КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
БУДІВНИЦТВА І АРХІТЕКТУРИ

Факультет геоінформаційних систем і управління територіями

Кафедра геоінформатики і фотограмметрії

ПОЯСНЮВАЛЬНА ЗАПИСКА
ДО АТЕСТАЦІЙНОЇ РОБОТИ
НА ЗДОБУТТЯ ОСВІТНЬОГО СТУПЕНЯ МАГІСТРА

на тему:

Дослідження засобів моделювання рельєфу в середовищі СКБД PostgreSQL/PostGIS

Дульський Володимир Сергійович

Київ 2024 р.

КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
БУДІВНИЦТВА І АРХІТЕКТУРИ

Факультет геоінформаційних систем і управління територіями

Кафедра геоінформатики і фотограмметрії

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____проф., д.т.н. Карпінський Ю.О.
“____” _____2024 року

ПОЯСНЮВАЛЬНА ЗАПИСКА
ДО АТЕСТАЦІЙНОЇ РОБОТИ
НА ЗДОБУТТЯ ОСВІТНЬОГО СТУПЕНЯ МАГІСТРА

Дослідження засобів моделювання рельєфу в середовищі СКБД PostgreSQL/PostGIS

Виконав студент групи ГСТ-23
193 «Геодезія та землеустрій»
(спеціальність)
Геоінформаційні системи і технології
(спеціалізація)
Дульський Володимир Сергійович

Керівник: Лященко А.А., проф., д.т.н.

Ідентичність підтверджую

Київ 2024 р.

КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
БУДІВНИЦТВА І АРХІТЕКТУРИ

Факультет: Геоінформаційних систем та управління територіями
Кафедра: Геоінформатики і фотограмметрії
Освітній рівень: «магістр за ОПП»
Спеціальність: 193 «Геодезія та землеустрій»
Спеціалізація: Геоінформаційні системи і технології

ЗАТВЕРДЖУЮ
Декан факультету
_____доцент., к.т.н. Нестеренко О. В.
“____” _____2024 року

З А В Д А Н Н Я
ДО ВИКОНАННЯ АТЕСТАЦІЙНОЇ ВИПУСКНОЇ РОБОТИ
НА ЗДОБУТТЯ ОСВІТНЬОГО СТУПЕНЯ МАГІСТРА

Дульський Володимир Сергійович
(прізвище, ім’я, по батькові)

1. Тема роботи: Дослідження засобів моделювання рельєфу в середовищі СКБД PostgreSQL/PostGIS
затверджена наказом ректора КНУБА № від « »_____2024 року

2. Керівник роботи проф., д.т.н. Лященко Анатолій Антонович
(прізвище, ім’я та по батькові, науковий ступінь, вчене звання)

3. Строк подання студентом роботи до захисту: 04 листопада 2024 р.

4. Зміст пояснювальної записки за розділами:
Вступ
Розділ 1. Стан і тенденції розвитку засобів моделювання рельєфу в ГІС
Розділ 2. Методичні засади моделювання рельєфу в середовищі баз геопросторових даних
Розділ 3. Результати моделювання рельєфу для дослідних територій в середовищі СКБД PostgreSQL/PostGIS
Висновки
Список використаної літератури

5.Графічний матеріал за розділами
Розділ 1: Завдання, структура та обмеження проекту. Case-діаграма класифікації цифрових моделей рельєфу. Класифікація цифрових моделей рельєфу. Аналіз засобів моделювання рельєфу в сучасних ГІС та базах геопросторових даних (таблиця).
Розділ 2: Уніфіковані класи геопросторових об’єктів і типи даних для моделювання рельєфу в БГД. Класифікація растрових моделей в ГІС. Класифікація засобів опрацювання і аналізу GRID-моделі рельєфу в середовищі СКБД. Класифікація функцій GRID-моделей в середовищі PostgreSQL/PostGIS (таблиця). Структурно-функціональна схема ГІС моделювання рельєфу з використанням СКБД. Концептуальна модель бази геопросторових даних GRID-моделі рельєфу. Концептуальна модель бази геопросторових даних TIN моделі рельєфу.
Розділ 3: Технологічна схема опрацювання і аналізу GRID-моделей рельєфу в середовищі СКБД PostgreSQL/PostGIS. Технологічна схема створення і використання TIN рельєфу в середовищі СКБД PostgreSQL/PostGIS. Картосхема дослідної території та структура вихідних даних для моделювання рельєфу. Схеми алгоритмів прикладних SQL-функцій для створення і аналізу моделей рельєфу. Результати геоінформаційного моделювання рельєфу дослідної території.

7. Календарний план виконання роботи:

Назва етапів дипломного проекту (роботи)	Строк виконання етапів проекту
Розділ 1. Стан і тенденції розвитку засобів моделювання рельєфу в ГІС. Основні визначення, класифікація та сфери використання цифрових моделей рельєфу. Аналіз засобів моделювання рельєфу в сучасних ГІС та базах геопросторових даних. Структура, завдання та обмеження проекту. Висновки до розділу 1.	03.10.2024
Розділ 2. Методичні засади моделювання рельєфу в середовищі баз геопросторових даних. Уніфіковані класи геопросторових об'єктів і типи даних для моделювання рельєфу в БГД. Моделі і засоби опрацювання і аналізу GRID-моделі рельєфу в середовищі СКБД. Типи даних і засоби побудови TIN-моделі рельєфу в середовищі СКБД. Структурно-функціональна схема ГІС моделювання рельєфу з використанням СКБД. Висновки до розділу 2.	28.10.2024
Розділ 3. Результати моделювання рельєфу для дослідних територій в середовищі СКБД PostgreSQL/PostGIS. Вихідні дані, програмні засоби та стислий опис обчислювального експерименту на дослідну територію. Результати використання GRID-моделей рельєфу в середовищі СКБД PostgreSQL/PostGIS. Реалізація прикладної SQL функцій для побудови 3D моделей ліній з використанням GRID моделі рельєфу. Створення і використання TIN моделі рельєфу в середовищі СКБД PostgreSQL/PostGIS. . Висновки до розділу 3. Загальні висновки.	18.11.2024
Складання резюме та остаточне оформлення роботи	30.11.2024
Подання роботи на перевірку на плагіат	25.11.2024
Подання роботи на рецензування	10.12.2024
Попередній захист роботи на кафедрі	06.12.2024

8. Консультанти розділів атестаційної випускної роботи

Розділ	Прізвище, ініціали та посада консультанта	Перевірив	
		дата	підпис
Розділ 1.			
Розділ 2.			
Розділ 3.			

9. Дата видачі завдання 08 серпня 2024 р.

Зав. кафедри _____ Карпінський Ю.О.
(підпис) (прізвище та ініціали)

Керівник _____ Лященко А.А.
(підпис) (прізвище та ініціали)

Студент _____ Дульський В.С.
(підпис) (прізвище та ініціали)

<table>
<tr><td colspan="2">РЕЗЮМЕ (summary)
До атестаційної випускної роботи студента:</td><td colspan="3">Дульський Володимир Сергійович</td></tr>
<tr><td>Назва ВНЗ</td><td colspan="4">Київський національний університет будівництва і архітектури</td></tr>
<tr><td>Тема</td><td colspan="4">Дослідження засобів моделювання рельєфу в середовищі СКБД PostgreSQL/PostGIS</td></tr>
<tr><td>Освітній ступінь</td><td colspan="4">Магістр за освітньо-професійною програмою навчання</td></tr>
<tr><td>Факультет</td><td colspan="4">Геоінформаційних систем та управління територіями</td></tr>
<tr><td>Кафедри</td><td colspan="4">Геоінформатики та фотограмметрії</td></tr>
<tr><td>Спеціальність</td><td colspan="4">193 Геодезія та землеустрій</td></tr>
<tr><td>Спеціалізація</td><td colspan="4">Геоінформаційні системи і технології</td></tr>
<tr><td>Керівник</td><td colspan="4">Лященко Анатолій Антонович, д.т.н, професор</td></tr>
<tr><td rowspan="2">Обсяг роботи</td><td>пояснювальна записка, стор.</td><td>розділів</td><td colspan="2">креслень формату А4</td></tr>
<tr><td>135</td><td>3</td><td colspan="2"></td></tr>
<tr><td>Розділ 1</td><td colspan="4">Визначено основні поняття, класифікацію та сфери використання цифрових моделей рельєфу. Виконано порівняльний аналіз засобів моделювання рельєфу в сучасних ГІС та базах геопросторових даних. Приведено структуру, завдання та обмеження проєкту</td></tr>
<tr><td>Розділ 2</td><td colspan="4">Розроблено структурно-функціональну модель ГІС моделювання рельєфу з використанням об’єктно-реляційних СКБД, що ґрунтується на уніфікованих класах геопросторових об’єктів, відповідних спеціальних типах даних і функцій просторових розширень універсальних СКБД.</td></tr>
<tr><td>Розділ 3</td><td colspan="4">Розроблено технологічні схеми моделювання рельєфу з використанням GRID та TIN моделей в середовищі СКБД PostgreSQL/PostGIS на дослідну територію міста Києва. Викладено результати аналізу GRID моделі рельєфу з використанням даних SRMT на дослідну територію. Реалізовано прикладні SQL функції побудови 3D лінії на GRID-моделі поверхні та обчислення їх довжини. Визначено типові схеми SQL запитів та розроблено прикладні SQL функції побудови TIN моделей рельєфу, обчислення висоти довільної точки та побудови 3D лінії з використанням TIN-моделі рельєфу. Викладено результати тестування розроблених SQL функцій для створення і використання GRID та TIN моделей рельєфу на дослідну територію міста Києва.</td></tr>
<tr><td>Висновки по роботі:</td><td colspan="4">В СКБД PostgreSQL з просторовим розширенням PostGIS надається набір базових функцій, які забезпечують підтримку спеціальних типів даних для створення і зберігання в базах геопросторових даних цифрових моделей рельєфу типу регулярних сіток GRID, нерегулярних тріангуляційних мереж TIN та TIN з обмеженнями структурних ліній. Розроблені прикладні SQL функції формування реконструкції TIN моделей, інтерполяції висоти довільної 2D точки та побудови 3D ліній на поверхні TIN підвищують ефективність створення і використання цифрових моделей рельєфу в середовищі СКБД PostgreSQL/PostGIS при вирішенні практичних завдань. Коректність результатів використання розробленних функцій підтверджено на реальних наборах даних на дослідну територію міста Києва.</td></tr>
<tr><td colspan="5"><u>Ключові слова:</u> цифрова модель рельєфу, база геопросторових даних, СКБД PostgreSQL/PostGIS, нерегулярна мережа трикутників, ЦМР, TIN, GRID
<u>Keywors:</u> Digital Elevation Model, Geospatial Database, DBMS PostgreSQL/PostGIS, Triangular Irregular Network, DTM, DEM, TIN, GRID</td></tr>
</table>

Укладач: _____ / Дульський В.С./

Керівник: _____ /Лященко А.А./

«____» _____ 2024

ЗМІСТ

ВСТУП

Актуальність теми. Моделювання рельєфу важливо для вирішення практичних завдань в багатьох сфер діяльності, включаючи геоінформаційні системи (ГІС) просторового планування, моніторингу земель, гідрогафії, розроблення протизсувних та протиповеневих заходів тощо. В сучасних інструментальних ГІС пропонуються ефективні засоби для створення та моделювання цифрових моделей рельєфу (ЦМР). Разом з цим, сьогодні спостерігається тенденція до інтеграції баз геоданих (БГД) з універсальними системами керування базами даних (СКБД), що забезпечує ряд переваг, таких як незалежність даних від форматів та програмних засобів інструментальних ГІС, багатокористувацький доступ до гепросторових даних, зберігання і маніпулювання великими обсягами геопросторових даних тощо.

Традиційно в СКБД зберігаються переважно векторні дані, але останніми роками активно впроваджується підтримка растрових даних та спеціальних моделей 3D поверхонь, включаючи цифрові моделі рельєфу. Незважаючи на те, що в інструментальних ГІС вже існують розвинені засоби для роботи з ЦМР, дослідження засобів зберігання та опрацювання таких моделей в середовищі СКБД залишається актуальним. Це обумовлено потребою в інтеграції даних різних типів (векторних, растрових і ЦМР) в єдиному сховищі на основі СКБД та їх використання в інструментальних ГІС від різних виробників в залежності від прикладних задая і потреб кінцевих користувачів геоінформаціної продукції.

Об’єктно-реляційна СКБД PostgreSQL з просторовим розширенням PostGIS належить до найпоширеніших поміж СКБД з відкритим кодом, що використовуються в сучасних ГІС. PostgreSQL/PostGIS відповідає вимогам міжнародних стандартів і специфікацій в сфері географічної інформації, зокрема стандартам відкритого геопросторового консоціумуму OGC (Open Geospatial Consortium) та міжнародним стандартам серії ISO 19100 «Географічна інформація/геоматика».

Мета і завдання дослідження. *Метою роботи* є дослідження функціональних можливостей та ефективності використання СКБД

PostgreSQL/PostGIS для створення і використання цифрових моделей рельєфу на реальних обсягах даних при вирішенні типових прикладних задач.

Для досягнення цієї мети в роботі сформульовано та вирішено такі основні завдання:

– аналіз стану і тенденцій розвитку засобів цифрового моделювання рельєфу в ГІС та СКБД;

– аналіз уніфікованих базових типів геопросторових даних і функцій, що використовуються для моделювання рельєфу в базах геопросторових даних;

– проведення обчислювального експеременту зі створення і використання GRID та TIN моделей рельєфу в середовищі СКБД PosrgreSQL/PostGIS;

– розроблення прикладних SQL функцій для побудови 3D ліній з використанням GRID моделі рельєфу;

– розроблення прикладних SQL функцій для побудови та реконструкції трикутників 3D TIN моделі рельєфу;

– розроблення прикладних SQL функцій для інтерполяції висоти довільної 2D точки та побудови 3D ліній за TIN моделлю рельєфу.

Об’єкт дослідження: спеціальні типи даних та вбудовані функції для цифрового моделювання рельєфу в сучасних ГІС та СКБД;

Предмет дослідження: моделі і методи створення і використання цифрових моделей рельєфу в середовищі СКБД PostgreSQL/PostGIS.

Методи дослідження. Методологічну основу роботи складають:

монографічний метод опрацювання наукових публікацій, нормативних документів, що стосуються моделювання рельєфу та СКБД;

методи формалізації для розроблення і подання схем алгоритмів, технологічних моделей та моделей геопросторових даних з використанням IDF та UML діаграм;

методи реалізації прикладних SQL-функцій в середовищі об’єктно-реляційної СКБД;

методи моделювання, аналізу та візуалізації геопросторових даних в ГІС.

Новизна одержаних результатів. У роботі на реальних наборах даних проведено обчислювальні експерименти щодо створення цифрових моделей рельєфу в середовищі СКБД PostgreSQL/PostGIS та розроблено прикладні SQL функції для вирішення таких типових прикладних задач:

1) побудова 3D моделей ліній та визначення їх довжини на поверхні з використанням GRID моделі рельєфу;

2) формування набору трикутників TIN моделі рельєфу з використанням базових функцій PostGIS для тріангуляції Делоне та тріангуляції з обмеженнями;

3) реконструкція TIN моделі рельєфу за набором геопросторових даних структурних ліній рельєфу;

4) визначення висоти довільної 2D точки на поверхні з використанням TIN моделі рельєфу;

5) побудова 3D моделей ліній на поверхні заданої TIN моделлю рельєфу.

З практичної точки зору в роботі встановлено, що:

1) в PostGIS надаються ефективні засоби для підтримки і аналізу GRID моделей рельєфу на основі спеціального типу даних для растрових моделей та набору функцій побудови поверхонь морфологічних характеристик рельєфу;

2) базові функції PostGIS реалізують тріангуляцію Делоне та тріангуляцію Делене з обмеженнями з наданням результатів у форматі спеціального типу даних TIN як єдиної колекції трикутників.

Для ефективного використання базових функцій PostGIS зі стоворення TIN моделей в практичних задачах в роботі реалізована прикладні SQL функція перетворення колекції типу TIN в набір полігонів окремих трикутників та зберіганням їх в таблиці бази геопросторових даних з побудовою просторового індексу на множині трикутників для оптимізації доступу до них в прикладних задачах моделювання 3D об’єктів та аналізу рельєфу.

Вихідні інформаційні ресурси роботи. Для виконання роботи як вихідні дані були використані такі інформаційні ресурси:

1. фрагмент SRMT-моделі рельєфу на територію міста Києва;
2. шейп-файл адміністративних меж областей України та міста Києва;

3. шейп-файл вулично-дорожньої мережі міста із набору даних OpenStreetMap;

4. Набір ізоліній рельєфу на територію м. Києва в масштабі 1:200 000.

Розділ 1

СТАН І ТЕНДЕНЦІЇ РОЗВИТКУ ЗАСОБІВ МОДЕЛЮВАННЯ РЕЛЬЄФУ В ГІС

РОЗДІЛ 1. СТАН І ТЕНДЕНЦІЇ РОЗВИТКУ ЗАСОБІВ МОДЕЛЮВАННЯ РЕЛЬЄФУ В ГІС

1.1 Основні визначення, класифікація та сфери використання цифрових моделей рельєфу

1.1.1 Основні визначення та класифікація цифрових моделей рельєфу

Цифрова модель рельєфу в загальному сенсі у геоінформатиці визначається як цифрове уявлення рельєфу земної поверхні, створене на основі даних про рельєф та морфології місцевості.

В англомовній літературі згідно розрізняють цифрову модель висот (digital elevation model, DEM) та похідну цифрову модель рельєфу (digital terrain model, DTM), яка використовується для морфометричного аналізу. Ці терміни пов'язані з найменуванням і змістом американського стандарту на ЦМР та багатозначністю слова "terrain". Розвиток методів створення ЦМР через оброблення зображень на цифрових фотограмметричних станціях призвів до появи терміна "цифрова модель поверхні" (DSM). Серед них найбільш вживані такі інтерпретації визначення:

Згідно [12] цифрова модель рельєфу *(digital terrain model, DTM; digital elevation model, DEM; Digital Terrain Elevation Data, DTED)* – засіб цифрового представлення 3-мірних просторових об’єктів (поверхонь, рельєфів) у вигляді тримірних даних як сукупності висотних відміток або відміток глибин та інших значень аплікат у вузлах регулярної сітки з утворенням матриці висот, нерегулярної трикутної сітки (TIN) або як сукупності записів горизонталей (ізогіпси, ізобат) або інших ізоліній.

Згідно [28] цифрова модель висот (DEM) — це цифрове представлення поверхні землі, яке відображає лише голий ґрунт, виключаючи дерева, будівлі та інші поверхневі об'єкти.

Згідно [16] цифрова модель рельєфу (DEM) або цифрова модель поверхні (DSM) — це тривимірне зображення даних про висоти для відображення рельєфу або об'єктів на його поверхні, зазвичай планети, місяця або астероїда.

Ґрунтуючись на приведених визначень та на опрацьованих публікаціях, узагальнено класифікацію моделей ЦМР, схема якої подано на рис. 1.1.

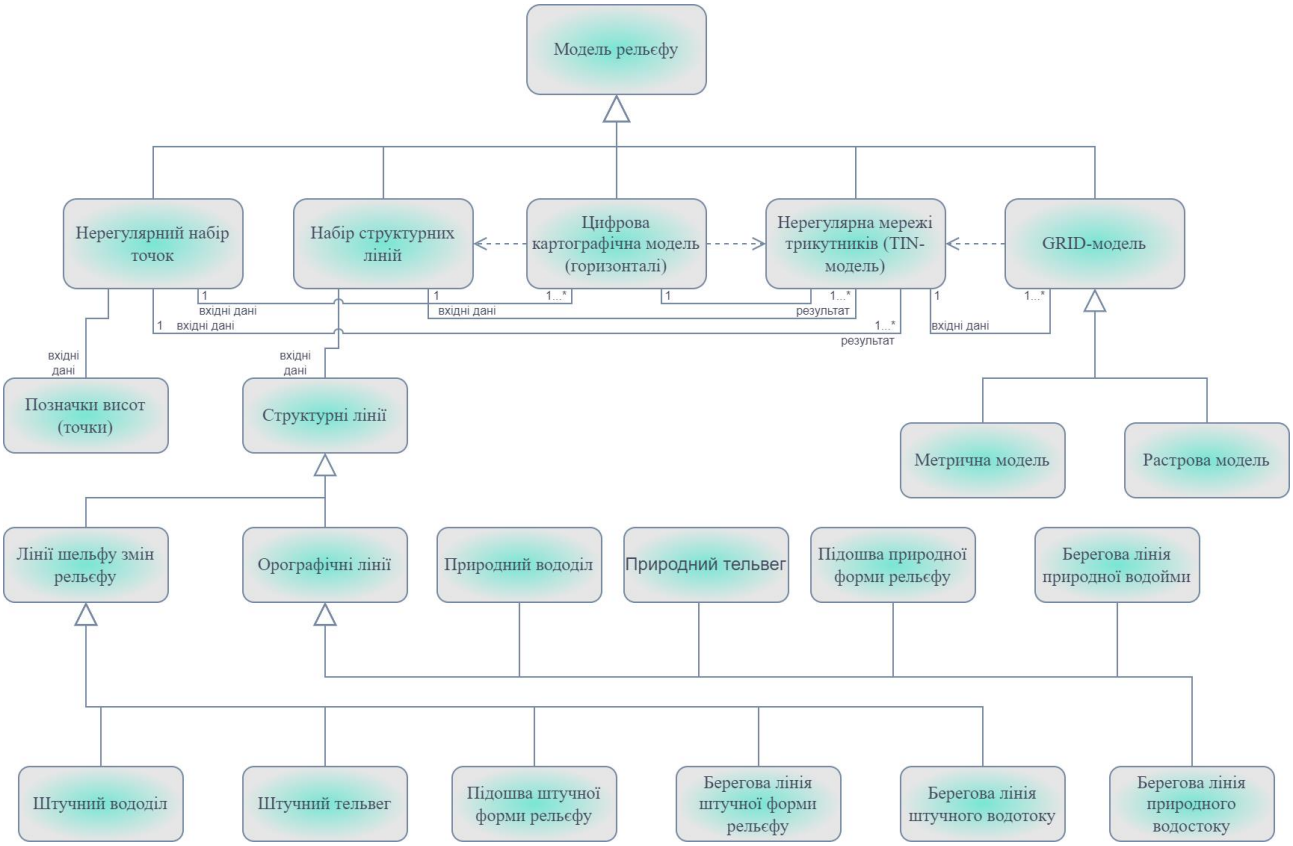


Рис. 1.1. Схема класифікація моделей рельєфу [5,9]

Нерегулярний набір точок: модель, у якій вихідні дані представлені як нерегулярно розташовані тривимірні точки (рис. 1.2).

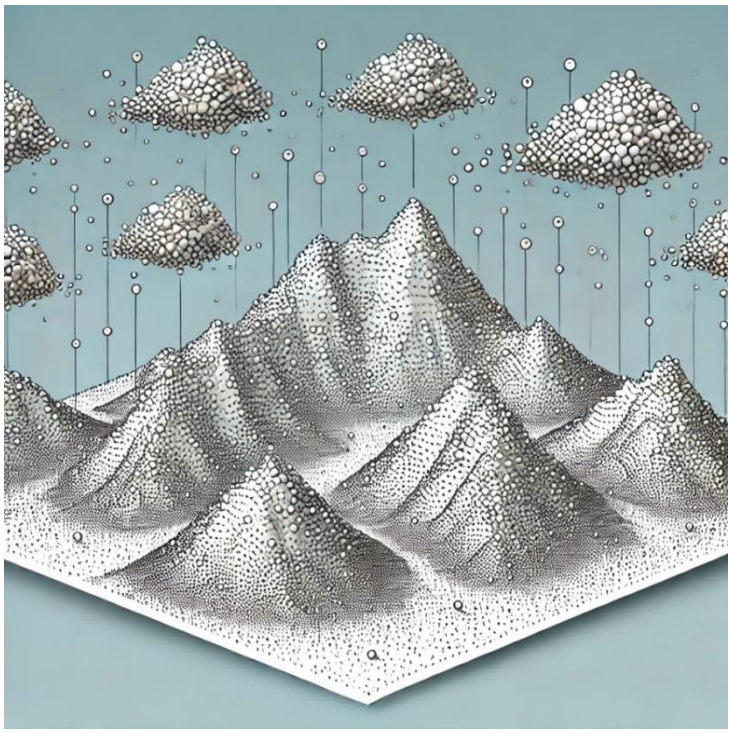


Рис. 1.2. Нерегулярний набір точок

Орографічно-тріангуляційна модель згідно [5] є комбінацією двох типів моделей: орографічної та тріангуляційної, що дозволяє точніше моделювати рельєф. Вона вирішує проблеми неоднозначності, характерні для інших моделей, завдяки інтеграції основних орографічних ліній у процес тріангуляції (рис. 1.3). Ребра тріангуляції повинні збігатися з орографічними лініями. Якщо такі лінії відсутні на певній ділянці, використовують класичну тріангуляцію Делоне. А сам процес моделювання відбувається через лінійну інтерполяцію висот всередині кожного трикутника

Рис. 1.3. Орографічно-тріангуляційна модель рельєфу

Картографічна модель: Сукупність ізогіпс (горизонталей та ізобат) – ліній, що з'єднують точки земної поверхні з однаковою абсолютною висотою [12](рис. 1.4). Вона не дозволяє отримати точні значення висот у будь-якій точці, але ефективна для загальних топографічних завдань.

TIN-модель (*Triangulated Irregular Network*): модель просторових даних, заснована на нерегулярній мережі трикутників, що апроксимує рельєф

багатогранною поверхнею з висотними відмітками (відмітками глибин) у вузлах мережі трикутників.

Рис. 1.4. Принцип побудови ізоліній картографічної моделі рельєфу

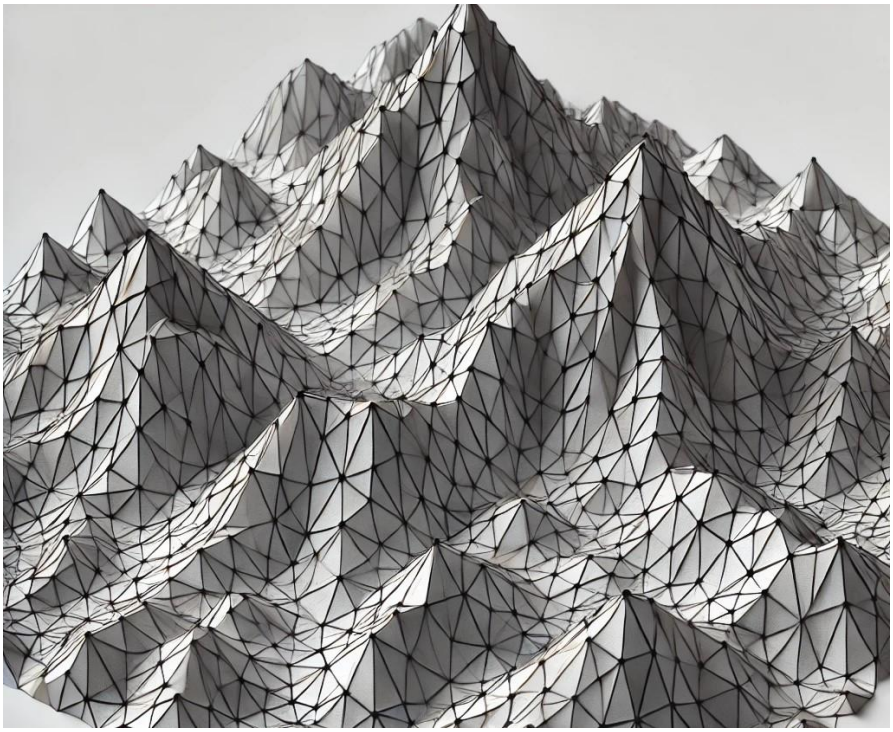


Рис. 1.5. TIN-модель рельєфу

TIN-модель (*Triangulated Irregular Network*): модель поверхні, заснована на нерегулярній мережі трикутників, що апроксимує рельєф багатогранною

поверхнею з висотними відмітками (відмітками глибин) у вузлах мережі трикутників. Згідно [21] вони здебільшого будуються за допомогою тріангуляції Делоне, але це не завжди так. Для даного набору точок можна побудувати різні тріангуляції. Тріангуляція Делоне (DT) множини точок S (рис. 1.5) визначається як тріангуляція S, така що жодна точка в S не лежить всередині описаного кола будь-якого іншого трикутника в тріангуляції DT максимізує мінімальний кут серед усіх можливих тріангуляцій, щоб уникнути довгих і тонких трикутників.

GRID-модель (матриця значень висот): цифрове подання рельєфу, яке відповідає растровій моделі поверхні як множині значень висоти у вузлах регулярної сітки (GRID-модель вузлова (рис. 1.6 а)) або в регулярно розташованих чарунках (комірках) однакового розміру та форми (GRID-модель чарункова (рис. 1.6 б))[12]. Переваги та недоліки використання цих методів наведено в таблиці 1.1

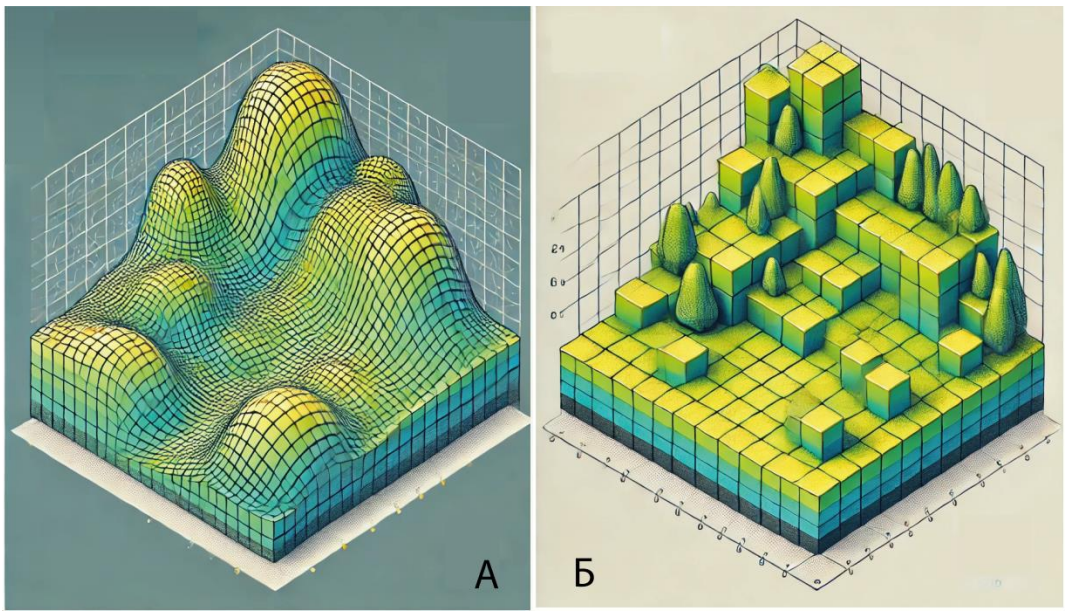


Рис. 1.6 GRID-моделі рельєфу (вузлова та чарункова)

В табл. 1.1 узагальнено порівняльні характеристики ЦМР різних типів. Заважимо, що в документації ArcGIS [14] наголошується, що TIN моделі не так широко доступні, як растрові моделі поверхонь, і, як правило, їх побудова та опрацювання дещо дорожче. Вартість отримання високоякісних вихідних даних може бути достатньо високою, а опрацювання TIN, із-за складності їх структур, дещо менш ефективна, ніж опрацювання растрових даних.

Таблиця 1.1 Порівняльні характеристики цифрових моделей рельєфу[9]

Модель	Переваги	Недоліки
Цифрова картографічна модель (горизонталі)	➢ невеликі об’єми збереження даних на носіях; ➢ топографічні карти є найбільш доступними з точки зору ціновоїполітики; ➢ виділяє основні риси рельєфу в масштабі карти, роблячи їх більш характерними для даного типу рельєфу; ➢ містить значно більше атрибутивної інформації про об'єкти місцевості	➢ великі трудові і фінансові затрати; ➢ недостатня точність моделювання; ➢ зображення рельєфу горизонталями не забезпечує отримання відмітки будь-якої точки з високою точністю
GRID	➢ швидке комп’ютерне опрацювання; ➢ легке зберігання і маніпулюванняданими; ➢ легко інтегрується з растровимибазами даних; ➢ отриманий рельєф місцевості плавний і має більш природний вигляд.	➢ топологічна невизначеність; ➢ займають більше місця на диску, ніж моделі TIN
TIN	➢ можливість опису поверхні різних рівнях роздільної здатності; ➢ ефективність зберігання даних на	➢ великі ресурсні затрати на опрацювання моделі; ➢ у багатьох випадках потрібна візуалізація і ручне керування мережею; ➢ виникає ефект терас, що виражається в появі «псевдо трикутників», що виникає при створенні, на основі горизонталей без урахування структурних ліній рельєфу(наприклад, по лінії днища V-подібних долин)
Структурна	➢ унеможливлює виникнення "псевдо трикутників" (які виникають там, де всі три вершини трикутника лежать на одній горизонталі), в результаті чого порушуються морфографія і морфометрія рельєфу, що моделюється, і відповідно знижується точність і якість самої моделі та її похідних; ➢ значно покращує якість і морфологічну правдоподібність ЦМР ; ➢ забезпечує практичне моделювання рельєфу великої розмірності	➢ додаткові витрати на підготовку 3D моделей набору структурних ліній рельєфу

1.1.2 Основні сфери застосування цифрових моделей рельєфу

Використання ЦМР є зручним методом для вирішення багатьох наукових та практичних завдань, які можуть виникати в тій чи іншій ситуації. Згідно [9] цифрову модель використовують для таких завдань:

1) інтерполяції висот в довільній точці місцевості;

2) отримання похідних морфометричних або інших даних, включаючи обчислення кутів ухилу та експозиції схилів;

3) побудови й аналізу зон видимості/невидимості;

4) побудови тривимірних зображень, профілів поперечного перетину;

5) оцінювання форми схилів через кривизну їх поперечного і поздовжнього перетину, вимірювану радіусом кривизни головного нормального перетину або її знаку, тобто опуклістю/увігнутістю поверхні;

6) формування ліній мережі тальвегів і вододілів, що утворюють каркасну мережу рельєфу, його структурних ліній або сепаратриса та інших особливих точок і ліній рельєфу: локальних мінімумів або западин та локальних максимумів або вершин, сідловини, брівок, ліній обривів та інших порушень "гладкості" поверхні, плоских поверхонь з нульовою крутістю;

7) побудови ізоліній на множині значень висот або тривимірній моделі поверхні;

8) автоматизації аналітичного відмивання рельєфу шляхом розрахунку відносної освітленості схилів при вертикальному, бічному або комбінованому освітленні від одного або більше джерел світла;

9) цифрового ортотрансформування при цифровому опрацюванні зображень та інших обчислювальних операціях в графоаналітичних побудовах;

10) трансформації вихідної моделі шляхом додавання нових даних;

11) розрахунку площ поверхні, розрахунку рівнів і площ затоплення і тощо.

ЦМР використовується в багатьох сферах життєдіяльності в тій чи іншій справі тому до них можливо віднести такі напрями:

ГІС – є важливим інструментом у багатьох сферах діяльності та використовується для різноманітних завдань, таких як аналіз місцевості, аналіз схилів і висот, створення карт та моделювання ландшафту.

Природоохоронна справа – використовуються для моніторингу змін ландшафтів, процесів ерозії ґрунтів, а також для планування заходів з охорони довкілля.

Сільське господарство – дозволяє моделювати та планувати заходи з запобігання водній та вітряній ерозії ґрунтів, а також запобігати їх деградації.

Туризм – можуть використовувати для прокладання оптимальних маршрутів, дослідження місцевості для визначення місць створення видових майданчиків.

Гідрографія – допомагають визначати межі басейнів та їх площі, моделювати ситуації затоплення внаслідок танення снігу або повеней, а також аналізувати глибини морського та річкового дна для забезпечення безпечного судноплавства.

Міське планування – використовуються для вертикального планування територій та побудови профілю рельєфу з метою вирівнювання ґрунтів для будівництва, проектування доріг, а також для розробки нових підземних комунікацій. Крім того, вони допомагають оцінювати обсяги робіт під час ремонту або будівництва, вибирати місця забудови, розробляти проєкти з облаштування фундаментів та забезпечення протизсувних заходів.

Археологія – використовуються для планування місць археологічних розкопок, аналізу розташування археологічних об'єктів, реконструкції історичних ландшафтів, а також для виявлення можливих стародавніх поселень.

1.2 Аналіз засобів моделювання рельєфу в сучасних ГІС та базах геопросторових даних

Аналіз засобів моделювання рельєфу в сучасних ГІС та базах геопросторових даних є актуальним і популярним напрямком досліджень, оскільки ці технології відіграють важливу роль у багатьох галузях.

Інструментарій для роботи з цифровими моделями рельєфу (ЦМР) надзвичайно широкий, і сучасні програмні рішення пропонують великий спектр функцій для їх створення, обробки та аналізу. В таблиці 1.2 наведено кілька популярних програмних продуктів, які широко використовуються для роботи з ЦМР, та їх основний функціонал для моделювання, аналізу та візуалізації рельєфу.

Таблиця 1.2 Функції підтримки ЦМР в сучасних ГІС та PostGIS[7]

Функції	**ArcGIS**	**MapInfo**	**Autodesk Map 3D**	**GeoMedia**	**QGIS**	**PostGIS**
Підтримка векторної моделі із Z-координатою	+	-	+	+	+	+
Підтримка регулярної моделі	+	+	+	+	+	+
Підтримка тріангуляційної моделі	+	+	+	-	+	+
Введення даних із геодезичних приладів	+	-	+	-	+	+
Фототріангуляція	+	-	+	+	+	-
Автоматична векторизація	+	+	+	+	+	-
Тривимірна візуалізація	+	+	+	-	+	+
Інтерполяція висот	+	+	-	+	+	+
Побудова профілів	+	+	+	+	+	+
Побудова ізоліній	+	+	-	+	+	+
Розрахунок експозиції схилів	+	+	-	-	+	+
Розрахунок об’ємів земляних робіт	-	+	+	-	+	+
Аналіз видимості	+	+	+	-	+	+
Побудова тальвегів та водорозділів	+	-	+	-	+	+

Однією з найпотужніших серед інструментальних ГІС для моделювання рельєфу є ArcGIS, яка забезпечує широкий спектр інструментів для роботи з тривимірними моделями поверхонь. Одним із ключових елементів цього

функціоналу є нерегулярні тріангуляційні мережі (TIN). TIN забезпечує точне моделювання рельєфу на основі тріангуляції Делоне, що дозволяє створювати мережу трикутників для відображення складних поверхонь землі. Така триангуляція мінімізує кількість "гострих" трикутників, що підвищує точність моделей рельєфу.

TIN будується на основі точкових даних, які можуть включати природні та штучні елементи рельєфу, такі як дороги, водні об'єкти чи гірські хребти. Інтеграція різних джерел даних, таких як точки, лінії та полігони, дозволяє створювати гнучкі моделі з високим рівнем деталізації. Крім того, ArcGIS підтримує фіктивну тріангуляцію Делоне, що допомагає зберігати конкретні геометричні елементи без зайвої деталізації, оптимізуючи обробку даних.

Однак, основною перевагою ArcGIS є terrain-набори даних, які дозволяють моделювати рельєф будь-якої складності та масштабу, враховуючи всі сучасні методи отримання даних. Terrain-набори можуть зберігати та обробляти різноманітні типи даних, зокрема Point cloud, растри, лінії розриву та GRID, що дозволяє будувати TIN як частину комплексного набору даних. Ці набори забезпечують високу продуктивність і ефективність при роботі з великими обсягами інформації.

Terrain-набори дозволяють використовувати пірамідну структуру для оптимізації даних, що забезпечує автоматичну зміну рівня деталізації залежно від масштабу або області аналізу. Це дає змогу працювати з великими територіями та швидко адаптувати модель до потреб користувача. Крім того, terrain-набори зберігають усі вихідні дані, дозволяючи зберігати повну інформацію про джерела даних та забезпечуючи гнучкість у їхньому використанні.

ArcGIS дозволяє виконувати складні аналітичні операції, такі як розрахунок нахилів, водозборів, об’єму земляних робіт та моделювання різних сценаріїв використання землі. Завдяки тому, що terrain-набори можуть зберігати та інтегрувати великі обсяги даних з різних джерел, цей інструмент є одним із найефективніших для комплексного аналізу рельєфу.

Останні публікації в контексті теми роботи, які було опрацьовано, можна розподілити за такими напрямам:

класифікація моделей рельєфу [3 - 5], [28];

удосконалення методів побудови рельєфу [1-5], [9], [27];

аналіз методів зберігання даних [13], [20], [27];

стандартизація моделей даних та функцій їх створення і використання [3, 4], [7], [11], [18], [23];

реалізація функцій створення та використання TIN моделей в середовищі СКБД [11], [13], [14], [20];

приклади практичного використання [1], [5], [8], [11], [27].

1.3 Структура, завдання та обмеження проекту

Структура роботи (рис. 1.6) відповідає послідовності етапів її виконання. Логічність структури роботи доводиться тим, що викладка матеріалу йде лаконічно і послідовно. В першому розділі послідовно викладено поняття по об’єкту дослідження (цифрова модель рельєфу) та його розуміння в науковій літературі. Після чого наводиться його класифікація та його сфери застосування із аналізом сучасного стану цифрового моделювання рельєфу із його тенденціями.

В другому розділі розглядається моделювання рельєфу в базах геоданих. Для початку наводяться уніфіковані класи геопросторових об’єктів і типи даних, що використовуються для моделювання рельєфу. Після чого детально розкривається GRID та TIN-моделі, їх особливостях, методах обробки та прикладах використання в системах керування базами даних (СКБД). Gодається структурно-функціональна схема ГІС моделювання рельєфу, що ілюструє взаємозв’язок компонентів системи.

У третьому розділі подаються вхідні дані використані в роботі з переліком використаних програмних засобів, потім надається технологічна схема опрацювання GRID-моделі в середовищі СКБД з подальшим опрацюванням етапів шо були визначені. Після відбулась реалізація прикладних SQL запитів для створення 3D моделей ліній використовуючи

SRTM модель. Та в кінці побудова опрацювання різних типів TIN моделі з реконструкцією окремої частини моделі, інтерполяцією на TIN-моделі за 2D точкою та 3D побудовою лінії на TIN.

