та сільське господарство, де сипучі матеріали необхідно зберігати або транспортувати у певному обмеженому просторі. Моделі розміщення куль допомагають передбачити поведінку матеріалу, оптимізувати пакування та зменшити ризики пошкодження продукту під час транспортування.

Таким чином, задача розміщення куль у тривимірному просторі є актуальною у багатьох сферах та вимагає використання математичних моделей, таких як методи молекулярної динаміки, імітаційного відпалу, алгоритмів послідовного розміщення або методів глобальної оптимізації ([16]). Завдяки розробці та вдосконаленню цих моделей і методів стало можливим вирішення складних задач розміщення в обмежених об’ємах з урахуванням усіх обмежень, що дозволяє підвищити ефективність систем, у яких вони застосовуються.

1.4 Проблеми, що виникають у адитивному виробництві

В адитивному виробництві (3D друку) економія матеріалів та часу на виробництво є критичними факторами, особливо коли потрібно виготовляти складні структури або об'єкти великих розмірів. В умовах, коли традиційне виготовлення суцільних об’єктів є надзвичайно ресурсомістким і призводить до збільшення ваги та виробничих витрат, виникає необхідність зробити об'єкти порожнинними, тобто зі структурами, які включають порожнини та внутрішні порожнини. Порівняно з традиційними методами виробництва, адитивне виготовлення дозволяє створювати такі порожнинні структури з мінімальними обмеженнями, що значно знижує використання матеріалів і час, необхідний на їх друк.

Порожнинна структура забезпечує зниження загальної маси виробу та матеріальних витрат. Окрім цього, за допомогою порожнин можна покращити теплообмінні характеристики виробів, що особливо важливо для деталей, які використовуються в умовах високих температур або під час охолодження. Ця властивість порожнинних конструкцій робить їх незамінними в різноманітних галузях, таких як автомобільна, аерокосмічна та медична промисловості, де важливо забезпечити як легкість конструкції, так і її надійність та функціональність. З іншого боку, при надмірному використанні матеріалів або друку суцільних структур збільшуються час і собівартість виробництва, що є невиправданими в умовах сучасного виробництва, орієнтованого на ефективність.

Для вирішення цієї проблеми під час 3D друку застосовуються різні підходи до створення порожнинних структур, які, наприклад, були приведені у [17]. Одним з них є використання структур із великою кількістю порожнин у вигляді куль, що розташовані у 3D-просторі за певним принципом. Задача оптимального розміщення куль дозволяє визначити найкращі варіанти розташування цих порожнин таким чином, щоб вони з одного боку забезпечували необхідну легкість і порожнинність, а з іншого – зберігали механічну стійкість і міцність конструкції. Цей підхід дозволяє уникнути надмірного використання ма-

теріалів та одночасно підвищує характеристики міцності об'єктів за рахунок рівномірного розподілу навантаження по усьому об'єму.

Розміщення куль у 3D просторі також надає можливість створення різних рівнів порожнинності залежно від вимог до функціональності виробу. Наприклад, для деталей, які мають витримувати високі навантаження, можна залишити більш густу структуру, тоді як в областях з меншими вимогами до міцності можна застосувати більш порожнинну структуру, що забезпечує додаткову економію матеріалів. Використання подібних структур у 3D друці дозволяє досягти компромісу між вагою, витратою матеріалів і механічною стійкістю виробу, забезпечуючи оптимальні показники для конкретних виробничих потреб.

Також є очевидним, що у деяких випадках можуть накладатися додаткові обмеження такі як введення зон, де порожнин має не бути взагалі, або зони де порожнини мають бути не сферичними. Але є одна додаткова умова, що є необхідною за будь-якого розміщення порожнин у формі куль під час 3D друку – це умова мінімальної відстані між кулями. Це необхідно, оскільки за відсутності цієї умови – кількість будівельного матеріалу конструкції буде занадто малою, щоб забезпечити її стійкість.

1.5 Формальна та змістовна постановка задачі

У контексті 3D-адитивного виробництва розглядається задача генерації порожнинних структур для зменшення ваги деталі, описаної опуклим багатогранником. Необхідно розмістити в межах цього багатогранника максимальну кількість куль однакового радіуса r, дотримуючись таких умов: відстань між кулями повинна бути не меншою за ξ (певне технологічне обмеження), жодна куля не повинна виходити за межі багатогранника, а кількість куль має бути максимальною для мінімізації ваги виробу. Задачу можна сформулювати як оптимізаційну задачу розміщення куль з урахуванням геометричних та технологічних обмежень.

Нехай задані рівні кулі $S_i = S$ з радіусом r, $i_i \in I_n$, де n – достатньо велике натуральне число. Куля S_i є моделлю порожнини у виробі адитивного виробництва. Розміщення куль S_i у просторі $\mathbb{R}^3$ визначають координатами їх центрів: $u = (u_1, u_2, ..., u_n)$, $u_i = (x_i, y_i, z_i)$, $i_i \in I_n$. Нехай також задана область розміщення C, що є перетином напівпросторів C_j, які задаються площинами P_j, заданих рівняннями $A_j x + B_j y + C_j z + D_j = 0$, $j \in J_m$, де m – кількість напівплощин:

$$C = \bigcap_{j=1}^{m} C_j,\ C_j = \{(x, y, z) \in \mathbb{R}^3 : A_j x + B_j y + C_j z + D_j \geq 0\}.$$

Задача формулюється наступним чином: Необхідно розмістити в області C максимальну кількість куль S_i без взаємних накладень з мінімальною відстанню між кулями, не меншою за ξ, та визначити їх координати центрів u_i, $i_i \in I_n$.

Математична модель задачі має такий вигляд:

$$N^* = \max_{u \in G} \sum_{i_i \in I_n} \Psi_i(u_i),$$

де $\Psi_i(u_i) = \begin{cases} 1, \text{якщо } \Phi_i(u_i) \geq 0 \\ 0, \text{в іншому випадку}, i_i \in I_n \end{cases}$;

$G = \left\{ u \in \mathbb{R}^3 : \Phi_{ij}(u_i, u_j) \geq 0, i < j \in I_n \right\}$;

$\Phi_{ij}(u_i, u_j) = (x_i - x_j)^2 + (y_i - y_j)^2 + (z_i - z_j)^2 - (2r + \xi)^2, i < j \in I_n$;

$\Phi_i(u_i)$ – Ф-функція кулі S_i та області С;

$$\Phi_i(u_i) = \min_{j \in J_m} \left(\frac{\left| A_j x_i + B_j y_i + C_j z_i + D_j \right|}{\sqrt{A_j^2 + B_j^2 + C_j^2}} - r \right).$$

Ця модель задовольняє задані умови і обмеження розглянутої задачі розміщення однакових куль S_i у заданій області C.

Отриманий виріб можна описати як

$$P = C \setminus \bigcup_{k=1}^{N^*} \text{int}(S_k(u_k^*)).$$

1.6 Постановка задач дослідження

Метою кваліфікаційної роботи є розробка та впровадження методів оптимального розміщення куль у заданій опуклій багатогранній області для забезпечення максимального використання простору та досягнення заданих характеристик конструкції.

Для досягнення поставленої мети потрібно виконати наступні етапи:

– провести системний аналіз задачі розміщення куль та визначити основні вимоги до заповнення опуклої області;

– ознайомитися із існуючими підходами та алгоритмами розв’язання задач оптимального заповнення 3D області кулями із заданими параметрами;

– визначити потреби до моделі розміщення куль у заданій області, включаючи вимоги до щільності заповнення, мінімальної відстані між кулями та обмежень на допустимі зони розміщення;

– провести дослідження та сформулювати вимоги для методів розміщення куль, що задовольняють визначені критерії та обмеження;

– описати розробку алгоритмічного підходу для вирішення задачі розміщення куль у опуклій багатогранній області відповідно до визначених вимог;

– реалізувати описаний алгоритм програмно;

– провести обчислювальний експеримент на тестових задачах та визначити ефективність реалізованого алгоритму.

2 ВИБІР ТА ОБҐРУНТУВАННЯ МЕТОДУ РОЗВ’ЯЗАННЯ

2.1 Загальний огляд методології вирішення задачі

Загальний алгоритм вирішення задачі розміщення куль будь-якої вимірності у заданому просторі зводиться до виконання трьох наступних кроків:

– побудова допустимих початкових точок – знаходження набору шуканих значень, який відповідає заданим обмеженням задачі і слугує початковою точкою для пошуку локального та глобального екстремуму функції цілі;

– локальна оптимізація – крок, на якому відбувається оптимізація шуканих параметрів таким чином, щоб функція цілі досягала локального екстремуму;

– глобальна оптимізація – на цьому кроці відбувається пошук наближення до глобального екстремуму функції цілі серед локальних екстремумів для знаходження найбільш вдалих значень параметрів, які необхідно знайти за умовами задачі.

Таким чином, побудова конкретної стратегії розв’язання поставленої задачі полягає у створенні алгоритму виконання даних трьох кроків. Для цього, у свою чергу, необхідно обрати відповідні методи побудови допустимих початкових значень, локальної оптимізації та глобальної оптимізації. Для того, щоб обрати найбільш ефективний метод нам необхідно:

– проаналізувати від яких особливостей, параметрів чи обмежень поставленої задачі може залежати вибір методу проходження кожного кроку розв’язання задачі;

– розглянути можливі методи для кожного з кроків, проаналізувати їхні сильні та слабкі сторони у рамках поставленої задачі;

– обрати для кожного кроку метод або набір декількох методів, які найбільш доцільно використовувати для поставленої задачі.

Результатом виконання цих кроків є чітке визначення стратегії розв’язання задачі.

2.2 Огляд особливостей поставленої задачі

Розглянемо особливості нашої задачі, що найбільше впливають на процес обрання стратегії розв’язку задачі:

– функція цілі визначає критерій якості. У задачах розміщення куль виділяють задачу розміщення заданих куль у контейнері найменшого розміру та задачу розміщення максимальної кількості (або об’єму) куль у заданому контейнері;

– вимірність простору контейнера та куль;

– радіуси куль;

– характеристики і просторова форма контейнера;

– наявність додаткових обмежень.

Для задачі, яка розглядається в роботі:

– функція цілі – максимальна кількість розміщених куль у заданому багатогранному опуклому контейнері;

– вимірність простору контейнера та куль – 3D;

– радіуси куль – усі кулі мають однаковий радіус, що є заздалегідь відомим;

– характеристики і форма контейнеру –контейнер є опуклим багатогранником;

– наявність додаткових обмежень – обмеження на мінімально допустиму відстань між кулями.

Тепер, після того, як були визначені ключові параметри задачі, що впливають на процес побудови стратегії розв’язку задачі, можна розпочати аналіз можливих методів і вибір найбільш прийнятних для розв’язання задачі.

2.3 Огляд методів побудови стартового розміщення куль

Розглянемо основні методи побудови стартового розміщення куль, що зазвичай використовуються у вирішенні задач подібних до нашої.

Метод випадкових розміщень зазвичай використовується на початкових етапах або для швидкого отримання початкових точок у випадках, коли необхідно отримати базову конфігурацію для подальших обчислень. Цей метод обирає випадкові значення координат центрів куль у просторі та перевіряє їх на відповідність умовам допустимості. Його основна перевага – простота та швидкість, що дозволяє оперативно створити множину точок для наступної оптимізації. Проте, через випадковий характер вибору, результати можуть бути далекими від оптимального розміщення, особливо у випадках, коли область має складні обмеження або форму. Метод випадкових розміщень добре підходить для задач з менш жорсткими вимогами до розміщення або для первинної оцінки можливих конфігурацій.

Метод ґратчастих розміщень є найефективнішим у ситуаціях, коли контейнер є опуклим і не має заборонених зон, а розміри куль суттєво менші за розміри контейнера. При такому підході центри куль розташовуються у вигляді регулярної структури, наприклад, гексагональної ґратки, що в тривимірному просторі дозволяє кожній кулі дотикатися до 12 сусідніх куль. Ґратчасте розміщення забезпечує високу щільність заповнення простору, особливо у великих контейнерах, де кожна куля може повторюватися багаторазово. Цей метод є оптимальним для випадків, коли відсутні складні обмеження і потрібно забезпечити високу рівномірність у розподілі об'єктів. Недоліком є обмежена гнучкість для контейнерів складної форми або при наявності зон заборони, що робить його менш ефективним за наявності таких обмежень.