Розділ 1	1.1	Основні визначення, класифікація та сфери використання цифрових моделей рельєфу	*Поняття про ЦМР*
	1.2	Аналіз засобів моделювання рельєфу в сучасних ГІС та базах геопросторових даних	*Сучасний стан використання ЦМР*
	1.3	Структура, завдання та обмеження проекту	*Структура роботи*
Розділ 2	2.1	Уніфіковані класи геопросторових об'єктів і типи даних для моделювання рельєфу в БГД	*Типи даних для моделювання рельєфу*
	2.2	Моделі і засоби опрацювання і аналізу GRID-моделі рельєфу в середовищі СКБД	*Засоби обробка GRID-моделей рельєфу*
	2.3	Типи даних і засоби побудови TIN-моделі рельєфу в середовищі СКБД	*Засоби обробки TIN-моделей рельєфу*
	2.4	Структурно-функціональна схема ГІС моделювання рельєфу з використанням СКБД	*Схема ГІС моделювання рельєфу*
Розділ 3	3.1	Вихідні дані, програмні засоби та стислий опис обчислювального експерименту на дослідну територію	*Вхідні дані та засоби їх опрацювання*
	3.2	Результати використання GRID-моделей рельєфу в середовищі СКБД PostgreSQL/PostGIS	*Опрацювання GRID-моделі рельєфу*
	3.3	Реалізація прикладної SQL функцій для побудови 3D моделей ліній з використанням GRID моделі рельєфу	*Реалізація SQL функцій для 3D моделей ліній*
	3.4	Створення і використання TIN-моделі рельєфу в середовищі СКБД PostgreSQL/PostGIS	*Опрацювання TIN-моделі рельєфу, реалізація прикладних SQL функцій*
Висновки		Аналіз результатів	*Загальні висновки*
Додатки	Б	Тести функцій на PL/pgSQL	*Розроблені функції*

Рис. 1.6 Структура роботи з результатами її етапів

Обмеження проекту:

обчислювальні експерименти приведено на територію міста Києва, рельєф якого за морфологічними характеристиками швидше за все можна класифікувати як рівнинний;

як вхідні дані для створення TIN моделі рельєфу на територію міста Києва, використано набір ізоліній рельєфу з просторовим розрізненням масштабу 1:200 000, що пояснюється відсутністю у відкритому доступі наборів даних більшої точності;

в усіх розроблених функціях на мові PL/pgSQL використовується статична структура експериментальної бази даних, що пояснюється часовими обмеженням на їх реалізацію, а їх розробка виконувалася з метою перевірки коректності пропонованих алгоритмів та реалізованих прикладних SQL функцій.

Висновки до розділу 1

В сучасних інструментальних ГІС надається широкий спектр засобів для створення і використання цифрових моделей рельєфу як у форматах регулярних сіток типу GRID, так і форматах полігонального покриття, зокрема нерегулярної мережі трикутників на основі класичної тріангуляції Делоне та тріангуляції Делоне з обмеженнями. Найбільш комплексні рішення щодо цифрового моделювання рельєфу із використанням TIN моделі надаються в програмних засобах ArcGIS. В документації ArcGIS [12], зокрема, зазначено, що TIN моделі зазвичай використовуються для моделювання невеликих областей з дуже високою точністю, наприклад в інженерних застосунках, де їх використання дозволяє провести обчислення планіметричної площі, площі поверхні та об'єму.

Максимально допустимий розмір TIN змінюється в залежності від наявних неперервних ресурсів пам'яті системи. При нормальних умовах роботи, наприклад в операційній системі Windows, максимальний розмір мережі TIN складає 10 – 15 мільйонів вузлових вершин. Незалежно від цього,

настійно рекомендується обмежити розміри кількома мільйонами з метою зручності використання та збільшення продуктивності системи. Більші розміри TIN краще всього реалізуються в ArcGIS за допомогою набору даних Terrain, в якій TIN може мати більш високе розрізнення (роздільну здатність) в тій частині, де поверхня вкрай нерівномірна або потрібна більша деталізація, і нижче розрізнення в місцях з однорідною поверхнею.

Проблематика функціональних можливостей та ефективності побудови цифрових моделей рельєфу в середовищі СКБД, зокрема в PostgreSQL/PostGIS, залишається малодослідженою, хоча спостерігається загальна тенденція до застосування технології систем баз даних для усіх типів даних, як важливого напряму розвитку ГІС та їх інтеграції із компонентами з інформаційних систем різного призначення в середовищі інфраструктури геопросторових даних та хмарних обчислень.

Розділ 2

МЕТОДИЧНІ ЗАСАДИ МОДЕЛЮВАННЯ РЕЛЬЄФУ В СЕРЕДОВИЩІ БАЗ ГЕОПРОСТОРОВИХ ДАНИХ

2 МЕТОДИЧНІ ЗАСАДИ МОДЕЛЮВАННЯ РЕЛЬЄФУ В СЕРЕДОВИЩІ БАЗ ГЕОПРОСТОРОВИХ ДАНИХ

2.1 Уніфікація класів геопросторових об’єктів і типів даних для моделювання рельєфу в БГД

У сучасному світі системи керування базами даних (СКБД) все більше використовуються для зберігання, аналізу та моделювання рельєфу, що вимагає їхньої уніфікації для забезпечення інтероперабельності, а також швидкого та ефективного обміну даними. Уніфікація геопросторових даних є основним завданням Міжнародної організації зі стандартизації (ISO) та Open Geospatial Consortium (OGC). У цьому розділі стисло розглянуто основні стандарти моделювання рельєфу, що були розроблені цими організаціями.

Стандарт ISO 19125 гармонізований із специфікаціями OGC Simple Feature Access (SFA), які визначають модель даних та інтерфейси для представлення простих геометричних об'єктів, таких як точки, лінії та полігони, у середовищах баз даних. Загальна архітектура SFA заснована на UML та нейтральна до платформи розподілених обчислень. Крім того, вона реалізує профіль просторової схеми, описаної в ISO 19107, забезпечуючи узгодженість між різними підходами до моделювання геометрії в ГІС.

ISO 19125-1 "Доступ до простих просторових об'єктів – Частина 1: Загальна архітектура" (Simple feature access - Part 1: Common architecture) встановлює загальну архітектуру для географічної інформації та визначає терміни, що використовуються в межах цієї архітектури. Також цей стандарт стандартизує назви та геометричні визначення для типів геометрії, зокрема ієрархію класів типу Geometry, визначену в ISO 19125-1:2004 та OGC (рис. 2.1).

Geometry (Геометрія) є основним базовим класом для всіх геометричних об'єктів, які використовуються в моделюванні просторових даних. Він визначає базові характеристики, якими можуть бути доповнені його підкласи. Від Geometry успадковуються такі основні класи:

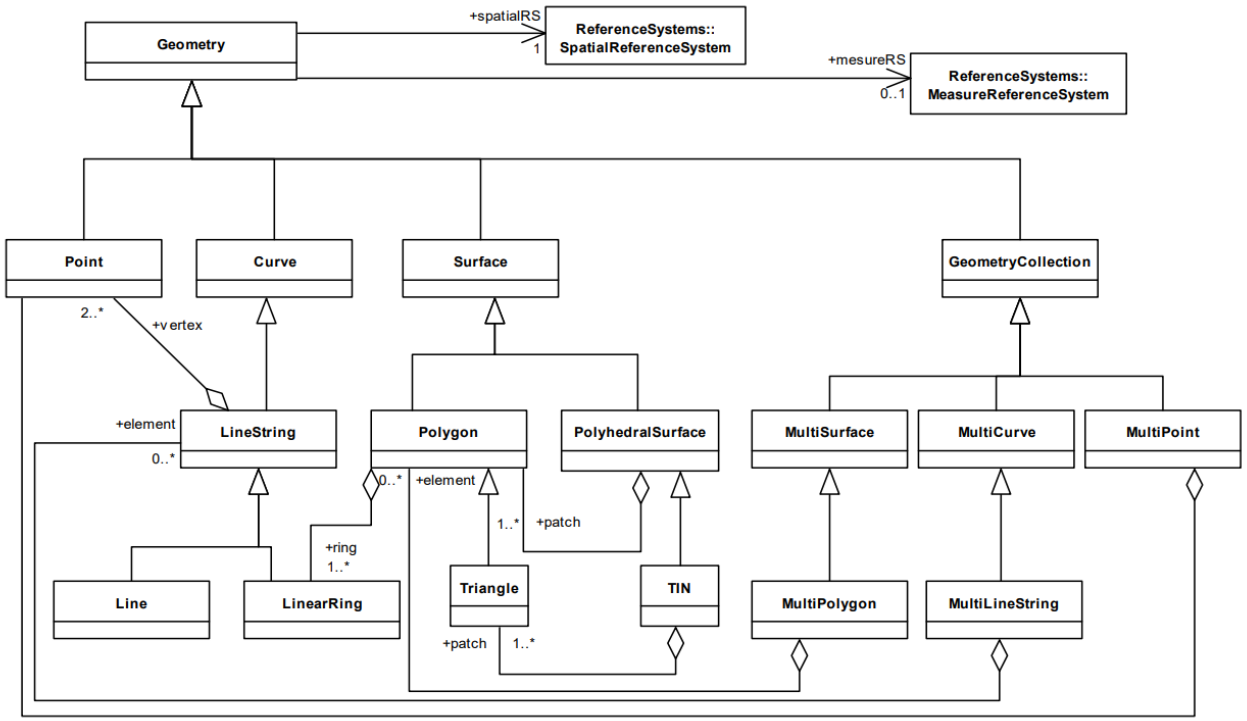


Рис. 2.1 Ієрархія класів геометрії згідно ISO 19125-1 і OGC [23]

Point (Точка): Використовується для представлення окремих точок у просторі, наприклад, місцезнаходження будівлі або дерева.

Curve (Крива): Представляє лінійні об'єкти, такі як дороги або річки. Основним підкласом є LineString, який дозволяє моделювати лінії, що складаються з послідовності точок. Line та LinearRing є специфічними варіантами LineString.

Surface (Поверхня): Використовується для моделювання двовимірних об'єктів, таких як території або озера. Підкласом Surface є Polygon, який описує замкнуті області. Далі є підкласи PolyhedralSurface і TIN (нерегулярна тріангуляційна мережа), які дозволяють представляти складні поверхні за допомогою набору полігонів або трикутників.

Класи MultiPoint, MultiCurve (наприклад, MultiLineString), MultiSurface (наприклад, MultiPolygon) дозволяють працювати з множинами точок, кривих або поверхонь як з єдиним об'єктом, що спрощує аналіз і зберігання комплексних даних.

В ISO 19125-2 «Географічна інформація. Доступ до простих просторових об'єктів - SQL опція» визначено вимоги до реалізації доступу до просторових об'єктів на основі SQL. Ця частина стандарту є логічним продовженням ISO

19125-1, який встановлює загальну архітектуру для роботи з просторовими об’єктами. В ISO 19125-2 визначено функції для роботи з геометричними об’єктами в середовищі SQL баз даних, що дозволяють виконувати просторові обчислення та аналіз. Серед основних функцій можна виділити ті, що дозволяють перевіряти просторові відношення між об’єктами (ST_Intersects, ST_Within), обчислювати відстані (ST_Distance), площі (ST_Area), або створювати буферні зони (ST_Buffer). Такі функції дозволяють ефективно використовувати просторові дані для вирішення практичних завдань, наприклад, у містобудуванні або управлінні природними ресурсами.

ISO 19125-2 також описує функції побудови об’єктів із складною геометрією, наприклад, з використанням функцій ST_Union для об’єднання об’єктів або ST_Difference для обчислення різниці між геометричними елементами об’єктів. Це відкриває можливості для просторового аналізу, де потрібно об'єднувати або розділяти території.

Стандарт ISO/IEC 13249-3 SQL/MM Spatial розширює класи просторових об’єктів для моделювання криволінійних об’єктів. зокрема, таких як ST_CircularString для моделювання дуг і ST_CompoundCurve для побудови геометричних об'єктів, що містять прямолінійні та криволінійні сегменти. Це дозволяє моделювати геометрії з більшою точністю і деталізацією, особливо в ситуаціях, коли необхідно працювати із об’єктами складної форми, наприклад, шляхи, що включають криві ділянки або дугоподібні форми.

Цей стандарт також розширює можливості для перевірки геометрій та перетворення координат, що дозволяє працювати з різними системами координат та забезпечує коректне просторове вирівнювання об’єктів. Підтримка тривимірних поверхонь, зокрема через класи ST_PolyhedralSurface та ST_TIN, робить SQL/MM Spatial ефективним інструментом для аналізу і моделювання тривимірних рельєфів та інших складних поверхонь. Структура класів геометрії за стандартом ISO/IEC 13249-3:201x(E) (2013) (рис. 2.2) відображає ієрархію класів, що включає підтримку як простих, так і складних геометричних об’єктів.

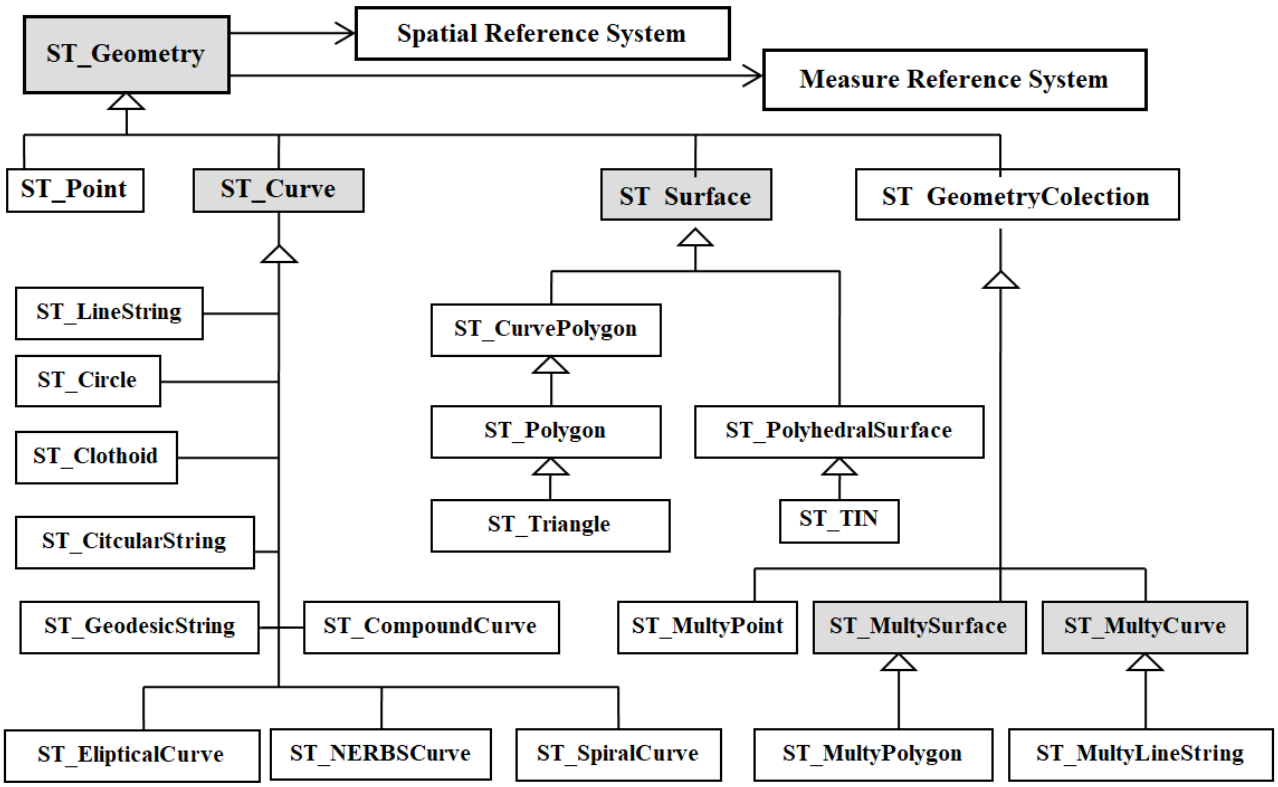


Рис. 2.2. Схема ієрархії класів типу Geometry за ISO/IEC 13249-3:201x(E) [18]

ISO 19107 Геометричні об'єкти та їх застосування в моделюванні рельєфу стандарт, що визначає геометричні об'єкти для просторового моделювання, включаючи їх структуру, атрибути та взаємозв'язки. Він забезпечує стандартизацію роботи з геометричними об'єктами, дозволяючи ефективно моделювати різні географічні особливості, що, своєю чергою, поліпшує сумісність даних та взаємодію між різними геоінформаційними системами (ГІС).

TIN розглядається як геометричний об'єкт, який може бути представлений векторними даними з зазначенням просторових характеристик, таких як координати та граничні лінії. TIN є абстракцією реального світу, пов'язаною з моделюванням рельєфу, що дозволяє точно описати топографію земної поверхні.

Згідно зі стандартом ISO 19107, просторові характеристики TIN виражаються через геометричні об'єкти. Головний клас для геометрії в цьому стандарті — GM_Object, який є абстрактним і описує набір об'єктів, що мають значення просторових атрибутів для опису географічних характеристик.

Екземпляри класу GM_Object можуть бути лише підтипами, такими як GM_Surface і GM_Point.

GM_Primitive — це абстрактний підклас GM_Object, що визначає геометричні об'єкти, які не містять своїх граничних точок, тобто вони є топологічно відкритими. Наприклад, поверхні підкласу GM_Primitive не включають свої граничні криві.

GM_Surface є основою для двовимірної геометрії, яка може бути побудована як у двовимірному, так і у тривимірному просторі координат. GM_Surface визначає лише орієнтовані поверхні, і орієнтація поверхні вибирає "верхній" напрямок — той, з якого зовнішня межа проходить проти годинникової стрілки. На основі GM_Surface побудовано інші підкласи, такі як GM_SurfacePatch та GM_Polygon, що описують однорідні частини поверхні та поверхні з граничними кривими.

Ієрархія класів, що описують поверхні та їх підкласи, зокрема GM_Surface, GM_Polygon, GM_PolyhedralSurface та GM_Tin, наведена на рис. 2.3

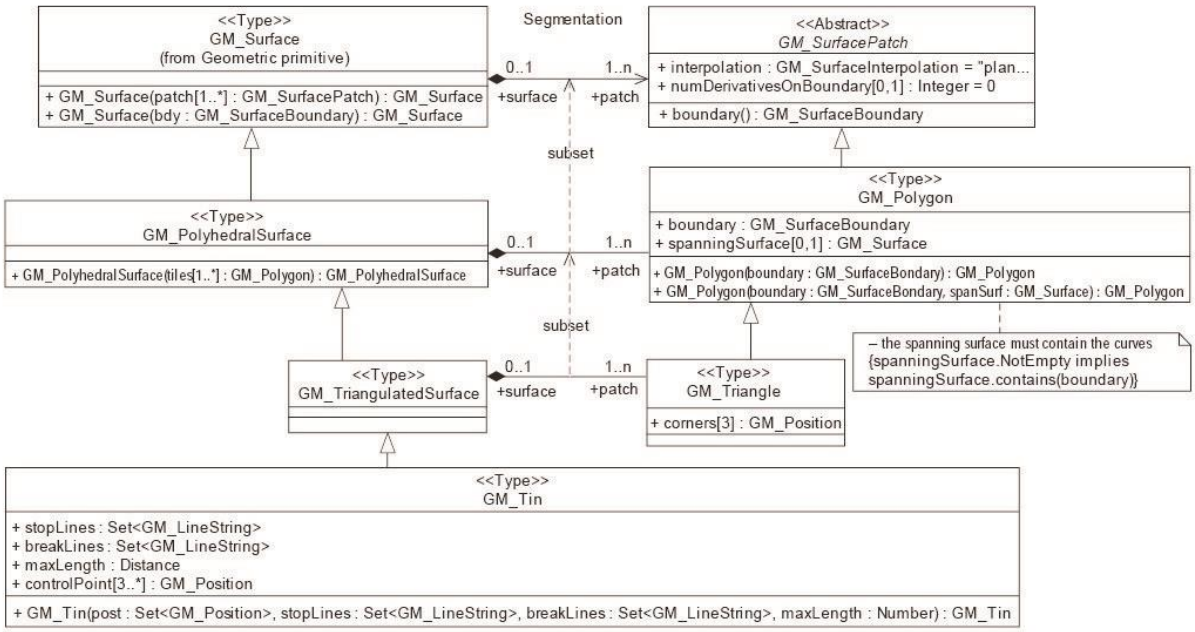


Рис. 2.3. Схема класів поверхонь і їх підкласів у стандарті ISO 19107 [4]

ISO 19123 Схема геометрії та функцій покриття визначає концептуальну схему для просторових об'єктів типу покриття, які моделюють географічні поля — явища реального світу, що неперервно змінюються в просторі. Покриття є

ключовою структурою даних у багатьох прикладних сферах, таких як дистанційне зондування Землі, метеорологія, батиметрія, картографування висот, ґрунтів, рослинності та інших природних ресурсів.

Термін покриття (coverage) походить із специфікацій OGC і стосується будь-якої структури даних, яка надає значення певної характеристики безпосередньо в точці простору. Покриття виступає як просторовий об'єкт та функція, яка визначає зв'язок між точками в просторі та доменом атрибутів для певного тематичного змісту. Прикладами покриттів є: растри (регулярні сітки), нерегулярні тріангуляційні мережі (TIN), точкові та полігональні структури.

Покриття може бути дискретним або неперервним. Дискретне покриття складається з геометричних об'єктів, які мають кінцеву кількість позицій. Кожен геометричний об'єкт відповідає єдиному запису атрибутів. Дискретне покриття відображає кожен об'єкт на певне значення, наприклад, тип ґрунту всередині полігона. В ISO 19123 визначено такі типи дискретного покриття:

CV_DiscretePointCoverage – набір нерегулярно розподілених точок та їх атрибутів.

CV_DiscreteCurveCoverage – покриття характеризується скінченною просторово-часовою областю, що складається з кривих, які можуть бути елементами мережі.

CV_DiscreteSurfaceCoverage – покриття складається з набору поверхонь.

CV_DiscreteSolidCoverage – дискретне об’ємне покриття складається з набору суцільних тіл, наприклад, океан або атмосфера можуть бути зображені у вигляді просторових об’ємів з діапазоном атрибутів, таких як температура і тиск у кожній вершині 3D-сітки.

CV_DiscreteGridPointCoverage – дискретна сітка як набір регулярно розподілених точок та їх атрибутів.

Неперервне покриття, навпаки, має область визначення, яка містить безліч позицій в координатному просторі. Воно дозволяє моделювати явища, які змінюються по всій площині або в об'ємі (наприклад, температурні поля чи рельєф). Для інтерполяції в ISO 19123 визначено такі методи:

CV_ThiessenPolygonCoverage – неперервне покриття полігонів Тіссена (діаграма локусів Вороного або теселяція (мозаїка) Дирихле), полігони покриття в ISO 19123 розглядають лише в двовимірному просторі.

CV_ContinuousQuadrilateralGridCoverage – неперервне покриття сіткою прямокутників.

CV_HexagonalGridCoverage – гексагональні сіткові покриття на основі правильних шестикутників.

CV_TINCoverage – покриття, утворене нерегулярною мережею трикутників (TIN), побудованих зазвичай за критерієм Делоне, яке може допускати лінійну й криволінійну інтерполяцію на елементі трикутної форми.

CV_SegmentedCurveCoverage – криволінійні сегментовані покриття використовуються для моделювання явищ, які змінюються по кривих, що можуть бути елементами мережі.

TIN, або нерегулярна тріангуляційна мережа, у ISO 19123 визначається як GM_Tin (рис. 2.5), що є GM_TriangulatedSurface, побудованою з використанням алгоритму Делоне або подібного алгоритму з урахуванням ліній розриву (breaklines), стоп-ліній (stoplines) і максимальної довжини сторін трикутників. На рис. 2.4 наведено процес створення TIN, що включає кроки додавання стоп-ліній та подальшої триангуляції з урахуванням ліній розриву.

Алгоритм Делоне забезпечує, що для кожного трикутника у мережі коло, що проходить через його вершини, не містить всередині жодних вершин інших трикутників. У випадку, якщо трикутники перетинають стоп-лінії або перевищують максимальну довжину сторони, ці трикутники видаляються з поверхні, утворюючи таким чином отвори на поверхні TIN (рис. 2.5).

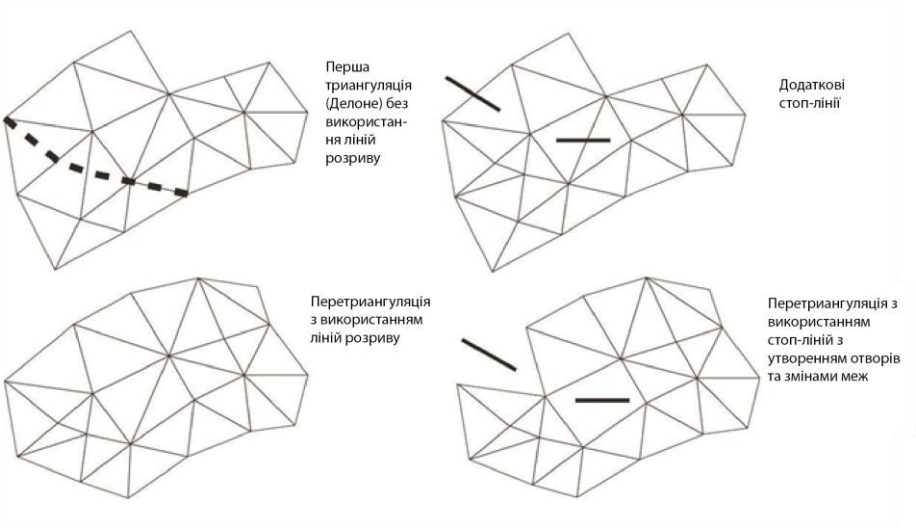


Рис. 2.5. Схема побудови TIN моделі згідно із ISO 1923 [3]

2.2 Моделі і засоби опрацювання і аналізу GRID-моделі рельєфу в середовищі СКБД

GRID-модель рельєфу є однією з найпоширеніших структур для представлення та аналізу цифрових моделей рельєфу (ЦМР) у геоінформаційних системах (ГІС) та системах керування базами даних (СКБД). Її реалізація в PostgreSQL/PostGIS базується на використанні типу даних **raster**. Нижче представлено узагальнену класифікацію растрових моделей рельєфу в ГІС.(рис. 2.6).

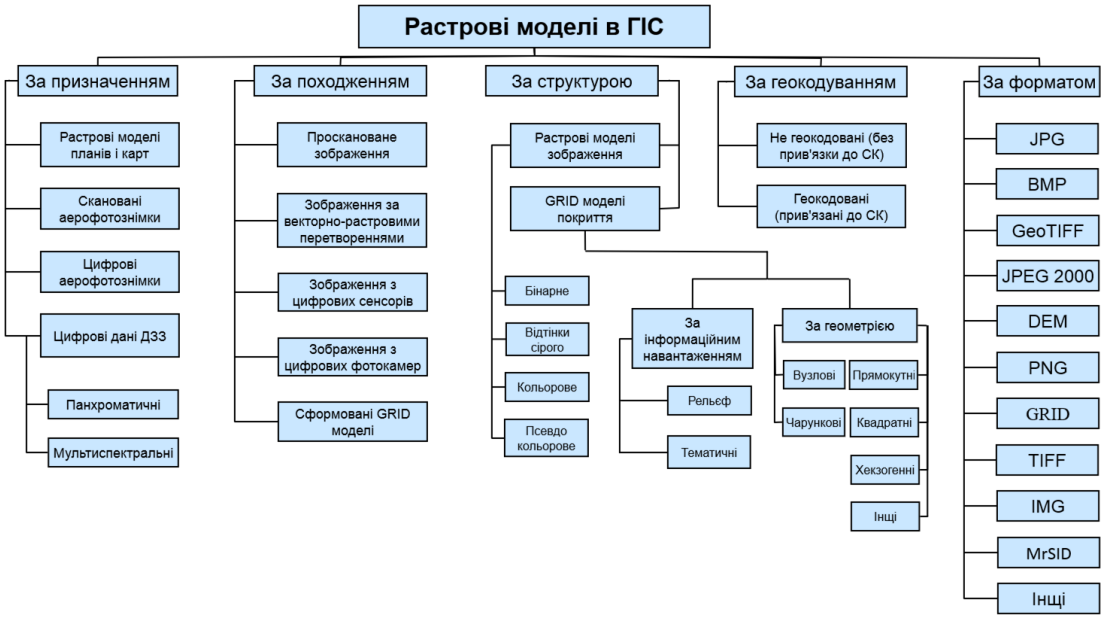


Рис. 2.6. Узагальнена класифікації растрових моделей[10]

У PostgreSQL/PostGIS растрові дані можна уявити як двовимірну матрицю або масив, де кожен осередок (піксель) має певне значення, яке

відображає фізичну величину. Щоб уникнути роботи з великими масивами даних одразу і зменшити навантаження на систему, виконується розбивка растрової моделі на блоки (тайли). Це прискорює доступ до конкретних частин растру й підвищує продуктивність системи.

При розбитті растру на блоки важливим параметром є крок растру (resolution), який визначає фізичний розмір одного пікселя. Крок може варіюватися в залежності від вимог до точності аналізу: наприклад, 10x10 метрів на піксель або 30x30 метрів.

Для зберігання блоків растру в базі даних створюється окрема регулярна таблиця, де кожен запис представляє окремий тайл растру. Кожен блок містить метадані (рис. 2.7), що дозволяють коректно обробляти та аналізувати дані. До таких метаданих належать: система координат (SRID), крок на піксель, перекоси (Δx, Δy), які використовуються для обчислення тригонометричних функцій, а також значення NULL для клітинок, в яких відсутні дані.

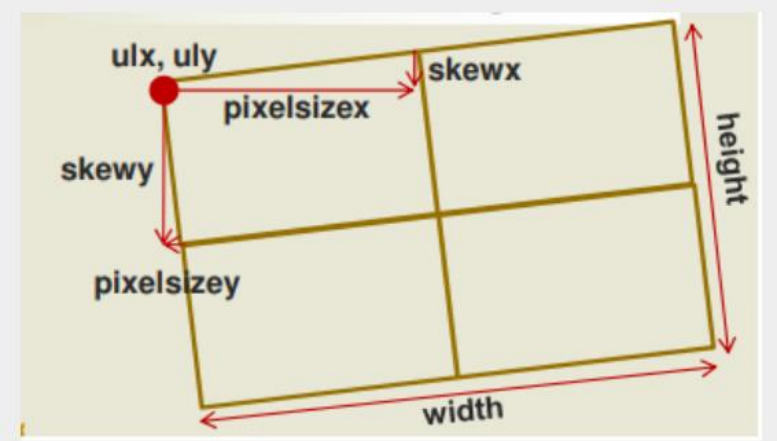


Рис. 2.7 Структура растрового блоку з параметрами перекосів та розмірів пікселів

PostgreSQL/PostGIS підтримує широкий набір функцій для роботи з растровими моделями, які вирішують різні завдання, пов'язані з аналізом рельєфу. Було розроблено UML-діаграму функцій опрацювання GRID-моделей у середовищі PostgreSQL/PostGIS (рис. 2.8).

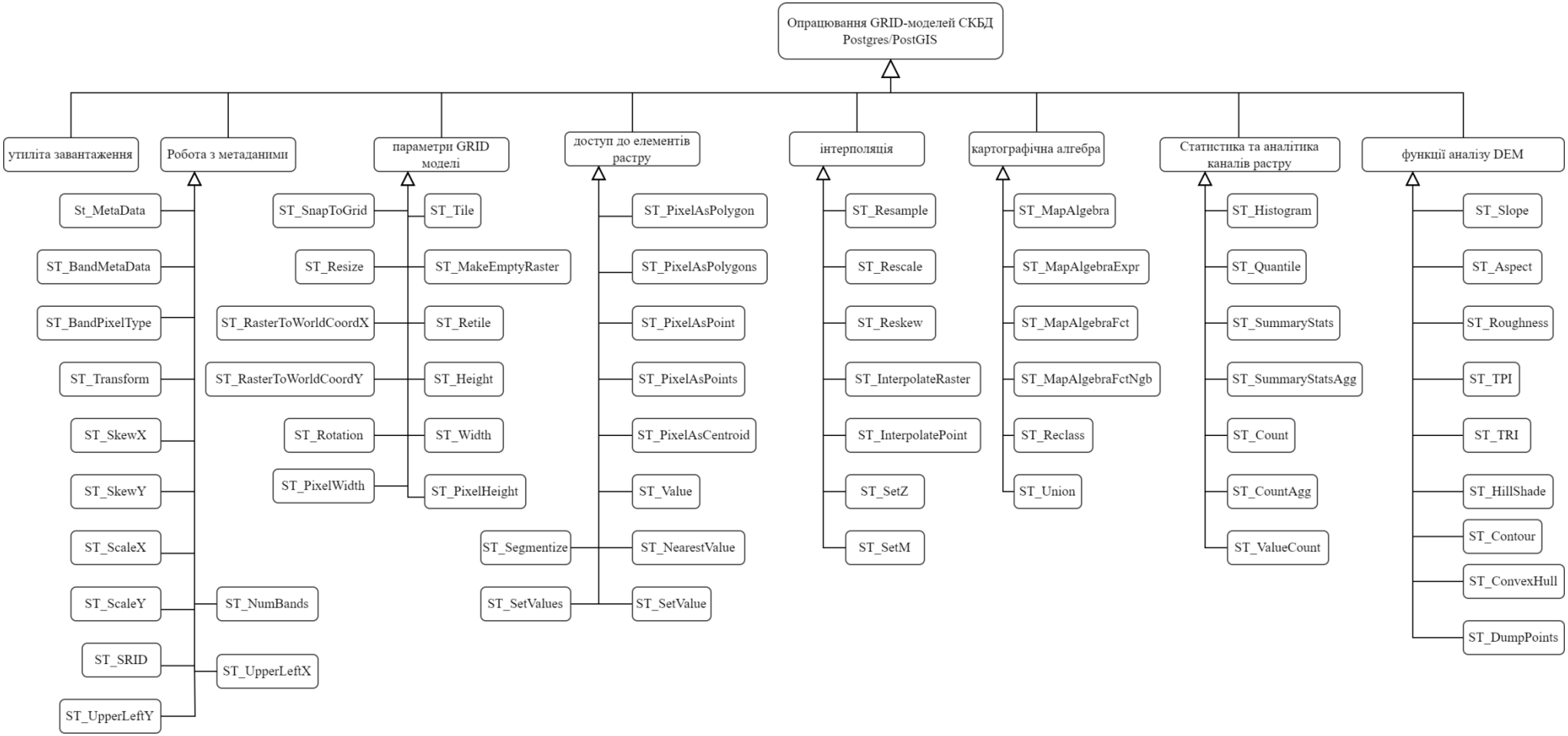


Рис. 2.8. Засоби опрацювання GRID-моделей в середовищі PostgreSQL/PostGIS

Розглянемо докладніше призначення функцій опрацювання растрових моделей за категоріями.

Функції роботи з метаданими (табл. 2.1) забезпечують доступ до загальної інформації про растр та його просторову прив’язку до референцної системи координат.

Таблиця 2.1. Функції роботи з метаданими[24]

Функції /PostGIS	Опис функції
ST_MetaData	Повертає базову метаінформацію про растровий об'єкт, таку як розмір пікселя, обертання (нахил), верхній, нижній лівий кут тощо. Повернені колонки: upperleftx | upperlefty | width | height | scalex | scaley | skewx | skewy | srid | numbands
ST_ScaleX/ ST_ScaleY	Повертають відповідно компоненти X та Y розміру пікселя в одиницях системи координат. У версіях WKTRaster вони мали назви ST_PixelSizeX для ширини пікселя та ST_PixelSizeY для висоти пікселя.
ST_SkewX/ ST_SkewY	Повертає X/Y-компонент нахилу сітки растру (або параметр обертання) географічної прив'язки
ST_SRID	Повертає ідентифікатор просторової прив'язки для ST_Geometry, визначений у таблиці spatial_ref_sys. Таблиця spatial_ref_sys каталогізує всі відомі просторові системи координат у PostGIS та використовується для перетворення між різними системами координат.
ST_UpperLeftX/ ST_UpperLeft Y	Повертає X/Y-координату верхнього лівого кута растра у проєктованій системі координат.

Функції доступу до параметрів GRID-моделі забезпечують доступ до таких параметрів як розміри растру, його нахил, ширина та висота пікселів.

Функції доступ до елементів растру (табл. 2.2) забезпечують доступ до окремих пікселів або їхніх характеристик.

Таблиця 2.2. Функції доступ до елементів растру[24]

Функції PostGIS	Опис функції
ST_Value	Повертає набір записів зі стовпцями value та count, які містять значення піксельної смуги та кількість пікселів у растровій плитці або покритті вибраної смуги. Якщо смуга (nband) не вказана, за замовчуванням використовується перша смуга. Якщо не вказані значення для пошуку (searchvalues), буде повернено всі знайдені значення пікселів у растрі або покритті. Якщо задано одне значення для пошуку, повертається ціле число, яке вказує кількість пікселів з цим значенням піксельної смуги.
ST_Segmentize	Повертає змінену геометрію/географію, яка не містить жодного сегмента, довжина якого перевищує max_segment_length. Довжина обчислюється у 2D. Сегменти завжди розбиваються на під сегменти однакової довжини. Для геометрії максимальна довжина задається в одиницях просторової системи координат. Для географії максимальна довжина задається в метрах. Відстані обчислюються на сфері, а додані вершини створюються вздовж сферичних дуг великих кіл, визначених кінцевими точками сегментів.
ST_PixelAsPoints	Повертає точкову геометрію для кожного пікселя растрового шару разом зі значенням, X та Y координатами кожного пікселя. Координати точкової геометрії відповідають верхньому лівому куту пікселя. Формат повернення запису: geom (тип — geometry), val (тип — double precision), x (тип — integer), y (тип — integer).
ST_PixelAsCentroid	Повертає центроїд (точкова геометрія) області, представленої пікселем.

Функції інтерполяції (табл.2.3) призначені для обчислення нових значень на основі існуючих даних, зокрема заповнення прогалин або уточнення даних.

Таблиця 2.3 Функції інтерполяції[24]

Функції PostGIS	Опис функції
ST_Resample	Модифікує растр із використанням зазначеного алгоритму ресемплінгу, нових розмірів (ширина та висота), кута сітки (gridx та gridy) та набору атрибутів геоприв'язки растру (scalex, scaley, skewx та skewy), які визначені або запозичені з іншого растру. Якщо використовується референцний растр, обидва растри повинні мати однаковий SRID. Нові значення пікселів обчислюються за допомогою одного з наступних алгоритмів ресемплінгу: NearestNeighbor, Bilinear, Cubic, CubicSpline, Lanczos, Max, Min.
ST_InterpolateRaster	Інтерполює ґраткову поверхню на основі вхідного набору тривимірних точок, використовуючи X- та Y-значення для позиціонування точок на сітці, а Z-значення точок як висоту поверхні. Доступно п'ять алгоритмів інтерполяції: обернена відстань, обернена відстань з найближчим сусідом, ковзне середнє, найближчий сусід і лінійна інтерполяція. Вхідні параметри: • input_points — точки для керування інтерполяцією. Допустимі будь-які геометрії з Z-значеннями, всі точки з вхідних даних будуть використані. • algorithm_options — рядок, що визначає алгоритм та його параметри у форматі, який використовується gdal_grid. • template — растровий шаблон для керування геометрією вихідного растру. Ширина, висота, розмір пікселя, просторове охоплення та тип пікселя будуть зчитані з цього шаблону. • template_band_num — за замовчуванням для керування вихідним растром використовується перша смуга шаблонного растру, але цей параметр можна налаштувати.
ST_InterpolatePoint	Повертає інтерпольоване значення вимірювання для лінійної виміряної геометрії в точці, найближчій до заданої.
ST_SetZ	Повертає геометрію з тими ж координатами X/Y, що й вхідна геометрія, і значеннями із растра як координату Z, обчислену з використанням алгоритму, вказаного в параметрі ресемплінгу: на "nearest" - для копіювання значень із комірки, у яку потрапляє кожна вершина; "bilinear" - для використання білінійної інтерполяції, яка враховує значення сусідніх комірок.
ST_SetM	Повертає геометрію з тими ж координатами X/Y, що й вхідна геометрія, і значеннями з растра, скопійованими у M-вимір з використанням вказаного алгоритму ресемплінгу. Параметр ресемплінгу можна встановити на "nearest" для копіювання значень із комірки, у яку потрапляє кожна вершина, або на "bilinear" для використання білянійної інтерполяції, яка враховує значення сусідніх комірок.

Функції картографічної алгебра (табл. 2.4) призначені для змінювати значення пікселів на основі математичних виразів або рекласифікації растру для подальшого аналізу.

Таблиця 2.4. Функції картографічної алгебри[24]

Функції PostGIS	Опис функції
ST_MapAlgebra	Версія з функцією зворотного виклику — повертає однодіапазонний растр на основі одного або кількох вхідних растрів, індексів смуг і заданої користувачем функції зворотного виклику.
ST_MapAlgebra	Версія з виразом — повертає однодіапазонний растр на основі одного або двох вхідних растрів, індексів смуг та одного або кількох заданих користувачем SQL-виразів.
ST_Reclass	Створює новий растр шляхом застосування валідної алгебраїчної операції PostgreSQL, визначеної через reclassexpr, до вхідного растру (rast). Якщо смуга не вказана, за замовчуванням використовується смуга 1. Новий растр матиме таку ж геоприв'язку, ширину та висоту, як і початковий растр. Смуги, які не були вказані, залишаться без змін. Дивіться reclassarg для опису валідних виразів рекласифікації. Смуги нового растру матимуть тип пікселя, визначений через pixeltype. Якщо передано reclassargset, то кожен reclassarg визначає поведінку кожної згенерованої смуги.
ST_Union	Повертає об'єднання набору растрових плиток у єдиний растр, що складається як мінімум з однієї смуги. Область отриманого растру відповідає області всього набору. У випадку перетину значення, що утворюється, визначається параметром uniontype, який може мати одне з наступних значень: LAST (за замовчуванням), FIRST, MIN, MAX, COUNT, SUM, MEAN, RANGE.

Функції статистики растрів (табл. 2.5) призначені для статистичного аналізу растру, зокрема для побудови гістограм розподілу значень пікселів та показників стандартної статистики (мінімальні максимальні та середні значеннях, стандартне відхилення тощо).

Таблиця 2.5. Функції статистики растру[24]

Функції PostGIS	Опис функції
ST_Histogram	Повертає набір записів, що складаються з мінімального (min), максимального (max) значення, кількості (count) та відсотка (percent) для заданої растрової смуги для кожного інтервалу. Якщо смуга (nband) не вказана, за замовчуванням використовується перша смуга.
ST_Quantile	Обчислює квантилі для растру або покриття растра у контексті вибірки або генеральної сукупності. Таким чином, значення може бути оцінене на рівні 25%, 50% або 75% перцентилю растру.
ST_SummaryStats	Повертає підсумкову статистику (summarystats), яка включає кількість (count), суму (sum), середнє (mean), стандартне відхилення (stddev), мінімальне (min) та максимальне (max) значення для заданої растрової смуги растру або покриття растру. Якщо смуга (nband) не вказана, за замовчуванням використовується перша смуга.
ST_Count	Повертає кількість пікселів у заданій смузі растру або покриття растру. Якщо смуга (nband) не вказана, за замовчуванням використовується перша смуга. Якщо exclude_nodata_value встановлено на true, будуть враховані лише пікселі, значення яких не дорівнює значенню nodata растру. Встановіть exclude_nodata_value на false, щоб врахувати всі пікселі.

Функції аналізу GRID моделі рельєфу (табл. 2.6) призначені для опрацювання растру як дискретної моделі рельєфу та обчислення таких морфометричних характеристик рельєфу як: нахили та орієнтацію схилів, освітлюваність та топографічний індекс.

Таблиця 2.6. Функції аналізу GRID моделі рельєфу [24]

Функції PostGIS	Опис функції
ST_Slope	Повертає нахил (за замовчуванням у градусах) для висотної смуги растру. Використовує алгебру карт і застосовує рівняння нахилу до сусідніх пікселів. units вказує одиниці вимірювання нахилу. Можливі значення: RADIANS, DEGREES (за замовчуванням), PERCENT. scale — це коефіцієнт, що визначає співвідношення вертикальних одиниць до горизонтальних. Для Feet:LatLon використовуйте scale=370400, для Meters:LatLon використовуйте scale=111120. Якщо interpolate_nodata встановлено в TRUE, значення для пікселів з NODATA у вхідному растрі будуть інтерпольовані за допомогою ST_InvDistWeight4ma перед обчисленням нахилу поверхні.

Продовження таблиці 2.6. Функції аналізу GRID моделі рельєфу

ST_Aspect	Повертає експозицію (за замовчуванням у градусах) для висотної смуги растру. Використовує алгебру карт і застосовує рівняння експозиції до сусідніх пікселів. units вказує одиниці вимірювання експозиції. Можливі значення: RADIANS, DEGREES (за замовчуванням). Якщо units = RADIANS, значення знаходяться в діапазоні від 0 до 2 * π радіан, виміряні за годинниковою стрілкою від півночі. Якщо units = DEGREES, значення знаходяться в діапазоні від 0 до 360 градусів, виміряні за годинниковою стрілкою від півночі. Якщо нахил пікселя дорівнює нулю, експозиція пікселя дорівнює -1.
ST_Roughness	Обчислює "шорсткість DEM, віднімаючи мінімальне значення від максимального для заданої області.
ST_HillShade	Повертає гіпотетичне освітлення для висотної смуги растру з використанням вхідних параметрів азимуту, висоти (альтитуди), яскравості та масштабу. Використовує алгебру карт і застосовує рівняння затінення рельєфу до сусідніх пікселів. Значення пікселів на виході знаходяться в діапазоні від 0 до 255. azimuth — значення від 0 до 360 градусів, виміряні за годинниковою стрілкою від півночі. altitude — значення від 0 до 90 градусів, де 0 градусів — на горизонті, а 90 градусів — безпосередньо над головою. max_bright — значення від 0 до 255, де 0 означає відсутність яскравості, а 255 — максимальна яскравість. scale — коефіцієнт, що визначає співвідношення вертикальних одиниць до горизонтальних. Для Feet:LatLon використовуйте scale=370400, для Meters:LatLon використовуйте scale=111120. Якщо interpolate_nodata встановлено на TRUE, значення для пікселів із NODATA у вхідному растрі будуть інтерпольовані за допомогою ST_InvDistWeight4ma перед обчисленням затінення рельєфу.
ST_ConvexHull	Обчислює опуклу оболонку геометрії — це найменша опукла геометрія, що охоплює всі геометрії у вхідних даних. Опуклу оболонку можна уявити як геометрію, що отримується при обгортанні набору геометрій гумкою. Це відрізняється від ввігнутої оболонки, яка аналогічна "усадці" навколо геометрій. Опуклу оболонку часто використовують для визначення зони впливу на основі набору точкових спостережень. У загальному випадку опукла оболонка — це полігон. Опукла оболонка для двох або більше колінеарних точок є лінійною геометрією (LineString) з двома точками. Опукла оболонка для однієї або більше ідентичних точок є точкою (Point). Це не є агрегатною функцією. Для обчислення опуклої оболонки набору геометрій використовуйте ST_Collect для їх об'єднання в колекцію геометрій

Продовження таблиці 2.6. Функції аналізу GRID моделі рельєфу

ST_TRI	Індекс шорсткості рельєфу (Terrain Ruggedness Index) обчислюється шляхом порівняння центрального пікселя з його сусідами, обчислення абсолютних значень різниць і обчислення середнього результату.
ST_TPI	Обчислює топографічний індекс позиції, який визначається як фокальне середнє з радіусом один мінус центральна комірка.
ST_Contour	Генерує набір векторних контурів ізоліній для вхідної растрової смуги, використовуючи алгоритм контурів GDAL. Якщо параметр fixed_levels містить ненульовий масив, параметри level_interval та level_base ігноруються. Вхідні параметри: ➢ rast: Растр, для якого генерується контур. ➢ bandnumber: Номер каналу, для якого генерується контур. ➢ level_interval: Інтервал висоти між згенерованими контурами. ➢ level_base: "База", відносно якої застосовуються контурні інтервали. Зазвичай це нуль, але може бути іншим значенням. Наприклад, щоб згенерувати контури через 10 м на висотах 5, 15, 25, ... значення LEVEL_BASE буде 5. ➢ fixed_levels: Інтервал висоти між згенерованими контурами. ➢ polygonize: Якщо true, будуть створені контурні полігони замість лінійних контурів. Повертаються записи з такими атрибутами: ➢ geom: Геометрія контурної лінії. ➢ id: Унікальний ідентифікатор, наданий контурній лінії GDAL. ➢ value: Значення растрового каналу, яке представляє лінія. Для вхідного DEM (цифрової моделі висот) це буде висота згенерованого контуру.

Для завантаження растрових моделей у PostGIS призначена утиліта raster2pgsql. Це засіб, що перетворює растрові файли на в образ SQL-інструкцій для створення растрових таблиць і завантаження даних у PostGIS. Він підтримує роботу з різними форматами растрових даних, що забезпечуються бібліотекою GDAL, і може створювати огляди растрів. Докладніше функції та параметри утиліти raster2pgsql розглянуто в розділі 3.

2.3 Типи даних і засоби побудови TIN-моделі рельєфу в середовищі СКБД

2.3.1 Типи даних для TIN моделі рельєфу

Для забезпечення побудова TIN моделі рельєфу в середовищі PostgreSQL/PostGIS ведені спеціальні типи даних зокрема TIN та Triangle, які є спеціалізованими підтипами геометрії, призначеними для моделювання рельєфу та інших тривимірних поверхонь.

TIN є колекцією суміжних трикутників, які моделюють тривимірну нерегулярну поверхню покриття. Цей тип геометрії дозволяє моделювати рельєфні структури та інші природні об'єкти з високою точністю, оскільки забезпечує гнучке розташування точок, що краще відповідає нерегулярному характеру природних форм, ніж регулярні сітки.

Triangle визначає трикутник як полігон з трьома різними неколінеарними вершинами, який використовується для представлення простих елементів поверхні. Він є базовою геометричною фігурою для створення складніших моделей.

Як було сказано різниця між ними закладається в тому, що Triangle просто набором трикутників, TIN може подаватися як колекція трикутників, що являється тривимірною нерегулярною моделлю покриття. Кожна з моделей TIN створюється тільки один запис в середовищі СКБД PostgreSQL/PostGIS.

У середовищі СКБД PostgreSQL/PostGIS реалізовані такі інструменти для роботи з TIN: ST_DelaunayTriangles, ST_ConstrainedDelaunayTriangles та ST_TriangulatePolygon.

2.3.2 Функція тріангуляції Делоне

Функція *ST_DelaunayTriangles* обчислює тріангуляцію Делоне для вхідного набору вершин геометрії 2D або 3D точок. Тріангуляція Делоне забезпечує, що жодна точка не лежить всередині кола, описаного навколо жодного з трикутників, що дозволяє створювати максимально рівносторонні трикутники. Це мінімізує ймовірність створення "вузьких" або "довгих" трикутників, що часто важливо для аналізу даних, моделювання поверхонь і

створення цифрових моделей рельєфу. Параметр tolerance дозволяє коректувати точність, об'єднуючи близькі точки, що покращує стабільність розрахунків. Геометрія результату обмежена опуклою оболонкою вхідних точок — найменшою опуклою областю, що охоплює всі точки.

Стандартне подання функцій

geometry: ST_DelaunayTriangles(geometry g1, float tolerance = 0.0, int4 flags = 0);

де:

1. geometry g1: представляє собою вхідний набір точок
2. float tolerance = 0.0: представляє мінімальну відстань між точками
3. int4 flags = 0: представляє визначає формат результату:

- 0 – GEOMETRYCOLLECTION трикутних полігонів (за замовчуванням);
- 1 – MULTILINESTRING, що представляє ребра тріангуляції;
- 2 – TIN (Тріангульована Нерегулярна Мережа).

Функція підтримує тривимірну (3D) геометрію, зберігаючи z-координати, що робить її корисною для моделювання тривимірних поверхонь, таких як цифрові моделі рельєфу. Підтримуються також трикутники та TIN, що може бути корисним для аналізу складних поверхонь, наприклад, змін рельєфу або гідрологічного аналізу.

Нижче наведено приклади використання функції ST_DelaunayTriangles для різних варіантів TIN, побудованих із різними параметрами для вхідного набору точок. Вхідний набір точок рис. 2.9

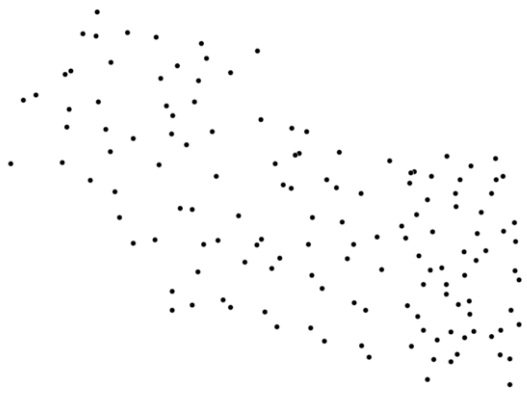


Рис. 2.9. Тестовий набір вхідних точок

Запит на створення TIN моделі рельєфу як колекції трикутних полігонів (GEOMETRYCOLLECTION=0)

```
CREATE TABLE tin0 AS WITH pts AS
(
SELECT geom AS pt
FROM point_test
)
SELECT ST_DelaunayTriangles(ST_Union(pt::geometry), 0.0, 0) AS the_geom
FROM pts;
```

Результат подання запиту на створення тріангуляції Делоне як колекції трикутних полігонів TIN моделі подано на рис. 2.10.

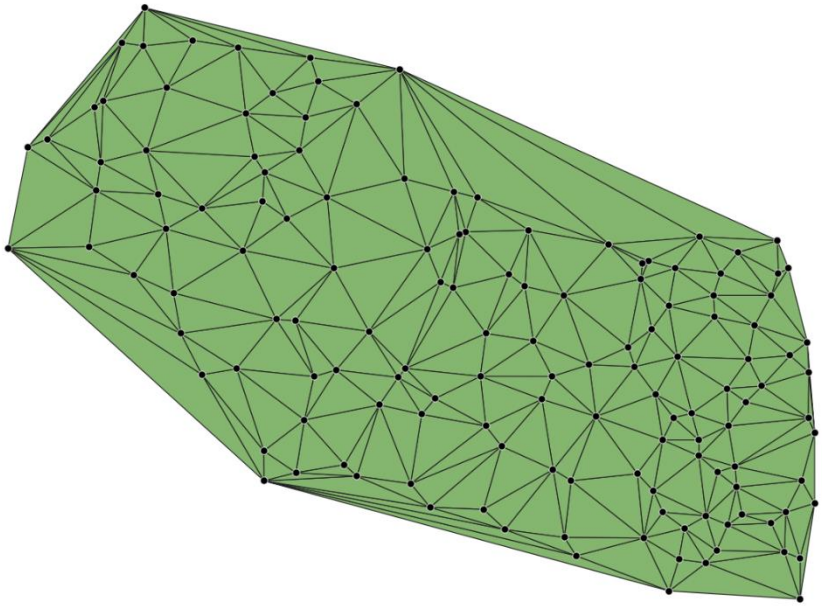


Рис. 2.10. Відображення результату запиту на створення тріангуляції Делоне як колекції трикутних полігонів TIN моделі

Запит на створення TIN моделі рельєфу як набору ребр тріангуляції (MULTILINESTRING=1):

```
CREATE TABLE tin1 AS WITH pts AS
(
SELECT geom AS pt
FROM point_test
)
SELECT ST_DelaunayTriangles(ST_Union(pt::geometry), 0.0, 1) AS the_geom
FROM pts;
```

Результат запиту на створення тріангуляції Делоне як набору ребр тріангуляції подано на рис. 2.11

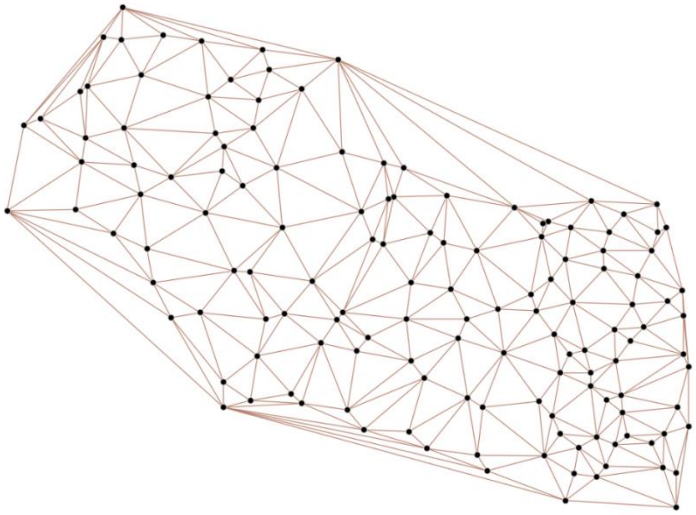


Рис. 2. 11. Відображення результату запиту на створення тріангуляції Делоне як набору ребр тріангуляції

Запит на створення TIN моделі рельєфу як тривимірної нерегулярної мережі (TIN=2):

```
CREATE TABLE tin2 AS WITH pts AS
(
SELECT geom AS pt
FROM point_test
)
SELECT ST_DelaunayTriangles(ST_Union(pt::geometry), 0.0, 2) AS the_geom
FROM pts;
```

Результат подання запиту на створення тріангуляції Делоне як тривимірної нерегулярної мережі подано на рис. 2.12.

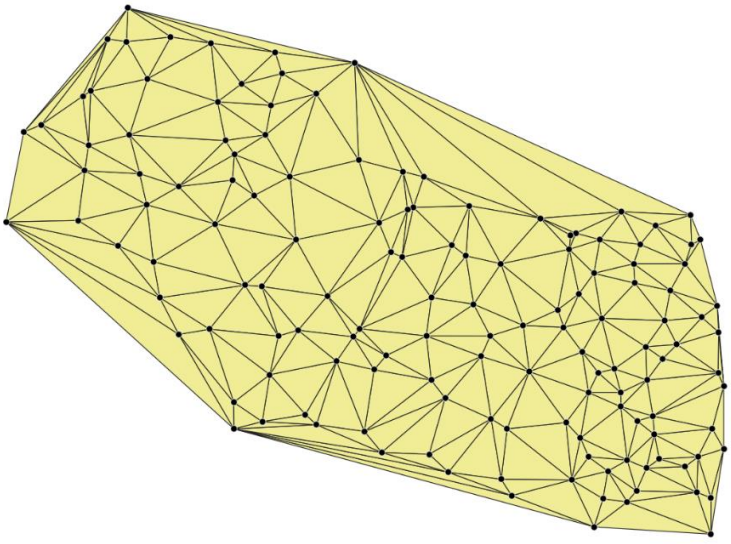


Рис. 2.12. Відображення результату запиту на створення тріангуляції Делоне як колекції трикутників типу TIN

Результатом кожного запиту став один запис у відповідних таблицях, стовпчик геометрів якої містить колекцію сукупності за відповідним типом даних трикутників, ребр або TIN.

2.3.3 Функція тріангуляції Делоне з обмеженнями

Функція ST_ConstrainedDelaunayTriangles призначена для побудови тріангуляції Делоне з урахуванням обмежень, що задаються структурними лініями або контурами полігонів. Обмеження полягає в тому, що відрізки структурних ліній і обмежувальних контурів полігонів обов'язково включаються як ребра тріангуляції, незалежно від того, чи виконуються для ребр трикутників тест описаного кола Делоне. Це робить тріангуляцію більш точною для моделювання геометричних об'єктів із врахуванням їх форми та зв'язків. Результат цієї функції представлений у вигляді TIN, що дозволяє точно моделювати та аналізувати тривимірні поверхні, наприклад, для створення цифрових моделей рельєфу. Функція підтримує тривимірні дані (z-координати). Запит на створення TIN з обмеженнями:

```
CREATE TABLE tin_const AS
WITH pts AS (
   SELECT geom AS pt FROM point_test),
lts AS (SELECT geom AS cl FROM gidro_line)
SELECT ST_ConstrainedDelaunayTriangles(ST_Collect(ST_Union(pt::geometry),
ST_Union(cl::geometry))) AS the_geom
FROM pts, lts;
```

Результат подання запиту подано на рис. 2.13 та рис. 2.14.

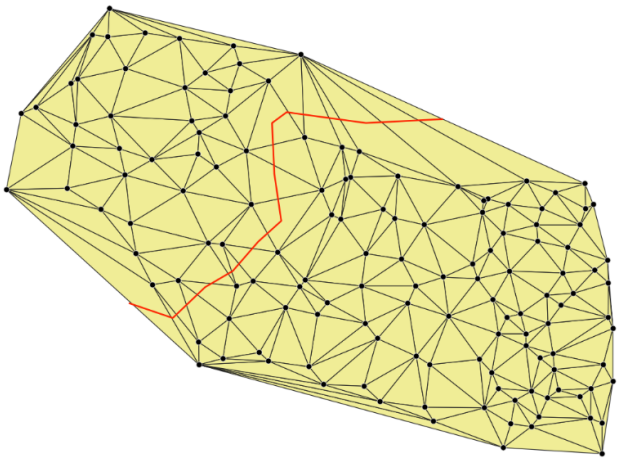


Рис. 2.13. Структурна лінія для обмеження тріангуляції Делоне

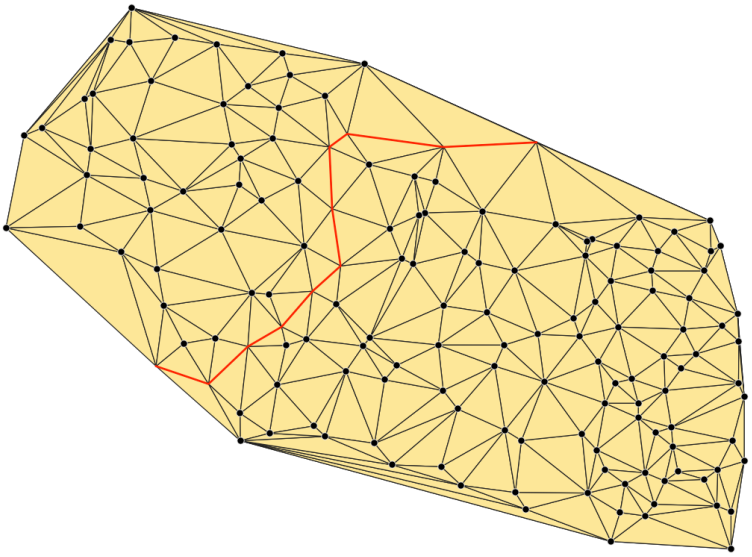


Рис. 2.14. Відображення результату запиту на тріангуляцію з обмеженнями, в якій відрізки структурної лінії увійшли до складу ребер TIN

2.3.4 Функції тріангуляції полігонів

Функція *ST_TriangulatePolygon* обчислює обмежену тріангуляцію Делоне для полігонів, враховуючи наявні отвори та мультиполігони. Це означає, що полігони можуть бути поділені на трикутники таким чином, щоб максимально враховувати форму та внутрішні області полігона. Обмежена тріангуляція Делоне — це набір трикутників, які формуються з вершин полігону та повністю покривають його, забезпечуючи максимальний внутрішній кут у всіх можливих тріангуляціях. Це гарантує "найкращу якість" тріангуляції полігону, роблячи результат придатним для точного аналізу та обчислення геометрії складних фігур. Ця функція особливо корисна для роботи з полігонами, що мають отвори або кілька компонентів, забезпечуючи коректний поділ на трикутники для подальшого моделювання. Запит на тріангуляцію полігону:

```
CREATE TABLE tin_poligon AS
WITH pts AS (
   SELECT geom AS pt
   FROM poligon
)
SELECT ST_TriangulatePolygon(pt) AS the_geom
FROM pts;
```

Результат запиту на тріангуляцію полігону подано на рис. 2.15.

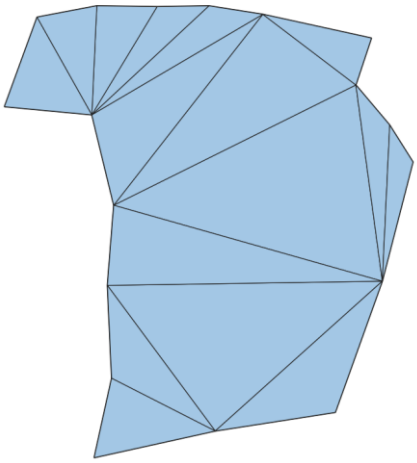


Рис. 2.15. Відображення результату запиту на створення тріангуляції полігону

2.4 Структурно-функціональна схема ГІС моделювання рельєфу з використанням СКБД.

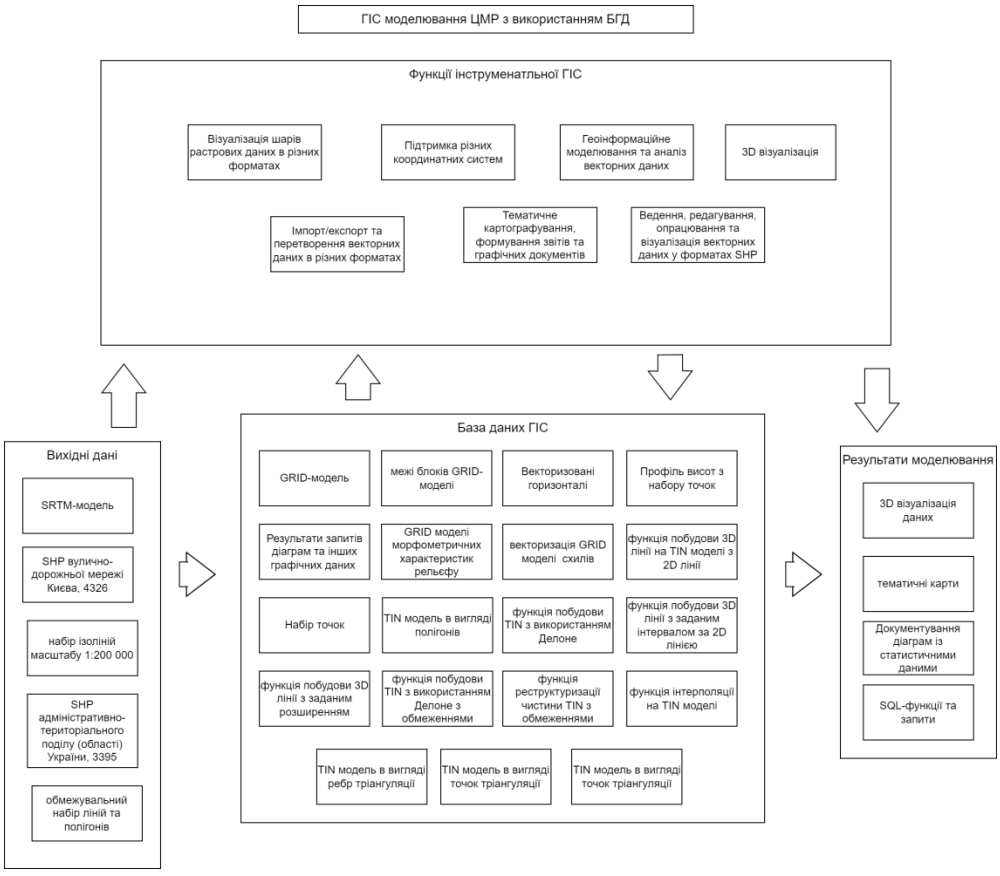


Рис. 2.14. Структурно-функціональна схема ГІС ЦМР з використанням БГД

Структурно-функціональна схема — це засіб графічного моделювання, що дозволяє зобразити складну систему, структуру та функції окремих її елементів у вигляді взаємопов'язаних блоків. Основною метою такої схеми є

створення чіткого уявлення про організацію системи: які основні складові вона включає, як вони взаємодіють між собою, та які функції виконує кожна з них. У контексті ГІС моделювання рельєфу, схема відображає взаємодію компонентів, таких як бази даних, програмне забезпечення для обробки геоданих, аналітичні модулі, та інтерфейси користувача, які спільно дозволяють моделювати рельєф та виконувати просторовий аналіз.

Висновки до розділу 2

Розглянуто уніфікацію класів геопросторових об’єктів і типів даних, які забезпечують сумісність та ефективність обміну геопросторовими даними у базах даних. В сучасних баз даних уніфіковані дані ґрунтуються на таких стандартах ISO 19125, ISO 19107, та ISO 19123, які визначають структури та моделі для представлення геометричних об’єктів, таких як точки, лінії та полігони.

Основні типи даних для побудови цифрових моделей рельєфу в PostGIS є типи даних *TIN* та *triangle* для TIN моделей та **raster** для регулярного покриття типу GRID.

В реалізації PostgreSQL і його розширеннях для GRID-моделі надаються прикладні функції для опрацювання даних, які забезпечують: роботу з метаданими, доступу до параметрів GRID-моделі, доступу до елементів растру, інтерполяції, операції картографічної алгебри, статистики растру.

Для TIN моделювання в середовищі PostgreSQL/PostGIS надаються три базові функції, які забезпечують побудову як класичної тріангуляції Делоне, так і тріангуляцію з обмеженнями та побудову TIN в середині полігонів.

Розроблена структурно-функціональна модель ГІС моделювання рельєфу з використанням СКБД PostgreSQL/PostGIS, реалізація якої забезпечить комплексне дослідження засобів створення та опрацювання як GRID-моделі, так і TIN моделі рельєфу з обмежувальною тріангуляцією включно.

Розділ 3

РЕЗУЛЬТАТИ МОДЕЛЮВАННЯ РЕЛЬЄФУ ДЛЯ ДОСЛІДНОЇ ТЕРИТОРІЇ В СЕРЕДОВИЩІ СКБД POSTGRESQL/POSTGIS

3 РЕЗУЛЬТАТИ МОДЕЛЮВАННЯ РЕЛЬЄФУ ДЛЯ ДОСЛІДНОЇ ТЕРИТОРІЇ В СЕРЕДОВИЩІ СКБД POSTGRESQL/POSTGIS

3.1 Вихідні дані, програмні засоби та стислий опис обчислювального експерименту на дослідну територію

В якості досліджуваної території було вибрано територію міста Київ з прилеглими до неї територіями. Як вхідні дані для обчислювальних експериментів на територію міста були використані такі набори даних:

1. Фрагмент SRTM-моделі
2. Шейп-файл адміністративних меж областей України, SRID 3395
3. Шейп-файл вулично-дорожньої мережі міста, SRID 4326
4. Набір ізоліній масштабу 1:200 000

SRTM-модель: Обраний фрагмент глобальної GRID-моделі рельєфу SRTM (Shuttle Radar Topography Mission) представляє цифрову модель поверхні, що покриває територію міста Київ з прилеглими до неї територіями (рис. 3.1 – 3.2). Один блок SRTM (N50E030 у випадку роботи) покриває територію розміром 1x1 градус, що містить 1201x1201 пікселів. Кожен піксель моделі представляє висотне значення з розрізненням 3 секунди дуги в системі відліку WGS84 для географічних координат та EGM96 GEOID для висотних значень. Для території міста Київ фізичний розмір пікселя становить приблизно 58.5 м по паралелі та 92.7 м по меридіану. Для країн Євразії, згідно з [17,26], абсолютна похибка по висоті становить 6,2 метри, а відносна помилка складає 8,7 метрів.

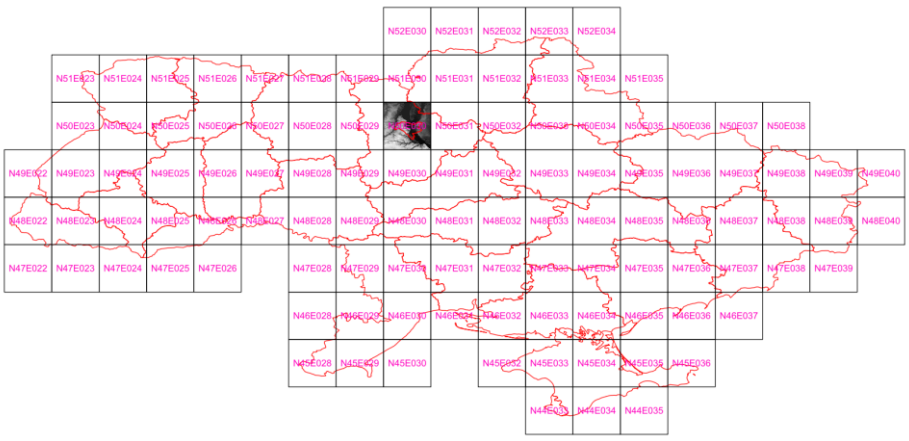


Рис. 3.1 Покриття SRTM моделі для всієї території України

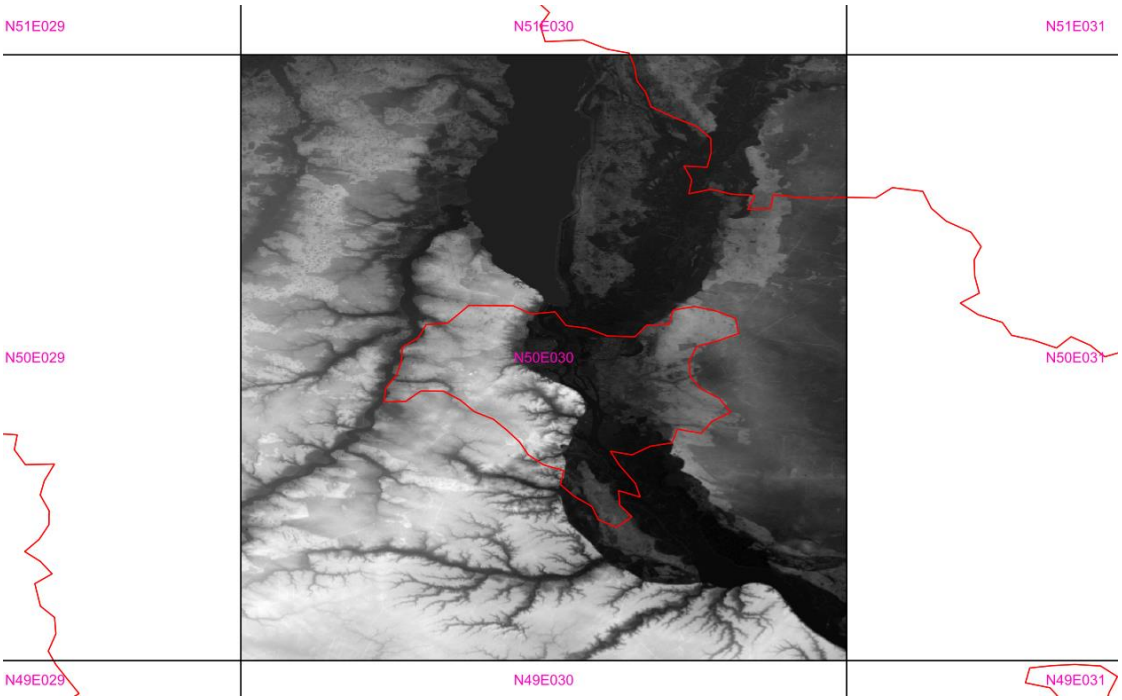


Рис. 3.2. Блок SRTM-моделі використаний в роботі

Шейп-файл адміністративних меж областей України у системі координат SRID 3395 (Рис. 3.3).

Рис. 3.3. Адміністративні межі областей території України

Набір ізоліній топографічної карти масштабу 1:200 000 у системі координат WGS 84 (SRID 4326) (рис. 3.4).

Рис. 3.4. Набір ізоліній в системі координат WGS 84 (SRID 4326)

Шейп-файл вулично-дорожньої мережі міста Києва вивантажений з OSM у системі координат WGS 84 (SRID 4326) (рис. 3.5).

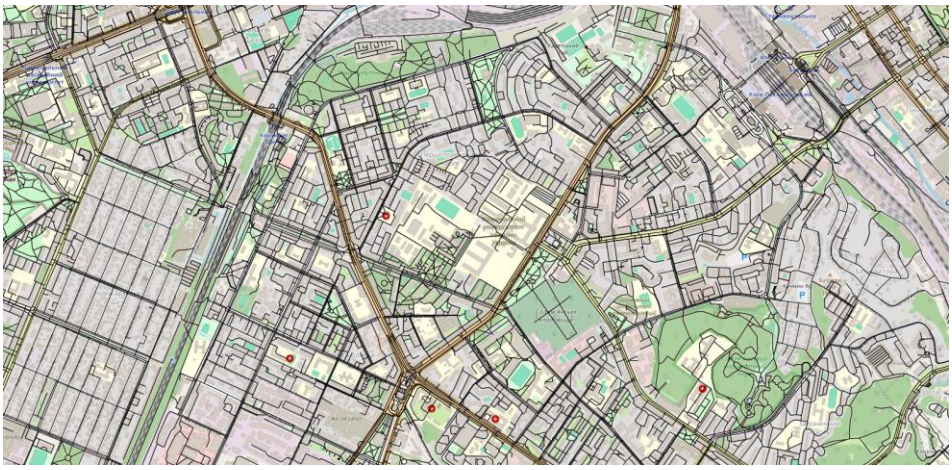


Рис. 3.5. Вулично-дорожня мережа міста Київ з OpenStreetMap

Експеримент виконувався в середовищі PostgreSQL 16 з використанням таких бібліотек: postgis версії 3.4.1, postgis_raster та таких програм:

1) pgAdmin 4 – інтерфейсна утиліта для інтерактивного доступу і керування базою даних PostgreSQL;

2) QGIS – інструментальна ГІС для візуалізації та аналізу геопросторових даних;

3) pgShapeLoader – утиліта для імпорту векторних даних формату Shapefile до бази даних PostgreSQL/PostGIS;

4) raster2pgsql – утиліта для завантаження растрових даних у PostgreSQL/PostGIS.

Моделювання GRID-моделі рельєфу в середовищі PostgreSQL/PostGIS виконувалося в такій послідовності:

1) ***завантаження шейп-файлів*** шарів вулично-дорожньої мережі та адміністративно-територіального поділу в БД з використанням *pgShapeLoader*;

2) ***завантаження SRTM моделі*** рельєфу в базу даних з використанням утиліти *raster2pgsql*;

3) ***створення рамок блоків растру*** з використанням функції ST_ConvexHull PostGIS для визначення межі кожного блоку растру *SRTM*;

4) ***3D-візуалізація SRTM растру*** в середовищі *QGIS*;

5) ***побудова ізоліній та їх 3D-візуалізація***: використання функції *ST_Contour* для створення таблиць ізоліній *contours* та *contours_u* з SRTM-моделі з її подальшою 3D візуалізацією в середовищі *QGIS*;

6) ***інтерполяція висот точок з побудовою профілю висот***: використання функції *ST_Value* для дослідження методів інтерполяції та функції *ST_Z* для створення таблиці *transect*, що містить точковий набір з інтерпольованими висотами точок профілю;

7) ***аналіз розподілу висот*** з використання функції *ST_Histogram* для створення тимчасового іменованого набору даних *hist*, що містить поділ по градаціям значень висот;

8) ***обчислення ухилів рельєфу*** з використанням функції *ST_Slope* для створення растру *dem_slope*, який містить значення нахилу для кожної чарунки растру поверхні;

9) ***обчислення орієнтації схилів*** **з** використанням функції *ST_Aspect* для створення растру *dem_aspect*, який містить орієнтацію схилів для кожної клітинки досліджуваної території;

10) ***обчислення гіпотетичного освітлення рельєфу*** **з** використанням функції *ST_HillShade* для створення растру *dem_hillshade*, який містить значення інтенсивності гіпотетичного освітлення для кожної клітинки досліджуваної території;

11) ***обчислення топографічного індексу поверхні*** з використанням функції *ST_TPI* для створення растру *dem_tpi*, який містить значення обчисленого TPI в кожній комірці моделі поверхні;

12) ***побудова векторної моделі ухилів рельєфу*** на основі рекласифікації растру *dem_slope* з використанням функції *ST_Reclass* та функції *ST_DumpAsPolygons* для автоматичної векторизації рекласифікованого растру та створенням таблиці векторної полігональної моделі ухилів *dem_reclassmpoly*.

3.2 Результати використання GRID-моделей рельєфу в середовищі СКБД PostgreSQL/PostGIS

3.2.1 Технологічні схеми дослідного використання GRID-моделі рельєфу

GRID-модель в середовищі СКБД PostgeSQL/PostGIS опрацьовувалась за технологічною схемою (рис. 3.6 – 3.7), яка складається з таких етапів:

А1) завантаження SRTM модель в СКБД за допомогою raster2pgsql;

А2) створення рамок для блоків растру з ST_ConvexHull;

А3) 3D візуалізація растрових блоків та растру в QGIS;

А4) побудова ізоліній на основі SRTM-моделі з ST_Contour та її 3D візуалізацією;

А5) інтерполяція висоти точки за допомогою ST_Value та створення профілю рельєфу з ST_Z;

А6) загальний аналіз рельєфу деталізовано на рис. 3.2 з етапами Б1-Б6:

Б1) побудови гістограми висот GRID-моделі;

Б2) обчислення ухилів рельєфу GRID-моделі;

Б3) обчислення орієнтацію схилів GRID-моделі;

Б4) обчислення гіпотетичного освітлення GRID-моделі;

Б5) обчислення топографічного індексу GRID-моделі;

Б6) проведення рекласифікації ухилів з перетворенням в векторну форму.

Докладніше зміст етапів та їх результати описано в пунктах 3.2.2 – 3.2.7.

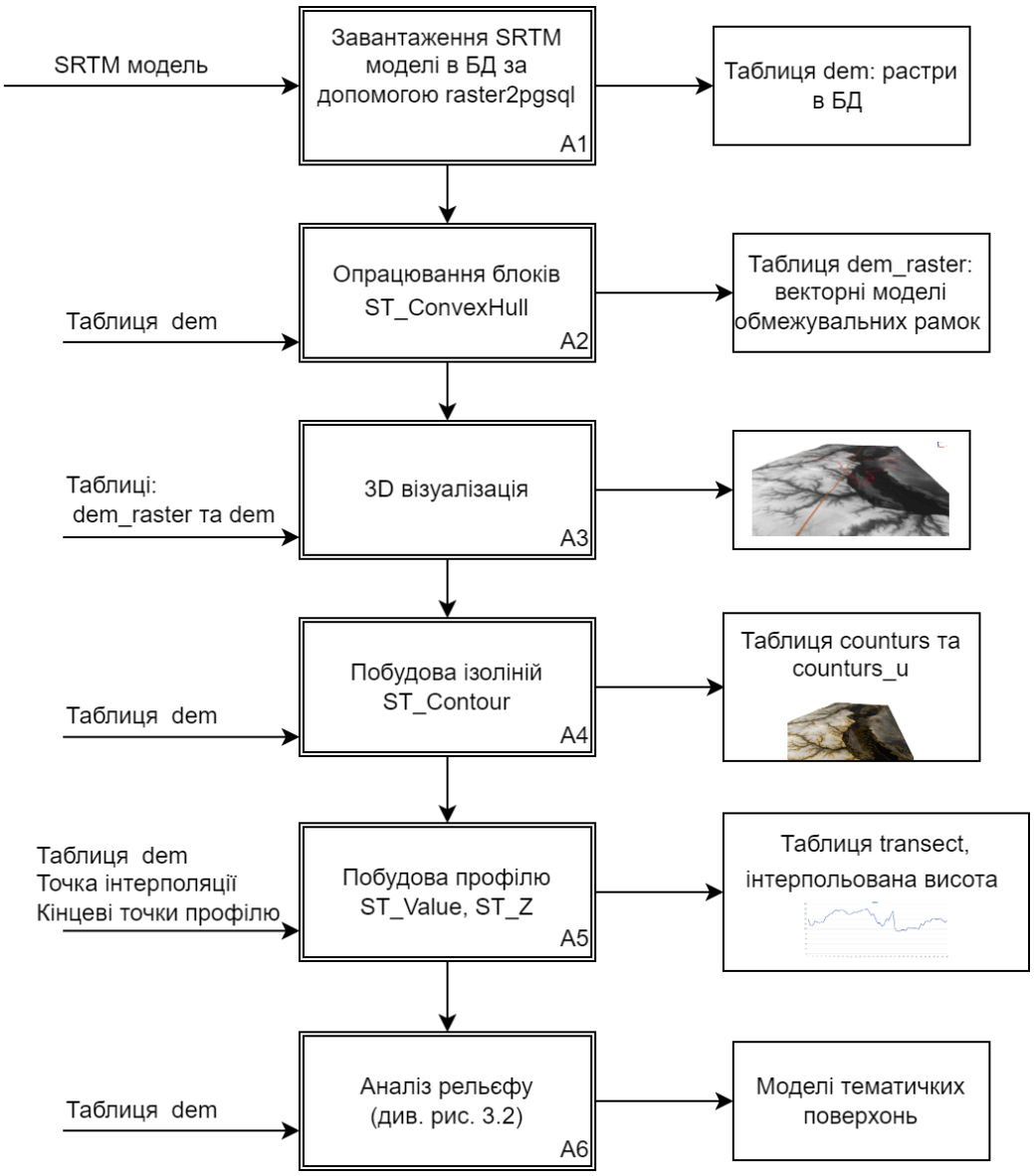


Рис. 3.6. Технологічна схема дослідницького використання GRID-моделі рельєфу

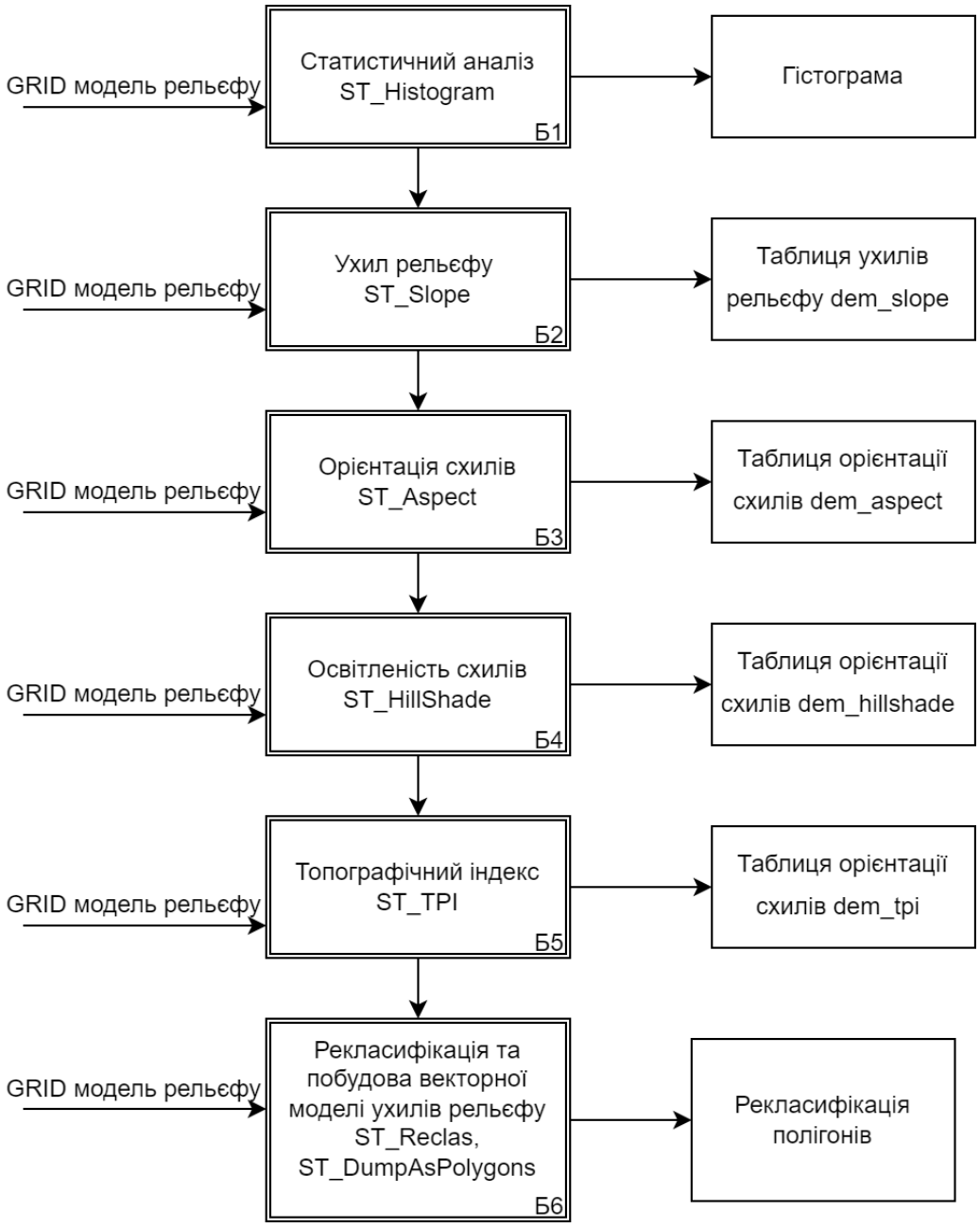


Рис. 3.7. Технологічна схема аналізу рельєфу з використанням GRID моделі

3.2.2 Завантаження GRID-моделі

Для завантаження растру в БД для попереднього опрацювання растрових моделей в середовищі PostgreSQL/PostGIS призначена утиліта **raster2pgsql**. Вона дозволяє конвертувати растр у SQL-інструкції, що забезпечують додавання, зберігання та подальшу обробку растрових даних безпосередньо в базі даних.. У наведеній технологічній схемі нижче (рис. 3.8) докладно процес завантаження та опрацювання растрових даних з використанням утиліти raster2pgsql.