Жадібний алгоритм або метод оптимізації за групами змінних добре підходить для отримання допустимих початкових точок у випадках, коли важливо швидко знайти перспективне рішення. Цей алгоритм є модифікацією методів оптимізації, що працює з окремими групами змінних, і дозволяє розбивати задачу на послідовні етапи. Кожен з етапів полягає у розв’язанні задачі для частини змінних, що дозволяє уникати глибоких локальних мінімумів і досягати близьких до глобальних рішень.

Жадібний алгоритм використовується для задач, де важливий прийнятний час обчислень, наприклад, у реальному часі, та для початкових наближень до локальних і глобальних мінімумів із використанням мультистартових підходів. Завдяки здатності створювати наближені рішення, цей метод особливо корисний для складних контейнерів та розміщень з обмеженнями, де інші методи можуть виявитися недостатньо ефективними.

Метод оптимізації нев’язок застосовується в тих випадках, коли контейнер має заборонені зони, що накладає складні обмеження на конфігурацію куль у просторі. Цей метод формулюється як задача нелінійного програмування з обмеженнями, і для досягнення допустимого розміщення розв'язок має відповідати заданим обмеженням.

Метод побудови допустимих точок шляхом оптимізації нев’язок включає пошук координат, які не порушують обмеження, що задаються областю розміщення, та дозволяє виконати розміщення навіть у складних контейнерах зі специфічними зонами заборони. Наприклад, якщо область має сегменти, недоступні для розміщення через присутність іншого обладнання, оптимізація нев’язок допомагає обійти такі області та розмістити кулі в допустимих точках. Цей метод забезпечує високу точність отриманих рішень та є ефективним для задач з жорсткими вимогами до допустимості розміщення.

2.4 Огляд методів локальної оптимізації

Розглянемо основні методи локальної оптимізації стартового розміщення куль, що зазвичай використовуються у вирішенні задач подібних до вирішуваної.

Комбінований метод послідовного квадратичного програмування, реалізований у вигляді розв’язувача SLSQP (Sequential Least Squares Programming), ефективно працює в задачах нелінійного програмування з помірною кількістю змінних (приблизно до 500) та обмежень. За його допомогою основна задача

наближається до розв’язання через послідовність задач квадратичного програмування, що дозволяє ефективно обробляти як лінійні, так і нелінійні обмеження. Метод SLSQP особливо підходить для задач середньої розмірності, оскільки забезпечує швидку збіжність завдяки використанню інформації про градієнти функції цілі та обмежень. Цей метод є універсальним і простим у реалізації, що робить його зручним для задач оптимізації, які не потребують надто високої точності, але вимагають обробки різних типів обмежень. Він добре підходить для задач, де швидкість і легкість налаштування є важливими чинниками.

Метод можливих напрямків, зі своїми модифікаціями, застосовується для задач, де кількість змінних перевищує кілька тисяч. Цей метод зводить задачу нелінійного програмування до послідовності задач лінійного програмування, визначаючи напрямок руху для покращення цільової функції та збереження допустимості розв’язків за всіма обмеженнями. Програми HOPDM та BPMPD підтримують обчислення для задач великої розмірності та можуть обробляти сильно розріджені матриці, що часто виникає при розміщенні об'єктів у просторі. Метод можливих напрямків добре підходить для задач із великою кількістю змінних і розрідженими матрицями, де важливо не тільки знайти оптимальний розв’язок, але й забезпечити швидку збіжність.

Метод оптимізації за групами змінних є різновидом покоординатного спуску та особливо ефективний для задач із дуже великою кількістю змінних (понад 10 000). У цьому методі всі змінні розділяють на групи, і на кожному кроці обирається одна група змінних, що розглядається як активна, тоді як інші залишаються сталими. Для розв’язання задачі за кожною групою змінних застосовують розв’язувач IPOPT або метод можливих напрямків. Такий підхід дає наближений локальний екстремум, який можна покращувати, змінюючи вибір груп. Метод оптимізації за групами змінних зручний для задач великої розмірності, де процес розв’язання вимагає розбиття змінних на групи для зниження обчислювального навантаження.

Метод спуску на основі аналізу множників Лагранжа використовується для задач із обернено опуклими та лінійними обмеженнями і потребує попере-

днього розрахунку множників Лагранжа. Він працює в парі з методом оптимізації за групами змінних і стратегією активного набору обмежень. Цей метод дозволяє рухатися межею області допустимих розв'язків, поступово послаблюючи активні обмеження. Такий підхід продовжується доти, доки всі множники Лагранжа не стануть додатними, що свідчить про досягнення екстремуму. Метод спуску на основі множників Лагранжа добре підходить для задач з обмеженнями, які потрібно враховувати в процесі оптимізації, оскільки дає змогу ефективно рухатися в межах допустимих рішень, не виходячи за межі області розв'язків.

2.5 Огляд методів глобальної оптимізації

Розглянемо основні методи глобальної оптимізації стартового розміщення куль, що зазвичай використовуються у вирішенні задач подібних до вирішуваної.

Метод гілок та меж є одним з класичних методів, який базується на розбитті множини допустимих розв'язків на підмножини меншого розміру, що утворюють дерево розв'язків. На кожному етапі неперспективні підмножини відтинаються, що дозволяє суттєво скоротити кількість варіантів порівняно з повним перебором усіх можливих рішень. Цей метод підходить для задач з неперервними змінними, а також для задач типу KP (задача про рюкзак), де змінні мають дискретні значення. Він зручний у випадках, коли потрібно розглянути всі можливі варіанти з подальшим відсіюванням, особливо в задачах розміщення з простими умовами допустимості.

Метод околів, що звужуються, ефективний для задач, де важливо зосередитися на статистичних властивостях функції цілі. У цьому методі організовується спрямований перебір послідовностей куль або гілок дерева розв'язків. Для цього визначаються околи, які поступово звужуються на основі статистичної інформації, що накопичується під час виконання алгоритму. Якщо функція

цілі не покращується, радіус околу зменшується, що дозволяє зосередитися на більш перспективних розв’язках. Цей метод доцільний у задачах HSOA (задача оптимального розміщення куль), а також для побудови допустимих початкових точок. Він забезпечує високу ефективність у порівнянні з випадковим пошуком, зокрема методом Монте-Карло.

Метод гомотетичних перетворень застосовується для задач IIPP (задача розміщення однакових об'єктів) і використовує змінні метричні характеристики куль. За його допомогою задача зводиться до серії задач нелінійного програмування з лінійними функціями цілі, де відбуваються гомотетичні перетворення об'єктів. Завдяки цьому, локальні екстремуми визначаються в крайніх точках області допустимих рішень, а кінцевий розв’язок наближається до глобального максимуму. Метод гомотетичних перетворень є корисним у випадках, коли потрібно знайти наближене значення глобального екстремуму з використанням лінійних перетворень, що спрощує розрахунки.

Метод спрямованого перебору локальних екстремумів, або Jump Algorithm, особливо підходить для задач ODP (задача розміщення з відкритими вимірами). Метод базується на поступовому переході між локальними екстремумами задачі, коригуючи радіуси куль таким чином, щоб вони зайняли максимальний об'єм області розміщення. Спершу радіуси змінюються, щоб знайти локальний екстремум, після чого поновлюються до початкових значень з урахуванням нового розташування. Метод особливо ефективний, якщо радіуси куль рівномірно розподілені, а при неоднорідних розмірах куль ефективність знижується. Для задач із неоднорідними розмірами краще підходить метод околів, що звужуються.

Метод мультистарту працює разом із іншими методами локальної й глобальної оптимізації, окрім методу гілок та меж. Його основна ідея – вибір найкращого з локальних екстремумів як наближення до глобального екстремуму. Метод мультистарту є ефективним у випадках, коли задачі потребують багаторазового розв’язання з різних початкових точок для досягнення кращого результату.

2.6 Вибір методів вирішення задачі

Підсумовуючи усе описане вище, для вирішення нашої задачі доцільно обрати наступні методи:

– метод випадкового розміщення для початкової генерації позицій куль. Метод випадкового розміщення є простим у реалізації та швидким, що дозволяє швидко отримати різноманітні початкові розміщення куль для подальшої оптимізації. Він особливо важливий у випадках, коли критичним є час обчислення, наприклад, у практичних задачах;

– метод SLSQP обрано для локальної оптимізації. Він дозволяє працювати з лінійними і нелінійними обмеженнями, а також його реалізація є ефективною з точки зору швидкодії при малій чи середній кількості змінних (до приблизно 500), що і є необхідним у нашій задачі;

– метод мультистарту обрано для глобальної оптимізації. Завдяки багаторазовому запуску локального оптимізатора з різних початкових точок, цей метод дозволяє уникати обмежень локальних мінімумів і наближатися до глобального екстремуму.

Також слід зазначити, що для великої кількості куль і у випадках, коли контейнер має форму, що дозволяє агрегацію куль у ґратки (наприклад, не витягнутий, а більше схожий на кулю), доцільно використовувати ґратчасті початкові розміщення. Але, у випадку, коли кількість куль є малою або середньою (до приблизно ста куль), більше підходить метод випадкового розміщення.

Поєднання цих методів забезпечує простоту, ефективність та гнучкість у вирішенні задач розміщення куль, незалежно від геометричної складності контейнера.

2.7 Реалізаційна математична модель

Перейдемо до реалізаційної математичної моделі, яку адаптовано для

майбутньої комп'ютерної реалізації.

Як було зазначено вище, для нашої задачі розміщення однакових куль у контейнері фіксованого розміру, алгоритм послідовно виконує кроки локальної оптимізації для N, $N+1$, $N+2$ тощо, де N – деяке число початкової кількості розміщуваних куль, до тих пір поки додання додаткової кулі не робить задачу їх розміщення неможливою до розв'язання. Для визначення, чи можливо розміщення заданого набору куль, використовують функцію штрафу $Penalty(n)$, $n = N,\ N+1,\ N+2, \ldots$.

Функцію штрафів визначають як розмір штрафу за порушення умов пакування куль. Ця функція набуває значення 0, якщо розміщення допустиме, або розміру штрафу – в іншому разі, коли:

– куля виходить за межі контейнера;

– куля перетинається з іншою кулею або відстань між ними менша за задану мінімально допустиму відстань між кулями.

Таким чином, функцію штрафів можна представити як суму штрафів за вихід куль за межі контейнера $Penalty_{bound}$ та штрафів за перетин куль $Penalty_{\text{int}er}$. Далі слід перейти до формального визначення функції штрафів.

Нехай задані рівні кулі $S_i = S$ з радіусом r, $i \in I_n$, де n – задане натуральне число. Куля S_i є моделлю порожнини у виробі адитивного виробництва. Розміщення кулі S_i у просторі $\mathbb{R}^3$ визначається координатами їх центрів: $u = (u_1, u_2, u_3 ... u_n)$, $u_i = (x_i, y_i, z_i)$, $i_i \in I_n$. Нехай також задана область розміщення C, що є перетином напівпросторів C_j, які задаються площинами P_j, описуваними рівняннями $A_j x + B_j y + C_j z + D_j = 0$, $j \in J_m$:

$$C = \bigcap_{j=1}^{m} C_j,\ C_j = \{(x, y, z) \in \mathbb{R}^3 : A_j x + B_j y + C_j z + D_j > 0\},$$

де m – кількість напівплощин.

При цьому відстань між кулями не може бути меншою за ξ.

Таким чином, функція штрафів має наступний вигляд:

$$Penalty(n) = Penalty_{bound} + Penalty_{inter},$$

де $Penalty_{bound} = \sum_{i=1}^{n}\sum_{j=1}^{m} \max(-(\frac{A_j x_i + B_j y_i + C_j z_i + D_j}{\sqrt{A_j^2 + B_j^2 + C_j^2}} - r), 0)$;

$$Penalty_{inter} = \sum_{i=1}^{n}\sum_{j=i+1}^{n} \max((2r+\xi)^2 - (x_i - x_j)^2 + (y_i - y_j)^2 + (y_i - y_j)^2, 0).$$

Для фіксованої кількості куль n задача полягає у пошуку таких значень $u_i = (x_i, y_i, z_i)$, $i \in I_n$, за яких функція штрафів набуває найменшого значення:

$$Penalty(n) \to \min.$$

Функція штрафу дає змогу звести задачу умовної оптимізації до послідовності задач безумовної оптимізації для різних значень n.