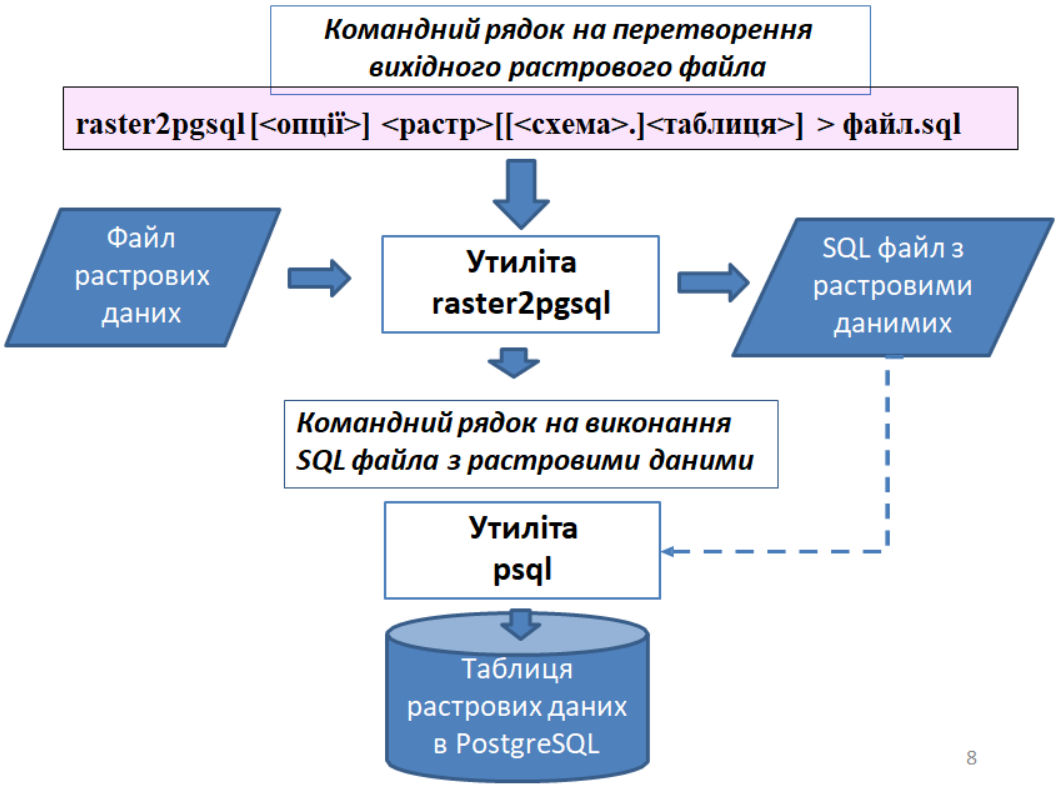


Рис. 3.8 Технологічна схема завантаження та обробки растрових даних у PostgreSQL/PostGIS за допомогою утиліти raster2pgsql [10]

Виклик та ведення параметрів для утиліти PostgreSQL здійснюється в режимі «командного рядка» операційної системи (рис. 3.9). Команда містить такі параметри:

<опції> — параметри для управління даними растру (параметри використані в роботі наведено в таблиці 3.1);

<растр> — шлях до файлу растрових даних, який необхідно завантажити.

<схема>.<таблиця> — назва схеми та таблиці, де буде збережено растр у базі даних.

Таблиця 3.1 Параметри використаних опцій завантаження растру

Ключ та параметри	Стислий зміст призначення ключів та значень параметрів
-s 4326	Призначити вихідному растра заданий SRID.
-C	Створити нову таблицю і заповнити її з растрами
-F	Додати стовпчик із назвою файла
-t 600x600	Розрізати растр на тайли розміром 600x600 пікселів, щоб вставити кожний в записи таблиці.

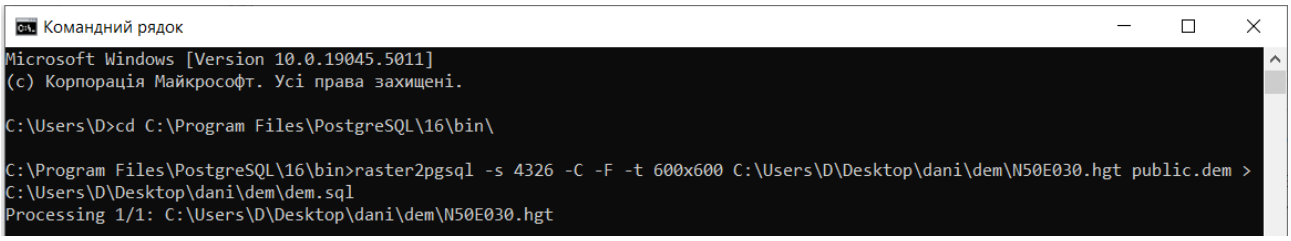


Рис. 3.9 Завантаження растрових даних у PostgreSQL за допомогою утиліти raster2pgsql

Результатом виконання утиліти *raster2pgsql* є файл із SQL описом растру, який містить послідовність операторів для створення таблиці бази даних та команди вставки в таблицю записів з блоками растру у відповідності до параметрів, визначених в командному рядку запуску утиліти. Завантаження та запуск на виконання команд SQL-файлу здійснюється в програмі pgAdmin 4, в якій реалізовано зручний інтерфейс інтерактивного доступу до сервера СКБД PostgreSQL (рис. 3.10). В результаті опрацювання SQL опису растру в базі даних створюється таблицю із записами блоків растру (рис. 3.11) із такими атрибутами:

1) *rid* (первинний ключ): ідентифікатор блоку растру;

2) *filename* (тип text): ім’я вихідного файлу SRTM як джерела даних для блоку растру;

3) *rast* (тип даних raster): набір значень пікселів блоку растру.

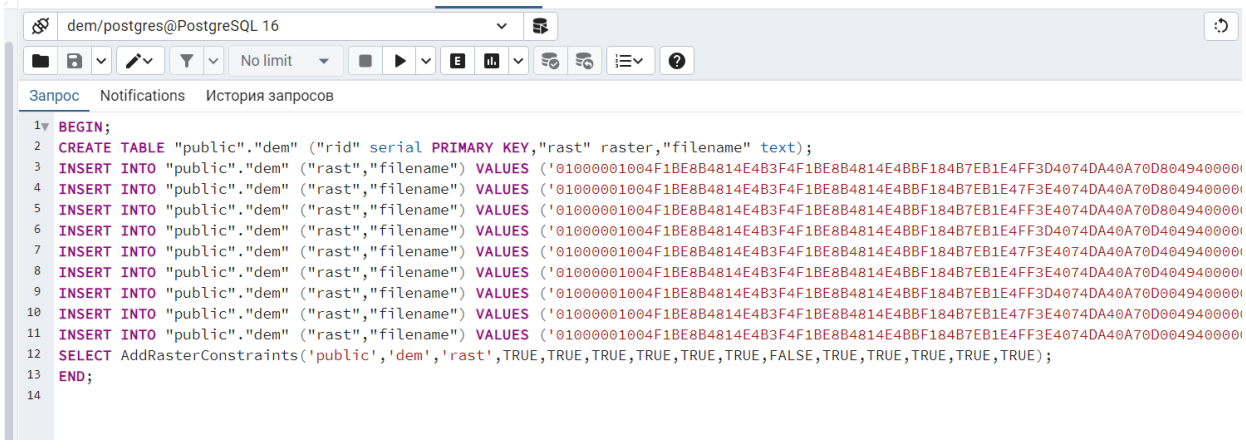
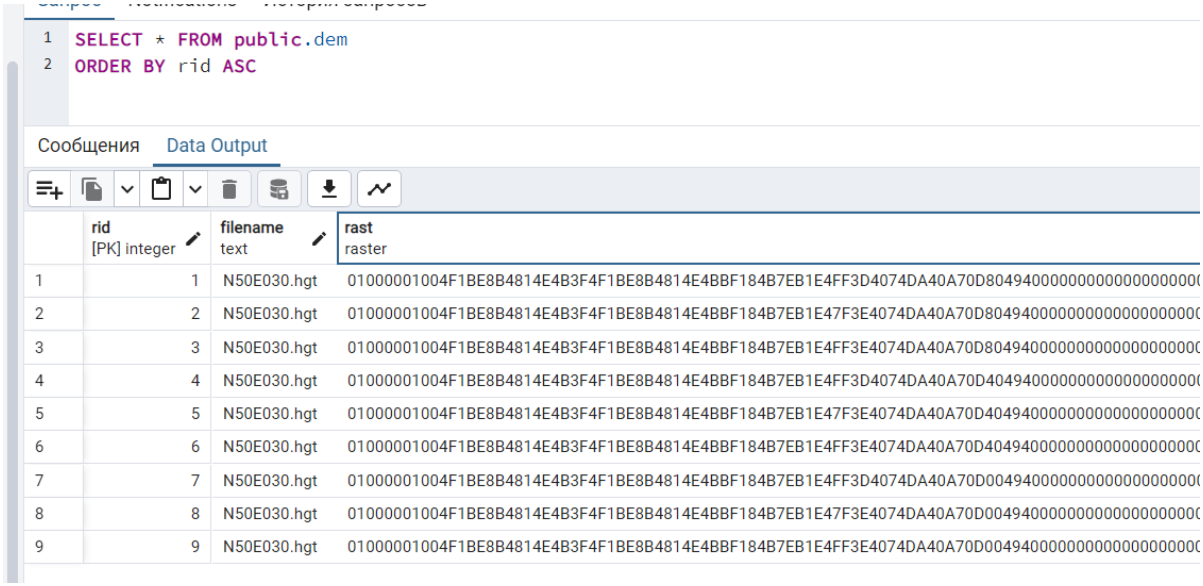


Рис. 3.10. Завантажений SQL-файл в pgAdmin 4



```
SELECT * FROM public.dem
ORDER BY rid ASC
```

Сообщения | Data Output

	rid [PK] integer	filename text	rast raster
1	1	N50E030.hgt	01000001004F1BE8B4814E4B3F4F1BE8B4814E4BBF184B7EB1E4FF3D4074DA40A70D80494000000000000000000(
2	2	N50E030.hgt	01000001004F1BE8B4814E4B3F4F1BE8B4814E4BBF184B7EB1E47F3E4074DA40A70D80494000000000000000000(
3	3	N50E030.hgt	01000001004F1BE8B4814E4B3F4F1BE8B4814E4BBF184B7EB1E4FF3E4074DA40A70D80494000000000000000000(
4	4	N50E030.hgt	01000001004F1BE8B4814E4B3F4F1BE8B4814E4BBF184B7EB1E4FF3D4074DA40A70D40494000000000000000000(
5	5	N50E030.hgt	01000001004F1BE8B4814E4B3F4F1BE8B4814E4BBF184B7EB1E47F3E4074DA40A70D40494000000000000000000(
6	6	N50E030.hgt	01000001004F1BE8B4814E4B3F4F1BE8B4814E4BBF184B7EB1E4FF3E4074DA40A70D40494000000000000000000(
7	7	N50E030.hgt	01000001004F1BE8B4814E4B3F4F1BE8B4814E4BBF184B7EB1E4FF3D4074DA40A70D00494000000000000000000(
8	8	N50E030.hgt	01000001004F1BE8B4814E4B3F4F1BE8B4814E4BBF184B7EB1E47F3E4074DA40A70D00494000000000000000000(
9	9	N50E030.hgt	01000001004F1BE8B4814E4B3F4F1BE8B4814E4BBF184B7EB1E4FF3E4074DA40A70D00494000000000000000000(

Рис. 3.11 Результат завантаження растрових даних в таблицю DEM у PostgreSQL

При завантаженні растру автоматично заповнюється та оновлюється таблиця *raster_columns* системного каталогу у PostgreSQL/PostGIS із метаданими про растрові стовпці усіх таблиць бази даних. Створення й опрацювання метадани здійснюється автоматично при будь-яких операціях, що впливають на структуру растрових таблиць (додавання, оновлення або вилучення растрів).

3.2.3 Побудова меж для блоків растру

Межі блоків растру будуються як полігональні прямокутники, що охоплюють блоки растру. В експериментальній базі даних векторна модель меж блоків реалізована як вид (View) *dem_rasters*, який містить запит на формування віртуальної таблиці з ідентифікаторами та межами блоків растру. *dem_rasters* як віртуальну таблицю можна використовувати в інших запитах та для візуалізації шару меж в ГІС. Межа блоку формується з використанням функції *ST_ConvexHull*, яка обчислює опуклу оболонку вхідного блоку растру.

Запит виду:

```
CREATE OR REPLACE VIEW dem_rasters AS
SELECT rid, ST_ConvexHull(rast) AS geom
FROM dem;
```

В дослідній базі даних запит виду *dem_rasters* надає межі дев’яти растрових блоків.

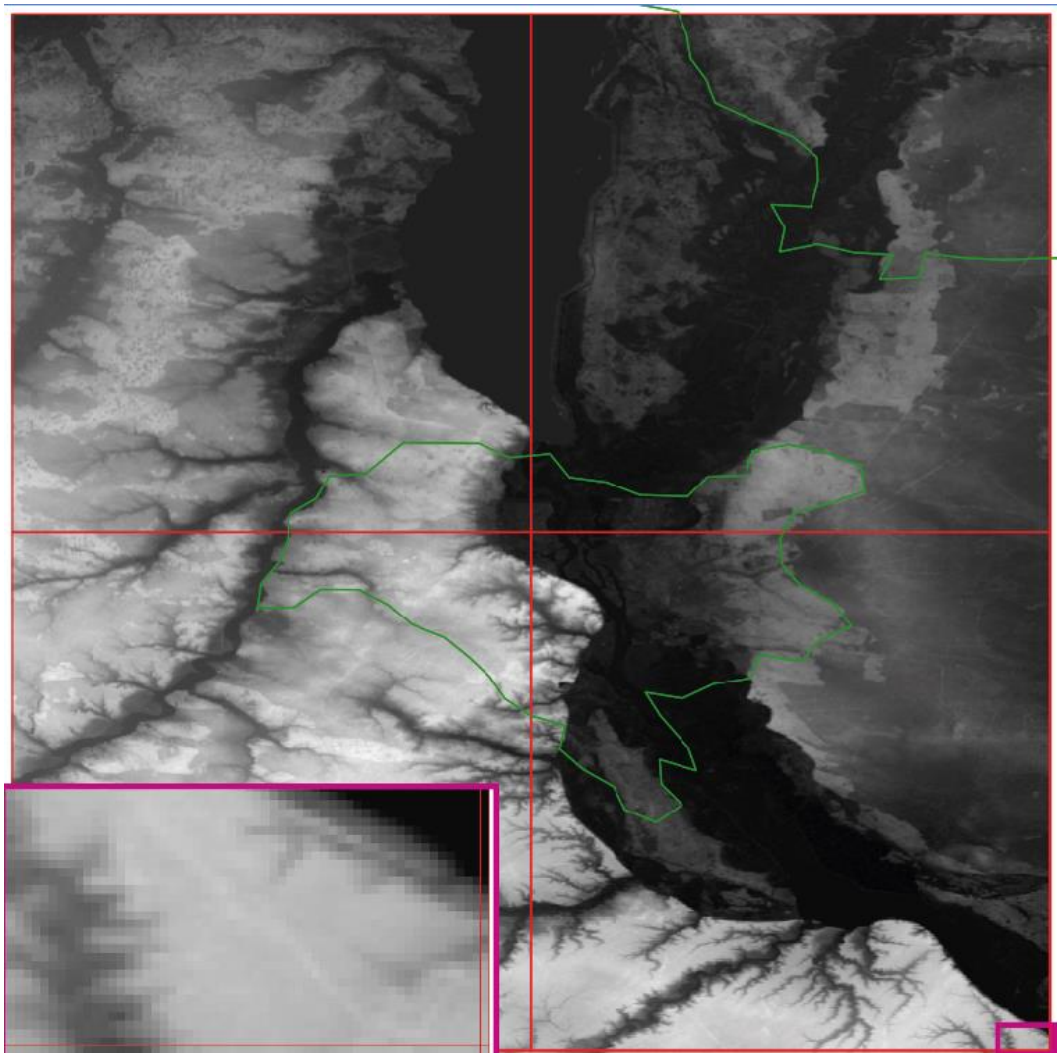


Рис. 3.12. Візуалізація меж растрових блоків у QGIS із виду *dem_rasters*

3.2.4 Візуалізація тривимірної моделі рельєфу

Візуалізації тривимірної форми рельєфу використовуються засоби створення 3D-карти у середовищі QGIS (рис. 3.13, 3.14).

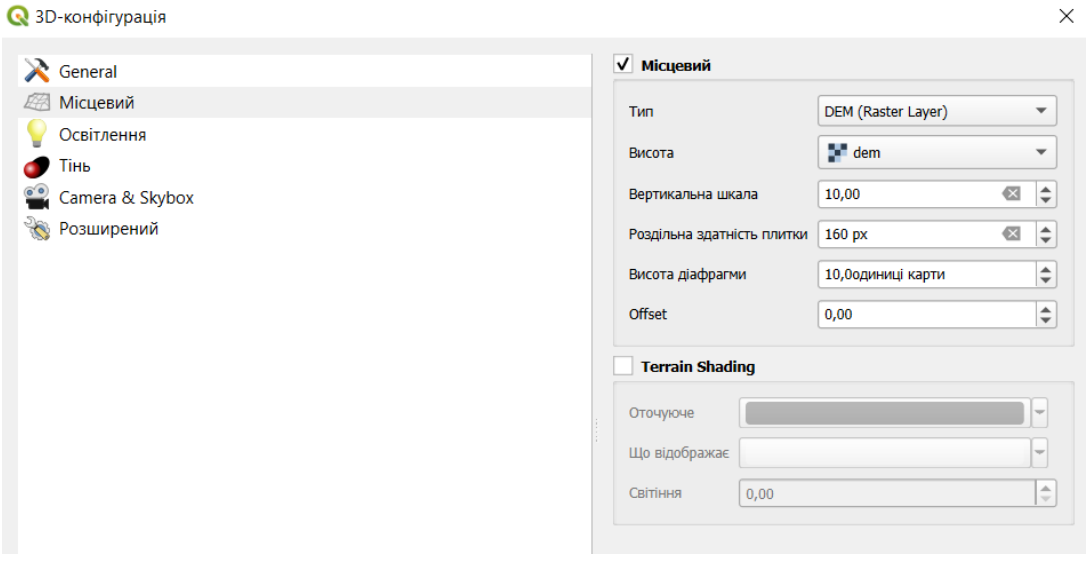


Рис. 3.13. Налаштування параметрів 3D-візуалізації шару DEM у QGIS

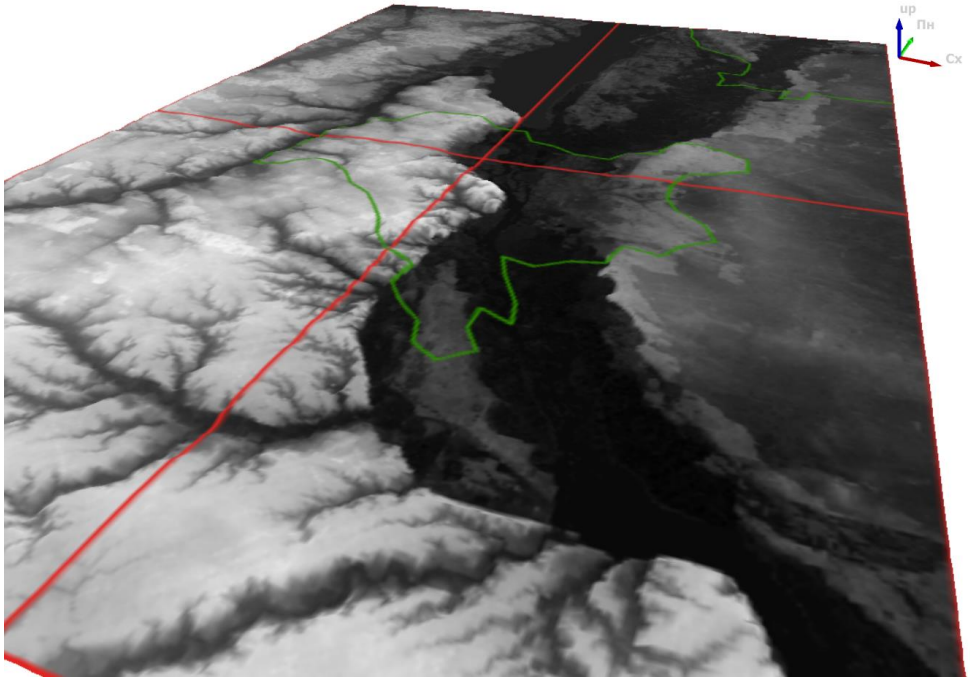


Рис. 3.14. 3D-візуалізація DEM моделі в QGIS

Функція 3D-візуалізації є доступною починаючи з версії QGIS 3.0 і доступна в меню: *Вид – Вікно 3D карти – Створити 3D перегляд карти*.

3.2.5 Побудова ізоліній

Побудова ізоліній здійснена з використанням функції «Контур» - ST_Contour, який генерує набір векторних контурів ізоліній для вхідної растрової смуги, використовуючи алгоритм контурів GDAL. Функція ST_Contour має вхідні параметри: rast (растр, для якого генерується контур), bandnumber (номер каналу, для якого генерується контур), level_interval (інтервал висоти між згенерованими контурами).

Повертаються записи з такими атрибутами: geom (геометрія контурної лінії), id (унікальний ідентифікатор, наданий контурній лінії GDAL), value (значення растрового каналу, яке представляє лінія). Запит на побудову ізоліній:

```
CREATE TABLE contours AS
SELECT (ST_Contour(
 rast,
 bandnumber => 1,
 level_interval => 10)).*
 FROM dem;
```

В результаті виконання запиту створюється таблиця *contours*, яка містить контури ізоліній (рис. 3.15). Запит виконався за 896 мсек, і було згенеровано 17401 записів.

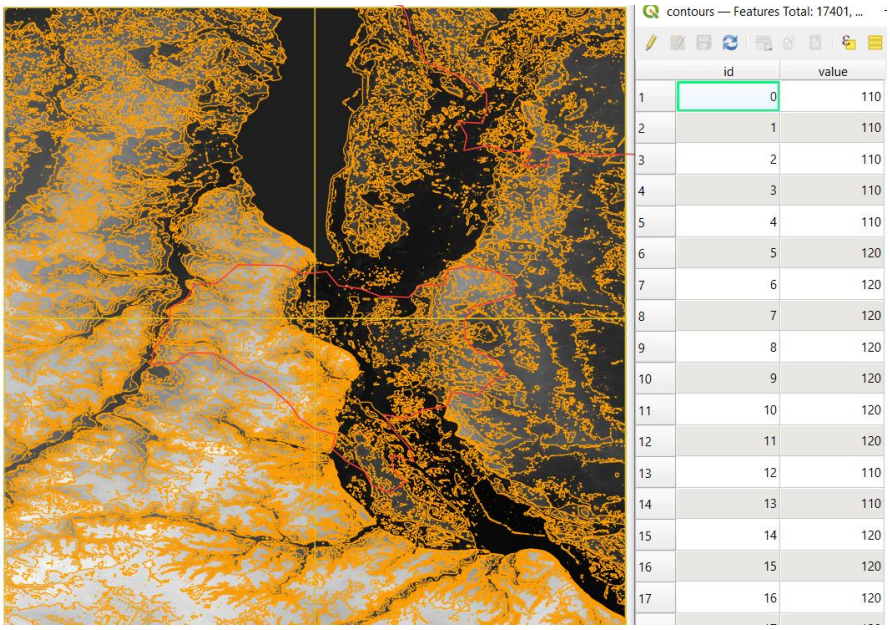


Рис. 3.15. Результат побудови ізоліній в середовищі PostgreSQL/PostGIS

При детальному аналізі зображення виявлено, що контури ізоліній між окремими блоками не співпадають (рис. 3.16) через зберігання растру блоками в БГД.

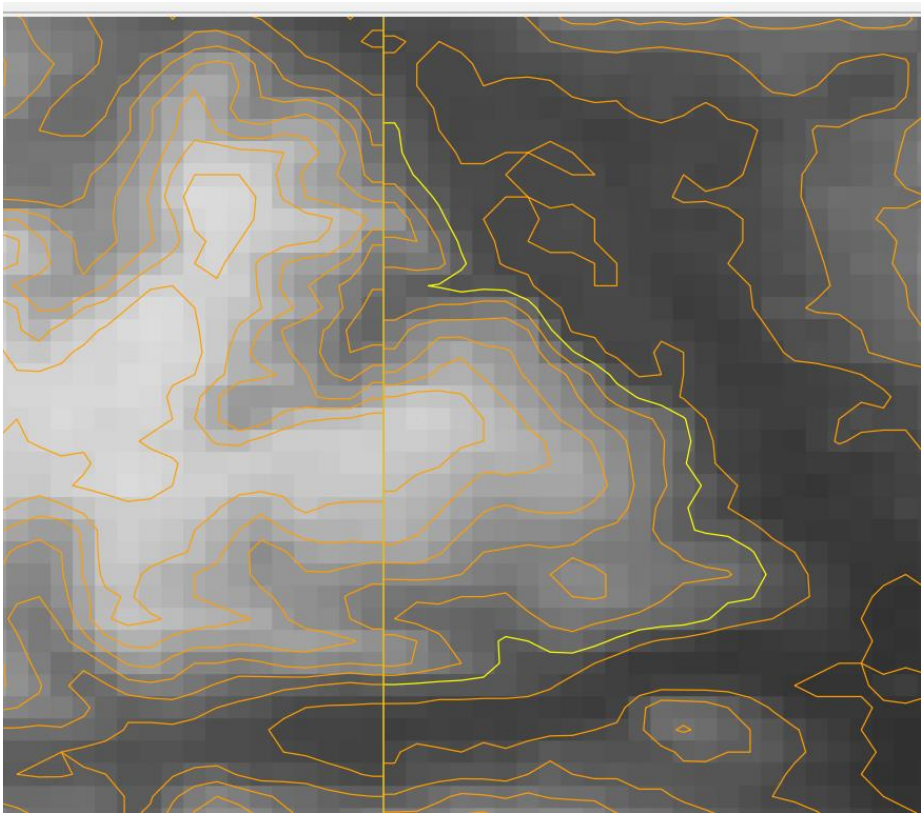


Рис. 3.16. Неузгодженість контурів ізоліній між блоками GRID-моделі

Для забезпечення цілісності контурів по всій площі між блоками до запиту було додано функцію ST_Union, що об’єднує всі блоки в один масив під час виконання запиту.

Модифікований запит побудови ізоліній з використанням функції ST_Union:

```
CREATE TABLE contours_u AS
SELECT (ST_Contour(
ST_Union(rast),
bandnumber => 1,
level_interval => 10)).*
FROM dem;
```

В результаті виконання запиту створюється таблиця *contours_u*, яка містить контури ізоліній, що узгоджені між блоками (рис. 3.17). Запит виконаний за 1 сек 238 мсек та було згенеровано 16782 записів.

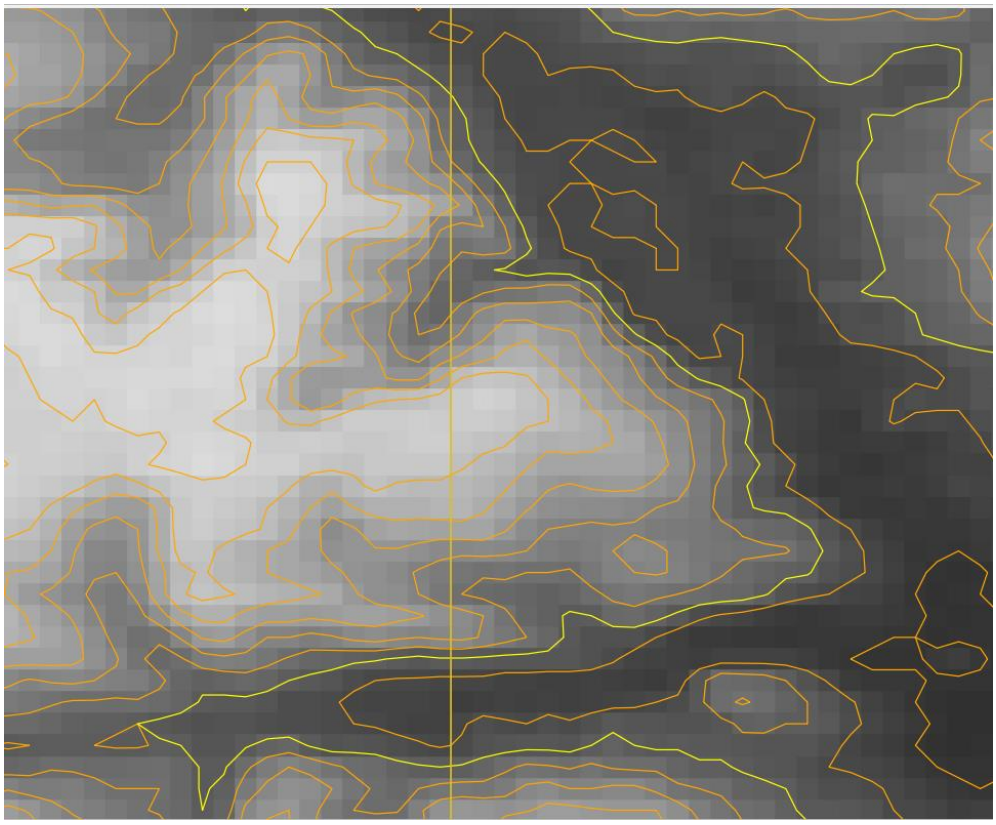


Рис. 3.17. Узгоджені контури ізоліній ліній для об’єднання блоків GRID-моделі з використанням функції ST_Union

Отримані контури ізоліній таблиці *contours_u,* були додані до створеної 3D карти (рис. 3.18) з налаштуваннями 3D-конфігурації рис. 3.13

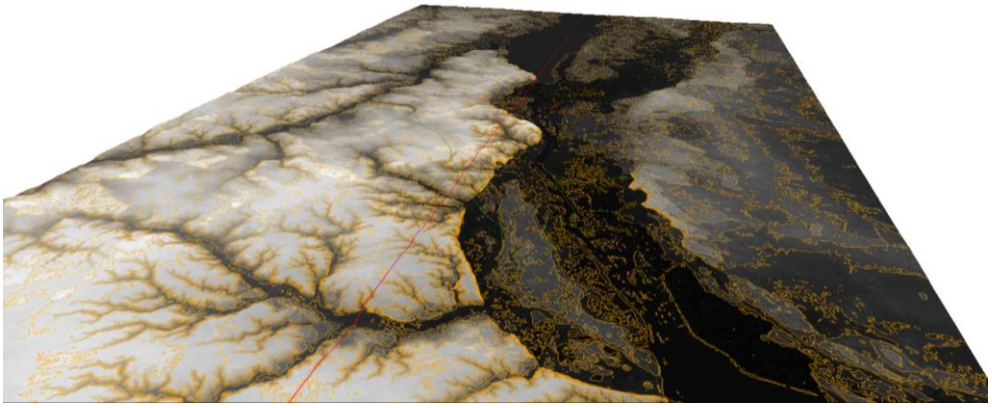


Рис. 3.18. Візуалізація 3D моделі DEM з горизонталями в QGIS

3.2.6 Побудова профілю висот

Для профілю висот необхідно визначитись з методом інтерполяції висот точки. В роботі розглянуто два методи інтерполяції для визначення висоти — метод найближчого сусіда та білінійна інтерполяція. Для інтерполяції висоти в одній точці в PostGIS надається функція ST_Value (rast, band, geom, resample), яка повертає обчислене за растром значення висоти у точці із заданими просторовими координатами. Параметри функції ST_Value:

rast — растрове поле, яке містить растрові дані, з яких буде отримано значення;

band — номер каналу растру. В запиті використовується значення 1, що означає перший канал;

geom — геометрична точка, для якої потрібно отримати значення пікселя;

resample — параметр, що задає метод інтерполяції.

Сутність методу найближчого сусіда полягає у визначені найближчого пікселя до заданої точки та повернення відповідного значення висоти.

Сутність білінійної інтерполяції полягає в обчисленні значення висоти, враховуючи значення у чотирьох сусідніх вершинах комірки. Спочатку виконується лінійна інтерполяція вздовж осі X між парами сусідніх пікселів, що розміщені на однакових координатах Y, а потім виконується лінійна інтерполяція вздовж осі Y як результат значення висоти у заданій точці.

Для інтерполяції в одній точці була вибрана позиція з координатами BL (30.46753, 50.42641) на території КНУБА. У запиті використано оператор WITH, який створює тимчасовий іменовану набір координат точки pt, що

формується функцією-конструктором ST_GeomFromText та доступний в межах поточного SQL-запиту. Запит виконання інтерполяції за найближчим сусідом:

```
WITH pt AS (
SELECT ST_GeomFromText('POINT(30.46753 50.42641)',4326) AS geom)
SELECT ST_Value(rast, 1, geom) AS value
FROM dem, pt
WHERE ST_Intersects(ST_ConvexHull(rast), geom);
```

В результаті за методом найближчого сусіда висота становить 186 м, оскільки найближчий піксель мав значення 186.

Запит виконання інтерполяції за методом білінійної інтерполяції:

```
WITH pt AS (
SELECT ST_GeomFromText('POINT(30.46753 50.42641)',4326) AS geom)
 SELECT ST_Value(rast, 1, geom, resample => 'bilinear') AS value
FROM dem, pt WHERE ST_Intersects(ST_ConvexHull(rast), geom);
```

В результаті білінійної інтерполяції висота становить 186.63 м.

Для побудови профілю з кроком 1000 метрів, обрано лінію, що лежить приблизно на серединній паралелі міста Києва (рис. 3.10). У запиті використано функцію ST_SetZ, яка забезпечує перетворення 2D-точки в 3D-точку з координатою Z, значення якої інтерполюється за обраним методом.

Запит побудови профілю з кроком між точками:

```
CREATE TABLE transect AS
WITH transect AS (
SELECT ST_Segmentize(ST_GeogFromText('LINESTRING(30.20 50.41, 30.80
50.41)'), 1000)::geometry AS geom ), rast AS (
SELECT ST_Union(dem.rast) AS rast FROM dem JOIN transect ON
ST_Intersects(transect.geom, ST_ConvexHull(dem.rast)) ), z AS (
SELECT (ST_DumpPoints(ST_SetZ(rast.rast, transect.geom, resample =>
'bilinear'))).* FROM rast CROSS JOIN transect )
SELECT round(ST_Z(geom)) AS z, geom FROM z;
```

В результаті виконання запиту створюється таблиця *transect*, що містить 65 точкових об’єктів з обчислиними висотами (рис. 3.19), на основі яких побудовано графік висотного профілю (рис. 3.20)

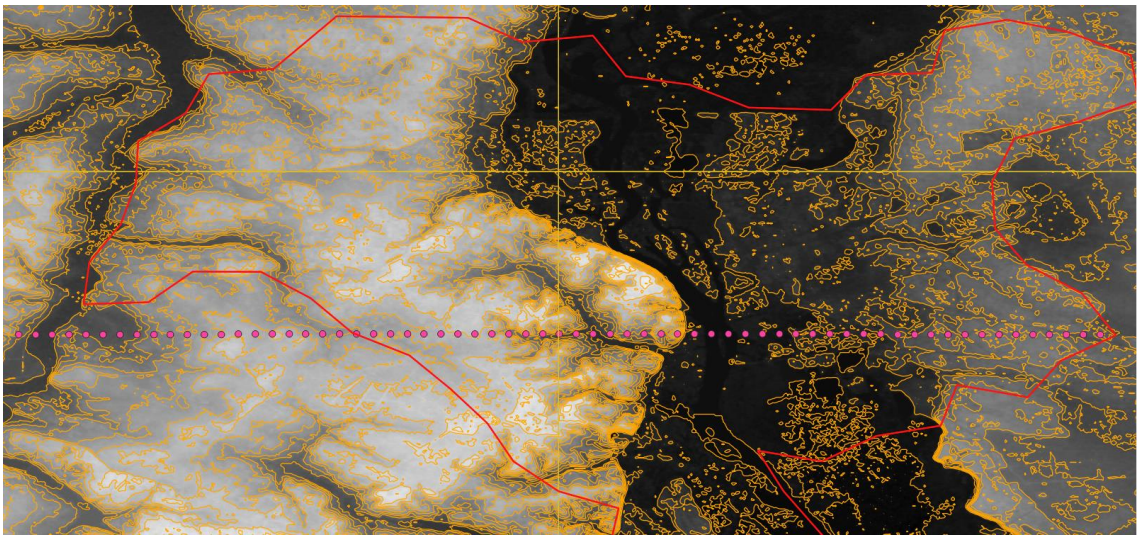


Рис. 3.19. Результат створенню профілю проілюстрований в QGIS

В pgAdmin IV є зручний засіб побудови графіків, який дозволяє будувати лінійні діаграми, накопичені лінійні діаграми, стовпчикові діаграми, накопичені стовпчикові діаграми та кругові діаграми за обраними значеннями осей X та Y. В рамках роботи була побудована лінійний графік профілю висот (рис. 3.15) із пустим значенням осі X та значенням Z осі Y.

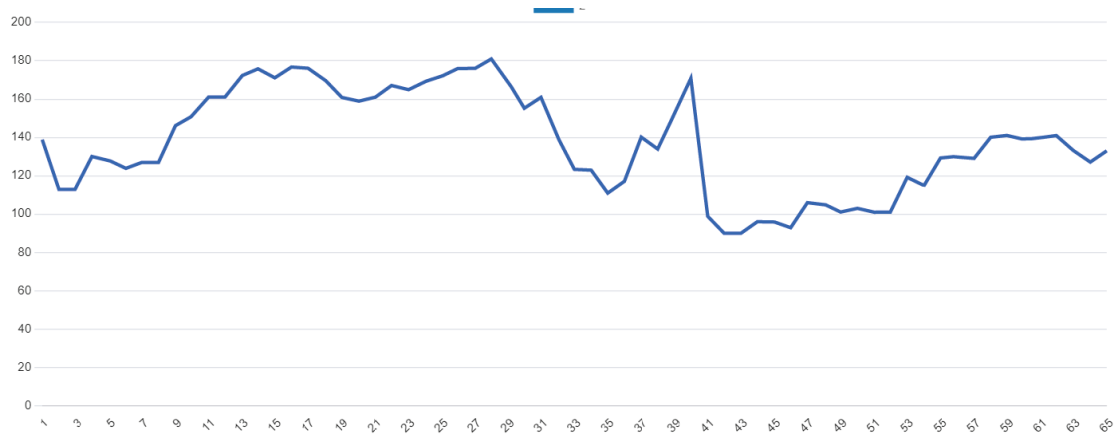


Рис. 3.20. Графік обчисленого профілю висот

В таблиці 3.3 узагальнені характеристики виконання запитів опрацювання SRTM моделі в середовищі СКБД PostgreSQL/PostGIS, що містить 1 442 401 пікселів.

Наведені узагальнені дані засвідчують, що функції, які реалізовані для опрацювання GRID-моделі в PostGIS є ефективними. Найскладніший запит виконався трохи більше ніж за одну секунду, при цьому растр містив 1 442 401 пікселів, що свідчить про високий рівень продуктивності та оптимізацію використаних функцій.

Таблиця 3.3. Узагальнення виконання запитів опрацювання GRID моделі

Назва таблиці	Назва основної функції	Назва додаткових функцій	Число створених елементів моделі	Час виконання
dem_rasters	ST_ConvexHull	відсутня	9	53 мсек
counturs	ST_Contour	відсутня	17401	896 мсек
counturs_u	ST_Contour	ST_Union	16782	1 сек 238 мсек
pt (найближчий сусід)	ST_Value	ST_GeomFromText, ST_Intersects, ST_ConvexHull	1	52 мсек
pt (білінійна інтерполяція)	ST_Value	ST_GeomFromText, ST_Intersects ST_ConvexHull	1	93 мсек
transect	ST_Z	ST_Segmentize, ST_GeogFromText, ST_Union, ST_Intersects, ST_ConvexHull, ST_DumpPoints	65	284 мсек

3.2.7. Аналіз рельєфу

Аналіз рельєфу складається з наступних етапі: побудова гістограми висот GRID-моделі; обчислення ухилів рельєфу GRID-моделі; обчислення орієнтації схилів GRID-моделі; обчислення гіпотетичного освітлення GRID-моделі; обчислення топографічного індексу GRID-моделі; побудовою векторної моделі ухилів рельєфу.