Висновки за розділом 2

У другому розділі було розглянуто методологію вирішення задачі розміщення куль у тривимірному просторі, а також особливості поставленої задачі, які впливають на вибір стратегії її розв'язання. Проаналізовано сучасні методи побудови стартового розміщення, локальної та глобальної оптимізації, їх сильні та слабкі сторони.

Для реалізації поставленої задачі обрано комбінацію методів:

– метод випадкового розміщення для генерації початкових позицій сфер, що забезпечує простоту та швидкість;

– метод SLSQP для локальної оптимізації, який ефективно працює з обмеженнями і забезпечує швидку збіжність;

– метод мультистарту для глобальної оптимізації, що дозволяє уникнути локальних мінімумів і досягти глобального екстремуму.

Отримані результати аналізу дозволили сформувати стратегію вирішення задачі, яка поєднує простоту, точність та ефективність. Запропоновані методи забезпечують адаптивність до різних типів контейнерів та обмежень, що підтверджує їх доцільність для вирішення задач розміщення куль. Це створює основу для програмної реалізації обраної стратегії.

3 РЕАЛІЗАЦІЯ ПРОГРАМИ

3.1 Аналіз функціональних і нефункціональних вимог до системи

Перед розробкою програмного забезпечення для задачі розміщення куль у заданій тривимірній області необхідно визначити функціональні та нефункціональні вимоги до системи. Це дозволить сформувати чітке уявлення про можливості, які повинна забезпечувати система, а також визначити основні критерії, які впливатимуть на її ефективність та зручність використання.

Функціональні вимоги описують основні вимоги до системи, які забезпечать виконання головної задачі. Серед них слід виділити наступні ключові вимоги:

– імпорт вхідних даних: система повинна дозволяти завантаження параметрів області розміщення, а саме геометричних характеристик контейнера у вигляді набору нормалізованих площин у 3D просторі. Вхідними даними також є радіус куль, що розміщуються, а також мінімальна допустима відстань між кулями;

– генерація початкового розміщення: система повинна автоматично будувати початкове допустиме розміщення куль у тривимірному просторі на основі заданих площин контейнеру;

– оптимізація розміщення: система повинна виконувати процес локальної та глобальної оптимізації для наближення розміщення куль до цільового показника максимізації кількості куль;

– візуалізація результату: система повинна забезпечувати наочне відображення отриманого розміщення куль у межах тривимірного контейнера, з можливістю перегляду результатів під різними кутами та масштабами;

– збереження результатів: система повинна дозволяти збереження результатів роботи у вигляді файлу з параметрами розміщення куль та стартовими обмеженнями задачі, а також з візуальним представленням результату (зображення або 3D-модель);

– виведення метрик ефективності: система повинна надавати аналітичну інформацію щодо щільності розміщення куль, використаного об’єму та часу виконання оптимізації.

Нефункціональні вимоги стосуються продуктивності, надійності та зручності роботи системи. Основні вимоги до системи наступні:

– продуктивність: програма повинна виконувати обчислення для задач великої розмірності (з кількістю куль понад сто) за прийнятний час;

– масштабованість: система повинна підтримувати роботу з різними обсягами даних, починаючи від невеликих задач до великих тривимірних областей із високим числом куль;

– зручність інтерфейсу: користувачеві має бути забезпечений зручний інтерфейс для налаштування параметрів розміщення, запуску алгоритмів оптимізації та перегляду результатів;

– портативність: програма має працювати на різних операційних системах (Windows, Linux), а також підтримувати можливість інтеграції з іншими інструментами чи програмними пакетами.

Таким чином, аналіз функціональних та нефункціональних вимог до системи дозволяє визначити ключові завдання для реалізації програми. Важливо, щоб програмний продукт був ефективним, масштабованим та зручним у використанні, а також забезпечував високоякісну візуалізацію отриманих результатів.

Також необхідно зазначити, що система розглядається не як кінцевий продукт, а як модуль, що потім буде використаний іншими більш спеціалізованими системами. Тому інтерфейс взаємодії та імпорту/експорту даних слід обирати саме для такого контексту. Таким чином, для основного застосунку розрахунку позицій куль імпорт на експорт даних доцільно реалізувати методом зчитування та запису у JSON файл для імпорту та експорту відповідно, а інтерфейс має бути інтерфейсом програмного рядку без графічної складової. Засіб візуального представлення результату доцільно реалізувати окремо від засобу обчислення позицій куль. Він також буде здійснювати імпорт даних за допомогою

зчитування результатів обчислення з файлу, але матиме графічний інтерфейс користувача.

На цьому аналіз вимог до системи можемо вважати закінченим.

3.2 Вибір технологій створення програми

3.2.1 Вибір середовища для реалізації програми розміщення куль

На етапі вибору середовища для реалізації програми розміщення куль важливо порівняти найбільш популярні мови програмування та математичні пакети, щоб визначити найефективніший та найзручніший інструмент для вирішення поставлених задач. Критеріями порівняння є наявність математичних бібліотек, простота використання, ефективність обчислень, можливість інтеграції у фінальні програмні продукти, а також вартість ліцензій.

Одним із відомих математичних пакетів є Wolfram Mathematica, що має широкий набір функцій для математичного моделювання, симуляцій та візуалізації. Ця система надає зручні вбудовані функції для роботи з геометричними об'єктами та оптимізаційними задачами. Однак Wolfram Mathematica є комерційним продуктом, і його використання вимагає придбання ліцензії, що може стати перешкодою при подальшому використанні програми. Крім того, хоч цей пакет і є потужним для розв’язання наукових задач, він не є універсальною мовою програмування, а отже, код, написаний у Mathematica, важко інтегрувати у складніші програмні системи різних арехітектур таких як мобільні додатки, WEB-сервери тощо.

MATLAB також є популярним інструментом для наукових обчислень та інженерних задач. MATLAB пропонує потужні засоби для роботи з матрицями, оптимізації та геометрії, а його додаткові інструменти (Simulink та інші модулі) дозволяють моделювати системи різної складності. Проте MATLAB, як і Wolfram Mathematica, є платним продуктом. Його обмеження у вигляді ліцен-

зійної політики роблять його менш привабливим для розробників, які прагнуть створити масштабоване і безкоштовне рішення. Ще одним недоліком є те, що MATLAB не є повноцінною мовою програмування для промислових систем, що обмежує його інтеграцію у фінальні програмні продукти.

Розглянемо універсальні мови програмування такі як C++ та Java. Вони широко використовуються для розробки програмних продуктів завдяки їхній швидкодії та можливостям низькорівневого доступу до ресурсів. Проте робота з математичними задачами у C++ та Java вимагає додаткових зусиль, оскільки ці мови не мають такого різноманіття бібліотек для наукових обчислень, як інші інструменти. Для виконання геометричних операцій або оптимізаційних задач розробнику доводиться використовувати сторонні бібліотеки або писати власні алгоритми, що значно збільшує час розробки.

Оптимальним рішенням є Python, який поєднує переваги гнучкої мови програмування та наявність багатого набору бібліотек для математичних та наукових обчислень. Python є безкоштовною мовою з відкритим вихідним кодом, що робить його доступним для широкого кола розробників. Завдяки бібліотекам, таким як NumPy [18], SciPy [19], Matplotlib, SymPy та Pandas, Python дозволяє ефективно виконувати складні математичні обчислення, оптимізацію, роботу з геометричними об’єктами, а також візуалізацію результатів. Для вирішення задач локальної оптимізації у Python існує бібліотека scipy.optimize, яка має широкий набір методів для нелінійного програмування. Крім цього, у Python доступні такі потужні бібліотеки, як PyTorch або TensorFlow, що можна використовувати для розв'язання оптимізаційних задач у контексті машинного навчання якщо це буде необхідно на подальших етапах розвитку системи.

Ще однією перевагою Python є його висока інтегрованість із іншими системами. Код, написаний у Python, легко може бути інтегровиний у готові програмні продукти, оскільки ця мова широко використовується для розробки прикладного програмного забезпечення, WEB-сервісів та систем автоматизації. Також мова Python є кросплатформною, що дозволяє додаткам написаним за допомогою Python бути запущеною на усіх сучасних популярних операційних си-

стемах таких як Windows, Linux та MacOS. На відміну від математичних пакетів на кшталт Wolfram Mathematica чи MATLAB, Python є повноцінною мовою програмування, що дозволяє використовувати розроблені алгоритми як частину більшого програмного продукту.

Таким чином, на основі порівняння математичних пакетів та мов програмування було обрано Python як середовище для реалізації програми розміщення куль. Python є безкоштовним, має широкий набір бібліотек для вирішення математичних задач та забезпечує зручність інтеграції у фінальні програмні рішення. Це робить його найоптимальнішим вибором для реалізації поставленої задачі. У нашому випадку використовуватиметься версія Python 3.7.4.

3.2.2 Вибір бібліотеки пакету для реалізації програми розміщення куль

Для реалізації алгоритму розміщення куль у тривимірному просторі необхідно обрати відповідну бібліотеку для математичних обчислень, яка забезпечить високу ефективність, точність та гнучкість у розв'язанні задач оптимізації. При виборі бібліотеки важливо враховувати наявність вбудованих методів для локальної оптимізації, підтримку роботи з нелінійними обмеженнями, можливість інтеграції з іншими інструментами Python, а також продуктивність під час обробки великих обсягів даних.

Однією з найвідоміших бібліотек для наукових обчислень є NumPy, яка є основою для більшості математичних операцій у Python. NumPy надає можливість ефективно працювати з багатовимірними масивами та матрицями, виконуючи базові операції лінійної алгебри, але ця бібліотека не містить високорівневих алгоритмів для оптимізації, особливо для задач із нелінійними обмеженнями. Тому для реалізації задачі розміщення куль NumPy може бути використана лише як допоміжний інструмент для базових математичних операцій.

Іншою потужною бібліотекою є SymPy, яка призначена для символьних обчислень. SymPy дозволяє формулювати математичні моделі аналітично та

виконувати символьне диференціювання й спрощення функцій. Однак SymPy не призначена для чисельної оптимізації та роботи з великими обсягами даних. Її обчислювальна продуктивність є відносно низькою для задач, що потребують чисельного розв'язання нелінійних рівнянь та оптимізації.

Бібліотека CVXPY орієнтована на задачі опуклої оптимізації. Вона дозволяє формулювати задачі у зручному декларативному вигляді та підтримує розв’язання лінійних, квадратичних і опуклих задач. Проте задачі розміщення куль часто мають нелінійний характер та можуть включати не лише опуклі, але й неопуклі обмеження. Це робить CVXPY менш придатною для реалізації алгоритмів у контексті досліджуваної задачі.

TensorFlow та PyTorch є популярними бібліотеками для роботи з нейронними мережами та машинним навчанням, що також містять базові функції для оптимізації. Проте ці бібліотеки більше орієнтовані на розв'язання задач глибокого навчання, а їх застосування для задач розміщення куль було б надлишковим через складність налаштування та нераціональне використання ресурсів.

SciPy є розширенням NumPy і надає потужний набір інструментів для наукових обчислень, включно з методами оптимізації, інтегрування, інтерполяції та обробки сигналів. Її модуль scipy.optimize є ключовим для задач оптимізації, оскільки він містить широкий спектр алгоритмів для локальної та глобальної оптимізації. Для задач нелінійного програмування scipy.optimize підтримує методи, такі як методи Ньютона, методи градієнтного спуску, алгоритм обмеженої оптимізації BFGS (L-BFGS-B), а також глобальні методи, включно з диференційною еволюцією та алгоритмом прямого пошуку. Важливою перевагою є наявність зручного інтерфейсу для задання цільових функцій та обмежень, що дозволяє ефективно реалізувати алгоритми розміщення куль у 3D просторі.

Порівняно з іншими бібліотеками, scipy.optimize є найкращим вибором для реалізації алгоритму через кілька ключових переваг:

– підтримка широкого набору методів оптимізації, як локальних, так і глобальних, що забезпечує гнучкість при розв’язанні задач різної складності;

– зручність інтеграції з іншими бібліотеками Python, такими як NumPy

для обчислень і Matplotlib для візуалізації;

– висока продуктивність під час чисельних обчислень, що забезпечує швидке виконання алгоритмів оптимізації навіть для великих наборів даних;

– доступність та відкритий код, що дозволяє використовувати бібліотеку без обмежень і витрат на ліцензії.

Таким чином, для реалізації програми розміщення куль у заданій тривимірній області було обрано бібліотеку scipy.optimize, яка поєднує в собі потужний інструментарій для оптимізації, високу продуктивність та зручність використання. У нашому випадку використовуватиметься версія SciPy 1.15.1.

3.2.3 Вибір технологій для реалізації програми візуалізації результату

Для реалізації програми візуалізації результатів розміщення куль необхідно обрати відповідний інструмент, що забезпечить зручне і наочне відображення тривимірних об’єктів у межах обмеженої 3D області. Основними критеріями для вибору технології є кросплатформовість, продуктивність, гнучкість у налаштуванні, можливість інтеграції з іншими програмами та доступність.

Одним із варіантів є Python-бібліотека Matplotlib, яка підтримує базову візуалізацію тривимірних об’єктів за допомогою модуля mpl_toolkits.mplot3d. Ця бібліотека є зручною для візуалізації результатів наукових обчислень, дозволяє відображати поверхні та точки у тривимірному просторі. Проте функціональність Matplotlib для роботи з 3D-графікою є обмеженою. Вона не підтримує налаштування освітлення, текстурування та експорт візуалізації у вигляді 3D моделей, що робить її менш підходящою для складних 3D-візуалізацій.

Іншим варіантом є використання Python-бібліотеки VTK (Visualization Toolkit), яка є потужним інструментом для побудови та візуалізації 3D-моделей. VTK дозволяє створювати тривимірні об'єкти, виконувати обчислення на сітках, а також будувати складні візуалізації, включно з анімаціями. Ця бібліотека активно використовується для наукової візуалізації, зокрема у медицині та ін-

женерії. Проте, як і більшість інструментів для Python, VTK орієнтована на офлайн-середовище і потребує встановлення спеціального програмного забезпечення. VTK є Python-обгорткою бібліотеки для C++, що використовує нативні засоби системи для проведення рендеру, що може бути проблемою при використанні даної бібліотеки у деяких операційних системах. Використання VTK вимагає додаткових зусиль для створення зручного інтерфейсу та інтеграції з іншими інструментами, що робить цей варіант менш зручним для кросплатформових рішень.

Для задач візуалізації результатів у реальному часі з інтерактивною взаємодією надзвичайно перспективним є використання WEB-технологій, а саме JavaScript/TypeScript у поєднанні з бібліотекою three.js. JavaScript є мовою програмування, яка підтримується практично в усіх сучасних WEB-браузерах та серверних середовищах, що забезпечує кросплатформовість і доступність програми для різних користувачів без необхідності встановлення додаткового програмного забезпечення. TypeScript, як надбудова над JavaScript, забезпечує строгий контроль типів і покращену легкість візуального сприйняття коду, що особливо важливо при розробці великих проєктів.

Three.js [20] є популярною бібліотекою для роботи з тривимірною графікою у WEB-середовищі за допомогою WebGL [21]. Ця бібліотека дозволяє легко створювати, налаштовувати та візуалізувати 3D-об'єкти у браузері. Three.js підтримує текстурування, освітлення, анімацію, камеру та інтерактивне управління моделями, що дозволяє реалізувати як базову візуалізацію результатів розміщення, так і створення повноцінної 3D-сцени. Також three.js має значну кількість варіантів експорту створеної сцени у 3D моделі різних форматів. Окрім цього, бібліотека є безкоштовною та має відкритий вихідний код, що робить її доступною для будь-яких розробників.

Використання JavaScript/TypeScript та three.js забезпечує кілька ключових переваг. По-перше, програма стає кросплатформовою, оскільки WEB-браузери є доступними на всіх основних операційних системах (Windows, MacOS, Linux) та мобільних пристроях. Це означає, що користувач може переглядати резуль-

тати візуалізації незалежно від платформи чи пристрою. По-друге, three.js дозволяє не лише інтерактивно візуалізувати результати оптимального розміщення куль у 3D просторі, але й експортувати створену модель у форматах, які можуть бути використані для подальшого 3D-друку або інтеграції з іншими програмами для тривимірного моделювання. При цьому експорт 3D моделі можна проводити як у WEB-браузері так і у серверному середовищі.

Таким чином, серед усіх розглянутих варіантів було обрано JavaScript/TypeScript у поєднанні з бібліотекою three.js для реалізації програми візуалізації результатів. Такий підхід є кросплатформовим, зручним для інтеграції та забезпечує ефективну, гнучку та доступну візуалізацію тривимірних об'єктів у реальному часі, що повністю відповідає вимогам до проєкту.

3.3 Реалізація програми розміщення куль

Спочатку розглянемо загальну структуру програму. Діяльність програми починається з імпорту початкових параметрів даних. Далі необхідно створити цикл на деяку кількість ітерацій (у нашому випадку – на 20, це число було підібрано емпірично). На кожній з цих ітерацій програма буде шукати оптимальне розміщення куль при заданих параметрах. Це необхідно, бо під час знаходження оптимального розміщення куль задіяні випадкові процеси, що у кожному окремому випадку може заважати знаходженню глобального екстермуму, а запуск одного і того же коду пошуку оптимального розміщення куль допомагає нівелювати ефект випадковості і знайти глобальне оптимальне рішення. Таким способом реалізується глобальна оптимізація методом мультистарту. На кожній ітерації пошук оптимального розміщення куль проводиться наступною послідовністю дій:

– генерація початкового рішення, що складається з одного центру кулі обраного випадковим чином;

– це рішення оптимізується;

– якщо рішення розміщення куль допустиме, то до нього додається один центр кулі згенерований випадковим чином і програма повертається на попередній крок;

– якщо рішення розміщення куль не є допустимим, то останнє допустиме рішення повертається як оптимальне.

Після завершення цього циклу глобальної оптимізації, програма виводить фінальне оптимальне рішення у JSON файл. Блок-схема даного алгоритму зображена на рис 3.1.

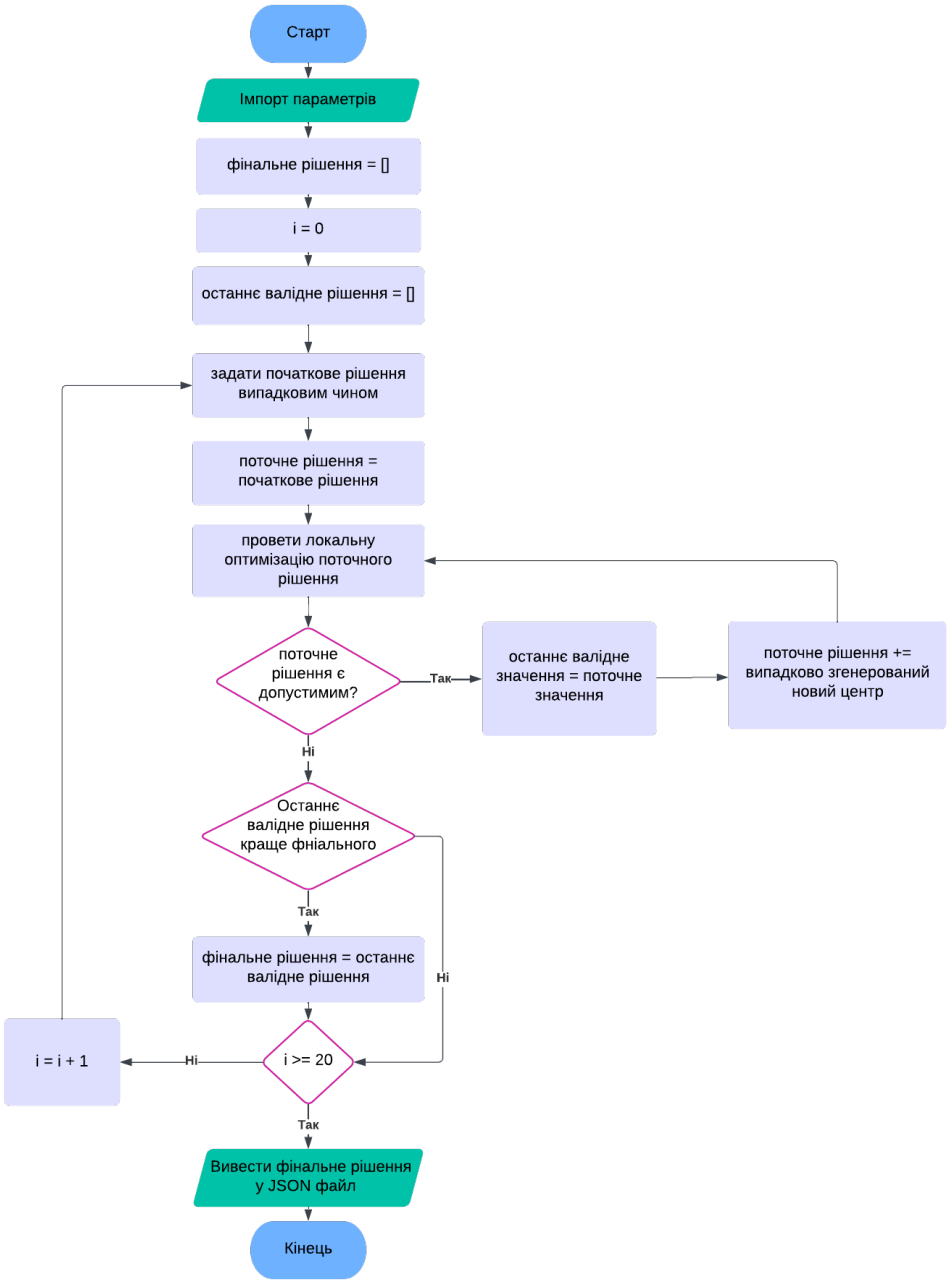

Рисунок 3.1 – Блок схема алгоритму програми розміщення куль

Далі розглянемо код розробленого застосунку на мові програмування Python. Вихідний код даної програми наведений у додатку А. Спочатку оглянемо функції, що наявні у даному файлі.

Функція import_data_from_json(filepath) відповідає за завантаження вхідних параметрів задачі з JSON-файлу. Основна мета – зчитати вихідні параметри задачі:

– список площин у вигляді їхніх коефіцієнтів (A, B, C, D);

– радіус куль;

– мінімальну відстань між кулями.

Слід зазначити, що функція зчитування даних не відповідає за валідацію даних, бо це не є у межах розроблюваної програми.

Функція distanceSquared(center1, center2) обчислює квадрат евклідової відстані між двома центрами куль, що задані координатами їх центрів. Ця функція використовується для перевірки перекриття між кулями при їх розміщенні.

Функція distance_from_sphere_to_plane(center, plane) розраховує відстань від центра кулі до заданої площини. Дана відстань набуває додатких значень якщо центр кулі знаходить з того півпростору, у котрий напрямлена нормаль даної площини і від’ємних у іншому випадку. Ця функція використовується у функціях для обчислення штрафів за вихід куль за межі контейнера.

Функція boundary_penalty_function(center, planes) обчислює штраф за вихід кулі за межі області. Для кожної площини перевіряється відстань від центра кулі до площини. Якщо куля виходить за межі, додається штраф, пропорційний відхиленню. При цьому штраф помножується на деяке достатньо велике число (у даному випадку на 100). Це зроблено для того, щоб алгоритм оптимізації у першу дотримувався меж контейнеру, а не неперетину куль.

Функція objective_function(centers_flat, num_spheres) – це цільова функція, яка мінімізується під час оптимізації. Вона враховує два компоненти:

– штраф за перекриття куль, обчислений як сума квадратів відхилень між центрами куль з поправкою на необхідну мінімальну відстань між ними;

– штраф за вихід куль за межі області, розрахований за допомогою

boundary_penalty_function.

Ця функція повертає загальний штраф, який має бути мінімізований до нуля функцією оптимізації у випадку коли розміщення куль при заданих параметрах контейнеру, кількості куль, радіусів куль і їх радіусів є можливим.

Функція single_optimization(initial_centers, num_spheres) використовує метод SLSQP з бібліотеки scipy.optimize для оптимізації положень куль. Початкові координати куль задаються у вигляді плоского масиву, який після оптимізації перетворюється назад у тривимірний формат. Результатом є оптимізовані координати центрів куль та значення цільової функції.