Побудова гістограми висот GRID-моделі виконувалася для визначення домінуючої розподілу висот на території Києва та його околиць. Для цього використовується функція побудови гістограми висот – ST_Histogram (raster, nband), де:

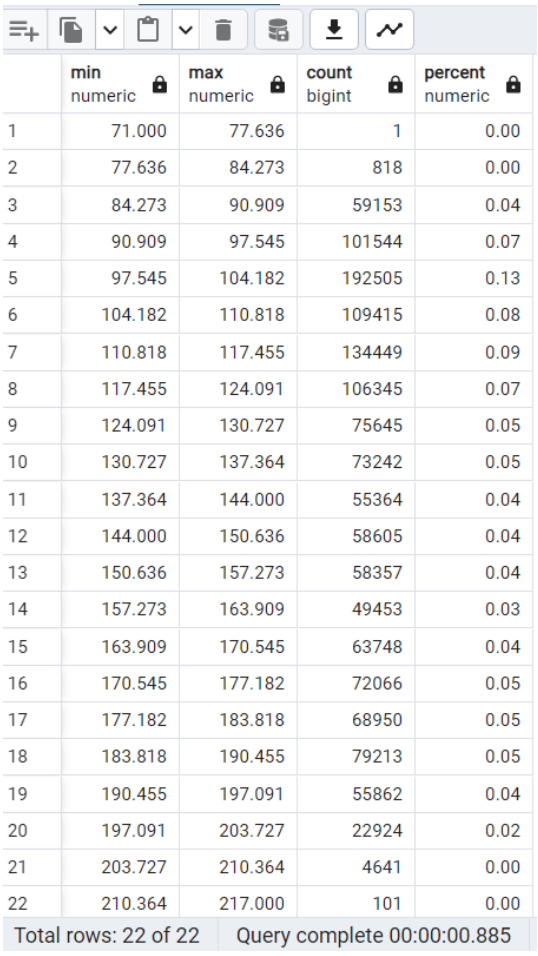
raster rast: вхідна растрова модель, яке передається функції;

integer nband =1: номер каналу в моделі растру.

Запит побудови гістограми висот:

```
WITH hist AS ( SELECT (ST_Histogram(ST_Union(rast),1)) .*
from dem
WHERE filename='N50E030.hgt'
)
SELECT
round(min::numeric,3) AS min,
round(max::numeric,3) AS max, count,
round(percent::numeric,2) AS percent
FROM hist
ORDER BY min;
```

В результаті запиту створюється набір даних гістограми, що містить 22 градації поділу значень висот з їх кількістю та відсотком до зальної кількості точок растру (рис.3.21). Використовуючи отриманий результат побудовано в середовищі qgAdmin 4 графічне подання гістограм (рис. 3.22).



	min numeric	max numeric	count bigint	percent numeric
1	71.000	77.636	1	0.00
2	77.636	84.273	818	0.00
3	84.273	90.909	59153	0.04
4	90.909	97.545	101544	0.07
5	97.545	104.182	192505	0.13
6	104.182	110.818	109415	0.08
7	110.818	117.455	134449	0.09
8	117.455	124.091	106345	0.07
9	124.091	130.727	75645	0.05
10	130.727	137.364	73242	0.05
11	137.364	144.000	55364	0.04
12	144.000	150.636	58605	0.04
13	150.636	157.273	58357	0.04
14	157.273	163.909	49453	0.03
15	163.909	170.545	63748	0.04
16	170.545	177.182	72066	0.05
17	177.182	183.818	68950	0.05
18	183.818	190.455	79213	0.05
19	190.455	197.091	55862	0.04
20	197.091	203.727	22924	0.02
21	203.727	210.364	4641	0.00
22	210.364	217.000	101	0.00

Total rows: 22 of 22 Query complete 00:00:00.885

Рис. 3.21. Таблиця розподілу висотних значень території Києва та його околиць

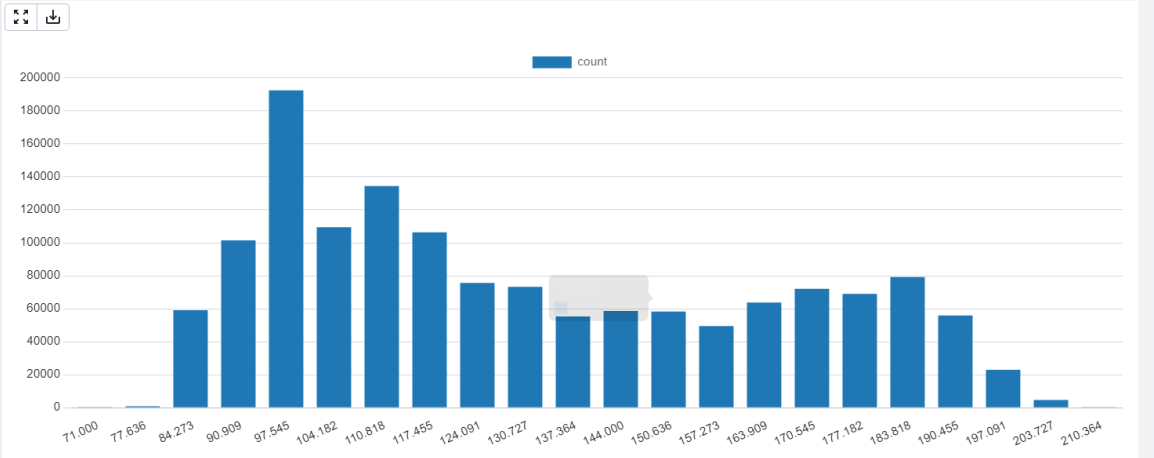


Рис. 3.22. Гістограма висотних значень території

На основі аналізу встановлено, що домінуючі висоти для досліджуваної території лежать у діапазоні від 97 м до 125 м, причому найбільша кількість пікселів зосереджена на висотах 97–110 м. Це вказує на переважно рівнинний рельєф із незначними підвищеннями. Підвищення висоти до 150–190 м представлені меншою кількістю точок, що вказує на наявність окремих пагорбів. Висоти понад 190 м є нетиповими для території, підтверджуючи загальний характер помірного рівнинного рельєфу з окремими висотними аномаліями.

Побудова моделі нахилу рельєфу здійснювалась з використанням функції PostGIS – ST_Slope (rast, nband, pixeltype, units, scaling), де:

rast::raster — растр вхідної GRID-моделі рельєфу.

nband::int — номер каналу у растрі в растрі, за замовчуванням 1

pixeltype::text — тип даних пікселів у вихідному растрі, '32BF' є форматом даних із плаваючою точкою з 32-бітною точністю.

units::text — одиниці виміру вихідного значення нахилу, використовується 'DEGREES' в градусах.

scaling::double precision — масштабний коефіцієнт, що визначає розмір пікселів у метрах на градус, виконується – 111 120 (для екватора).

Алгоритм роботи визначення нахилу рельєфу згідно [14,24] ґрунтується на аналізі висоти кожної чарунки GRID-моделі та її сусідніх клітинок, з використанням рухомого вікна розміром 3 х 3 чарунки(рис. 3.23). Вікно

переміщується по всій площі растру, і в кожному положенні обчислюється нахил центральної клітинки, за висотами сусідніх клітинок.

a	b	c
d	e	f
g	h	i

Рис. 3.23 Схема рухомого вікна 3 х 3 для обчислення нахилу

Для кожного положення цього вікна обчислюється нахил центральної клітинки, за висоти її восьми сусідніх клітинок.

Швидкість зміни висоти у напрямку X (формула 3.1) обчислюється як різниця між висотами сусідніх клітинок уздовж горизонтальної осі, тоді як швидкість зміни у напрямку Y(формула 3.2) обчислюється як різниця висот між клітинками уздовж вертикальної осі. На основі цих градієнтів обчислюється найкрутіший спуск від центральної клітинки, (формула 3.3) що дає можливість визначити її нахил. Значення нахилу може бути виражене у градусах, радіанах або у відсотках, залежно від параметрів, встановлених під час обчислення.

$$[dz/dx] = \frac{(c + 2f + i) - (a + 2d + g)}{8 * cellsize} \tag{3.1}$$

$$[dz/dy] = \frac{(g + 2h + i) - (a + 2b + c)}{8 * cellsize} \tag{3.2}$$

$$Slope\_rad = ATAN(\, z_{factor} * \sqrt{\left[\frac{dz}{dx}\right]^2 + \left[\frac{dz}{dy}\right]^2}\,) \tag{3.3}$$

Запит обчислення нахилу рельєфу:

```
CREATE TABLE dem_slope as
SELECT rid, ST_Slope(rast::raster,
'1'::int,'32BF'::text,'DEGREES'::text,'111120'::double precision) AS rast
FROM dem
WHERE filename='N50E030.hgt'
```

В результаті виконання запиту створюється таблиця *dem_slope*, яка містить значення нахилу кожної чарунки в растрі (рис. 3.24-3.25), час обробки склав 16 секунд і 767 мілісекунд.

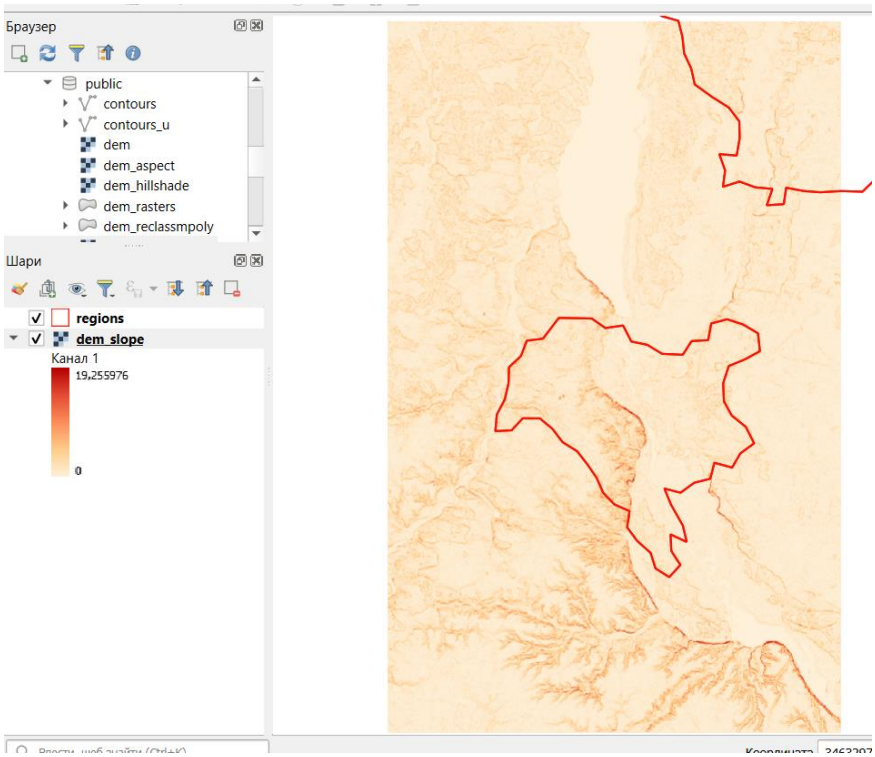


Рис. 3.24. Візуалізація растру нахилу в QGIS в системі координат 3395

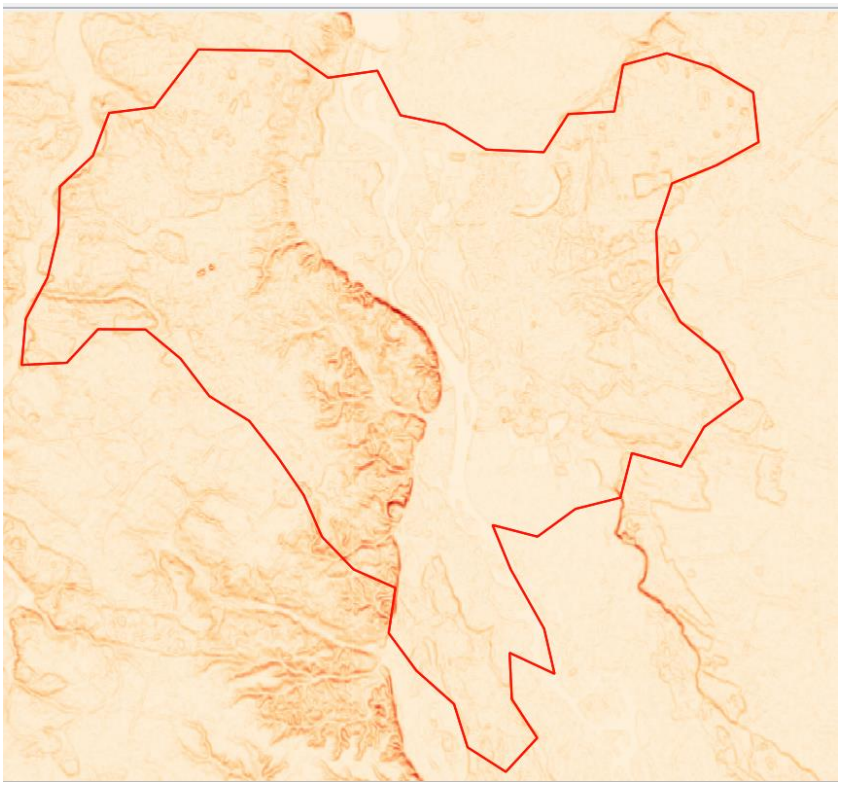


Рис.3.25 Фрагмент зображення растру нахилу для території м.Київ

Градації кольорів растру на рис. 3.24-3.25 від світло-бежевого до насиченого червоного представляють різні значення нахилу. Значення нахилу коливаються в межах від 0 до 19.26 градусів. Світліші відтінки вказують на пологі ділянки, тоді як більш насичені відтінки представляють крутіші схили.

Візуально більшість території представлена світлими відтінками, що вказує на м'який або слабкий нахил для більшої частини території.

Побудова моделі орієнтації схилів GRID-моделі виконувалась з використанням функції – ST_Aspect, де:

rast::raster – растр вхідної GRID-моделі рельєфу.

nband::int – номер каналу в растрі, для якого обчислюється аспект, за замовчуванням 1.

pixeltype::text – тип даних пікселів у вихідному растрі, '32BF' означає формат даних із плаваючою точкою з 32-бітною точністю.

units::text – одиниці виміру вихідного значення аспекту. Використовується 'DEGREES' для градусів.

return_from_north::boolean – параметр, що визначає, чи вимірювати аспект як кут від північного напрямку (1 – TRUE`) або від східного напрямку (0 – FALSE).

Алгоритм визначення орієнтації схилів згідно [14,24] ґрунтується на визначенні напрямку, в якому схил має найкрутіший спуск. Функція ST_Aspect використовує рухоме вікно розміром 3 x 3 чарунка (рис. 3.23) і обчислює градієнти по осі X (формула 3.1) і по осі Y (формула 3.2). Після обчислення градієнтів алгоритм визначає аспект з допомогою формули формули 3.4.

$$Aspect = arctan\left(\frac{\Delta y}{\Delta x}\right) \quad (3.4)$$

Отримане значення показує напрям, в якому відбувається найкрутіший спуск, і може бути виражене у градусах або радіанах. Значення аспекту обчислюється від півночі по годинниковій стрілці, якщо аспект виражається у градусах, то значення будуть у діапазоні від 0 до 360 градусів, де 0 градусів відповідає півночі, 90 градусів — сходу, 180 градусів — півдню, а 270 градусів — заходу.

Якщо нахил центральної клітинки дорівнює нулю (тобто рельєф є горизонтальним), значення аспекту для такої клітинки визначається як -1, оскільки немає чіткого напрямку схилу. Цей результат допомагає

ідентифікувати ділянки, які є абсолютно плоскими, і для яких напрямок спуску не має сенсу.

Запит обчислення орієнтації схилів:

```
CREATE TABLE dem_aspect as
SELECT rid,
ST_Aspect(rast::raster,'1'::int,'32BF'::text,'DEGREES'::text,'1'::boolean)
AS rast
FROM dem
WHERE filename='N50E030.hgt'
```

В результаті виконання запиту створено таблицю *dem_aspect*, що містить растр (рис. 3.26 – 3.27) із значеннями орієнтації схилів для кожної клітинки досліджуваної території. Час виконання запиту склав 21 секунду та 376 мілісекунд.

На рис. 3.26, 3.27 проілюстровано розподіл орієнтацій схилів з використанням градації кольорів від світло-оранжевого до темно-червоного. Градація починається зі світло-бежевого кольору, яким позначені рівнинні ділянки та ділянки, орієнтовані на північ. Далі кольори переходить до світло-оранжевого для схилів, орієнтованих на схід, помаранчевого — для схилів, близьких до півдня, і до насичено-червоного — для схилів, орієнтованих на захід

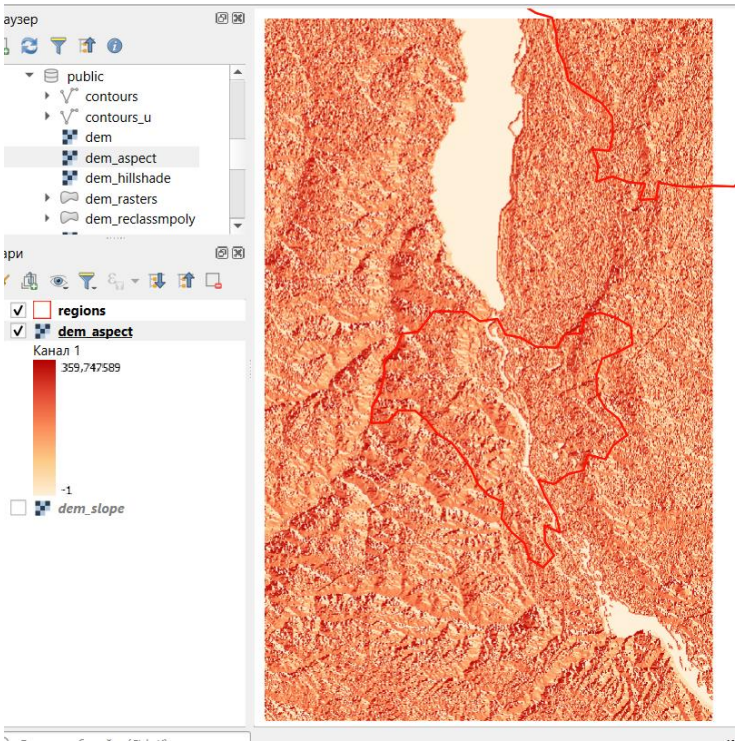


Рис. 3.26. Візуалізація растру орієнтації схилів в QGIS в системі координат 3395

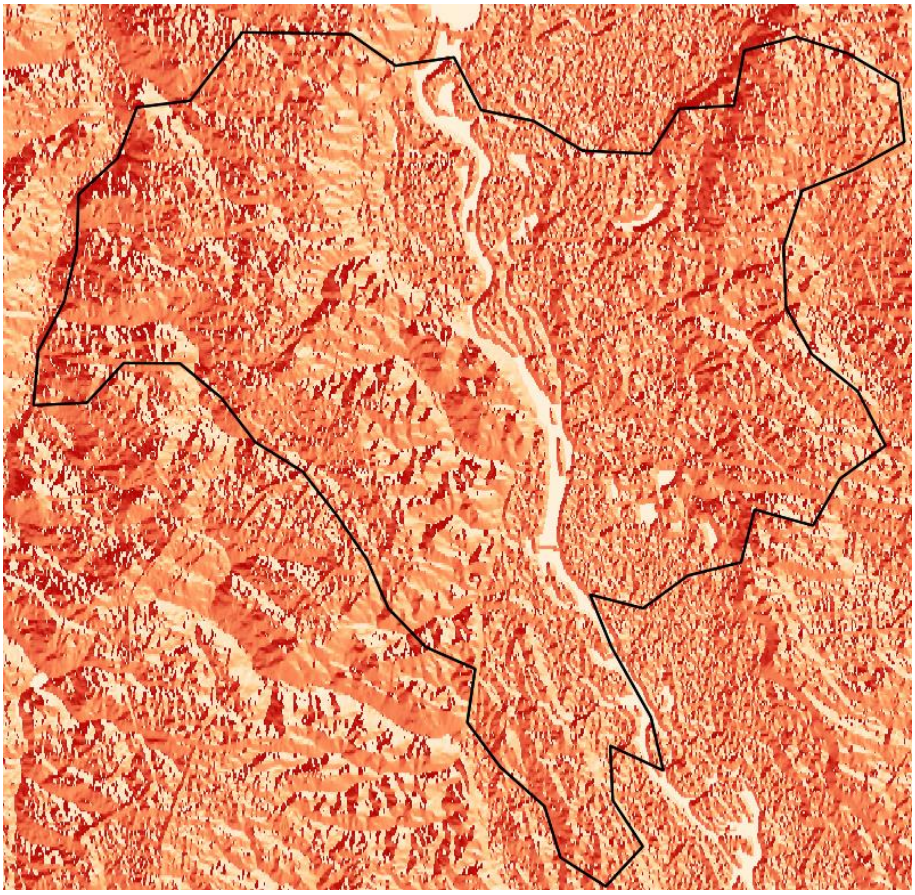


Рис. 3.27 Фрагмент зображення растру орієнтації схилів для території м. Київ

Побудова моделі освітлюваності схилів здійснюється з використанням функції PostGIS – ST_HillShade(rast, nband, pixeltype, azimuth, altitude, brightness, z_factor, compute_shadow). Параметри функції ST_HillShade:

rast::raster – растр вхідної GRID-моделі рельєфу.

nband::int – номер каналу в растрі, для якого обчислюється освітленість, за замовчуванням 1.

pixeltype::text – тип даних пікселів у вихідному растрі, '32BF' означає формат даних із плаваючою точкою з 32-бітною точністю.

azimuth – азимутальний кут, що вказує напрямок джерела світла, в градусах, становить 315° він відповідає північно-західному напрямку.

altitude – кут висоти, який вказує висоту джерела світла над горизонтом у градусах від 0 до 90°. Становить 45° – середня висота світла над горизонтом.

brightness – максимальне значення яскравості для вихідного растру, становить 255 відповідає максимальній яскравості.

z_factor – коефіцієнт затемнення, що використовується для регулювання контрасту тіні, становить 1.0 стандартний контраст.

compute_shadow – параметр, що визначає, чи використовувати додаткове затінення, становить FALSE додаткове затінення не використовується.

Алгоритм ST_HillShade згідно [14,24] полягає у визначенні, рівня освітленості для кожної клітинки GRID-моделі рельєфу залежно від моделювання положення джерела світла. Це дозволяє імітувати вплив сонячного світла на рельєф і отримати зображення, яке наочно показує тіні, схили та височини.

Алгоритм використовує інформацію про азимут (напрямок, з якого приходить світло (сонце)), висоту сонця (кут підйому сонця над горизонтом) та характеристики рельєфу (нахил і аспект), щоб визначити, наскільки інтенсивно сонячне світло впливатиме на кожну клітинку.

Спочатку для кожної клітинки растру обчислюються нахил і аспект. Нахил визначає крутизну схилу, тоді як аспект показує, у якому напрямку нахилений схил. На основі нахилу та аспекту кожної клітинки, а також параметрів азимута і висоти сонця, обчислюється, рівень освітленості клітинки за рівням:

$$Hillshade = 255.0 * ((cos(Zenith\_rad) * cos(Slope\_rad)) + (sin(Zenith\_rad) * sin(Slope\_rad) * cos(Azimuth\_rad - Aspect\_rad))) \quad (3.5)$$

Значення освітленості для кожної клітинки може бути у діапазоні від 0 до 255, де 0 означає повну тінь, а 255 – максимальне освітлення. Таким чином, клітинки, які знаходяться під прямим освітленням сонця, мають високі значення, а ті, що знаходяться у тіні або на віддаленому схилі – низькі значення

Запит побудови растру гіпотетичного освітлення:

```
CREATE TABLE dem_hillshade as
SELECT rid ST_HillShade(rast::raster, 1, '32BF'::text, 315, 45, 255, 1.0, FALSE),
AS rast
FROM dem
WHERE filename='N50E030.hgt'
```

У результаті виконання було створено таблицю dem_hillshade, що містить растр гіпотетичного освітлення,(рис. 3.28 – 3.29) із значеннями гіпотетичного освітлення для кожної клітинки досліджуваної території. Час обробки склав 16 секунд та 282 мілісекунди.

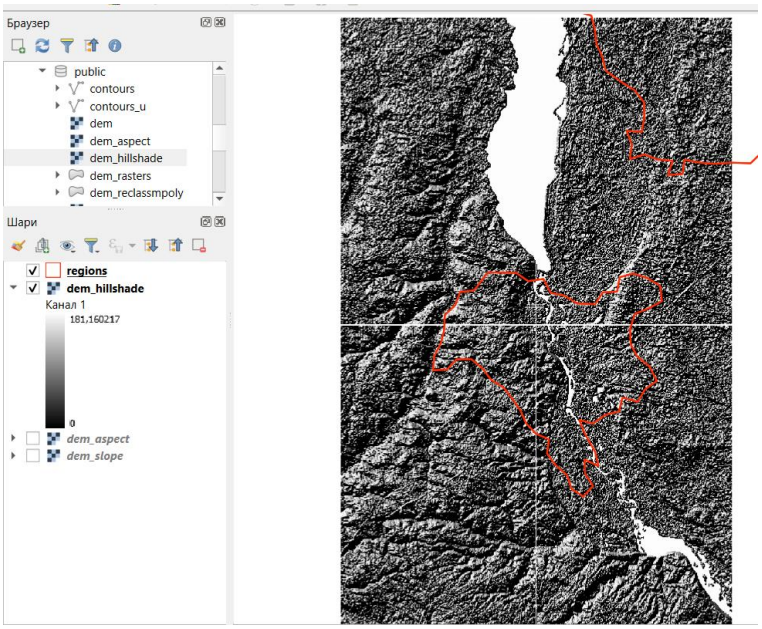


Рис. 3.28 Візуалізація растру гіпотетичного освітлення в QGIS в системі координат 3395

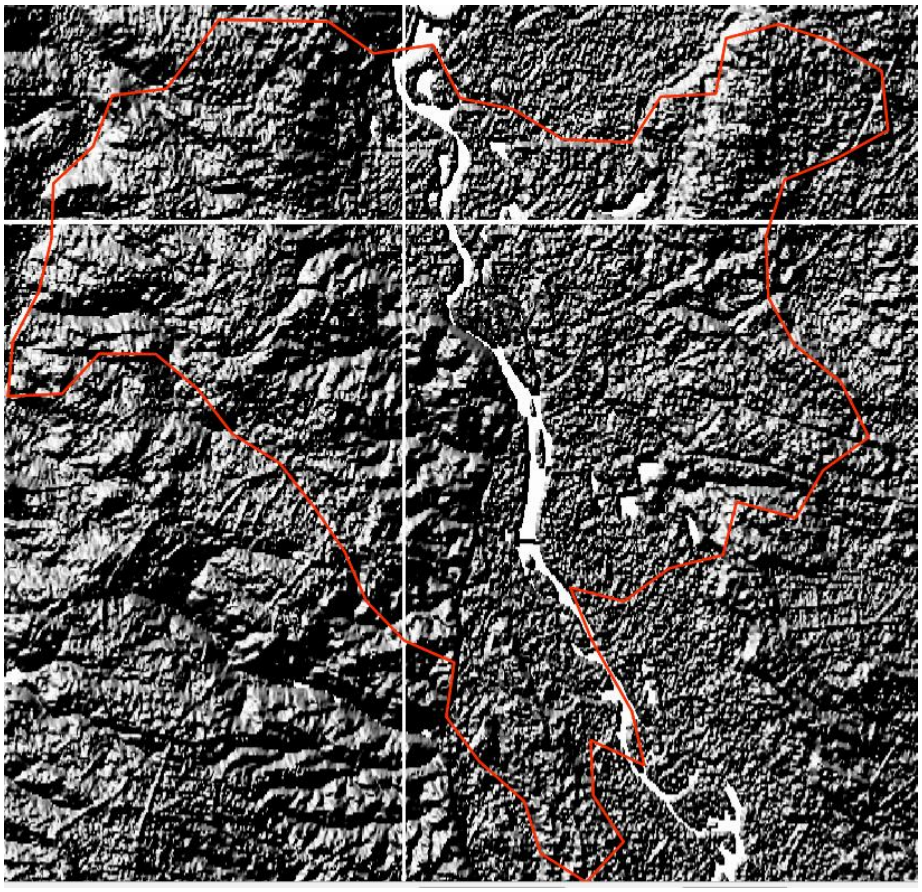


Рис. 3.29 Фрагмент зображення растру гіпотетичного освітлення для території м.Київ

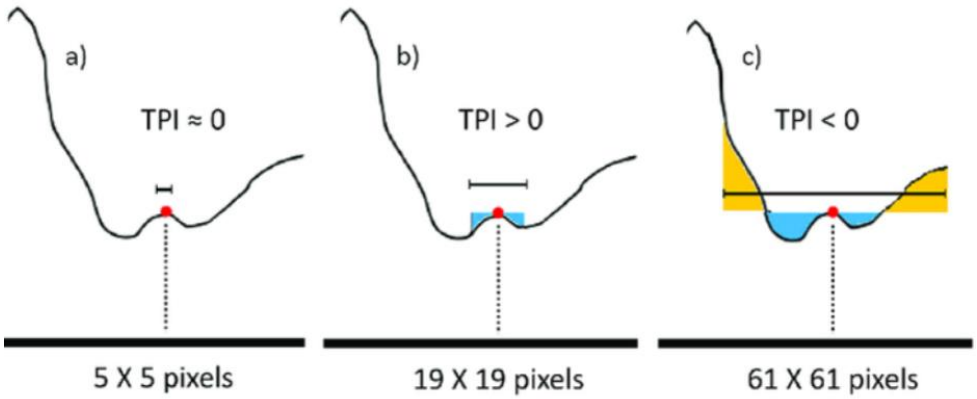
На рис. 3.28 – 3.29 продемонстровано освітленість на територію з північного заходу під середнім кутом у 45°. Де темніші ділянки відповідають затіненій місцевості при обраних параметрах освітленості, а світлі – територію без затінення.

Побудова індексу топографічного положення виконуються з використанням функції PostGIS ST_TPI (rast, nband) де:

a.rast – растр вхідної GRID-моделі рельєфу.

nband – номер каналу в растрі, для якого обчислюється TPI, за замовчуванням 1

Індекс топографічного положення (Topographic position index – TPI) згідно [15] обчислюється шляхом порівняння висоти кожного пікселя з його сусідніми пікселями. Кількість сусідів, з якими проводиться порівняння (тобто розмір околу), має великий вплив на результат (наприклад, 5×5, 19×19 або



61×61(рис. 3.30).

Рис. 3.30. Залежність значення TPI від масштабу аналізу [15]

TPI зі значенням нуль або близько до нуля – рівнинна ділянка або неперервний схил.

TPI із додатнім значенням відповідає положенню центрального пікселю значно вищого за оточуючі ділянки, що відповідає гребням або пагорбам.

TPI із великим від’ємним значенням для центрального пікселя значно нижчий за оточуючі ділянки тобто знаходиться на дні долини або яру.

Запит для побудови TPI:

```
CREATE TABLE dem_tpi AS
SELECT rid, ST_TPI (a.rast, 1) AS rast
FROM dem AS a
WHERE filename='N50E030.hgt'
```

У результаті виконання запиту створюється таблиця *dem_tpi*, що містить растр обчисленого TPI (рис. 3.31, 3.32). Час обробки склав 10 секунд та 519 мілісекунд. Основним кольором растру, який переважає на зображенні, є нейтральний або сірий відтінок. Це вказує на те, що більшість ділянок мають значення TPI, близькі до нуля. Такий результат свідчить про рівнинні або майже рівнинні поверхні, де немає значних підвищень або знижень рельєфу

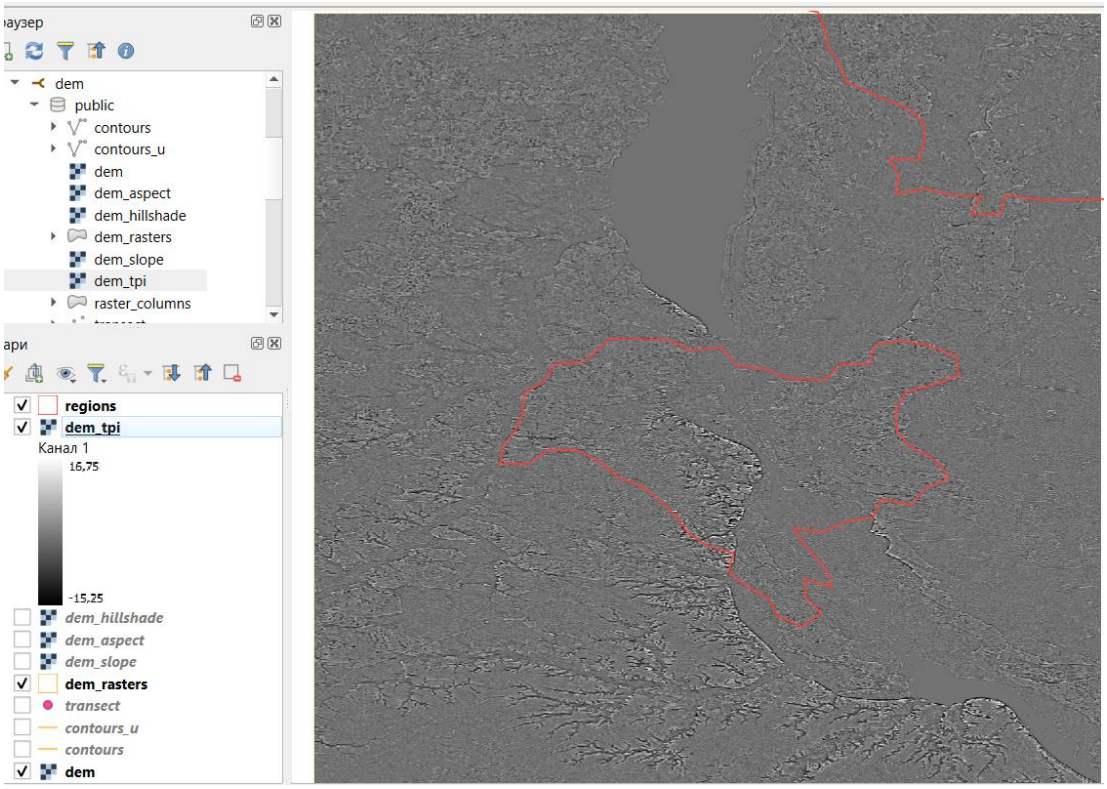


Рис. 3.31. Обчислений індекс топографічного положення

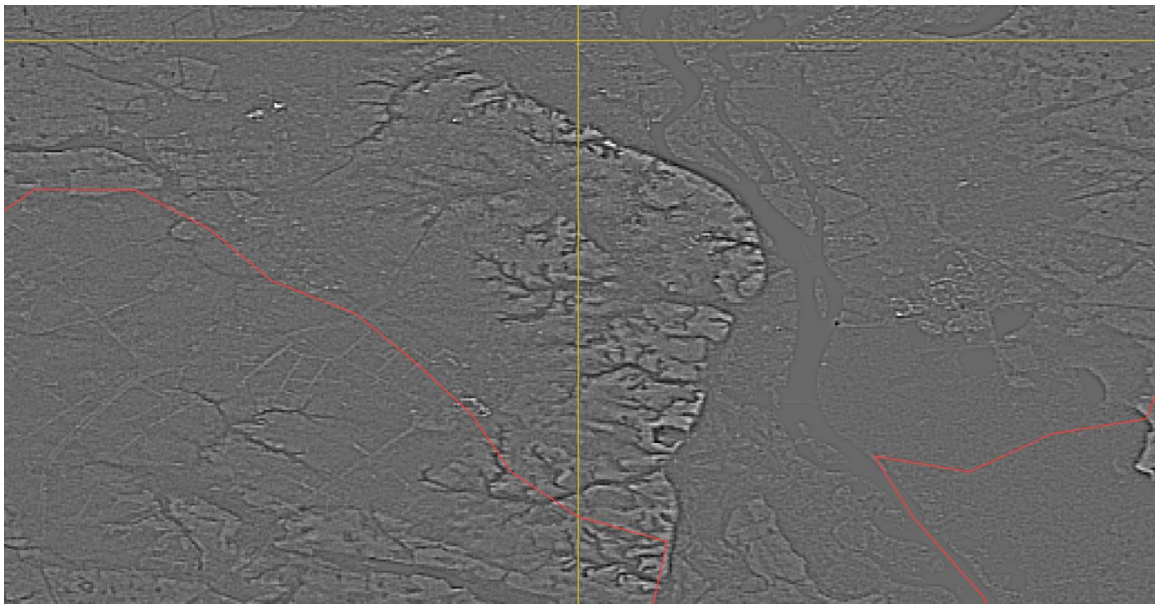


Рис. 3.32. Збільшений фрагмент обчисленого TPI

Побудова растрової моделі ухилів рельєфу здійснюється з використанням функції перетворенням растр в вектор (полігон). Для отримання візуально прийнятної векторної моделі ухилів рельєфу доцільно попередньо виконати рекласифікацію вхідної GRID-моделі. Для цього використана функція ST_Histogram для таблиці *dem_slope*, що проводить аналіз розподілу значень нахилу і забезпечити коректний поділ рельєфу на класи.

Запит обчислення гістограми висот для *dem_slope*:

```
WITH hist AS ( SELECT (ST_Histogram(ST_Union(rast),1)) .*
from dem_slope )
SELECT
round(min::numeric,3) AS min,
round(max::numeric,3) AS max, count,
round(percent::numeric,2) AS percent
FROM hist
ORDER BY min
```

Результатом виконання запиту є набір даних гістограм, що містить 22 градації поділу значень кутів нахилів з їх кількістю та відсотком зайнятості на растрі. Використовуючи отриманий результат побудовано в середовищі qgAdmin 4 графічне подання гістограм. (рис. 3.33)

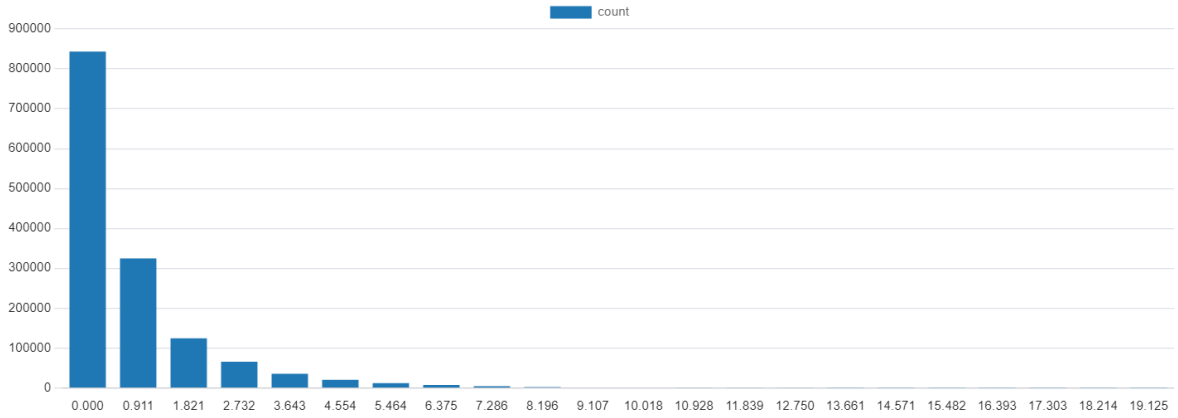


Рис. 3.33. Розподіл значень нахилу рельєфу для чарунок GRID моделі

Візуально на гістограмі значення *count* помітно, що розподіл значень має значну нерівномірність, з великою кількістю чарунків, що мають низькі значення нахилу (0 - 2). Значення з великим значенням нахилу від загального числа практично відсутнє, що свідчить про обмежену кількість крутих схилів або вершин. Найбільше чарунок має значення нахилу, близьке до 0. Для рекласифікації нахилів рельєфу був обраний поділ на 6 класів, з наступними інтервалами:

Клас 1 (0 - 1)

Клас 2 (1 - 2)

Клас 3 (2 - 5)

Клас 4 (5 - 10)

Клас 5 (10 - 15)

Клас 6 (>15)

Запит рекласифікації растру з збереженням в векторній формі:

```
CREATE TABLE dem_reclassmpoly AS
SELECT val, geom as geom
FROM (
  SELECT poly.*
  FROM dem, LATERAL
  ST_DumpAsPolygons(
    ST_Reclass(
      (ST_SLOPE(rast::raster, '1'::int, '32BF'::text, 'DEGREES'::text,
'111120'::double precision))::raster,
      '1'::int, '0-1):1, 1-2):2, 2-5):3, 5-10):4, 10-15):5, [15-90):6'::text, '32BF'::text
    )
```

```
    ) AS poly
    WHERE dem.filename = 'N50E030.hgt'
) AS foo;
```

У результаті виконання SQL-запиту рекласифікації було повернуто 103074 значень (рис. 3.34-36). Запит завершився успішно за 17 секунд та 731 мілісекунд.

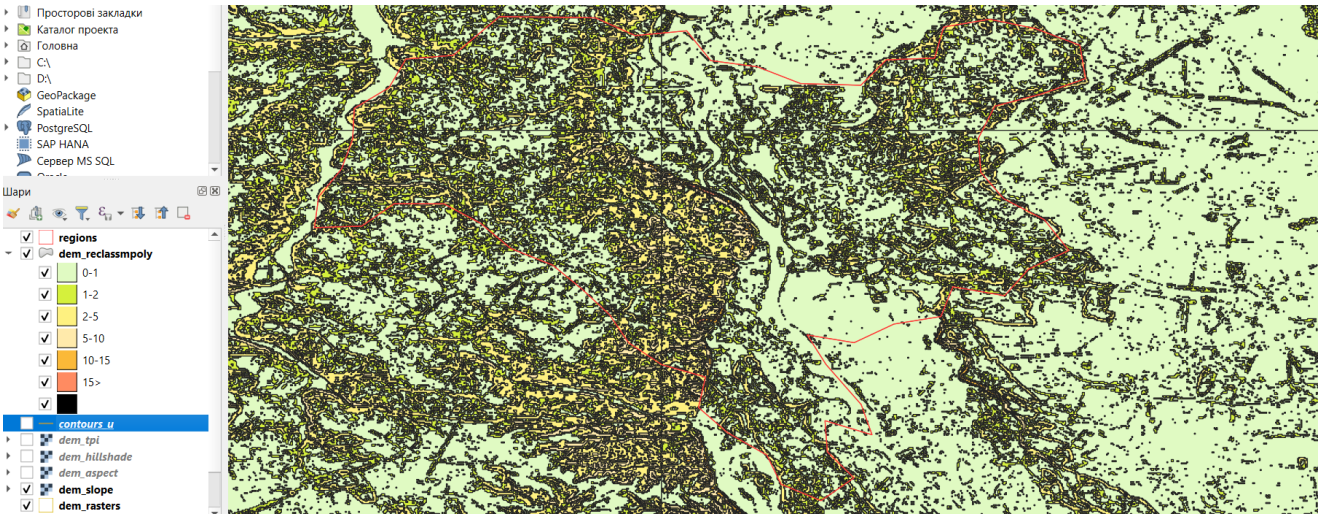


Рис. 3.34 Рекласифікована векторна модель нахилу поверхні

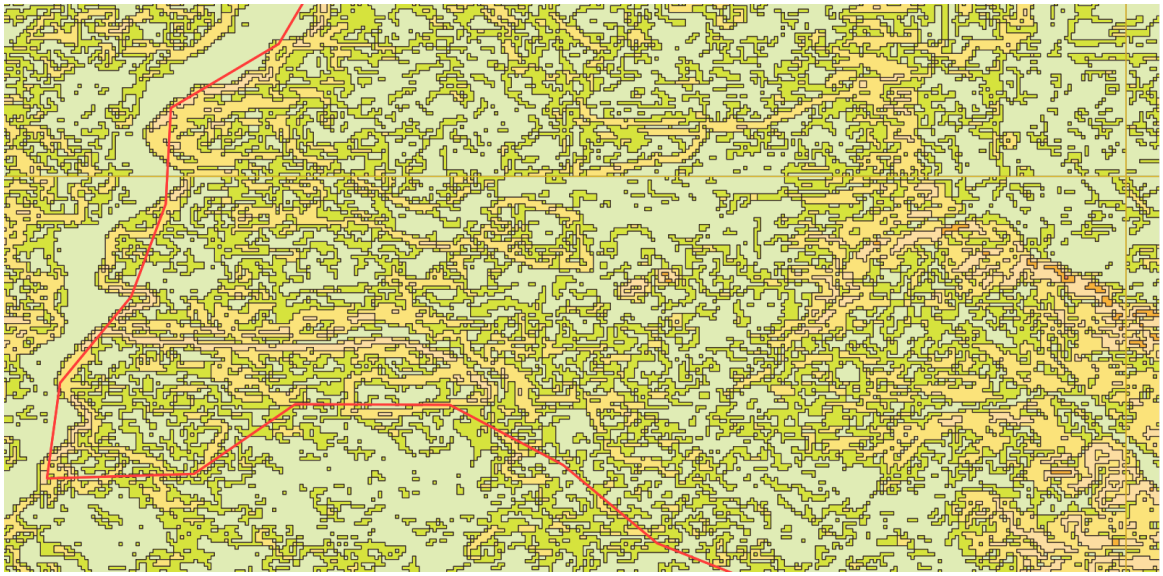


Рис. 3.35. Фрагмент рекласифікованої поверхні правого берегу Дніпра

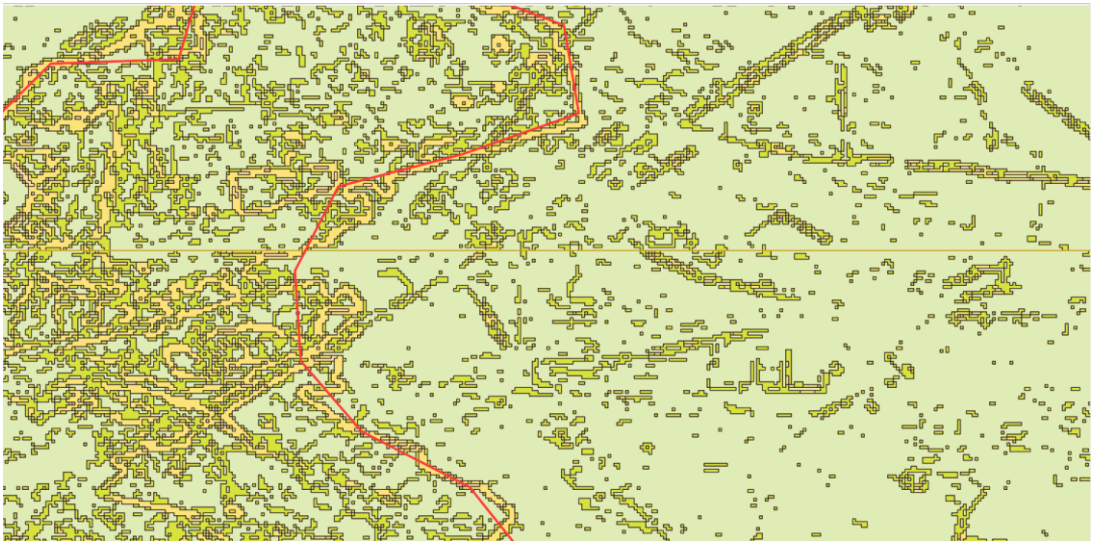


Рис. 3.36. Фрагмент рекласифікованої поверхні лівого берегу Дніпра

3.3 Реалізація прикладної SQL функцій для побудови 3D моделей ліній з використанням GRID моделі рельєфу

3.3.1 Призначення та можливі варіанти реалізації функції

Функція призначена для побудови тривимірної моделі лінії *line3d* для вихідної ламаної двовимірної лінії *line2d* типу LINESTRING з використанням GRID моделі рельєфу для визначення значень координати z (висоти) для точок *line3d*, проєкції яких POINT (x,y) належать вхідній лінії.

Функція формування 3D-ліній може скласти основу для створення тривимірних моделей з 2D-лінійних об’єктів і контурів 2D-полігонів у тривимірному просторі. Це дозволяє моделювати лінійні та контурні полігональні об'єкти з використанням GRID-моделі рельєфу.

Виходячи з [22] роботи можливо розглядати 3 варіанти вирішення задач:

За точністю визначення положення *line3d* на поверхні рельєфу, проекція якої відповідає вихідній лінії *line2d,* можливі три варіанти моделювання [22] :

1) з просторовим розрізненням вихідної лінії шляхом визначення значень координати z лише для точок вершин вихідної лінії;

2) з просторовим розрізненням GRID-моделі рельєфу шляхом визначення координат (x, y, z) для множини точок лінії *line3d*, що утворюються в результаті перетину вихідної лінії *line2d з чарунками* GRID-моделі рельєфу;

3) з просторовим розрізненням кроку сегментації вхідної лінії *line2d* для визначення набору точок (x, y) для яких визначаються значення координати z для тривимірної моделі лінії *line3d.*

Розглянемо типові схеми запитів для реалізації приведених вище варіантів моделювання тривимірної лінії *line3d* за її проекцією *line2d* з використанням GRID-моделі рельєфу.

Моделювання з роздільною здатністю вхідної 2d геометрії забезпечує отримання значення висоти лише у вершинах вхідної лінії (рис. 3.37).

Перевагою методу є збереження оригінальної роздільної здатності геометрії, тобто кількості точок, що формують лінію.

Недоліком методу є втрачання значної частина 3D-інформації через форму рельєфу яка не відтворюється повністю, значення висоти (Z) доступні лише в точках, а між ними рельєф залишається невідомим.

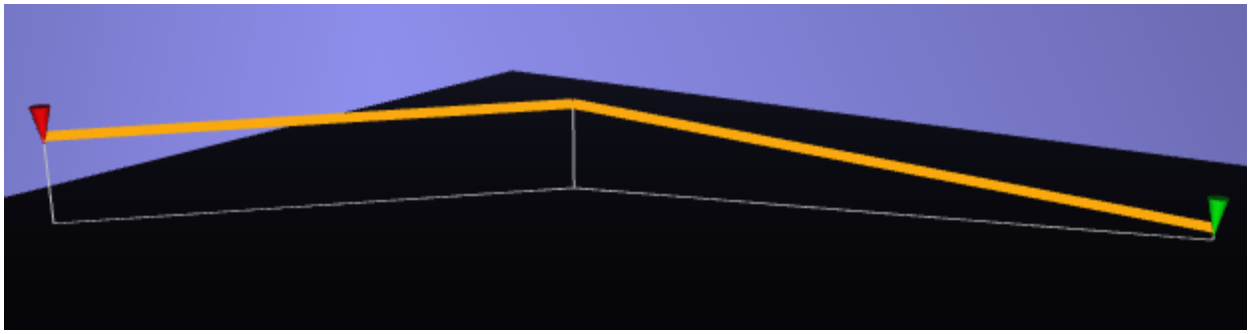


Рис. 3.37. 3D лінії за варіантом просторового розрізнення вхідної 2D лінії [22]

Запит побудови 3D лінії методом моделювання з роздільною здатністю:

```
WITH line AS
  -- Починаємо задану лінію
  (SELECT 'SRID=32632;LINESTRING (348595 4889225,352577 4887465,354784 4883841)'::geometry AS geom),
 points2d AS
  -- Витягування заданих точок
  (SELECT (ST_DumpPoints(geom)).geom AS geom FROM line),
 cells AS
  -- Отримання висоти з DEM для кожної точки
  (SELECT p.geom AS geom, ST_Value(mnt.rast, 1, p.geom) AS val
   FROM mnt, points2d p
   WHERE ST_Intersects(mnt.rast, p.geom)),
  -- Ініціалізація 3D-точки
 points3d AS
```

```
  (SELECT ST_SetSRID(ST_MakePoint(ST_X(geom), ST_Y(geom), val), 32632) AS
geom FROM cells)
-- Побудова 3D-лінії з 3D-точок
        SELECT ST_MakeLine(geom) FROM points3d;
```

Результатом є побудова 3D-лінії, яка базується на трьох заданих точкових вершин лінії. Цей підхід не забезпечує повноцінне моделювання 3D-лінії для точного розрахунку істинної довжини з використанням DEM моделі, оскільки враховуються лише визначені точки, без відображення рельєфу між ними.

Моделювання з просторовим розрізненням GRID забезпечує побудову 3D лінії *на* множині точок що утворюються в результаті перетину вихідної лінії *line2d* з чарунками GRID-моделі рельєфу із значенням висоти відповідної чарунки растру (рис. 3.38).

Перевагою методу є повне використання даних цифрової моделі рельєфу (DEM), що дозволяє максимально точно врахувати висоту та деталі рельєфу на всій довжині лінії, забезпечуючи більш реалістичну і деталізовану тривимірну модель.

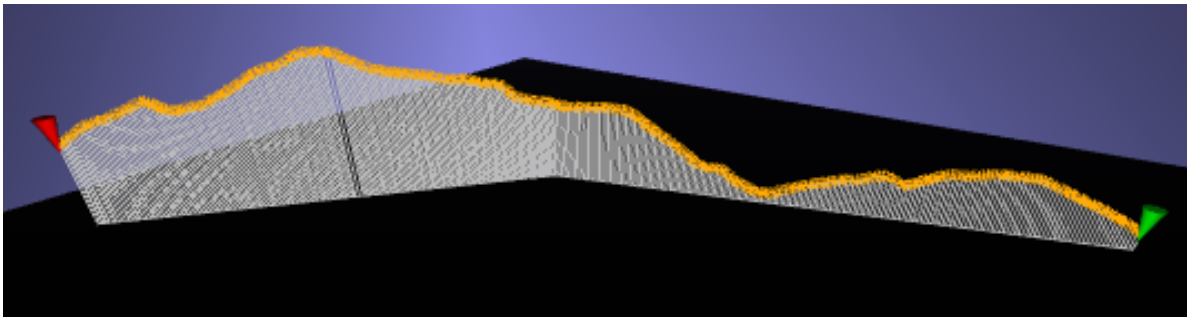


Рис. 3.38. Результат побудови 3D лінії з просторовим розрізненням GRID [22]

Недоліком методу є можливе значне збільшення кількості точок у геометрії лінії через роздільну здатність растру, що може призвести до надмірної деталізації та збільшення обсягу обчислень. Крім того, для ефективної роботи необхідно використовувати індексацію чарунок, щоб пришвидшити процес пошуку та обробки даних, особливо при роботі з великими DEM. Відсутність належної індексації може значно уповільнити виконання запитів і збільшити навантаження на систему.

Запит побудови 3D лінії з просторовим розрізненням GRID:

```
WITH line AS
  -- Починаємо задану лінію
  (SELECT 'SRID=32632;LINESTRING (348595 4889225,352577 4887465,354784 4883841)'::geometry AS geom),
 cells AS
  -- Отримання значення висоти з DEM для кожної пересіченої комірки
  (SELECT ST_Centroid((ST_Intersection(mnt.rast, line.geom)).geom) AS geom,
  (ST_Intersection(mnt.rast, line.geom)).val AS val
   FROM mnt, line
   WHERE ST_Intersects(mnt.rast, line.geom)),
  -- Створення 3D-точки та впорядковуємо їх вздовж лінії
 points3d AS
  (SELECT ST_SetSRID(ST_MakePoint(ST_X(cells.geom), ST_Y(cells.geom), val), 32632) AS geom
   FROM cells, line
   ORDER BY ST_Distance(ST_StartPoint(line.geom), cells.geom))
-- Побудова 3D-лінії на основі створених 3D-точок
SELECT ST_MakeLine(geom) FROM points3d;
```

Результатом є модель 3D-лінії на основі GRID, де визначається висота для кожної чарунки, з якою перетинається лінія. Однак, відсутність індексації чарунок призводить до того, що функцію ST_Intersection потрібно виконувати для кожної чарунки, що значно збільшує час виконання запиту.

Моделювання з просторовим розрізненням інтервалу сегментації вхідної 2D лінії забезпечує обчислення значення висоти для точок 3D лінії в точках сегментації вхідної 2D лінії з заданим крок (рис. 3.39).

Перевагою методу є можливість чітко контролювати інтервал (крок), через який будується лінія.

Недоліком методу є складність у визначенні оптимального кроку для різних форм поверхні.

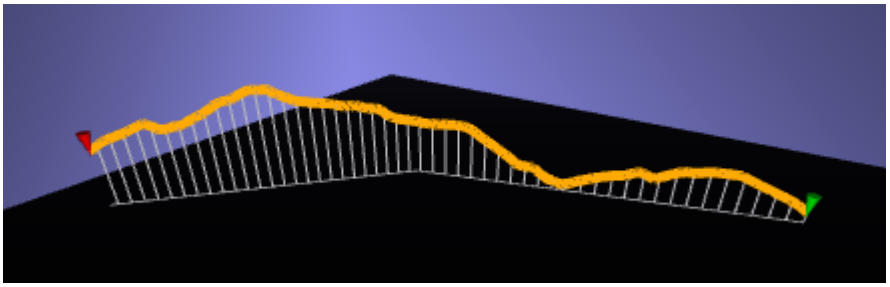


Рис. 3.39. Результат побудови 3D лінії з заданим кроком [22]

Запит побудови 3D лінії з заданим кроком:

```
WITH line AS
  -- Починаємо задану лінію
  (SELECT 'SRID=32632;LINESTRING (348595 4889225,352577 4887465,354784 4883841)'::geometry AS geom),
 linemesure AS
  -- Додавання міри до лінії для визначення кроків
  (SELECT ST_AddMeasure(line.geom, 0, ST_Length(line.geom)) as linem,
        generate_series(0, ST_Length(line.geom)::int, 50) as i
   FROM line),
 points2d AS
  (SELECT ST_GeometryN(ST_LocateAlong(linem, i), 1) AS geom FROM linemesure),
 cells AS
  -- Отримання висоти для кожної точки
  (SELECT p.geom AS geom, ST_Value(mnt.rast, 1, p.geom) AS val
   FROM mnt, points2d p
   WHERE ST_Intersects(mnt.rast, p.geom)),
  -- Ініціалізація 3D-точок
 points3d AS
  (SELECT ST_SetSRID(ST_MakePoint(ST_X(geom), ST_Y(geom), val), 32632) AS geom FROM cells)
-- Побудова 3D-лінії
SELECT ST_MakeLine(geom) FROM points3d;
```

Результатом є побудова 3D-лінії з використанням заданого кроку, що надає менш точну модель порівняно з методом "з розрізненням GRID". Однак, при правильному виборі кроку цей метод дозволяє скоротити час моделювання, оскільки аналізуються лише ті чарунки, які відповідають координатам сегментованої 2D.

Виходячи з очевидного недоліку першого варіанту моделювання 3D лінії з роздільної здатністю вихідної 2D лінії в роботі здійснено реалізацію і дослідження функцій моделювання 3D лінії за варіантами 2 та 3.

3.3.2 Реалізація SQL функції побудови 3D моделей ліній з просторовим розрізненням GRID рельєфу

Узагальнений алгоритм розробленої функції *_line3d(line2d geometry)*, в якій реалізовано варіант визначення точок місцеположення 3D лінії на поверхні

з просторовим розрізненням GRID моделі рельєфу (рис. 3.40) можна описати такими основними блоками (кроками)

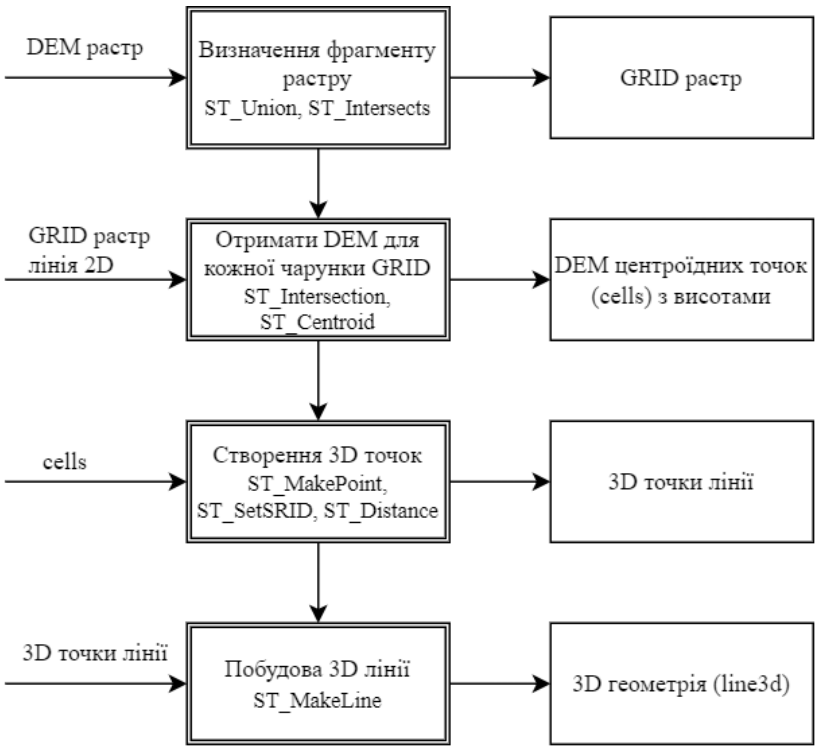


Рис. 3.40. Загальна схема алгоритму функції line3d(line2d geometry)

Функцію реалізовано на мові процедурної мови PL/pgSQL ОР СКБД PostgreSQL з використанням таких функцій PostGIS:

1. ST_Union(dem.rast) – об’єднання фрагментів растру в єдиний масив при виконанні визначеної в запиті умови;

2. ST_Intersects(g1, g2) – перевіряє, чи перетинаються дві геометрії (чи фрагмент растру g1 перетинається з лінією g2).

3. ST_ConvexHull(dem.rast) – обчислює опуклу оболонку для заданого растру, щоб визначити межі його області.

4. ST_Intersection(grid.rast, line2d) – визначає область перетину між фрагментом растру grid.rast та лінією line2d.

5. ST_Centroid(geom) – обчислює центроїд геометрії перетину.

6. ST_MakePoint(x, y, z) – створює 3D точку з координатами X, Y та значенням висоти Z.

7. ST_SetSRID(geom, srid) – встановлює систему координат для геометрії 4326.

8. ST_Distance(geom1, geom2) – обчислює відстань між двома геометріями (використовується для впорядкування точок за відстанню від початкової точки лінії).

9. ST_MakeLine(geom) – створює лінію з набору точок.

Текст функції line3d(line2d geometry):

Замінити на: Текст розробленої функції line3d(line2d geometry)подано в додатку Б.1.1.

Тестування функції _line3d(geometry) виконувалося на прикладі побудови 3D моделі для 2D ламаних ділянок осьових ліній вулиць з використанням набору даних OpenStreetMap, що завантажений в таблицю *str_test* бази даних *dem.* Для тестування виконувався запит побудови 3D моделі ділянки осьової лінії бульвару Тараса Шевченка із ідентифікатором gid=2:

```
SELECT ST_AsText (_line3d(str_test.geom))
FROM str_test
WHERE str_test.gid = 2;
Результат:_____
Successfully run. Total query runtime: 1 min 22 secs. 1 rows affected.
"LINESTRING Z (30.51266048541575 50.44344005226915 177,30.51333333229689
50.443325344695324 176,30.514344450253148 50.44315301221748 174)"
```

В результаті запиту для 2D лінії заданої ділянки вулиці функцією *_line3d(str_test.geom)* побудовано тривимірну лінію, яка в базі даних подається типом LINESTRING Z. Але, незважаючи на те, що ділянка була короткою, запит побудови її 3D моделі виконувався 1 хв. 22 сек.

Значний час побудови 3D моделі лінії з роздільною здатністю GRID моделі рельєфу зумовлений тим, що основу моделювання складає пошук точок перетину вихідної лінії з полігонами прямокутників пікселів растру GRID моделі. Ці прямокутники створюються в процесі обчислення для усього блоку растру без їх просторового індексвання. З цієї причини пошук прямокутників пікселів, з якими перетинається ламана лінія ділянки вулиці, потребує перебору та просторового аналізу прямокутників усіх пікселів растру, число яких може сягати кількох сотень тисяч. Незважаючи на найвищу точність моделювання положення точок лінії на поверхні з використанням роздільної здатності GRID-

моделі рельєфу, час виконання відповідної функції обмежує доцільність її практичного використання лише для невеликих наборів даних.

3.3.3 Реалізація SQL функцій побудови 3D моделей ліній з просторовим розрізненням заданого інтервалу

Узагальнений алгоритм розробленої функції ***_line3ds(line2d geometry, sgm_length float),*** в якій реалізовано варіант визначення точок місцеположення 3D лінії на поверхні з просторовим розрізненням заданого кроку сегментації вихідної 2D лінії (рис. 3.41).

Функцію реалізовано на мові процедурної мови PL/pgSQL ОР СКБД PostgreSQL з використанням таких функцій PostGIS:

ST_Segmentize(line2d::geography, sgm_length) – розбиває вхідну лінію (line2d) на менші сегменти, довжина кожного сегменту не перевищує задане значення (sgm_length).

ST_Union(dem.rast) – об’єднання фрагментів растру в єдиний масив для забезпечення цілісності GRID.

ST_Intersects(lsgm.geom, ST_ConvexHull(dem.rast)) – перевіряє, чи перетинається сегментована лінія (lsgm.geom) з опуклою оболонкою (ST_ConvexHull(dem.rast)) растру.

ST_ConvexHull(dem.rast) – обчислює опуклу оболонку для растру, що дозволяє визначити його межі.

ST_SetZ(grid.rast, lsgm.geom, resample => 'bilinear') – встановлює значення Z (висоти) для кожної точки сегментованої лінії (lsgm.geom), використовуючи білінійну інтерполяцію значень висоти з растру (grid.rast).

ST_DumpPoints(geom) – розпаковує геометрію (geom) у вигляді набору точок без дублювання.

ST_MakeLine(points3d.geom) – створює 3D лінію з множини 3D точок (points3d.geom), що були визначені з використанням даних про висоту рельєфу.

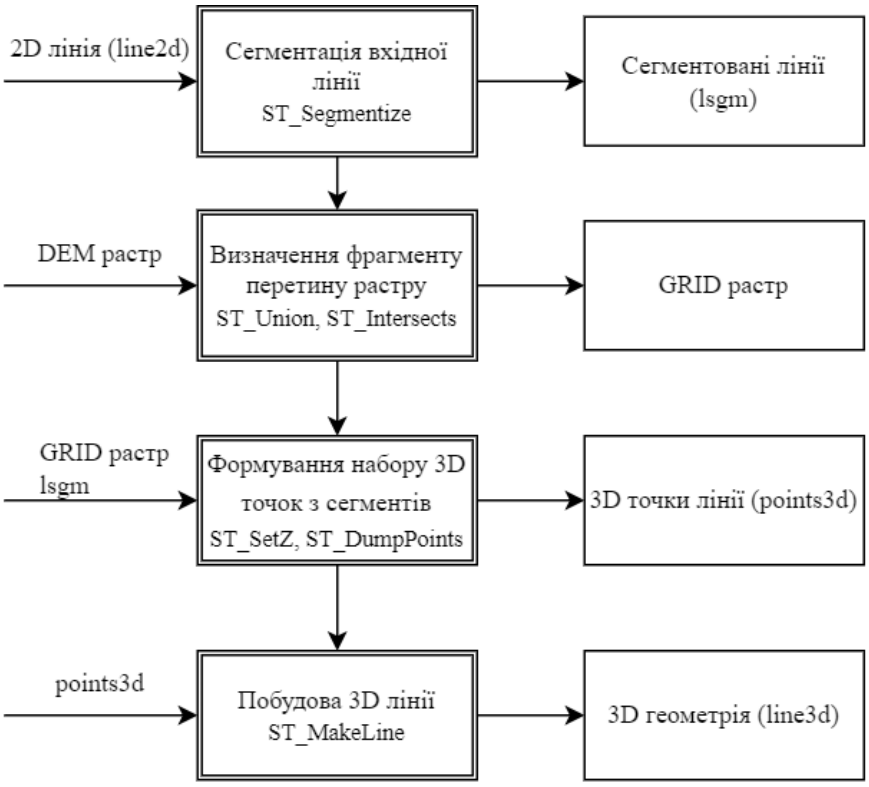


Рис. 3.41. Загальна схема алгоритму функції _line3ds(line2d geometry, sgm_length float)

Текст розробленої функції _line3ds(line2d geometry, sgm_length float) подано в додатку Б.1.2.

Тестування функції _line3ds( ) виконано на прикладі побудови 3D моделі для 2D ламаних ділянок осьових ліній вулиць з використанням набору даних OpenStreetMap, що завантажений в таблицю *str_test* бази даних *dem.* Для тестування виконувалась серія запитів побудови 3D моделі ділянок осьової лінії бульвару Тараса Шевченка. Запити та результати їх виконання подано нижче.

Запит 1:

```
SELECT ST_AsText (_line3ds(str_test.geom, 50.0))
FROM str_test
WHERE str_test.gid = 1;
```

Successfully run. Total query runtime: 676 msec. 1 rows affected.
"LINESTRING Z (30.4944945 50.4463704 135.23230745598255,30.4950213 50.4462878 135.64770059843354)"

Запит 2:

```
SELECT ST_AsText (_line3ds(str_test.geom, 50.0))
FROM str_test
WHERE str_test.gid = 2;
```

Successfully run. Total query runtime: 237 msec. 1 rows affected.

"LINESTRING Z (30.5124043 50.4434837 177.18480172959664,30.512513 50.4434652 176.83109708799662,30.513091051203713 50.44336665143174 176.3074718197053,30.5136691 50.4432681 175.3191794223899,30.5140208 50.4432082 174.52652400639252,30.514378550461107 50.4431472005484 174.0529849323125,30.5147363 50.4430862 173.68841033921473,30.5149389 50.4430517 173.39841768160096)"

Запит 3:

```
SELECT ST_AsText (_line3ds(str_test.geom, 50.0))
FROM str_test
WHERE str_test.gid = 3;
```

Successfully run. Total query runtime: 247 msec.1 rows affected.
LINESTRING Z (30.5171977 50.4426882 166.18419428162053,30.5172358 50.4426821 166.12184041923047,30.517305 50.442671 166.00678320001236,30.517657027912414 50.44261506621689 165.82510964980042,30.51800905499273 50.442559131371794 166.33218011575556,30.51836108124095 50.442503195464745 166.7997043405704,30.518713106657042 50.442447258495704 164.20794619809723,30.519065131241003 50.44239132046471 161.50200446996672,30.519417154992833 50.442335381371755 158.19653369107868,30.5197691779125 50.44227944121686 154.92033932534764,30.5201212 50.4422235 152.85062902398096).

За результатами тестування функції побудови 3D лінії з просторовим розрізненням заданого інтервалом сегментації вихідної 2D лінії для обчислення значення координат z точок лінії на поверхні за GRID моделлю рельєфу встановлено, що час моделювання на кілька порядків менше, порівняно з варіантом моделі з роздільною здатністю GRID (230 – 670 мілісекунд порівняно з 1 хв. 22 сек на одну лінію). Це зумовлено тим, що для інтерполяції значення z координати точок моделі піксели растру GRID вибираються безпосередньо за координатами точок сегментів лінії без великого числа просторових побудов та аналізу перетину лінії з чарунками растру. Для досягнення потрібної точності моделювання положення лінії на поверхні можна вибирати оптимальний крок сегментації, в залежності від характеристик форми рельєфу та ліній об’єктів моделювання. Для штучних лінійних об’єктів типу вулиці, трубопроводи тощо,

сегментації лінії крок в 50-100 м може бути цілком прийнятна для вирішення практичних завдань.

3.3.4 Обчислення довжини ліній на фізичні поверхні з використанням 3D моделей об'єктів і рельєфу

Обчислення довжини лінії на фізичній поверхні належить до типових операцій для прикладних задач, що потребують визначення розмірів об'єктів для обчислення обсягів робіт та/або витрат будівельних матеріалів. Будівництво, реконструкція, експлуатація вулиць та їх дорожнього покриття – прикладних завдань, які потребують обчислення геометричних параметрів вулиць на фізичній поверхні, а не в проєкції топографічного плану.

В бібліотеці PostGIS надається низка функцій для обчислення довжини 2D та 3D лінійних об'єктів, як на площині, так і на поверхні еліпсоїда з параметрами, визначеними в аргументах функції. Для практичних задач доцільно використовувати саме функцію обчислення довжини лінії на еліпсоїді *ST_LengthSpheroid(geometry, spheroid),* яка повертає значення довжини 2D або 3D лінії типу geometry або geography, обчислену на поверхні сфероїда (еліпсоїда), параметри якого задаються текстовим рядком. Для референцної системи координат WGS 84 параметри еліпсоїда в PostGIS можна подавати як: 'SPHEROID["GRS_1980",6378137,298.257222101]') або 'SPHEROID["WGS 84",6378137,298.257223563]').

Для обчислення довжини 3D ліній для таблиць з об'єктами, що містять їх 2D моделі, можна безпосередньо використовувати функцію *ST_LengthSpheroid(geometry, spheroid)* з викликом розробленої функції створення 3D моделі лінії *_line3ds(line2d geometry, sgm_length float)* як аргументу для параметру *geometry* у виклику функції *ST_LengthSpheroid*. Відповідні запити можна створювати, наприклад, за такою схемою:

```
SELECT ST_LengthSpheroid(_line3ds(geom,50.),
 'SPHEROID["WGS 84",6378137,298.257223563]')
FROM str_test
WHERE gid = 1;
```

З метою поліпшення зручності виконання подібних операцій в роботі на основі функції *geometry _line3ds(geom geometry, sgm_length float)* реалізовано функцію *double precision* ***_line3dslg****(geom geometry, sgm_length float)*, в якій для вхідної 2D лінії створюється 3D модель з просторовим розрізненням заданого інтервалу сегментації вхідної лінії та обчислюється її довжина на еліпсоїді WGS 84, яка й повертається як числовий результат функції. Текст функції подано в додатку Б.1.1.3.

Схема запиту для визначення довжини 3D ліній для таблиць з об'єктами, що містять їх 2D моделі, має такий вид:

```
SELECT line3dslg (geom, 50.) FROM str_test
WHERE gid = 1;
```

Тестування функції ***_line3dslg (…)*** виконувалося на тестовому наборі даних ділянок 2D осьових ліній вулиць міста Києва – таблиця str_test в базі даних із атрибутами *l_2d* і *l_3d* відповідно довжини 2D та 3D ліній ділянок вулиць на еліпсоїді, а також l_2dmc – довжини ділянок вулиць на площині, обчисленої за координатами в проєкції Меркатора.

Запит на оновлення значення атрибута *l_3d* довжини 3D ліній ділянок вулиць в таблиці *str_test:*

```
UPDATE str_test
SET l_3d = _line3dslg(str_test.geom, 50.);
_____________________________________________________
UPDATE 36
Query returned successfully in 6 secs 12 msec.
```

Запит на порівняння значень довжин ділянок вулиць таблиці str_test:

```
SELECT gid, name, l_2d::numeric(10,3) as l2d,
        l_3d::numeric(10,3) l3d,
        l_2dmc::numeric(10,3) as l2dmc,
        (l_3d-l_2d)::numeric(10,3) as delta,
        (l_2dmc-l_2d)::numeric(10,3) as delta_mc
        FROM str_test;
```

Результати обчислення довжини ділянок вулиць тестового набору геопросторових даних для м. Києва

№	Назва вулиці	Довжина ділянок вулиць, м			Різниця довжин	
		2D, на еліпсоїді	3D, на еліпсоїді	2D, на проєкції	на еліпсоїді	порівняно з L2D на проєкції
1	"Тараса Шевченка бульвар"	38.530	38.532	60.385	0.002	21.855
2	"Тараса Шевченка бульвар"	186.346	186.391	292.027	0.045	105.681
3	"Тараса Шевченка бульвар"	214.009	214.794	335.373	0.785	121.364
4	"Тараса Шевченка бульвар"	165.470	165.646	259.310	0.176	93.840
5	"Тараса Шевченка бульвар"	423.263	424.028	663.327	0.765	240.064
6	"Тараса Шевченка бульвар"	426.952	427.216	669.094	0.264	242.142
7	"Тараса Шевченка бульвар"	423.341	423.688	663.463	0.347	240.122
8	"Круглоуніверситетська "	299.311	299.940	469.053	0.629	169.742
9	"Круглоуніверситетська "	309.967	311.913	485.758	1.946	175.791
10	"Велика Васильківська "	61.192	61.194	95.833	0.002	34.641
11	"Велика Васильківська "	107.914	107.979	169.108	0.065	61.194
12	"Велика Васильківська "	236.857	236.969	371.018	0.112	134.161
13	"Велика Васильківська "	62.621	62.631	98.088	0.010	35.467
14	"Велика Васильківська "	207.367	207.643	324.938	0.276	117.571
15	"Велика Васильківська "	206.654	206.770	323.754	0.116	117.100
16	"Велика Васильківська "	112.034	112.042	175.455	0.008	63.421
17	"Велика Васильківська "	11.191	11.193	17.529	0.002	6.338
18	"Велика Васильківська "	56.650	56.670	88.735	0.020	32.085
19	"Велика Васильківська "	225.462	225.650	353.249	0.188	127.787
20	"Велика Васильківська "	221.595	221.650	347.176	0.055	125.581
21	"Велика Васильківська "	128.831	128.863	201.771	0.032	72.940
22	"Велика Васильківська "	243.779	244.155	381.828	0.376	138.049
23	"Велика Васильківська "	233.085	233.159	365.063	0.074	131.978
31	"Велика Васильківська "	11.433	11.433	17.915	0.000	6.482
24	"Велика Васильківська "	595.062	595.126	932.186	0.064	337.124
25	"Велика Васильківська "	222.298	222.589	348.307	0.291	126.009
26	"Велика Васильківська "	6.187	6.194	9.695	0.007	3.508
27	"Велика Васильківська "	62.215	62.274	97.486	0.059	35.271
28	"Велика Васильківська "	45.096	45.165	70.662	0.069	25.566
29	"Велика Васильківська "	107.423	107.581	168.261	0.158	60.838
30	"Велика Васильківська "	85.239	85.523	133.560	0.284	48.321
32	"Велика Васильківська "	29.901	29.906	46.855	0.005	16.954
33	"Велика Васильківська "	174.870	174.972	274.028	0.102	99.158

№	Назва вулиці	Довжина ділянок вулиць, м			Різниця довжин	
		2D, на еліпсоїді	3D, на еліпсоїді	2D, на проєкції	на еліпсоїді	порівняно з L2D на проєкції
34	"Велика Васильківська "	52.218	52.219	81.780	0.001	29.562
35	"Велика Васильківська "	90.727	90.738	142.092	0.011	51.365
36	"Велика Васильківська "	106.437	106.438	166.788	0.001	60.351

Запит для обчислення загальної довжини вулиць тестового набору:

```
SELECT name, sum(l_2d)::numeric(10,3) as l2d,
        sum(l_3d)::numeric(10,3) l3d,
        sum(l_2dmc)::numeric(10,3) as l2dmc,
        (sum(l_3d)-sum(l_2d))::numeric(10,3) as delta,
        (sum(l_2dmc)-sum(l_3d))::numeric(10,3) as delta_mc
        FROM str_test
        GROUP BY name;
```

Результат виконання запиту:

Назва вулиці	Довжина вулиць, м			Різниця довжини	
	2D, на еліпсоїді	3D, на еліпсоїді	2D, на проєкції	на еліпсоїді	порівняно з L2D на проєкції
"Круглоуніверситетська"	609.278	611.853	954.811	2.575	342.958
"Велика Васильківська"	3704.338	3706.724	5803.161	2.386	2096.437
"Тараса Шевченка бульвар"	1877.911	1880.295	2942.979	2.384	1062.684

За обчисленнями довжин ділянок вулиць тестового набору для міста Києва отримано очікувані результати, а саме: *незначну різницю (порядку одиниць метрів) між довжинами 2D і 3D ліній, визначених на еліпсоїді, та суттєву різницю (сотні і тисячі метрів) між довжинами на еліпсоїді на площині проєкції Меркатора.*

3.4 Створення і використання TIN моделі рельєфу в середовищі СКБД PostgreSQL/PostGIS

3.4.1 Технологічна схема дослідного створення і використання TIN моделі

TIN-модель в середовищі СКБД PostgeSQL/PostGIS опрацьовувалась за технологічною схемою (рис. 3.39), яка складається з таких етапів:

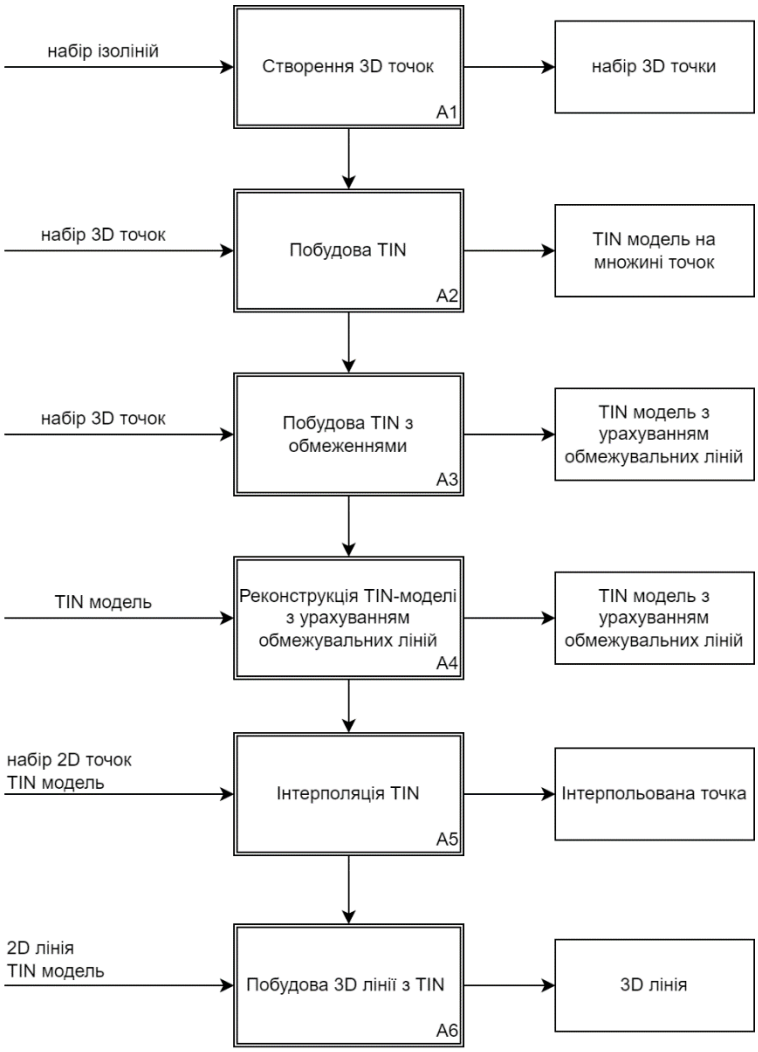


Рис. 3.42. Технологічна схема дослідницького використання TIN-моделі рельєфу

A1) отримання 3D точок на основі набору ізоліній;

A2) побудова TIN-моделі без обмежень для набору 3D точок з використанням функції PostGIS ST_DelaunayTriangles;

A3) побудова TIN-моделі з врахуванням обмежувальних ліній з використанням функції PostGIS ST_ConstrainedDelaunayTriangles;

A4) розробка і використання функції для реконструкції TIN-моделі Делоне за набором обмежувальних ліній;

A5) реалізація функції інтерполяції висоти довільної 2D точки з використанням TIN-моделі

A6) використання TIN-моделі для підняття 2D лінії до 3D, враховуючи рельєф.

Докладніше зміст етапів та їх результати описано в пунктах 3.4.2 – 3.4.7.

3.4.2 Формування набору 3D точок для TIN моделі

Для зберігання набору3D точок в системі координат WGS 84 (SRID=4326) в базі даних створюється таблиця *pts3d* за таким запитом:

```
CREATE TABLE pts3d(
        gid serial PRIMARY KEY,
        geom geometry(POINTZ,4326) );
```

Формування набору 3D точок для TIN моделі виконувалося з використанням набору ***36 412*** 2D точок, який створено за точками контурів ізоліній рельєфу топографічної карти масштабу 1:200 000 на дослідну територію (рис. 3.43). Набір 2D точок завантажено в таблицю *point_kyiv* бази даних. Атрибут **hg** в таблиці *point_kyiv* містить значення висоти перерізу рельєфу ізолінії, контуру якого належить точка.

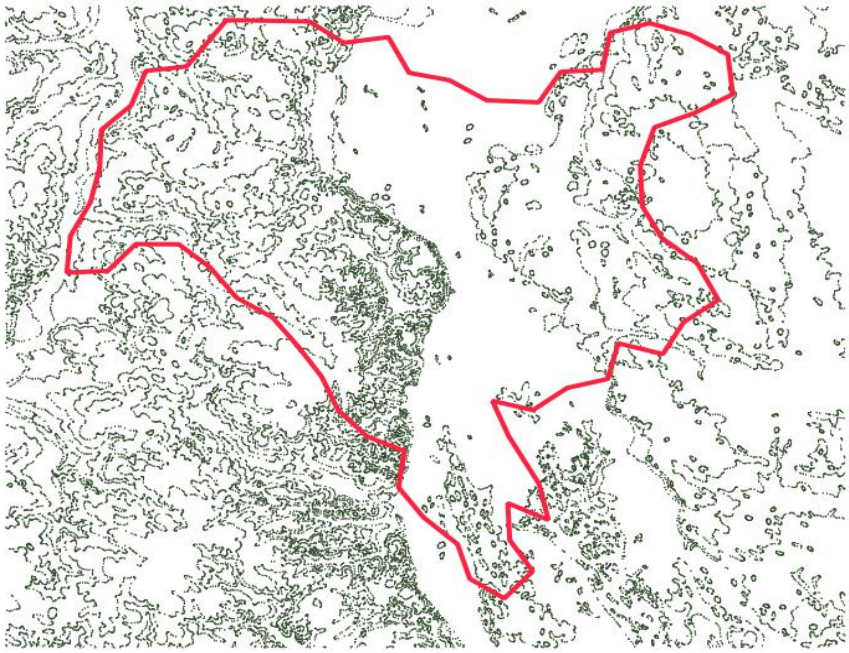


Рис. 3.43. Зображення набору із 36 412 точками для формування TIN моделі рельєфу на дослідну територію міста Києва

Набір 3D точок для таблиці *pts3d* формувався за таким запитом:

```
INSERT INTO pts3d (gid, geom)
SELECT gid, ST_Force3DZ(p.geom, p.hg)
FROM point_kyiv AS p;
```

Основу запиту на створення набору 3D точок складає виконання функції PostGIS *ST_Force3DZ(geom, zvalue)*, яка для вхідної 2d геометрії, заданої в параметрі geom, повертає 3D геометрію, в якій 2d точки вхідної геометрії

розширені координатою Z із значенням, що задано вхідним параметром *zvalue y* запиті це аргумент *p.hg*.