Функція make_optimal() реалізує ітеративний процес збільшення кількості куль у кубі. Починаючи з однієї кулі, функція виконує оптимізацію і поступово додає нові кулі до області, доки штраф за вихід або перекриття не стане більше нуля. Цей підхід дозволяє динамічно знаходити максимально можливу кількість куль у межах заданих параметрів контейнеру та куль.

Функція export_data_to_json(filepath, planes, sphere_radius, min_distance, optimized_centers) відповідає за запис отриманих результатів у JSON-файл. Вона зберігає вихідні параметри задачі (площини контейнеру, радіус куль та мінімальну допустиму відстань між кулями) разом із координатами оптимізованих центрів куль. Такий набір даних у одному файлі є повним для подальшої візуалізації результатів роботи програми.

3.4 Реалізація програми візуалізації результату

Програма для візуалізації результатів розміщення куль реалізована з використанням JavaScript/TypeScript та бібліотеки three.js. Основна мета – наочно представити сферичні об’єкти, оптимально розташовані у заданому контейнері, разом із візуалізацією обмежувальних площин і допоміжних елементів, таких як координатні сітка та осі. Вихідний код даної програми наведений у додатку А. Далі розглянемо функції даної програми.

Функція loadVisualizationData(filepath) відповідає за зчитування даних, що були експортовані програмою розміщення куль, з JSON-файлу. Вхідні параметри функції включають шлях до JSON-файлу. Функція завантажує дані про площини контейнеру, радіус куль, мінімальну допустиму відстань між кулями та координати оптимізованих центрів куль і повертає їх у вигляді об’єкту JavaScript.

Функція getSphereMaterial() створює та повертає матеріал для відображення куль у сцені. Використовує THREE.MeshStandardMaterial для задання кольору, прозорості та відбивної здатності, що забезпечує зручний для візуального сприйняття вигляд куль.

Функція getContainerMaterial() створює матеріал для контейнера, використовуючи напівпрозорість для демонстрації внутрішнього розташування куль. Використовує аналогічний підхід до створення матеріалу, як і у випадку куль, із параметрами, що підходять для візуалізації контейнера.

Функція createHalfSpaceObject(plane) приймає параметри площини (коефіцієнти A, B, C, D) та створює тривимірний об’єкт, що відповідає напівпростору, обмеженому цією площиною. Використовує інструменти бібліотеки three.js для створення геометрії та матеріалу об’єкта, який можна використовувати у подальших булевих операціях.

Функція createSphereObject(center, radius) створює тривимірний об’єкт кулі за заданими параметрами: координатами центру та радіусом. Використовує THREE.SphereGeometry для генерації геометрії кулі та getSphereMaterial() для додавання матеріалу.

Функція performBooleanIntersection(object1, object2) виконує булеву операцію перетину між двома тривимірними об’єктами. Використовує бібліотеку, three-bvh-csg, щоб отримати результат як новий об’єкт, що є спільною частиною вхідних об’єктів.

Функція createAllSpheres(centers, radius) створює масив 3D-об’єктів куль, використовуючи координати центрів і радіус куль, зчитаних із JSON-файлу. Вона викликає createSphereObject для кожного центру та об’єднує всі створені

об’єкти в один масив.

Функція createContainerObject(planes) створює тривимірний об’єкт контейнера, виконуючи послідовні булеві операції перетину для всіх напівпросторів, створених функцією createHalfSpaceObject. Отриманий об’єкт представляє фінальну область контейнера.

Функції для допоміжних об'єктів:

– createGridHelper(): створює координатну сітку за допомогою THREE.GridHelper, що полегшує орієнтування у сцені;

– createAxisHelper(): створює осі координат за допомогою THREE.AxesHelper для демонстрації основних напрямків.

Функція createScene(planes, centers, radius) створює тривимірну сцену та додає всі необхідні об’єкти:

– контейнер, створений функцією createContainerObject;

– кулі, отримані з createAllSpheres;

– допоміжні об’єкти, такі як координатна сітка та осі;

– використовує камеру, джерела світла (THREE.PointLight, THREE.AmbientLight) та сцену для забезпечення якісної візуалізації. Слід зазначити, що необхідно використовувати саме ортогональну камеру THREE.OrthographicCamera а не перспективну THREE.PerspectiveCamera, бо таким чином забезпечується більш наочне відображення куль на фоні координатних сіток та осей.

Функція exportSceneToFile(filepath, scene, format) зберігає 3D-сцену в заданому форматі (наприклад, GLTF або OBJ) у файл. Використовує бібліотеку three.js для експорту 3D-об'єктів, що дозволяє зберегти сцену для подальшого використання у сторонніх додатках чи 3D-друці.

Висновки за розділом 3

У третьому розділі виконано програмну реалізацію алгоритмів розміщен-

ня куль у заданій тривимірній області. Проведено аналіз функціональних і нефункціональних вимог до системи, що дозволило визначити основні критерії ефективності та зручності використання. Обґрунтовано вибір Python як середовища для реалізації програми через його потужний інструментарій для математичних обчислень, доступність та легкість інтеграції у складні програмні продукти. Для візуалізації результатів обрано JavaScript/TypeScript у поєднанні з бібліотекою three.js, що забезпечує кросплатформовість і якісну тривимірну графіку.

Реалізовано програму, яка включає етапи імпорту даних, генерації початкового розміщення, ітеративної оптимізації методом мультистарту з використанням локальної оптимізації (метод SLSQP) та експорту результатів. Виведені дані дозволяють проводити візуалізацію отриманих результатів за допомогою розробленого додатка на three.js, який інтерактивно відображає структури контейнера і куль.

Розроблене програмне забезпечення є ефективним інструментом для розв'язання задач розміщення куль, адаптованим до різних умов задачі, та забезпечує наочне представлення результатів. Виконана реалізація створює основу для подальшого вдосконалення системи.

4 ТЕСТУВАННЯ РОЗРОБЛЕНОЇ ПРОГРАМИ

4.1 Огляд методики тестування

Метою тестування є перевірка ефективності та точності програми для розміщення куль у межах різних тривимірних контейнерів. Для цього будуть проведені обчислення з використанням трьох різних форм контейнерів:

– куб, що є базовою формою із шістьма обмежувальними площинами;

– тетраедр, що є базовою формою із чотирма площинами;

– контейнер складної форми, визначений 56 напівплощинами, що створюють нетривіальну багатогранну область.

Для кожного з випадків тестування буде проведено аналіз таких параметрів:

– радіус куль – основний геометричний параметр, що впливає на кількість куль, які можна розмістити у заданому контейнері;

– мінімальна допустима відстань між кулями – визначає вимоги до розташування куль, забезпечуючи відсутність перекриття та дотику між ними;

– час виконання обчислень – кількість секунд, затрачених програмою для розміщення куль з оптимізацією їх положення;

– кількість розміщених куль – число куль, які успішно розміщені у контейнері з урахуванням заданих параметрів;

– порожнинність кінцевого об'єкта у 3D – відношення об’єму куль до загального об’єму контейнера, яке відображає щільність розміщення порожнин у формі куль всередині контейнера.

Формула порожнинності виглядає наступним чином:

$$Porosity = \frac{N * \frac{4}{3}\pi r^3}{V_{container}}$$

де N – кількість розміщених сфер;

r – радіус куль;

$V_{container}$ – об’єм контейнеру, що обчислюється за 3D моделлю.

Методика тестування передбачає використання різних значень радіусів куль та мінімальної допустимої відстані між ними для кожного з контейнерів. Це дозволяє оцінити вплив цих параметрів на ефективність алгоритму, щільність розміщення та обчислювальні ресурси, необхідні для виконання задачі.

Результати тестування будуть зафіксовані у вигляді таблиць, що допоможуть знайти залежності між параметрами. Особлива увага буде приділена аналізу порожнинності кінцевих структур та часу обчислень, що дозволить оцінити придатність програми для задач різної складності. Порівняння результатів для трьох типів контейнерів дозволить перевірити універсальність програми та її здатність працювати з формами різної геометричної складності.

4.2 Тестування для випадку кубу

Тестування програми розміщення куль було проведено для контейнера у формі кубу з розміром сторони чотири умовні одиниці. В межах тестування розглянуто три сценарії з різними радіусами куль, мінімальними допустимими відстанями між кулями та іншими параметрами. Результати тестування приведені у таблиці 4.1.

Таблиця 4.1 – Результати тестування програми для випадку кубу

№	Радіус куль	Мінімальна відстань	Час обчислень (с)	Кількість куль	Порожнинність (%)
1	0.9	0.1	0.418	6	39.97
2	0.4	0.05	34.99	39	19.49
3	0.3	0.001	126.12	68	13.64

У першому сценарії використано великі кулі з достатньо великою мінімальною відстанню між ними. Завдяки цьому обчислення виконуються дуже швидко і порожнинність є високою. Відповідне розміщення куль зображено на рис. 4.1.

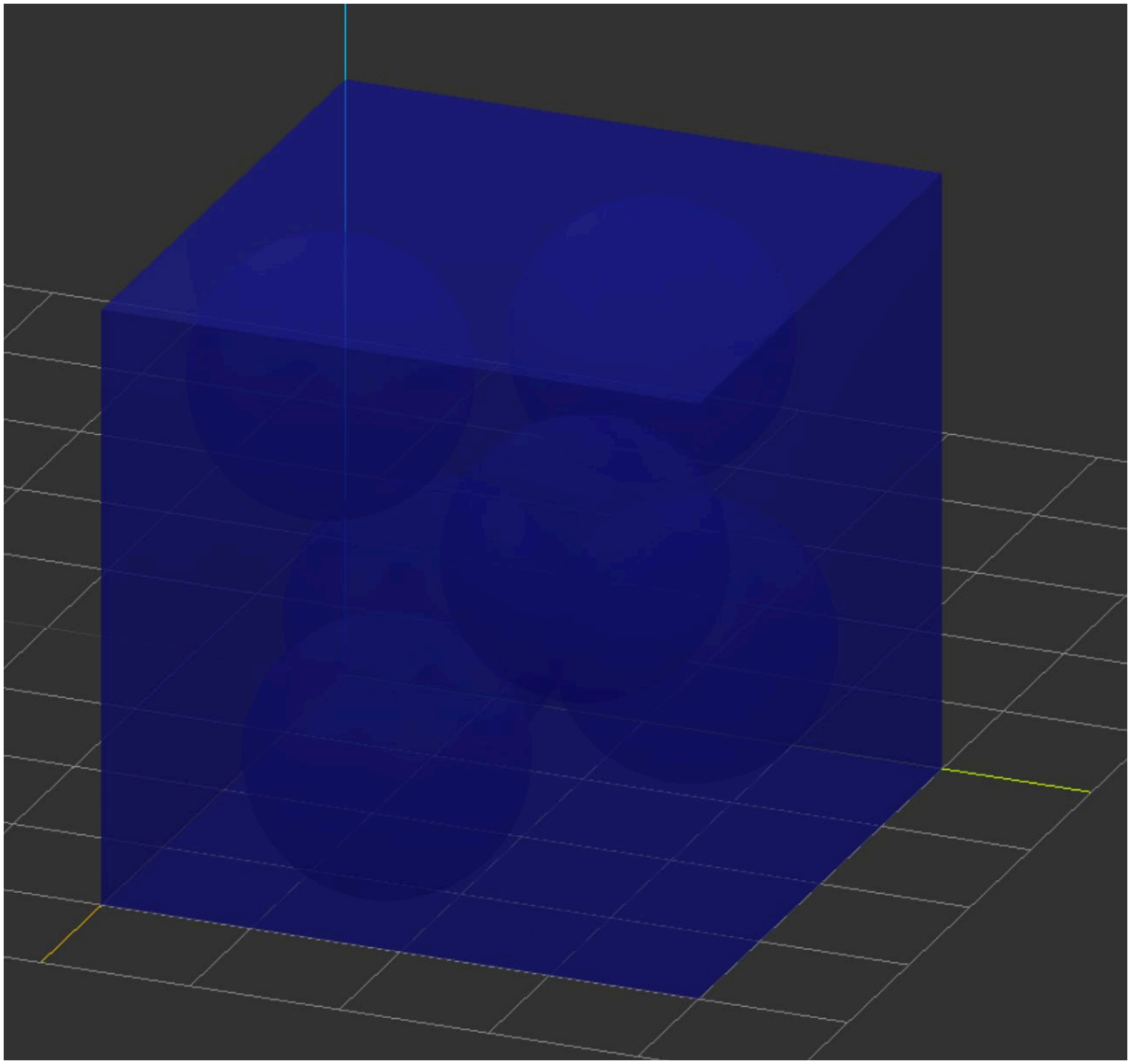

Рисунок 4.1 – Розміщення великих куль у кубі

У другому сценарії розміщення, який зображений на рис. 4.2, середній радіус куль та помірна мінімальна відстань дозволяють розмістити значно більше об'єктів у межах кубу. Час обчислень суттєво збільшується порівняно зі сценарієм 1, що пояснюється більшою кількістю куль та складнішими взаємодіями між ними.