Запит для тестування вмісту таблиці pts3d:

```
Select st_astext(geom) from pts3d
LIMIT 4;
Результат:_____
"POINT Z (30.742868999999992 50.595043000000004 120)"
"POINT Z (30.742466 50.594674999999995 120)"
"POINT Z (30.741374000000004 50.594024000000005 120)"
"POINT Z (30.74094 50.593279 120)"
```

3.4.3 Реалізація функції формування набору 3D трикутників TIN моделі рельєфу

Для побудови TIN моделі в PostGIS надається базова функція тріангуляції Делоне *ST_DelaunayTriangles* (див. п. 2.3.2).

```
geometry:ST_DelaunayTriangles(geometry g1,float tolerance = 0.0,int4 flags=0);
```

Функція *ST_DelaunayTriangles* обчислює тріангуляцію Делоне на множині вершин вхідної геометрії *g1*, яка повинна бути задана як єдиний набір 2D або 3D точок типу *MULTIPOINT*. Параметр *tolerance* забезпечує можливість задавати допуск мінімальної відстані між вхідними точками для виключення «близьких» вхідних вершин із розгляду, що покращує надійність процесу у деяких ситуаціях.

Функція повертає результат як єдину колекцію геометричних об’єктів мережі трикутників, тип яких залежить від значення вхідного параметра *flags*, а саме:

0 – GEOMETRYCOLLECTION трикутних полігонів (за замовчуванням);

1 – MULTILINESTRING ребер тріангуляції;

2 – TIN тріангуляції, що подається спеціальним типом даних TIN для поверхні нерегулярної мережі трикутників як колекції трикутників, визначених типом даних *Triangles* (трикутники).

Таблицю *tin_dln* з одним записом, що містить колекцію трикутних полігонів TIN поверхні ***для набору 36 412*** 3D точок із таблиці *pts3d*, можна створити з використанням такого простого запиту:

```
CREATE TABLE tin_dln AS
WITH pts AS
(
SELECT geom AS pt
FROM pts3d
)
SELECT ST_DelaunayTriangles(ST_Union(pt::geometry), 0.0, 0) AS the_geom
FROM pts;
```

Варто зазначити, що час виконання запиту 808 msec, але результат подається одним записом в таблиці *tin_dln* як колекція із 71 940 3D трикутників. Цю колекцію зручно використовувати для подання результату функціє *ST_DelaunayTriangles*, оскільки забезпечується незалежність функції від структури бази даних, в якій буде зберігатись і використовуватися TIN модель. Але для ефективного використання TIN моделі в задачах морфологічного аналізу рельєфу та/або побудови 3D об'єктів необхідно забезпечити прямий доступ до моделі кожного трикутника TIN з використанням просторового індексу для окремих трикутників, а не для їх колекції як єдиного об'єкта.

З цією метою в роботі реалізовано прикладну SQL функцію ***_tin2triangles***, яка опрацьовує колекцію трикутників тріангуляції Делоне із таблиці *tin_dln*, послідовно «витягує» окремі екземпляри трикутників із колекції та записує їх в таблицю *tin_dln3d* як окремі полігони.

Створення таблиці ***tin_dln3d*** для зберігання 3D полігонів трикутників TIN моделі у разі використання системи координат WGS 84 (SRID = 4326) може бути виконано, наприклад, за таким запитом:

```
CREATE TABLE tin_dln3d(
        gid serial PRIMARY KEY,
```

```
                geom geometry(POLYGONZ,4326) );
```

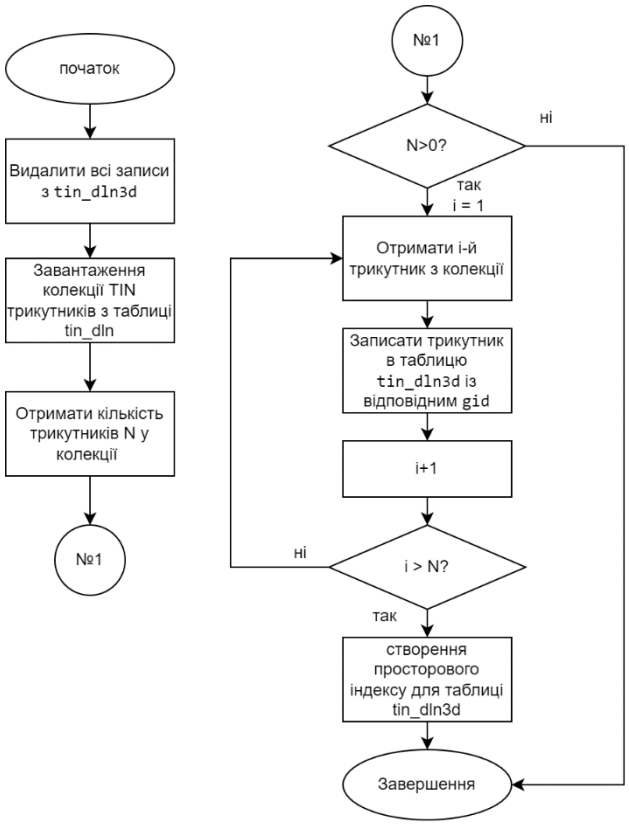


Рис. 3.44. Загальна схема алгоритму функції *tin_dln()* зі створенням набору трикутників тріангуляції Делоне для набору 3d точок

Основні блоки алгоритму функції (рис. 3.44) призначені для:

завантаження єдиної колекції TIN трикутників із таблиці *tin_dln*;

розпаковка колекції 3D полігонів трикутників зі зберіганням моделі кожного полігону в окремому записі таблиці *tin_dln3d;*

створення просторового індексу для таблиці *tin_dln3d* за схемою R-дерева (ієрархії прямокутників) загального пошукового дерева GIST (Generalized Search Trees) PostGIS для атрибута геометрії трикутників таблиці *tin_dln3d.*

Текст реалізації функції *tin2triangels()* на мові PL/pgSQL приведено в додатку Б, п. Б.2.1. Тестування функції виконувалося на перетвореній колекції типу TIN Делоне тріангуляції для рельєфу на дослідну територію м. Києва. В результаті виконання функції *tin2triangels()* в таблицю *tin_dln3d* записано 71 940 полігонів трикутників. Наявність полігонів трикутників в таблиці *tin_dln3d* можна проконтролювати за зображенням електронної карти TIN моделі (рис.3.46).

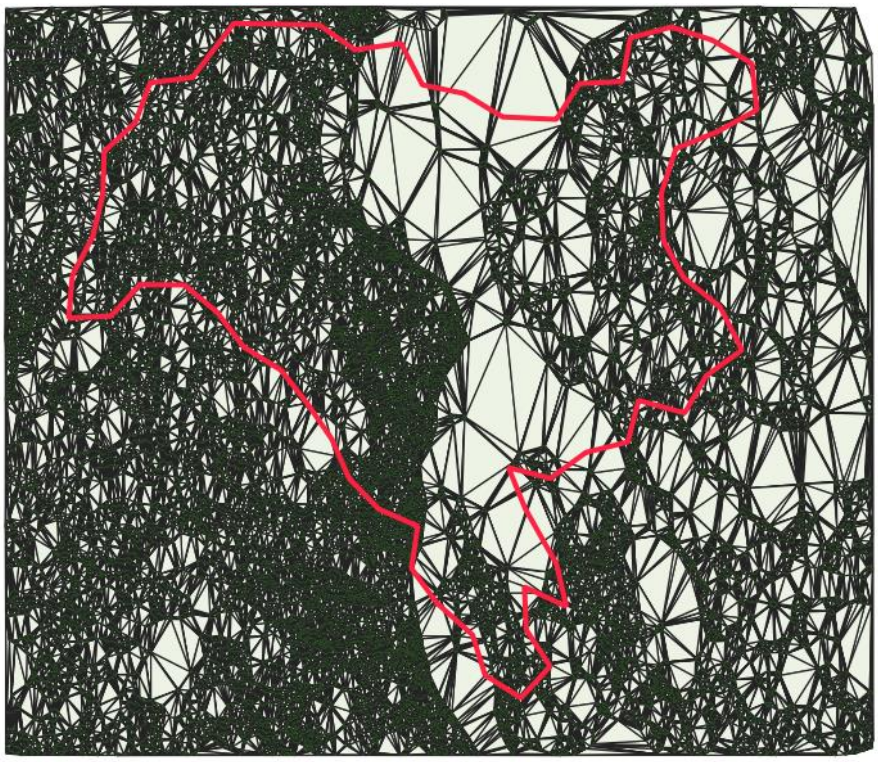


Рис. 3.46. Зображення 71 940 трикутників TIN моделі рельєфу на дослідну територію м. Києва

Наявність саме 3D полігонів можна перевірити також за таким запитом:

```
SELECT ST_AsText(geom) FROM tin_dln3d
LIMIT 2;
```

Результат запиту:

"POLYGON Z ((3359953.2963066157 6516840.598508985 150,3359912.1080950224 6515197.716442611 150,3360044.021691612 6516718.179751677 150,3359953.2963066157 6516840.598508985 150))"

"POLYGON Z ((3359953.2963066157 6516840.598508985 150,3360044.021691612 6516718.179751677 150,3360421.6174043827 6516686.700938486 150,3359953.2963066157 6516840.598508985 150))"

Час виконання функції на цьому тестовому наборі порядку 15 -20 хв., що набагато більше за час виконання запиту (808 мілісекунд) на створення власне колекції трикутників TIN з використанням функції *ST_DelaunayTriangles,* що розглядався вище в цьому підрозділі. Це вказує на те, що левова частка часу при побудові TIN моделі як набору полігонів окремих трикутників витрачається на розпаковку колекції типу TIN на окремі трикутники та їх зберігання в таблиці бази даних. Разом з цим, підвищення ефективності багаторазового використання TIN моделі як набору полігонів окремих

трикутників у задачах аналізу рельєфу, компенсує одноразові додаткові витрати на створення такої моделі.

3.4.4 Реалізація функції формування набору 3D трикутників TIN моделі рельєфу з обмеженнями

Побудову TIN моделі для набору точок і точок структурних ліній та/або точок контурів обмежувальних полігонів в Postgis забезпечує базова функція *geometry ST_ConstrainedDelaunayTriangles (geometry g1)* (див. п.2.3.3), яка реалізована з використанням бібліотеки SFCGAL та доступна у просторовому розширенні *postgis_sfcgal*, яке потрібно підключити до БГД після розширення postgis.

```
geometry ST_ConstrainedDelaunayTriangles(geometry g1);
```

На вхід функції *ST_ConstrainedDelaunayTriangles* у разі її використання для створення 3D TIN моделі рельєфу, необхідно подати колекцію геометричних елементів g1 як набір мультитипних даних точок, структурних ліній та контурів обмежувальних полігонів.

Як і *ST_DelaunayTriangles,* функція *ST_ConstrainedDelaunayTriangles* створює TIN модель з одним записом, що містить колекцію трикутних полігонів TIN поверхні для вхідної колекції точок, точок обмежувальних ліній або точок контурів обмежувальних полігонів. Створена TIN модель з обмеженнями обов'язково містить трикутники, сторони яких збігаються з відрізками заданих обмежувальних геометричних елементів (структурних ліній та/або контурів полігонів). Це дає можливість створити TIN модель, що відповідає морфології реального рельєфу, оскільки геометричні елементи обмежень використовуються для моделювання особливих місць зміни форми рельєфу (хребти, тальвеги долин, берегові лінії водотоків та інших гідрографічних об'єктів, насипи, обриви тощо).

Оскільки результатом виконання функції тріангуляції з обмеженнями *ST_ConstrainedDelaunayTriangles* є TIN модель у форматі колекції трикутних полігонів TIN поверхні, а для ефективного використання моделі в прикладних задачах аналізу рельєфу і моделювання 3D об'єктів потрібні набір полігонів

окремих трикутників, то формування TIN моделі з обмеженнями доцільно виконувати в два етапи:

1) виконання запиту на створення геометричної колекції типу TIN з використанням функції *ST_ConstrainedDelaunayTriangles;*

2) виконання функції типу

В залежності від комбінації наборів вхідних даних, що використовується для створення TIN моделі з обмеженнями, можна визначити такі три типові схеми запитів попереднього опрацювання вхідних даних та виклику функції *ST_ConstrainedDelaunayTriangles* для побудови TIN моделі:

1) ***запит типу PLT*** для побудови TIN моделі з обмеженнями для набору вихідних точок *points* і набору обмежувальних ліній *constr_line*:

```
CREATE TABLE tin_dlc AS
WITH pts AS
(SELECT geom AS pt FROM points),
lts AS
(SELECT geom AS cl FROM constr_line)
SELECT ST_ConstrainedDelaunayTriangles(ST_Collect(ST_Union(pt::geometry),
ST_Union(cl::geometry))) AS geom
FROM pts, lts;
```

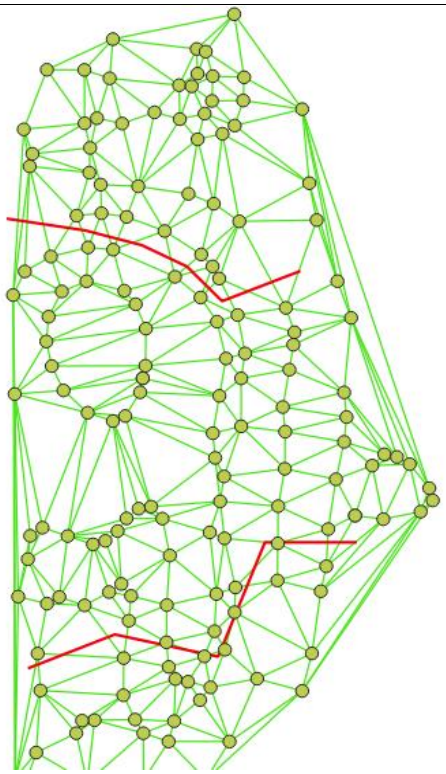
2) ***запит типу PPT*** для побудови TIN моделі з обмеженнями для набору вхідних точок *points* і набору обмежувальних полігонів *constr_polygon* :

```
CREATE TABLE tin_dlc AS
WITH pts AS
(SELECT geom AS pts FROM points),
pls AS
(SELECT geom AS cp FROM constr_polygon)
SELECT ST_ConstrainedDelaunayTriangles(ST_Collect(ST_Union(pt::geometry),
ST_Collect(ST_Union(cp::geometry))) AS geom
FROM pts, pls;
```

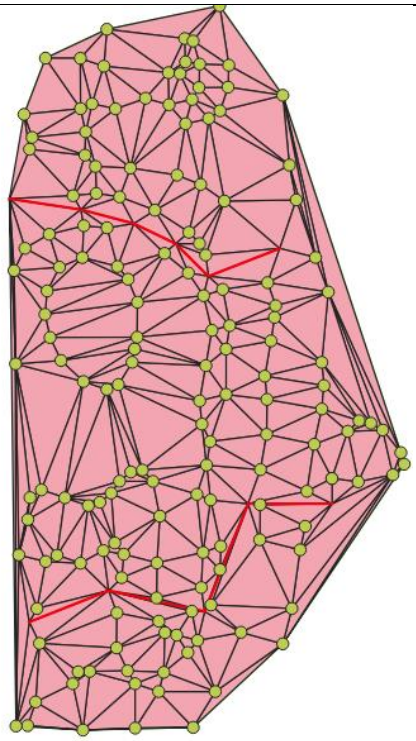
3) ***запит типу PLPT*** для побудови TIN моделі з обмеженнями для набору вхідних точок *points*, набору обмежувальних ліній *constr_line* і набору обмежувальних полігонів *constr_polygon*:

```
CREATE TABLE tin_dlc AS
WITH pts AS
(SELECT geom AS pts FROM points),
lts AS
(SELECT geom AS cl FROM constr_line),
pls AS
(SELECT geom AS cp FROM constr_polygon)
SELECT ST_ConstrainedDelaunayTriangles(ST_Collect(ST_Union(pt::geometry),
ST_Collect(ST_Union(cl::geometry),ST_Union(cp::geometry)))) AS geom
FROM pts, lts, pls;
```

Зображення тестових наборів вихідних даних та результати виконання запитів на побудову TIN моделі рельєфу з обмеженнями проілюстровано на рисунках 3.47 та 3.48.

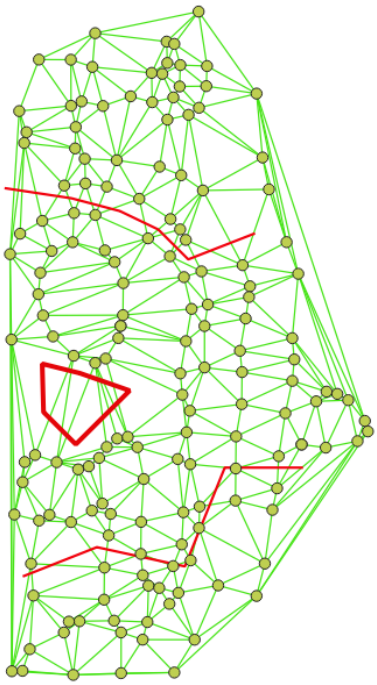


Набір вхідних точок і структурних ліній рельєфу

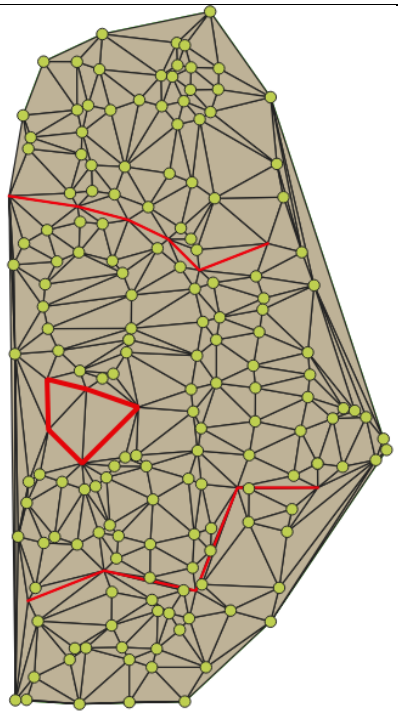


Колекція трикутників TIN моделі рельєфу з обмеженнями

Рис. 3.47. Вхідні дані та результати запиту на побудову TIN моделі рельєфу з обмеженнями структурних ліній



Набір вхідних точок, структурних ліній рельєфу і обмежувального полігону



Колекція трикутників TIN моделі рельєфу з обмеженнями

Рис. 3.48. Вхідні дані та результати запиту на побудову TIN моделі рельєфу з обмеженнями структурних ліній та полігону

Результати тріангуляції з обмеженнями за будь-яким поміж вищеописаних варіантів запитів потрібно опрацювати функцією типу *tin2triangels()* для перетворення колекції TIN в набір полігонів окремих трикутників із збереженням їх в таблиці трикутників TIN поверхні. Алгоритм цієї функції описано в п. 3.4.3, а її текст на мові PL/pgSQL приведено в додатку Б, п.Б.2.1. Текст функції можливо потрібно відредагувати у відповідності до конкретних імен вхідної таблиці з колекцією TIN та вихідної таблиці набору полігонів трикутників TIN моделі поверхні.

3.4.5 Реалізація функції реконструкції TIN-моделі за набором обмежень

Оновлення бази геопросторових даних з таблицею TIN моделлю рельєфу, що первинно була створена за технологічними схемами, запитами з використанням реалізованих функцій, що описані в пп. 3.4.3, 3.4.4, пов’язано із уточненням структурних ліній. Зважаючи на порівняно значні витрати часу на первинне формування набору трикутників TIN моделі (див. п. 3.4.3), оновлення

цього набору доцільно виконувати за схемою реконструкції раніше створеної TIN моделі на основі нового набору структурних ліній.

В загальній схемі реконструкції TIN моделі за набором структурних ліній можна виділити такі основні етапи опрацювання кожної вхідної структурної лінії (рис. х4):

1) визначення піднабору трикутників (далі ТЛС) вхідної TIN, з якими перетинається структурна лінія (рис. 49, а);

2) визначення контуру полігону області реконструкції на основі об'єднання трикутників піднабору ТЛС (рис. 49, б);

3) виконання функції *ST_ConstrainedDelaunayTriangles* для тріангуляції з обмеженнями області реконструкції з використанням геометричної колекції із полігону області реконструкції та власне структурної лінії, в результаті якої буде створена колекція TINR тріангуляції області реконструкції (рис. 49, в);

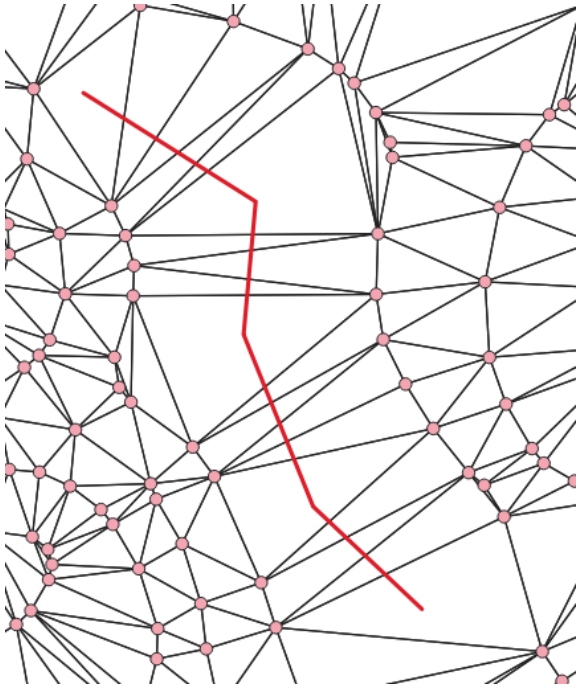
4) вилучення із первинної TIN моделі трикутників піднабору ТЛС;

5) долучення до первинної TIN моделі трикутників із колекції TINR, які повністю належать області реконструкції (рис. 49, г).

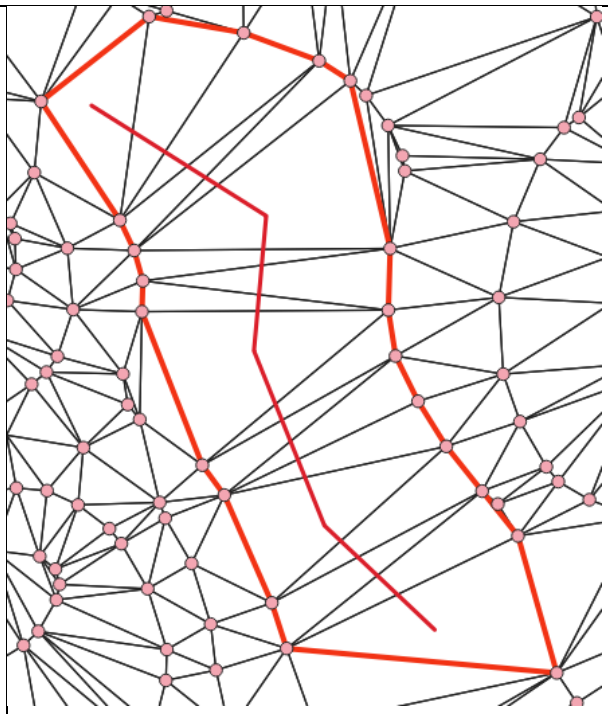
Виконання останнього етапу із визначеною умовою обумовлено тим, що колекція TINR тріангуляції області реконструкції, що створюється функцією *ST_ConstrainedDelaunayTriangles* для вхідної геометричної колекції із полігону області реконструкції та структурної лінії, може містити трикутники за межами області реконструкції, оскільки функція *ST_ConstrainedDelaunayTriangles* завжди створює колекцію трикутників TINR з межами опуклої оболонки для множини точок вхідної геометричної колекції. Отже, у разі, якщо полігон області реконструкції буде увігнутим (неопуклим) многокутником, то колекція TINR буде містити трикутники за межами області реконструкції, які є «зайвими» з точки зору задачі реконструкції TIN моделі за набором структурних ліній.

Для реалізації вищеописаної схеми реконструкції TIN моделі за набором структурних ліній в роботі розроблено прикладну функцію ***_rectin_cl(),*** яка здійснює реконструкцію TIN моделі із таблиці *tin_dln3d* за набором структурних ліній із таблиці *constr_line*. Результатом виконання функції

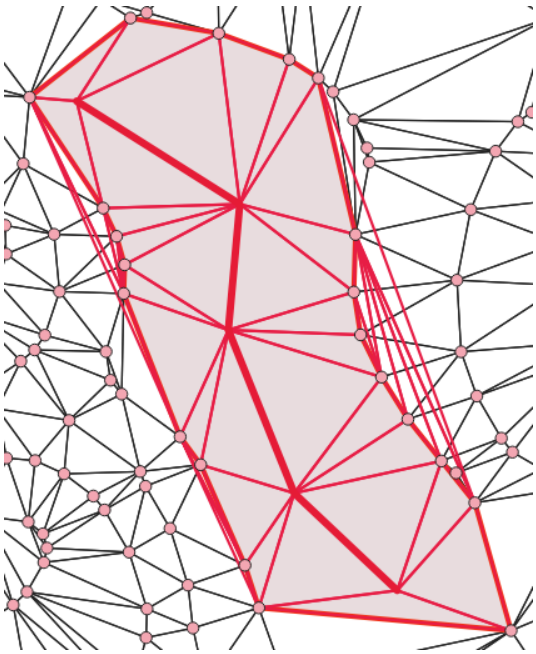
_rectin_cl() є оновлений набір трикутників TIN в таблиці tin_dln3d. Узагальнена схема алгоритму функції подано на рис. 3.50.



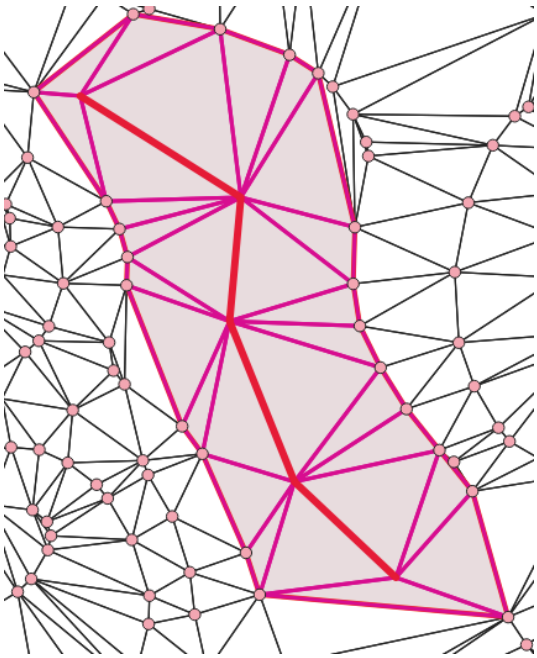
а) визначення піднабору трикутників TIN, що перетинаються вхідною структурною лінією



б) побудова полігону області реконструкції TIN з обмеженнями структурної лінії



в) тріангуляція з обмеженнями для геометричної колекції з полігоном області реконструкції та структурною лінією



г) оновлення TIN моделі трикутниками тріангуляції з обмеженнями, що містяться в середині полігону області реконструкції

Рис. 3.49. Ілюстрації до основних етапів реконструкції первинної TIN моделі за новим набором структурних ліній

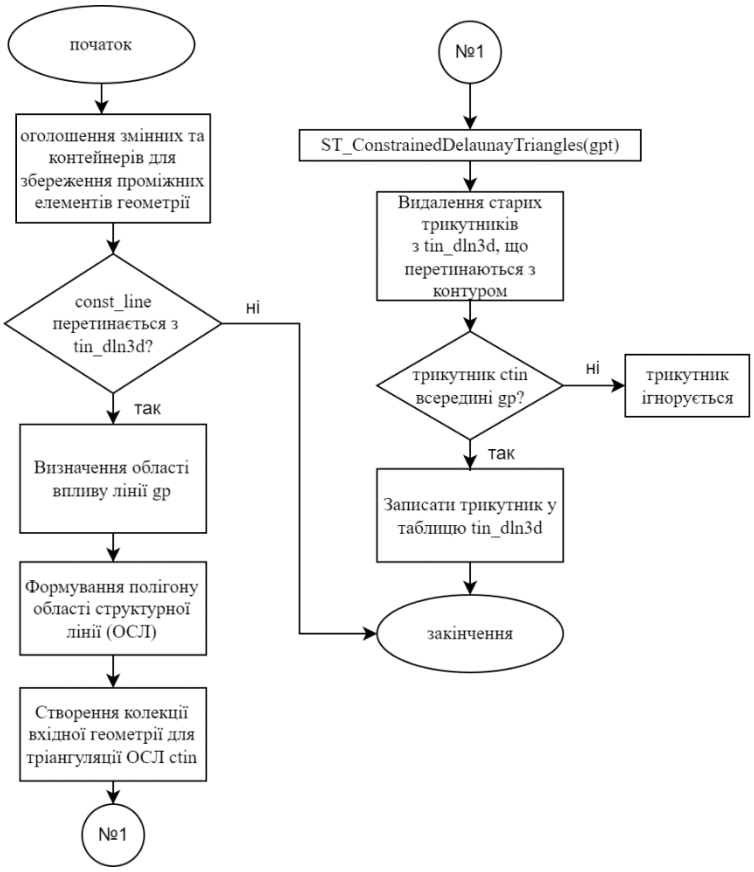


Рис. 3.50. Узагальнена схема алгоритму функції *_rectin_cl()* реконструкції TIN моделі рельєфу за набором структурних ліній

Текст реалізації функції *_rectin_cl()* на мові PL/pgSQL приведено в додатку Б п.Б.2.2. Тестування функції виконувалося на тестовому наборі структурних ліній та TIN моделі рельєфу на дослідну територію міста Києва. Фрагменти зображення, що отримані за результатами тестування функції *_rectin_cl()* подано зокрема на рис. 3.х3.

Варто наголосити, що найскладнішим завданням в створенні 3D TIN моделі рельєфу з обмеженнями є створення наборів вхідних даних для 3D моделей структурних ліній і контурів обмежувальних полігонів.

3.4.6 Реалізація функції інтерполяції висоти довільної 2D точки за 3DTIN моделлю рельєфу

В PostGIS не має функції, яка забезпечує визначення висоту довільної 2D точки за 3DTIN моделлю рельєфу. Користувачі PostGIS на форумах в інтернеті обговорюють питання застосування для цих цілей базову функцію визначення перетину двох геометрій ST_Intersection (g1, g2), яка повертає геометричну колекцію на множині спільних точок двох вхідних просторових об’єктів. У разі

вхідних об’єктів типу POIN і TIN результатом мала би бути 3D точка з інтерпольованим значенням координати Z. *Але в документації PostGIS вказано: «Функція ST_Intersection (g1, g2) підтримує 3d і не втрачає z-індекс. Однак результат обчислюється лише за допомогою XY. Отримані значення Z копіюються, усереднюються або інтерполюються». Реально для* TIN моделі ця функція повертає точки із середнім значенням висоти відносно вершин трикутника, з яким перетинається 2D точка, що для не задовольняє вимог більшості практичних задач використання для аналізу і моделювання. Тому постала задача реалізації прикладної функції для коректної (як мінімум лінійної) інтерполяції значення висоти точки за її 2D координатами та 3D координатами вершин трикутника, що містить цю точку.

Реалізована прикладна SQL функція ***_point_tin_value (triangle3d, point2d),*** яка повертає значення висоти для точки point2d за трикутником triangle3d поверхні TIN на основі визначення коєфіцієнтів рівняння площини трикутника за 3D координатами його вершин.

В узагальненій схемі алгоритму функції *_point_tin_value (,)* (рис. 3.51) визначено такі функціональні блоки:

Оголошення змінних координат вершин та вхідної точки з визначенням змінних коефіцієнтів: Підготовка змінних, які будуть використовуватись для обчислень, включаючи координати вершин трикутника та вхідної точки.

Отримання вершин трикутника з їх координатами: Використання функції ST_Points() для витягування геометрії вершин трикутника та обчислення їхніх координат.

Обчислення коефіцієнтів рівняння площини трикутника: Це основний обчислювальний блок, який визначає дискримінанти рівняння площини трикутника за координатами його 3D вершин:

$$a = (y_2 - y_1) * (w_3 - w_1) - (w_2 - w_1) * (y_3 - y_1) \quad (3.6)$$

$$b = (w_2 - w_1) * (x_3 - x_1) - (x_2 - x_1) * (w_3 - w_1) \quad (3.7)$$

$$c = (x_2 - x_1) * (y_3 - y_1) - (y_2 - y_1) * (x_3 - x_1) \quad (3.8)$$

Перевірка на належність точок трикутника на одній прямій: Перевірка належності точок трикутника одній прямій за детермінантом c. Якщо abs(c) < tolerance менше 0.0000001 то всі три точки лежать на одній прямій і не утворюють площину, тому функція повертає null.

Обчислення висоти для точки: Виконання обчислення висоти точки на основі значень дискримінанти рівняння площини трикутника і вхідних координат 2D точки:

$$z = \frac{\left(a * \left(x_p - x_1\right)\right) + \left(b * \left(y_p - y_1\right)\right)}{c} \tag{3.9}$$

Повернення значення висоти: Функція повертає обчислене значення висоти для заданої точки, якщо всі умови виконано.

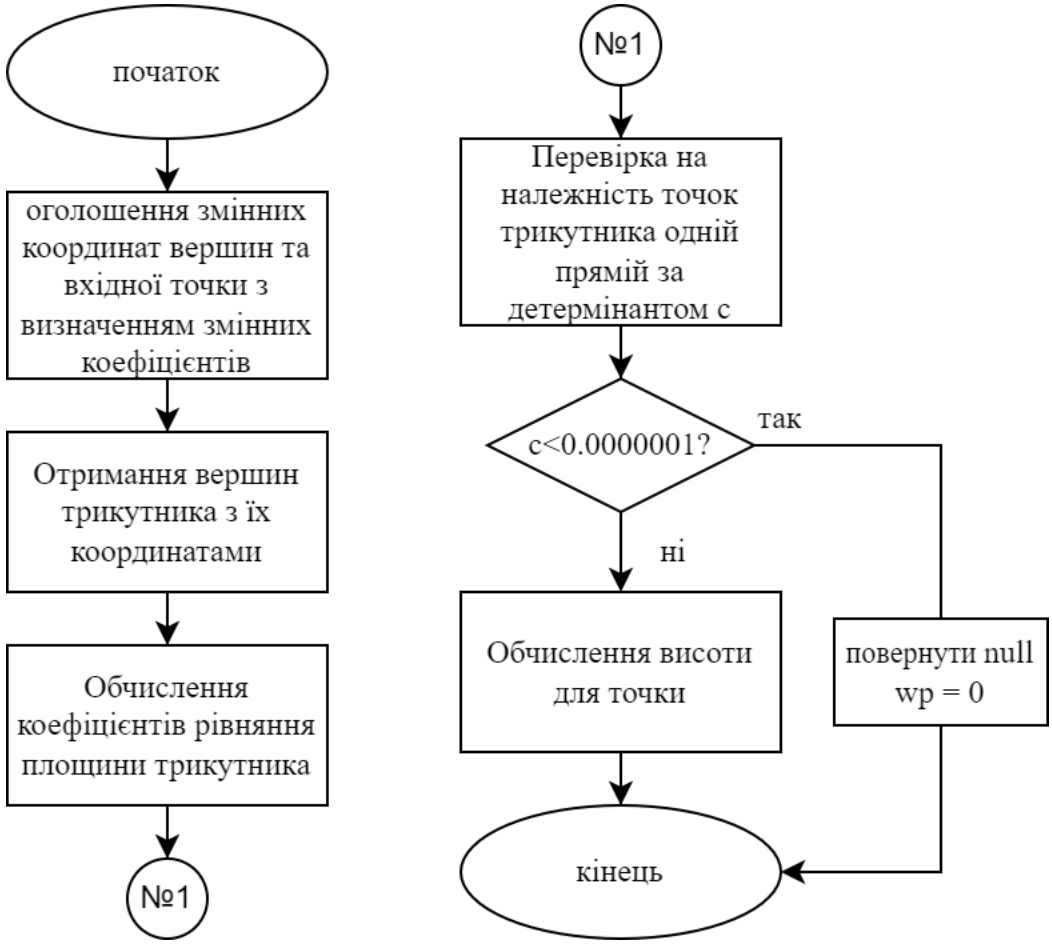


Рис. 3. 51. Загальне схема алгоритму функції інтерполяції значення висоти довільної 2D за 3D трикутником TIN моделі

Текст функції *_point_tin_value (,)* на мові PL/pgSQL приведено в додатку Б, п. Б.2.3.

Запит на тестування функції _point_tin_value(triangle3d, point2d)

```
SELECT _point_tin_value(
'TIN Z(
((1000.0 1000.0 100.0,2000.0 1800.0 500.0,
2000 1000.0 100.0,1000.0 1000.0 100.0)))'::geometry,
'POINT(1250 1100)'::geometry);
```

Результат - 150, що відповідає контрольному значенню.

Тестування функції також виконувалося *_point_tin_value (,)* в процесі тестування реалізованої в роботі функції *_line3ds_tin* для побудови 3D ліній з використанням 3D TIN моделі рельєфу (див. п. *3.4.7)*

3.4.7 Реалізація функції побудови 3D ліній з використанням 3DTIN моделлю рельєфу

Функція ***_line3ds_tin(line2d geometry, sgm_length float)*** призначена для побудови тривимірної моделі лінії *line3d* для вхідної ламаної двовимірної лінії *line2d* з використанням TIN моделі рельєфу з таблиці tin_dln3d. Геометрія 3D лінії обчислюється на основі 2D точок сегментації вхідної ламаної 2D лінії та використанні функції ***_point_tin_value ( )*** для обчислення координат z цих точок.

Схему алгоритму функції *_line3ds_tin* подано на рис.3.52.

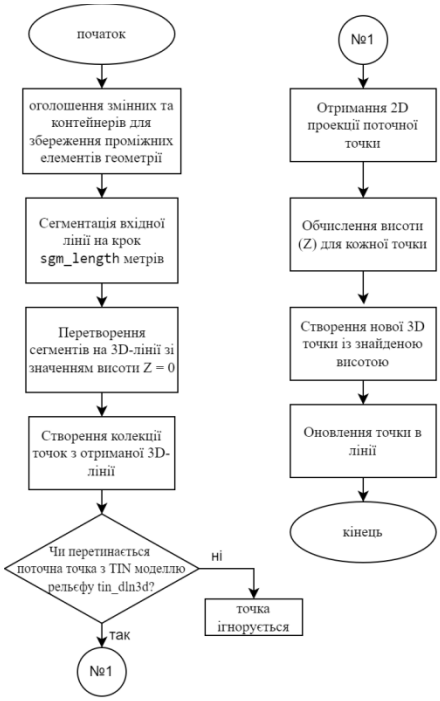


Рис. 3.52. Загальна схема алгоритму функції _line3ds_tin() для побудови 3D ліній на основі TIN-моделі рельєфу

Текст функції *_line3ds_tin* на мові PL/pgSQL приведено в додатку Б2, п.Б.2.4. Тестування функції виконувалося на прикладі запитів для побудови 3D моделей ділянок осьових ліній вулиць тестового набору вулиць міста Києва, обчислення довжини 3D осьових ліній. Виконано порівняльний аналіз результатів цих запитів з результатами аналогічних запитів, що виконувалися для тестування функції побудови 3D ліній з використанням GRID моделі рельєфу (див. п. 3.3.3). Тексти запитів, опис і аналіз результатів їх виконання приводиться далі в цьому підрозділі роботи.

Запити на побудову 3D осьових ліній вулиць з використанням тестового набору вулиць (рис. 3.53) та TIN моделі рельєфу на дослідну територію міста Києва.

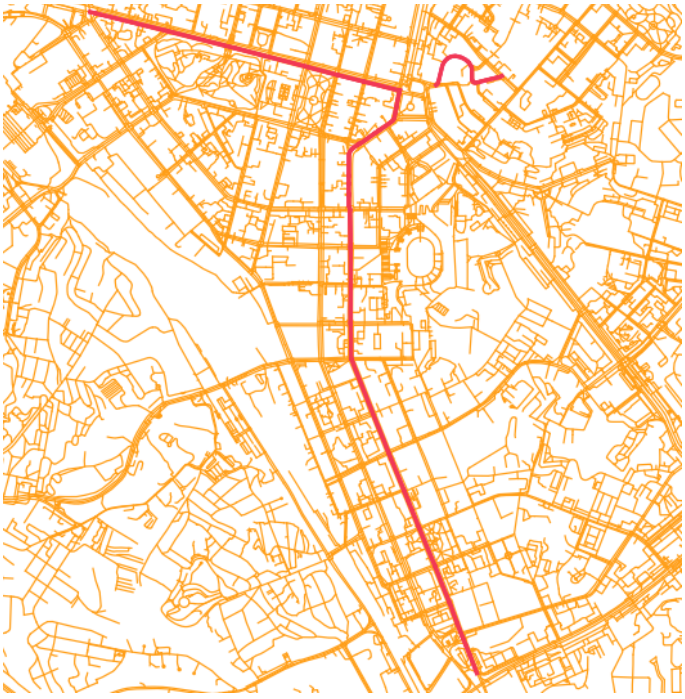


Рис. 3.53. Зображення тестового набору ділянок осьових ліній вулиць міста Києва (бульвар Тараса Шевченка, Велика Васильківська та Круглоуніверситетська вулиці)

Запит 1:

```
SELECT ST_AsText (_line3ds_tin(str_test.geom, 50.0))
FROM str_test WHERE str_test.gid = 1;
```

Результат:

```
Successfully run. Total query runtime: 196 msec
"LINESTRING Z (30.4944945 50.4463704 140,30.4950213 50.4462878 140)"
```

Запит 2:

```
SELECT ST_AsText (_line3ds_tin(str_test.geom, 50.0))
FROM str_test WHERE str_test.gid = 2;
```

Результат:

Successfully run. Total query runtime: 309 msec.

LINESTRING Z (30.5124043 50.4434837 165.64464487225808,30.512513 50.4434652 165.7377204179984,30.513091051203713 50.44336665143174 166.23224363079416,30.5136691 50.4432681 166.7267562316609,30.5140208 50.4432082 167.02779067467623,30.514378550461107 50.4431472005484 167.33382380277655,30.5147363 50.4430862 167.6398528660577,30.5149389 50.4430517 167.8132820324524)

Запит 3:

```
SELECT ST_AsText (_line3ds_tin(str_test.geom, 50.0))
FROM str_test WHERE str_test.gid = 3;
```

Результат:

Successfully run. Total query runtime: 482 msec.