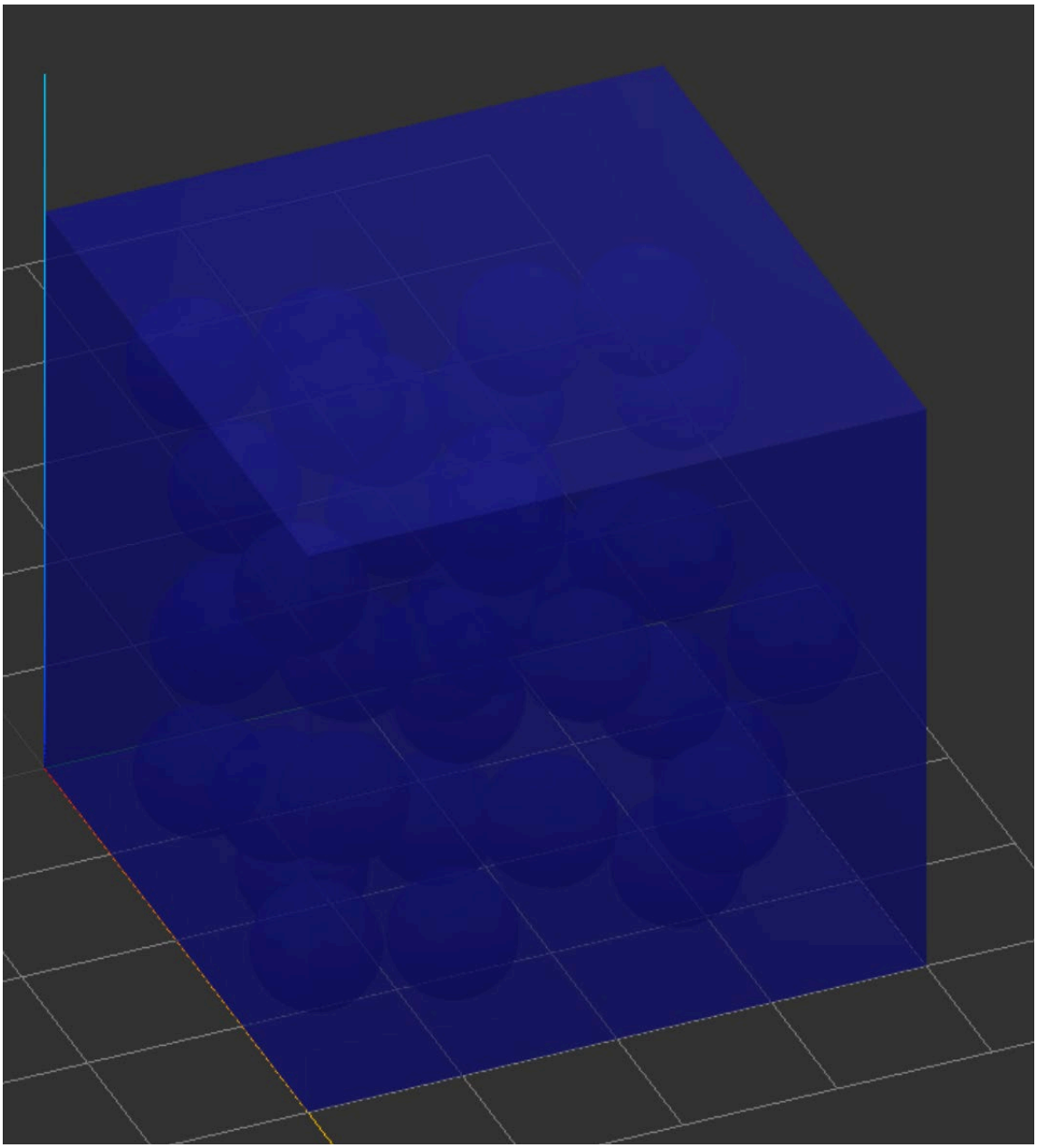

Рисунок 4.2 – Розміщення куль середнього розміру у кубі

Використання малих куль у третьому сценарії із дуже малою мінімальною відстанню між ними дозволяє максимально заповнити контейнер кулями. Однак час обчислень значно зростає через велику кількість куль і необхідність врахування багатьох взаємодій. Порожнинність є найнижчою, що свідчить про те, що попри велику кількість куль, їх малий розмір не дозволяє так ефективно зменшувати об’єм контейнеру, як у минулих сценаріях. Відповідне розміщення куль зображене на рис. 4.3.

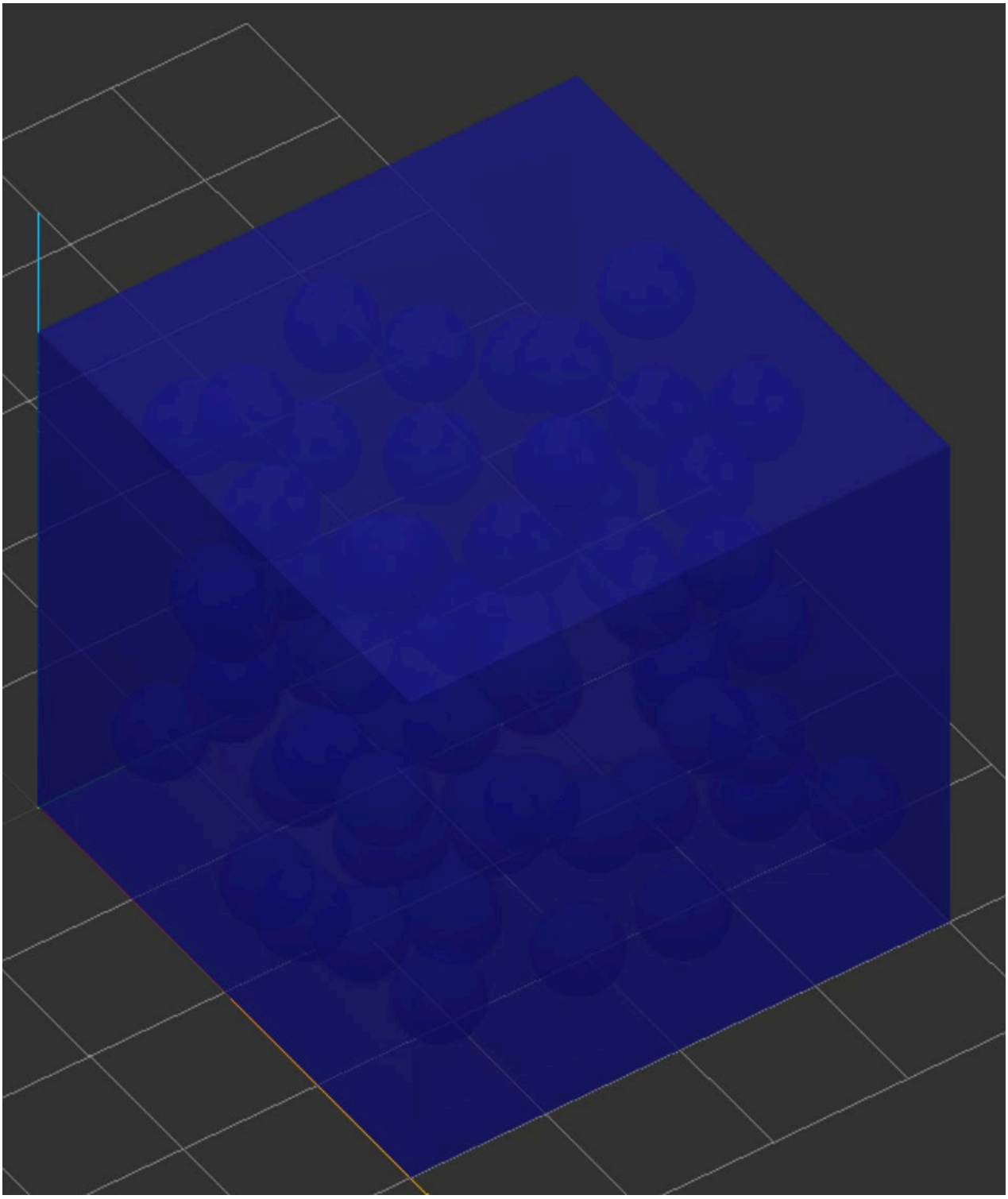

Рисунок 4.3 – Розміщення малих куль у кубі

Порівняння трьох сценаріїв демонструє, що використання якомога більшого радіусу куль та зменшення мінімальної відстані між ними дозволяє досягти вищої порожнинності контейнера. Також, такий варіант є більш ефективним з точки зору швидкодії програми.

4.3 Тестування для випадку тетраедра

Тестування програми розміщення куль було проведено для контейнера у формі тетраедра. Тестування охоплювало три сценарії з різними параметрами радіусів куль. Результати наведені у таблиці 4.2.

Таблиця 4.2 – Результати тестування програми для випадку тетраедру

№	Радіус куль	Мінімальна відстань	Час обчислень (с)	Кількість куль	Порожнинність (%)
1	0.4	0.01	0.223	4	31.20
2	0.2	0.01	3.039	18	15.46
3	0.12	0.01	73.288	58	10.28

У даному випадку залежність часу обчислення, порожнинності та кількості куль від їх радіусу є такою самою як і у випадку з кубом. Розміщення куль малого, середнього та великого розміру зображено на рис. 4.4 – 4.6 відповідно.

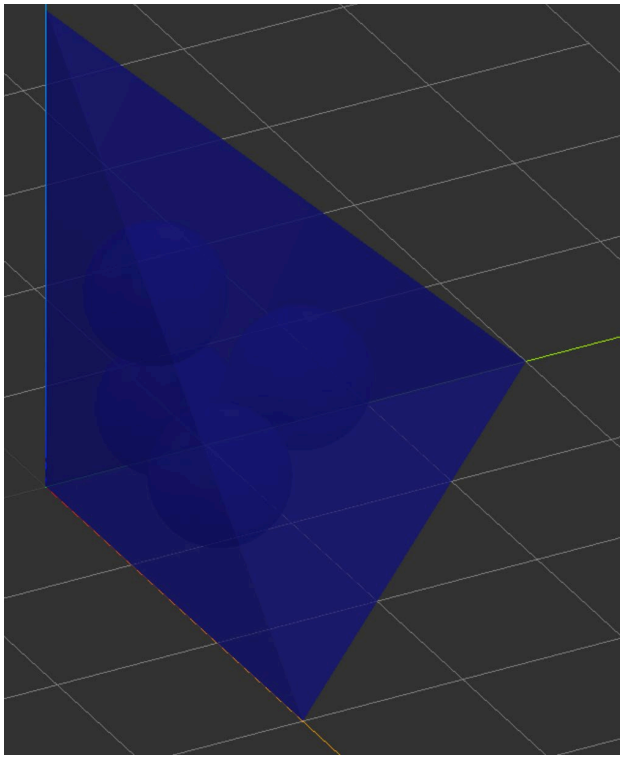

Рисунок 4.4 – Розміщення великих куль у тетраедрі

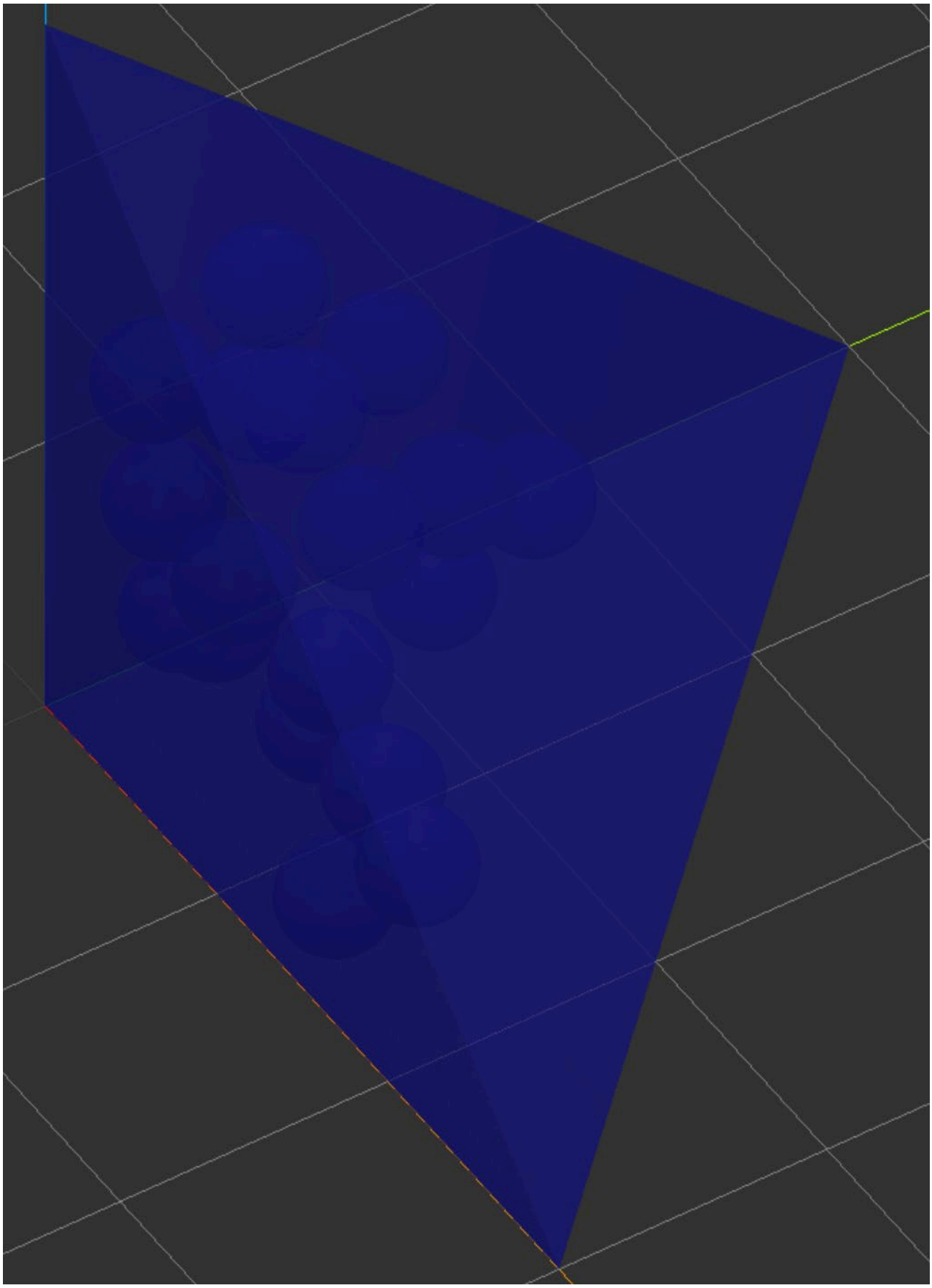

Рисунок 4.5 – Розміщення куль середнього розміру у тетраедрі

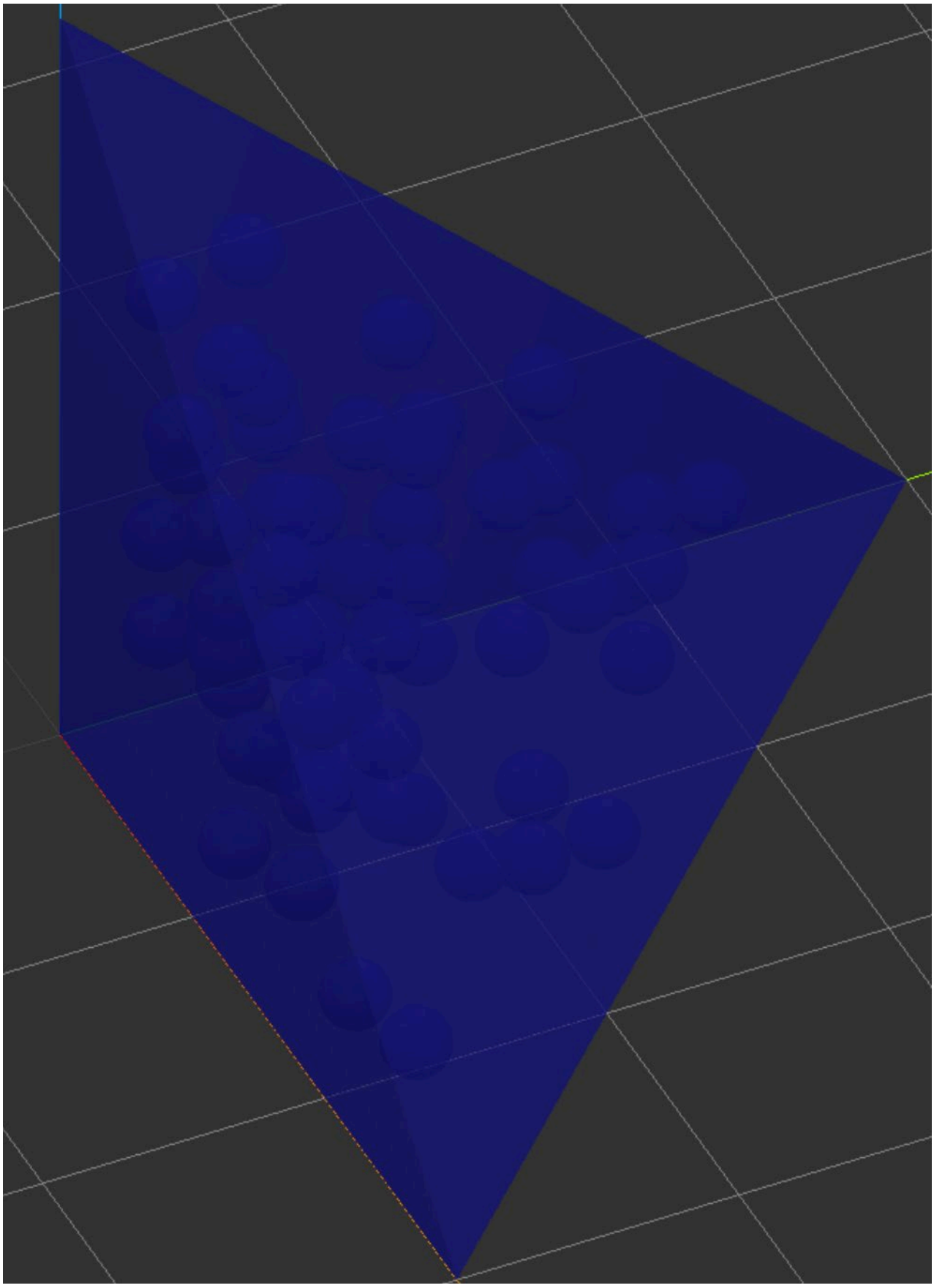

Рисунок 4.6 – Розміщення малих куль у тетраедрі

4.4 Тестування для випадку опуклого багатогранного контейнера довільної форми

Тестування програми розміщення куль було проведено для контейнера опуклого багатогранного контейнера довільної форми, що складається з 56-ти напівплощин. Тестування охоплювало три сценарії з різними параметрами радіусів куль. Результати наведені у таблиці 4.3.

Таблиця 4.3 – Результати тестування програми для випадку опуклого багатогранного контейнера довільної форми

№	Радіус куль	Мінімальна відстань	Час обчислень (с)	Кількість куль	Порожнинність (%)
1	0.4	0.01	5.997	13	19.50
2	0.3	0.01	51.670	36	23.54
3	0.2	0.01	209.417	69	12.14

У випадку опуклого багатогранного контейнера довільної форми, на відміну від попередніх форм, порожнинність при середньому розмірі куль (зображено на рис. 4.7) є більшою ніж при великому розмірі куль (зображено на рис. 4.8). Це пояснюється тим, що для даної конкретної форми у випадку великих куль залишалося забагато простору незаповненого порожнинами, який може бути заповнений при кулях меншого розміру. Даний результат важно екстраполювати на інші опуклі багатогранні контейнери довільної форми, тому для кожного є необхідність шукати оптимальний розмір куль. Також слід зазначити, що набагато більша кількість площин, у порівнянні з попередніми формами контейнера, вплинула на час обчислень, але усе ще кількість куль впливає на час обчислень значно більше. Розміщення куль малого розміру зображено на рис. 4.9.

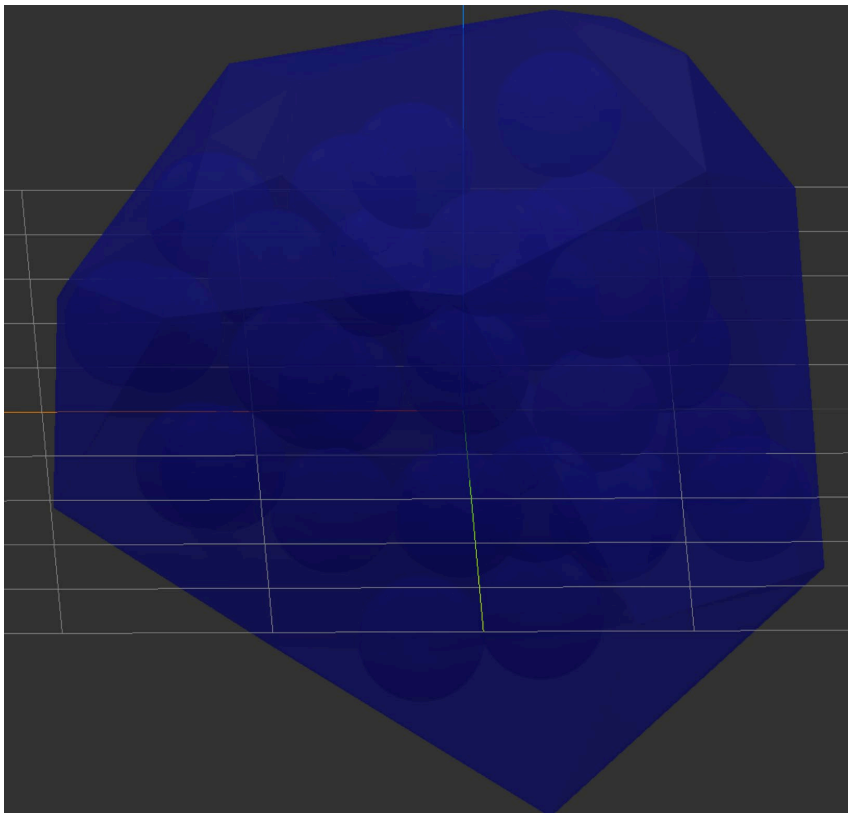

Рисунок 4.7 – Розміщення куль середнього розміру
у опуклому багатогранному контейнері довільної форми

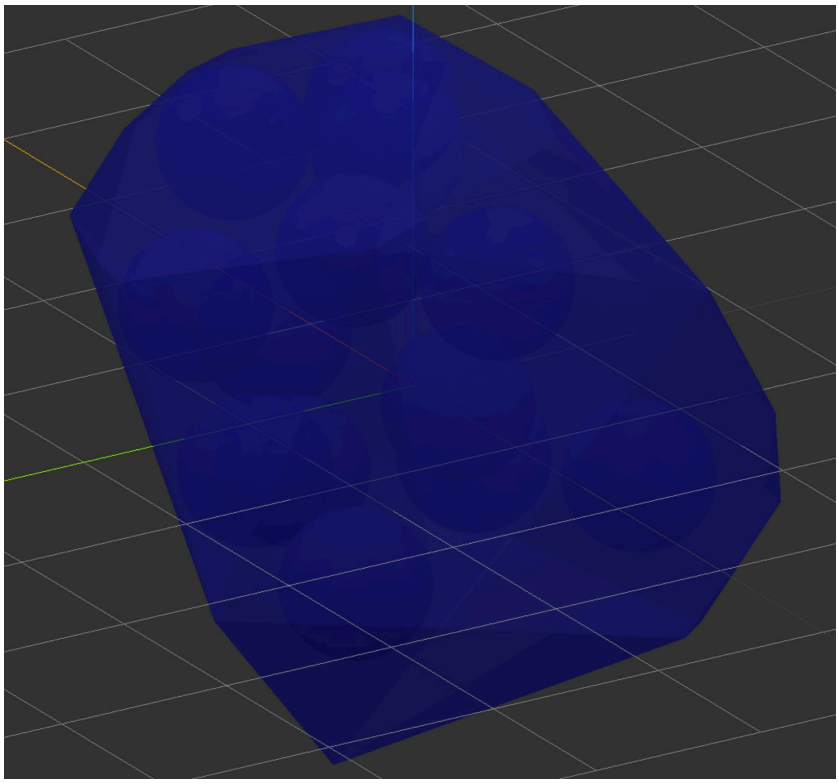

Рисунок 4.8 – Розміщення великих куль
у опуклому багатогранному контейнері довільної форми

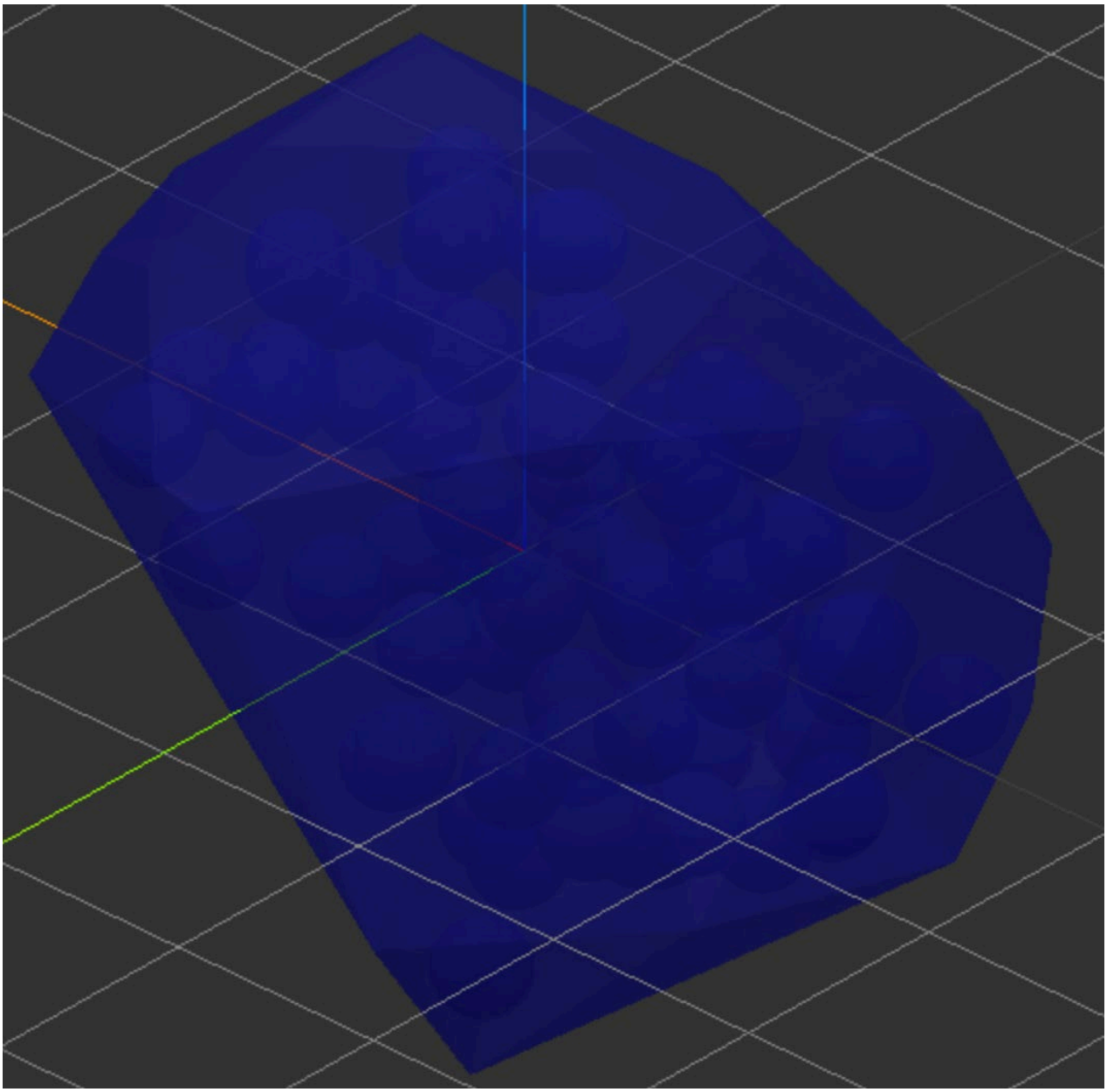

Рисунок 4.9 – Розміщення малих куль
у опуклому багатогранному контейнері довільної форми

Висновки за розділом 4

У ході тестування програми розміщення куль було проведено обчислення для трьох типів контейнерів: куб, тетраедр та контейнер нетривіальної форми, визначений 56 напівплощинами. У кожному випадку оцінювалися ключові параметри: радіус куль, мінімальна допустима відстань між ними, кількість роз-

міщених куль, час виконання обчислень та порожнинність отриманого розміщення.

Використання більших куль дозволило досягти більшої порожнинності контейнера. Також, таке розміщення потребує значно меншого часу на знаходження. Проте це не завжди є оптимальним варіантом, оскільки кулі великого розміру не завжди можуть бути розміщені у контейнери складної форми. Тому для кожного конкретного контейнеру необхідно окремо шукати оптимальний радіус куль. Також, використання великого радіусу порожнин у вигляді куль не завжди є фізично можливим у рамках адитивного виробництва та інших сценаріїв, для яких розроблюється дана програма.

Для всіх типів контейнерів час виконання зростав із збільшенням кількості куль. Це пояснюється тим, що задача розміщення однаковою кількості куль у контейнері належить до класу складності NP, тому складність вирішуваної задачі росте експоненційно до кількості розміщуваних куль.. Також, час обчислення зростав з кількістю площин, якими задається контейнер, але цей вплив був значно менший за вплив кількості розміщуваних куль.

Порожнинність залежала від розміру куль та щільності їх розташування. Найвища порожнинність спостерігалася для сценаріїв з великими кулями, а найнижча – для сценаріїв із малими кулями, що залишали велику частину контейнера без порожнин.

ВИСНОВКИ

У ході виконання кваліфікаційної роботи було досліджено та реалізовано підхід до вирішення задачі розміщення однакових куль у заданому тривимірному контейнері із використанням сучасних методів оптимізації. Основною метою було створення та програмна реалізація ефективного алгоритму для генерації порожнинних структур у формі куль із заданими геометричними параметрами, що може бути застосовано в адитивному виробництві та інших галузях.

Результатом виконання кваліфікаційної роботи є програма для обчислювання оптимального розміщення сфер у заданих умовах та програмний засіб візуалізації результатів обчислення. Обидві програмні засоби використовують сучасні технології і формати даних, тому можуть були легко інтегровані у інші інформаційні системи задля подальшої оптимізації процесів, для яких необхідно реалізовати обчислення. Результати роботи були проаналізовані для різних випадків розміру куль і форм контейнеру. В усіх випадках алгоритм успішно відпрацював і були отримані дані результатів розміщення куль, що відповідають заданим умовам і забезпечують необхідну порожнинність.

Програма реалізована у дуже загальному вигляді, тому вона може бути використана у будь-якій задачі, що потребує розміщення однакових куль із заданою мінімальної відстанню у заданому контейнері для забезпечення максимального об’єму зайнятого кулями. У першу чергу, практична цінність реалізованої програми може бути знайдена у оптимізації моделей у адитивному виробництві задля оптимізації витраченого матеріалу та часу виробництва.

Подальші дослідження за цією тематикою доцільно спрямувати на розширення можливостей програми, зокрема:

– розробку та реалізацію більш гнучких алгоритмів створення порожнин для реалізації більшої порожнинності, наприклад, із використанням куль різного радіусу;

– дослідження впливу геометрії порожнинних структур на механічні та теплові властивості кінцевих виробів і включення цих і подібних обмежень в

алгоритм генерації порожнин;

– інтеграцію алгоритму з сучасними інструментами моделювання для більш комплексного аналізу отриманих структур;

– адитивного виробництва, що потребують порожнинних структур для забезпечення легкості та економії матеріалів.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Rozhko D. Generating spherical void structures in bounded 3D-domains. *3-тя Міжнародна молодіжна науково-практична конференція «НАВЧАННЯ І ВИКЛАДАННЯ: у світі після війни». Зб. матеріалів форуму*. Т. 6, Ч. ІІ. Харків : ХНПУ, 2024. 248 с.

2. Kantorovich V.L. Several Facts Related to the Notes by L. V. Kantorovich Reproduced in This Volume. J *Math Sci* . 2006, Vol 133. P. 1384 –1387.

3. Mukhacheva E.A., Mukhacheva A.S. L. V. Kantorovich and Cutting-Packing Problems: New Approaches to Combinatorial Problems of Linear Cutting and Rectangular Packing. J *Math Sci* . 2006, Vol 133. P. 1504–1512.

4. Rvachev V.L., Stoyan Y.G. On the problem of optimal layout of circular patterns. *Cybern Syst Anal* 1, 1965, P. 80–84.

5. Rvachev V.L., Stoyan Y.G. Algorithm for solution of the problem of optimum layout of circular patterns with restrictions on the distances between patterns. *Cybern Syst Anal* 1, 1965. P. 79–85.

6. Bernal J. D. A Geometrical Approach to the Structure of Liquids. *Nature*. 1959. Vol. 183. P. 141–147.

7. Masalov V. M., Sukhinina N. S., Kudrenko E. A., Emelchenko G. A. Mechanism of formation and nanostructure of Stöber silica particles. *Nanotechnology*. Vol. 22. Iss. 27. P. 275 – 718.

8. Gan Y., Kamlah M, Reiman J. Computer Simulation of Packing Structure in Pebble Beds. *Nuclear Technology*. 2019. Vol. 85. Iss. 10-12. P. 1782–1787.

9. M. Hifi, L. Yousef, A local search-based method for sphere packing problems, Eur. J. Oper. Res. *European Journal of Operational Research*. 2019, P. 482–500.

10. N. Chernov, Y. Stoyan, T. Romanova, Mathematical model and efficient algorithms for object packing problem. *Computational Geometry*. 2010. Vol. 43. P. 535–553.

11. Z. Z. Zeng, W. Q. Huang, R. C. Xu, Z. H. Fu, An algorithm to packing un-

equal spheres in a larger sphere. *Advanced Materials Research*. 2012. Vol. 546–547. P. 1464–1469.

12. Titscher T., Unger J. F. Application of molecular dynamics simulations for the generation of dense concrete mesoscale geometries. *Computers & Structures*. 2015. Vol. 158. P. 274–284.

13. J. Wang, Packing of unequal spheres and automated radiosurgical treatment planning. *Journal of Combinatorial Optimization*. 1999. Vol. 3. P. 453–463.

14. A. Sutou, Y. Day, Global optimization approach to unequal sphere packing problems in 3D. *Journal of Optimization Theory and Applications*. 2002. Vol. 114. P. 671–694.

15. Radin C. Random close packing of granular matter. *Journal of Statistical Physics*. 2008. Vol. 131. Iss. 4. P. 567–573.

16 P. I. Stetsyuk. Theory and Software Implementations of Shor's r-Algorithms. *Cybernetics and Systems Analysis.* 2017. Vol. 5. P. 692-703.

17. Feng J., Fu J., Lin Z. et al. A review of the design methods of complex topology structures for 3D printing. *Vis. Comput. Ind. Biomed*. 2018. Vol 5.

18 NumPy About Us. URL: https://numpy.org/about (дата звернення: 13.11.2024).

19 SciPy About Us. URL: https://scipy.org/about (дата звернення: 23.12.2024).

20 Three.js Fundamentals. URL: https://threejs.org/manual/#en/fundamentals (дата звернення: 04.01.2025).

21 WebGL Specification. URL: https://registry.khronos.org/webgl/specs/latest/1.0 (дата звернення: 01.12.2024).