LINESTRING Z (30.5171977 50.4426882 166.9709731240445,30.5172358 50.4426821 166.95578373795584,30.517305 50.442671 166.9281382971058,30.517657027912414 50.44261506621689 165.96568787180505,30.51800905499273 50.442559131371794 164.83509355467973,30.51836108124095 50.442503195464745 163.70449912424507,30.518713106657042 50.442447258495704 162.5739045804989,30.519065131241003 50.44239132046471 161.44330992353423,30.519417154992833 50.442335381371755 160.312715153351,30.5197691779125 50.44227944121686 160,30.5201212 50.4422235 158.57377436592373)

В таблиці 3.4 приведено порівняння значень висоти точок 3D лінії ділянки осьової вулиці бульвар Тараса Шевченка у місті Києві, визначених з використанням GRID (див. результати запиту 3 в п. 3.3.3) і TIN моделі рельєфу (наведені вище результати запиту 3).

Таблиця 3.4. Порівняння значень висоти точок 3D лінії ділянки осьової вулиці бульвар Тараса Шевченка у місті Києві, визначених з використанням GRID і TIN моделі рельєфу

№	Координати точок				Відхилення Ztin - Zgrid
	Довгота	Широта	Zgrid, м	Ztin, м	
1	30.5171977	50.4426882	166.184	166.971	0,787
2	30.5172358	50.4426821	166.121	166.956	0,853

3	30.517305	50.442671	166.007	166.928	0,921
4	30.517657027912414	50.44261506621689	165.825	165.966	0,141
5	30.51800905499273	50.442559131371794	166.332	164.835	- 1,497
6	30.51836108124095	50.442503195464745	166.800	163.704	- 2,006
7	30.518713106657042	50.442447258495704	164.208	162.573	- 1,635
8	30.519065131241003	50.44239132046471	161.500	161.443	- 0,057
9	30.519417154992833	50.442335381371755	158.196	160.312	2,116
10	30.5197691779125	50.44227944121686	154.920	160.000	5,080
11	30.5201212	50.4422235	152.851	158.574	5,723

Результати статистичного аналізу абсолютного значення відхилення висоти точок |Ztin – Zgrid|:

мінімальне абсолютне значення відхилення: 0,057 м;

середнє значення абсолютного відхилення: 1.892 м;

середнє квадратичне відхилення: 2.598 м.

Відхилення між значеннями Ztin та Zgrid зумовлені такими двома основними чинниками:

1) різними системами відліку висот в GRID моделі на основі растру SRTM і TIN моделі на основі точок ізоліній рельєфу топографічної карти масштабу 1:200 000 в Балтійська системи висот 1977 року;

2) різним просторовим розрізненням цих моделей.

Разом з цим, отримані результати свідчать про:

коректність результатів розробленої функції ***_point_tin_value ( )*** для обчислення висоти довільної 2D точки з використанням TIN моделі;

коректність моделей 3D ліній, побудованих з використанням розроблених функції ***_line3ds*** та ***_line3ds_tin*** для побудови 3D ліній відповідно на GRID та 3D TIN моделях рельєфу.

Для дослідження точності власне TIN – моделі рельєфу потрібні вихідні дані з кращим просторовим розрізненням та набір контрольних точок з наперед визначеними значеннями висоти за польовими вимірюваннями.

Тестування функції ***line3ds_tin*** та опосередковано і функції ***_point_tin_value ( )*** також проведено на прикладах виконання запитів для

побудови 3D осьових ліній тестового набору вулиць міста Києва (рис. 3.53) та обчислення їх довжини.

Запити з побудови 3D осьових ліній вулиць та обчислення їх довжини з використанням TIN моделі рельєфу та розроблених функцій:

Запит 1:

```
UPDATE str_test_tin
SET l_3d_tin=ST_LengthSpheroid(_line3ds_tin (str_test_tin.geom,50.),
 'SPHEROID["WGS 84",6378137,298.257223563]');

_____________________
UPDATE 36 Query returned successfully in 10 secs 586 msec.
```

Запит 2:

```
-- порівняння довжин ділянок вулиць, обчислених на GRID та TIN моделях
SELECT gid, name, l_2d::numeric(10,3) as l2d,
        l_3d::numeric(10,3) l3dGRID,
        l_3d_tin::numeric(10,3) as l3dtin,
        (l_3d-l_3d_tin)::numeric(10,3) as delta
         FROM str_test_tin;
```

Результат запиту приведено в табл. 3.5.

Таблиця. 3.5. Порівняння довжин ділянок осьових ліній вулиць для тестового набору вулиць мю Києва

№	Назва вулиці	Довжина вулиці на еліпсоїді, м			L_3D grid – L_3D tin
		L_2D	L_3D grid	L_3D tin	
1	"Тараса Шевченка бульвар"	38.530	38.532	38.530	0.002
2	"Тараса Шевченка бульвар"	186.346	186.391	186.359	0.032
3	"Тараса Шевченка бульвар"	214.009	214.794	214.193	0.602
4	"Тараса Шевченка бульвар"	165.470	165.646	165.486	0.160
5	"Тараса Шевченка бульвар"	423.263	424.028	424.066	-0.038
6	"Тараса Шевченка бульвар"	426.952	427.216	427.030	0.185
7	"Тараса Шевченка бульвар"	423.341	423.688	423.479	0.209
8	"Круглоуніверситетська "	299.311	299.940	300.822	-0.882
9	"Круглоуніверситетська "	309.967	311.913	310.165	1.748
10	"Велика Васильківська"	61.192	61.194	61.192	0.002
11	"Велика Васильківська"	107.914	107.979	107.940	0.039
12	"Велика Васильківська"	236.857	236.969	236.857	0.112
13	"Велика Васильківська"	62.621	62.631	62.621	0.010
14	"Велика Васильківська"	207.367	207.643	207.630	0.012
15	"Велика Васильківська"	206.654	206.770	206.654	0.116
16	"Велика Васильківська"	112.034	112.042	112.034	0.008
17	"Велика Васильківська"	11.191	11.193	11.191	0.002
18	"Велика Васильківська"	56.650	56.670	56.650	0.020
19	"Велика Васильківська"	225.462	225.650	225.577	0.073

№	Назва вулиці	Довжина вулиці на еліпсоїді, м			L_3D grid –
20	"Велика Васильківська"	221.595	221.650	221.595	0.055
21	"Велика Васильківська"	128.831	128.863	128.831	0.032
22	"Велика Васильківська"	243.779	244.155	243.779	0.377
23	"Велика Васильківська"	233.085	233.159	233.085	0.074
31	"Велика Васильківська"	11.433	11.433	11.433	0.000
24	"Велика Васильківська"	595.062	595.126	595.062	0.065
25	"Велика Васильківська"	222.298	222.589	222.601	-0.013
26	"Велика Васильківська"	6.187	6.194	6.198	-0.005
27	"Велика Васильківська"	62.215	62.274	62.327	-0.053
28	"Велика Васильківська"	45.096	45.165	45.176	-0.012
29	"Велика Васильківська"	107.423	107.581	107.423	0.158
30	"Велика Васильківська"	85.239	85.523	85.369	0.154
32	"Велика Васильківська"	29.901	29.906	29.908	-0.002
33	"Велика Васильківська"	174.870	174.972	174.913	0.059
34	"Велика Васильківська"	52.218	52.219	52.218	0.000
35	"Велика Васильківська"	90.727	90.738	90.727	0.011
36	"Велика Васильківська"	106.437	106.438	106.438	0.000

Запит на обчислення загальної довжини вулиць тестового набору для 2D та варіантів 3D осьових ліні вулиць:

```
SELECT name, sum(l_2d)::numeric(10,3) as l2d,
        sum(l_3d)::numeric(10,3) l3d,
        sum(l_3d_tin)::numeric(10,3) as l3dtin,
        (sum(l_3d)-sum(l_3d_tin))::numeric(10,3) as delta
        FROM str_test_tin
        GROUP BY name;
```

Результат запиту зведено в табл. 3.6.

Таблиця. 3.6. Порівняння загальних довжин осьових ліній вулиць для тестового набору

№	Назва вулиці	Довжина вулиці на еліпсоїді, м			L_3Dgrid - L_3D tin
		L_2D	L_3D grid	L_3D tin	
1	"Круглоуніверситетська"	609.278	611.853	610.987	0.866
2	"Велика Васильківська"	3704.338	3706.724	3705.428	1.296
3	"Тараса Шевченка бульвар"	1877.911	1880.295	1879.143	1.153

Аналізуючи отримані результати побудови 3 D моделей ділянок осьових ліній вулиць тестового набору вулиць та обчислення їх довжини, можна зробити висновок про коректність роботи розроблених функцій.

Незважаючи на суттєві відхилення значень координат z в 3D моделях вулиць, отриманих з використанням GRID та TIN моделей рельєфу на дослідну територію міста Києва, різниця між довжинами змодельованих 3D ліній для цих

двох моделей рельєфу не суттєва, оскільки інтервал відхилення довжин 3D ліній, обчислених з використанням функції PostGIS *ST_LengthSpheroid* , складає від 0,886 до 1.153 метра.

Висновки до розділу 3

1. Результатом моделювання рельєфу в СКБД PostgreSQL/PostGIS стало підтвердження того, що PostGIS має широкий набір функцій для опрацювання GRID-моделей як растрового типу даних, в тому числі: обчислення меж боків растру, контурів ізоліній, апроксимацій значень у довільній точці на GRID-моделі, аналіз GRID-моделі з побудовою тематичних моделей морфологічних характеристик рельєфу: нахил і орієнтація схилів, гіпотетичне освітлення, індекс TPI.

2. Реалізовано прикладну SQL-функцію для побудови 3D-моделей ліній з використанням GRID-моделі рельєфу та визначення їх довжини, що підвищує ефективність використання PostgreSQL/PostGIS у прикладних задачах. Тестування функції побудови 3D ліній на поверхні за GRID моделлю, показало, що побудова 3D ліній із просторовим розрізненням GRID виконується повільно через великі витрати часу на геометричний аналіз перетину 2D-лінії з прямокутниками чарунок GRID-моделі. Оптимальним рішенням є метод побудови ліній із точністю до заданого інтервалу сегментації вхідної 2D-лінії, в якому інтерполяція висоти здійснюється за чарунками растру, координати яких обчислюються за координатами точок лінії без побудови і просторового аналізу перетину лінії з прямокутниками чарунок растру, що значно зменшує витрати на інтерполяцію висоти у вершинах лінії.

3. Базові функції PostGIS реалізують тріангуляцію Делоне та тріангуляцію Делоне з обмеженнями з вхідними параметрами геометричних колекцій наборів точок, обмежувальних ліній і контурів полігонів та наданням результатів у форматі спеціального типу даних TIN як єдиної колекції трикутників. Таке рішення є оптимальним з точки зору реалізації базових функцій як обчислювальних компонентів, незалежних від структури конкретної бази геопросторових даних. Разом з цим, такі формати вхідних даних і подання

результатів не забезпечують ефективного використання TIN моделі в прикладних задачах морфологічного аналізу рельєфу та побудови 3D моделей об’єктів.

4. В роботі реалізована прикладна SQL функція перетворення колекції типу TIN в набір полігонів окремих трикутників та зберігання їх в таблиці бази геопросторових даних з побудовою просторового індексу на множині трикутників для оптимізації доступу до них в прикладних задачах моделювання 3D об’єктів та аналізу рельєфу.

5. Реалізовані також прикладні SQL функції, що забезпечують:

реконструкцію раніше створених вхідних TIN моделей рельєфу за набором геопросторових даних структурних ліній рельєфу;

визначення висоти довільної 2D точки на поверхні з використанням TIN моделі рельєфу;

побудову 3D моделей ліній на поверхні заданої TIN моделлю рельєфу.

6. Коректність запропонованих алгоритмів та реалізованих функцій підтверджено результатами обчислювальних експериментів з побудови TIN моделі рельєфу на дослідну територію міста Києва, моделювання 3D ліній ділянок осьових ліній тестового набору вулиць та визначення їх довжини.

ВИСНОВКИ

1. У роботі на реальних наборах даних проведено обчислювальні експерименти щодо створення цифрових моделей рельєфу в середовищі СКБД PostgreSQL/PostGIS та встановлено що:

а) в PostGIS надаються ефективні засоби для підтримки і аналізу GRID моделей рельєфу на основі спеціального типу даних для растрових моделей та набору функцій побудови поверхонь морфологічних характеристик рельєфу;

б) базові функції PostGIS реалізують тріангуляцію Делоне та тріангуляцію Делене з обмеженнями з наданням результатів у форматі спеціального типу даних TIN як єдиної колекції трикутників, що є раціональним з точки зору реалізації вбудованих прикладних функцій тріангуляції.

2. Для підвищення ефективності використання базових функцій PostGIS моделювання рельєфу в прикладних задачах в роботі реалізовані прикладні SQL функції, які забезпечують:

а) побудову 3D моделей ліній та визначення їх довжини на поверхні з використанням GRID моделі рельєфу;

б) формування набору трикутників TIN моделі рельєфу з використанням базових функцій PostGIS для тріангуляції Делоне та тріангуляції з обмеженнями та побудовою просторового індексу для набору трикутників для оптимізації доступу до них в прикладних задачах моделювання 3D об’єктів та аналізу рельєфу;

в) реконструкцію існуючих TIN моделі рельєфу за набором геопросторових даних структурних ліній рельєфу;

г) визначення висоти довільної 2D точки на поверхні з використанням TIN моделі рельєфу;

д) побудову 3D моделей ліній на поверхні заданої TIN моделлю рельєфу.

3. Коректність запропонованих алгоритмів та реалізованих функцій підтверджено результатами обчислювальних експериментів з побудови TIN моделі рельєфу на дослідну територію міста Києва, моделювання 3D ліній ділянок осьових ліній тестового набору вулиць та визначення їх довжини..

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Бурштинська Х., Василиха І., Коваль П. Технологія побудови цифрової моделі рельєфу для створення плану дна ріки. Геодезія, картографія і аерофотознімання, випуск 69, 2007. – с. 135–144.

2. Геоінформаційні технології та інфраструктура просторових даних: у шести томах. Том 2: Системи керування базами геоданих для інфраструктури просторових даних. Навчальний посібник. / Кейк Д., Лященко А.А., Путренко В.В., Хмелевський Ю., Дорошенко К.С., Говоров М. - К.: Планета-Прінт, 2017. 456 с.

3. ДСТУ ISO 19123:2017 (ISO 19123:2005, IDT) Географічна інформація. Схема для геометрії і функцій покриття. К.: ДП «УкрНДНЦ» 2016.

4. ДСТУ 8774:2018 «Географічна інформація. Правила моделювання геопросторових даних». К.: ДП «УкрНДНЦ» 2017.

5. Карпінський Ю.О., Лященко А.А. Орографічно-триангуляційна цифрова модель рельєфу // Вісник геодезії та картографії. – 2000. - №3. - С. 28 - 32.

6. Карпінський Ю.О. Основи ГІС. Стандартизація географічної інформації: навч. посіб. / Ю.О. Карпінський, А.А. Лященко, Н.Ю. Лазоренко-Гевель. – Київ: КНУБА, 2021. – 152 с. ISBN 978-966-627-327-2.

7. Кузик З. Цифрові моделі рельєфу на територію курорту Східниця / З. Кузик // Вісник Національного університету «Львівська політехніка». Серія Геоінформаційні системи і моделювання рельєфу. – 2009. – № 71. – С. 24-35

8. Лазоренко-Гевель Н.Ю. Аналіз методів і моделей цифрового моделювання рельєфу в об’єктно-реляційних базах топографічних даних [Текст] / Н.Ю. Лазоренко-Гевель, Б.І. Денисюк // Управління розвитком складних систем. – 2016. – № 26. – С. 178 – 186

9. Майоров М.П. Дослідження методів інтелектуального аналізу для оцінки ландшафтних та кліматичних характеристик річкових басейнів : магістерська кваліфікаційна робота / М.П. Майоров. – Одеса: Одеський державний екологічний університет, 2021. – 77 с.

10. СОУ 742-33739540 0013:2010 – Правила цифрового опису рельєфу. Комплекс стандартів. База топографічних даних. — К.: Мінприроди України, 2010.

11. Al-Salami, A. M. (2009). TIN support in an open source spatial database. Master's thesis, International Institute for Geo-information Science and Earth Observation, Enschede, The Netherlands.

12. ArcGIS Pro. https://desktop.arcgis.com/ru/documentation/.

13. Deenik. T. (2021). Topographic Position Index (TPI). UBC Blog.: https://blogs.ubc.ca/tdeenik/2021/02/16/topographic-position-index-tpi/

14. Digital elevation model. Wikipedia. URL: https://en.wikipedia.org/wiki/Digital_elevation_model.

15. Farr, T. G., Rosen, P. A., et al. (2007). The Shuttle Radar Topography Mission. Reviews of Geophysics, 45(RG2004).

16. ISO/IEC 13249-3:2016 Information technology – Database languages – SQL multimedia and application packages – Part 3: Spatial. https://www.iso.org/standard/60343.html

17. QGIS. https://www.qgis.org

18. Kumara, K., Ledoux, H., & Stoter, J. (2016). Comparative analysis of data structures for storing massive TINs in a DBMS. The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences, Volume XLI-B2, 123–130. doi:10.5194/isprsarchives-XLI-B2-123-2016.

19. Kunah, O.M, Papka, O.S. (2016). Geomorhological ecogeographical variables defining features of ecological niche of common milkweed (Asclepias Syriacal.). Biological Bulletin of Bogdan Chmelnitskiy Melitopol State Pedagogical University. 6 (1), 243-275.

20. Leplatre, M. (2013). Drape lines on a DEM with PostGIS. URL: https://blog.mathieu-leplatre.info/drape-lines-on-a-dem-with-postgis.html

21. Open Geospatial Consortium. Implementation Specification for Geographic Information - Simple Feature Access - Part 1: Common Architecture, Version 1.2.1 / Editor: John R. Herring. – OGC 06-103r4. – 2011.

22. PostGIS 3.4.1 Manual. https://postgis.net/docs/index.html

23. PostgreSQL. https://www.postgresql.org/docs/current/index.html.

24. Rodriguez, E., Morris, C. S., et al. (2005). An assessment of the SRTM topographic products. JPL Publication, D31639, 143.

25. The Shuttle Radar Topography Mission. Reviews of Geophysics, 45, 2007. pp. 1 – 33. https://agupubs.onlinelibrary.wiley.com/doi/epdf/10.1029/2005RG000183

26. What is a digital elevation model (DEM)?. USGS. URL: https://www.usgs.gov/faqs/what-a-digital-elevation-model-dem.

ДОДАТОК А СЛАЙДИ ПРЕЗЕНТАЦІЇ

Київський національний університет будівництва і архітектури
Кафедра геоінформатики і фотограмметрії

Дослідження засобів моделювання рельєфу в середовищі СКБД PostgreSQL/PostGIS

Виконав студент групи ГіСТм-23
ДУЛЬСЬКИЙ Володимир Сергійович

Керівник: проф., д.т.н. ЛЯЩЕНКО Анатолій Антонович

Київ - 2024

Актуальність та мета робити

Актуальність теми. Моделювання рельєфу є ключовим аспектом багатьох сфер діяльності, таких як ГІС, міське планування та екологічний моніторинг. Сучасні ГІС забезпечують потужні засоби для створення цифрових моделей рельєфу (ЦМР), проте зростає тенденція до інтеграції баз геоданих з універсальними СКБД, що дозволяє працювати з векторними, растровими даними та ЦМР в одному середовищі. Це зменшує залежність від специфічного інструментарію ГІС і форматів даних. Об'єктно-реляційна СКБД PostgreSQL з просторовим розширенням PostGIS є однією з найпоширеніших СКБД з відкритим кодом, що використовується в сучасних ГІС. PostgreSQL/PostGIS також відповідає міжнародним стандартам і специфікаціям у сфері географічної інформації.

Метою роботи є дослідження функціональних можливостей та ефективності використання СКБД PostgreSQL/PostGIS для створення і використання цифрових моделей рельєфу на реальних обсягах даних при вирішенні типових прикладних задач.

Завдання дослідної роботи

1. аналіз стану і тенденцій розвитку засобів цифрового моделювання рельєфу в ГІС та СКБД;
2. аналіз уніфікованих базових типів геопросторових даних і функцій, що використовуються для моделювання рельєфу в базах геопросторових даних;
3. проведення обчислювального експеременту зі створення і використання GRID та TIN моделей рельєфу в середовищі СКБД PosrgreSQL/PostGIS;
4. розроблення прикладних SQL функцій для побудови 3D ліній з використанням GRID моделі рельєфу;
5. розроблення прикладних SQL функцій для побудови та реконструкції 3D трикутників TIN моделі рельєфу;
6. розроблення прикладних SQL функцій для інтерполяції висоти довільної 2D точки та побудови 3D ліній за TIN моделлю рельєфу.

Види ЦМР

Цифрова модель рельєфу (ЦМР) в загальному сенсі у геоінформатиці визначається як цифрове уявлення рельєфу земної поверхні, створене на основі даних про рельєф та морфології місцевості.

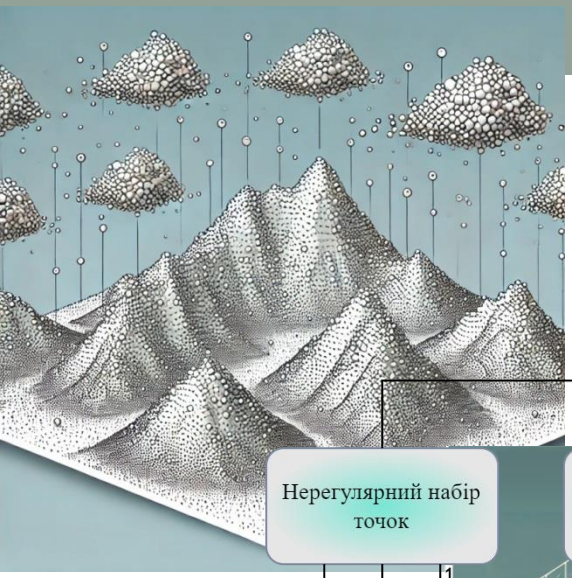
В англомовній літературі розрізняють такі формулювання ЦМР:

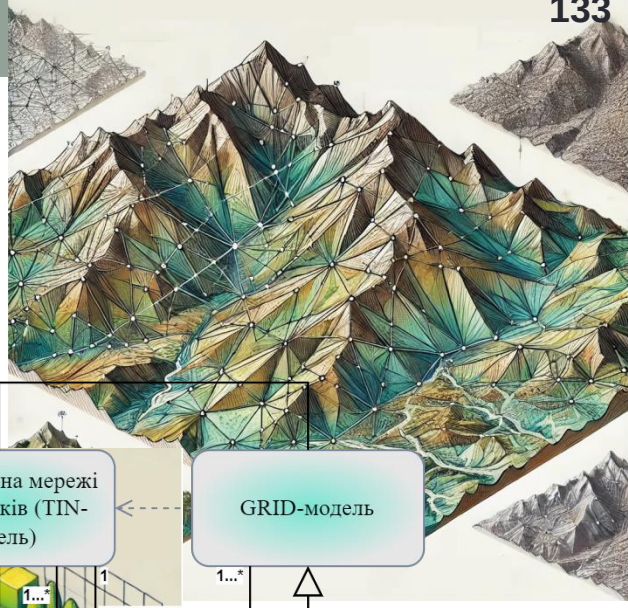
цифрова модель висот (digital elevation model, DEM) – це цифрове представлення поверхні землі, яке відображає лише голий ґрунт, виключаючи дерева, будівлі та інші поверхневі об'єкти;

цифрова модель рельєфу (digital terrain model, DTM) – містить інформацію про висоти тієї чи іншої місцевості без відображення об'єктів та зелених насаджень;

цифрова модель поверхні (digital surface model, DSM) – поняття з’явилось з появою фотограмметричних станціях, являє тривимірним представлення поверхні Землі, включаючи всі об'єкти та об'єкти, присутні на ній.

Типи ЦМР





Модель рельєфу

Нерегулярний набір точок

Набір структурних ліній

Цифрова картографічна модель (горизонталі)

Нерегулярна мережі трикутників (TIN-модель)

GRID-модель

1 вхідні дані

1 вхідні дані

1 вхідні дані

1...* вхідні дані

1 вхідні дані

1...* результат

1...* результат

1 вхідні дані

1...*

вхідні дані

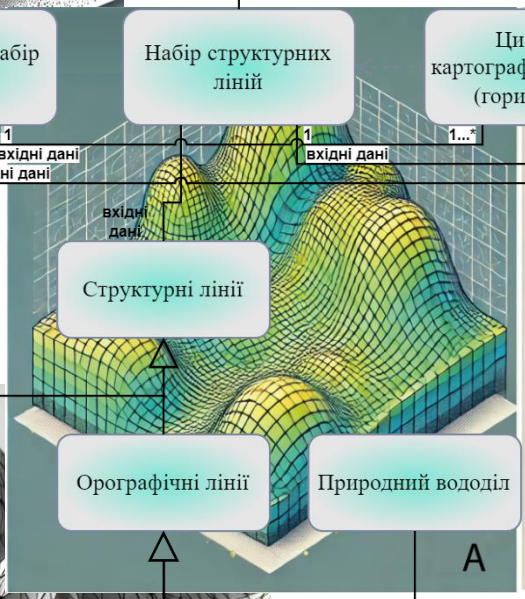
вхідні дані

Позначки висот (точки)

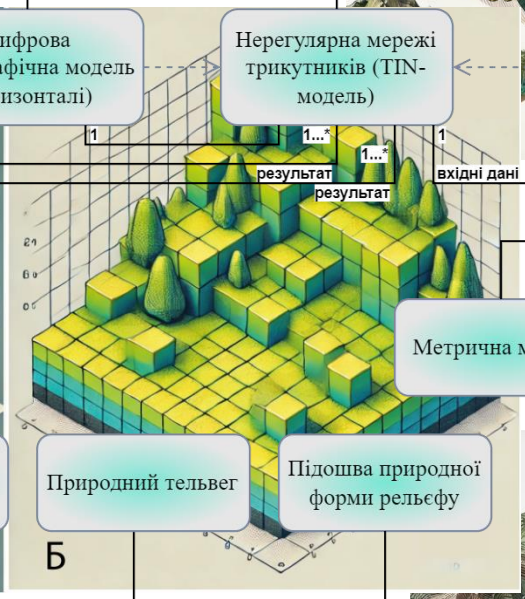
Структурні лінії

Метрична модель

Растрова модель



А



Б

Лінії шельфу змін рельєфу

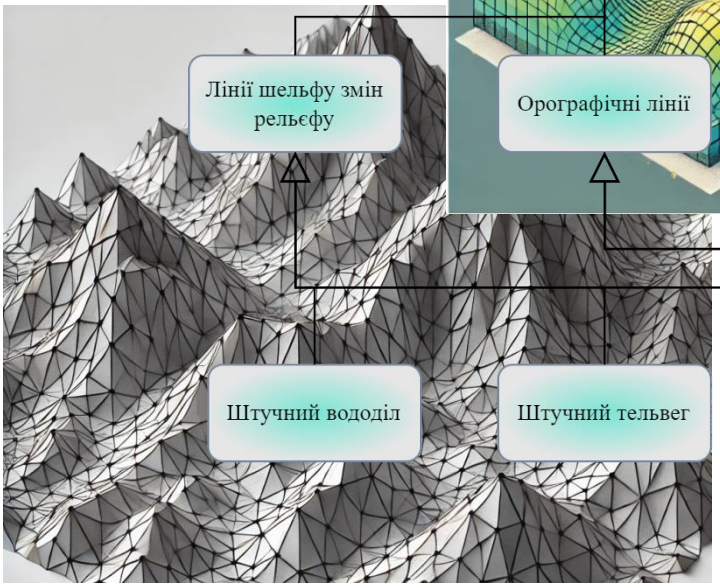
Орографічні лінії

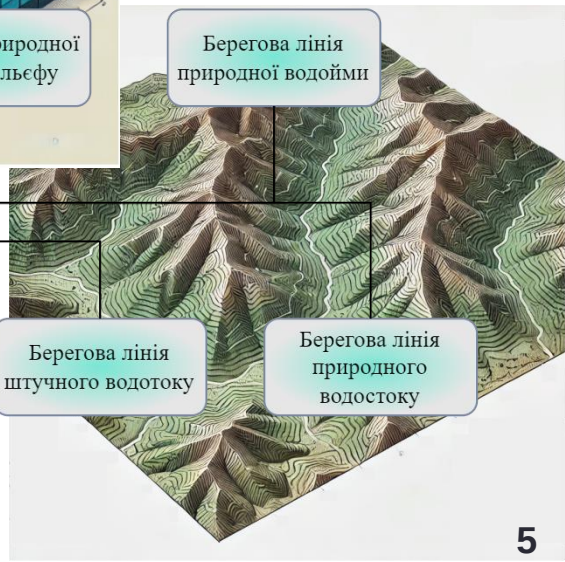
Природний вододіл

Природний тельвег

Підошва природної форми рельєфу

Берегова лінія природної водойми





Штучний вододіл

Штучний тельвег

Підошва штучної форми рельєфу

Берегова лінія штучної форми рельєфу

Берегова лінія штучного водотоку

Берегова лінія природного водостоку

Програмне забезпечення роботи з ЦМР

Функції	ArcGIS	MapInfo	Autodesk Map 3D	GeoMedia	QGIS	PostgreSQL
Підтримка векторної моделі із Z-координатою	+	-	+	+	+	+
Підтримка регулярної моделі	+	+	+	+	+	+
Підтримка тріангуляційної моделі	+	+	+	-	+	+
Введення даних із геодезичних приладів	+	-	+	-	+	+
Фототріангуляція	+	-	+	+	+	-
Автоматична векторизація	+	+	+	+	+	-
Тривимірна візуалізація	+	+	+	-	+	+
Інтерполяція висот	+	+	-	+	+	+
Побудова профілів	+	+	+	+	+	+
Побудова ізоліній	+	+	-	+	+	+
Розрахунок експозиції схилів	+	+	-	-	+	+
Розрахунок об’ємів земляних робіт	-	+	+	-	+	+
Аналіз видимості	+	+	+	-	+	+
Побудова тальвегів та вододілів	+	-	+	-	+	+

Загальні стандарти геоінформаційного моделювання

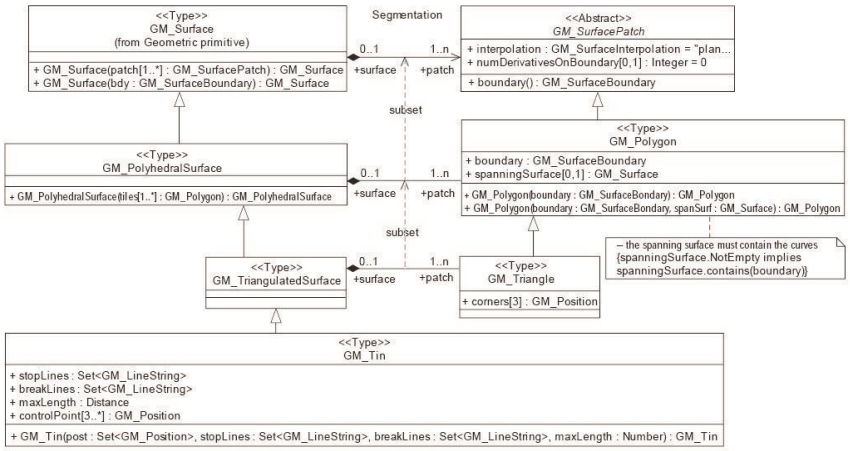
1. ISO 19107 Географічна інформація — Просторові схеми;
2. ISO 19125 Географічна інформація — Простий доступ до геопросторових об’єктів:

- ISO 19125-1: Частина 1 – Загальна архітектура;
- ISO 19125-2: Частина 2 – Опція SQL;

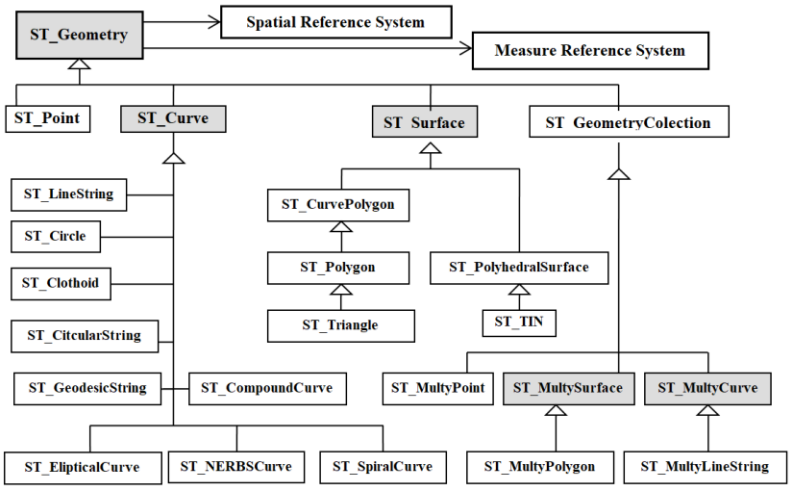
3. ISO 19123 Географічна інформація — Схема для геометрії та функцій покриттів;
4. Open Geospatial Consortium (OGC) Simple Feature Access;
5. ISO/IEC 13249-3 Інформаційні технології – Мови баз даних – SQL мультимедійні та прикладні пакети, Частина 3: Просторові дані;

Концептуальна модель класів об’єктів покриття та ієрархія класів об'єктів за стандартами ISO та OGC

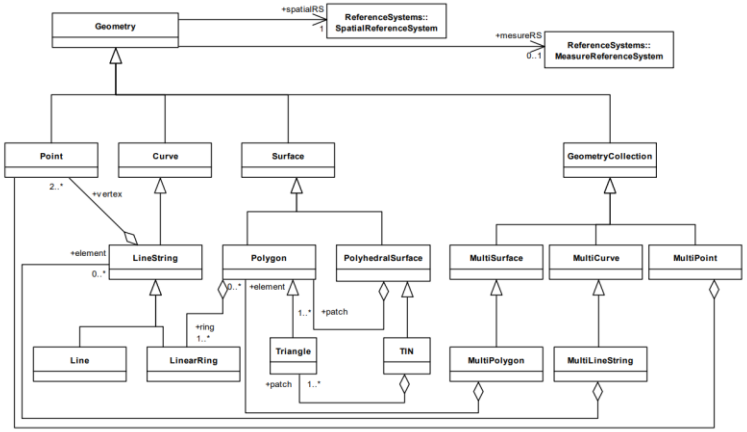
Концептуальна модель подання класів об’єктів для покриття за ISO 19123



Ієрархія класів об’єктів за ISO/IEC 13249-3



Ієрархія класів об’єктів за ISO 19125 та OGC



Типи даних для ЦМР в PostGIS

GRID:
raster

GRID-модель рельєфу в PostGIS реалізована на основі типу даних ***raster***, де растрові дані зберігаються у вигляді двовимірної матриці та розбиваються на блоки (тайли) для підвищення продуктивності і зручного доступу до окремих частин даних.

TIN:
Triangle/ TIN

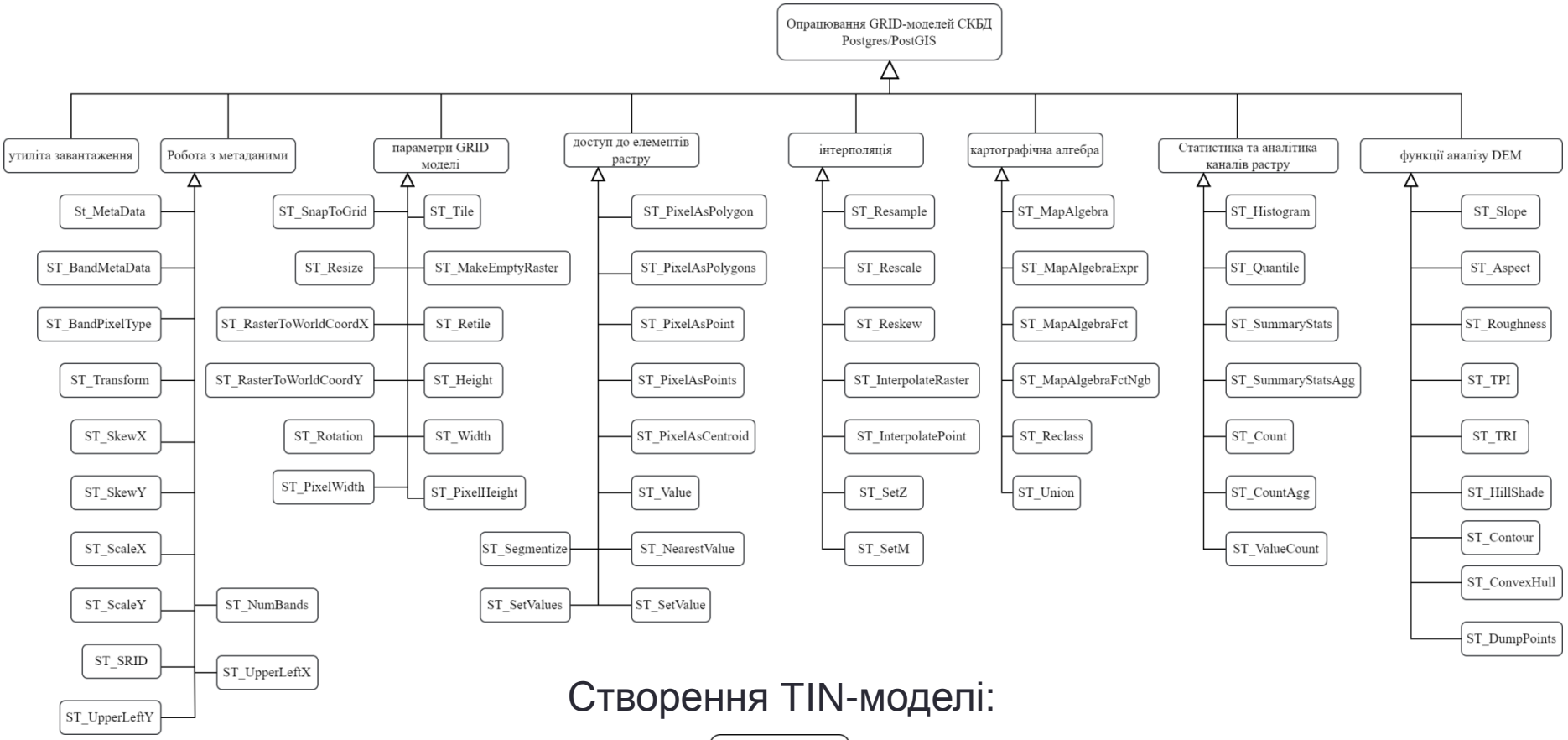
TIN подаэться типом даних ***Triangle*** та ***TIN.***

Triangle визначає трикутник як полігон з трьома неколінеарними вершинами.

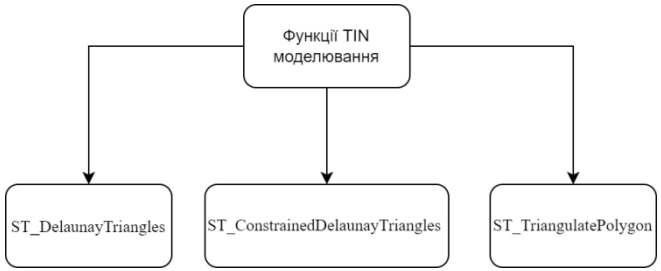
TIN є колекцією суміжних трикутників, які моделюють тривимірну нерегулярну поверхню покриття.

Функції опрацювання ЦМР в PostGIS

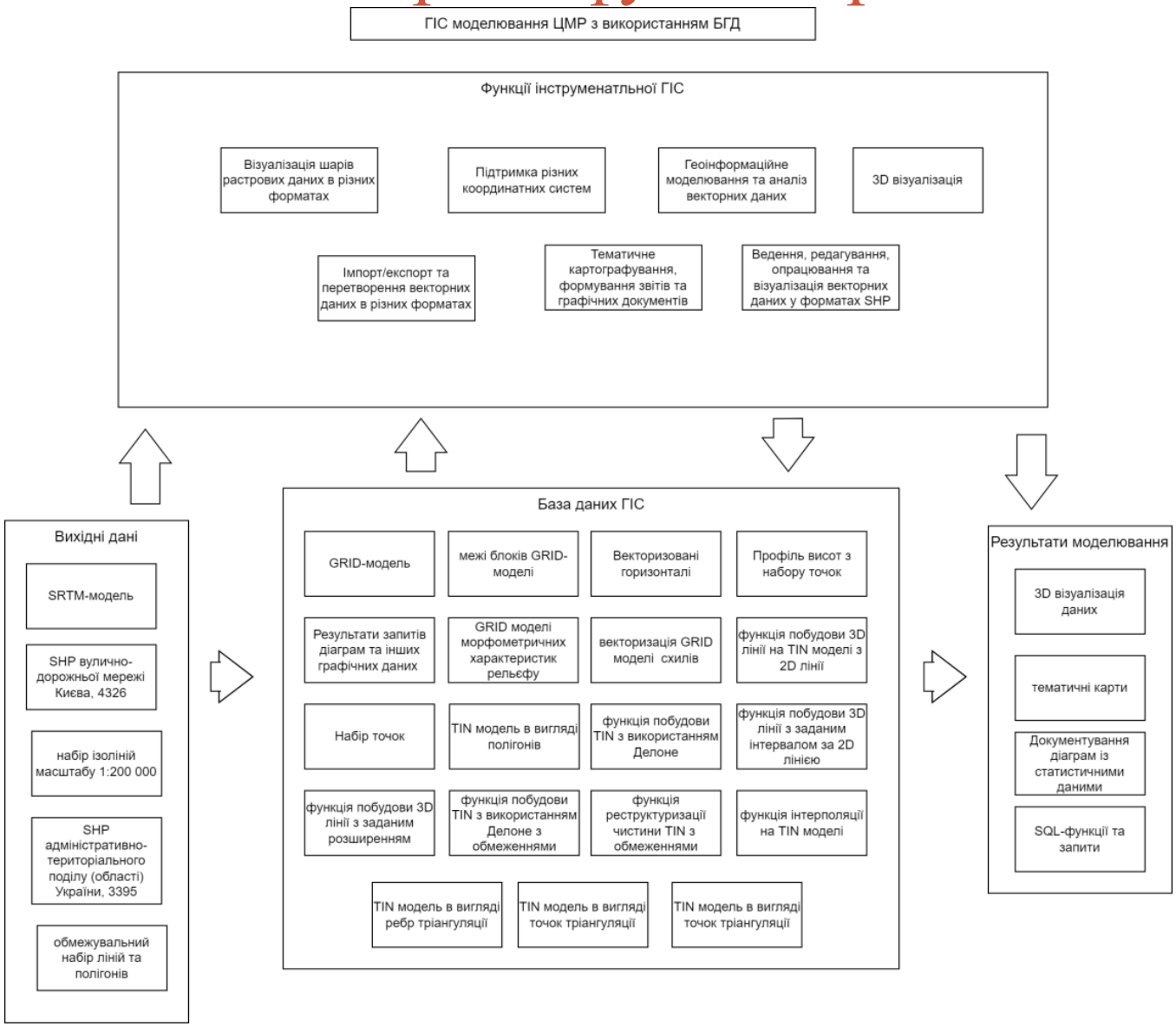
Опрацювання GRID-моделі:



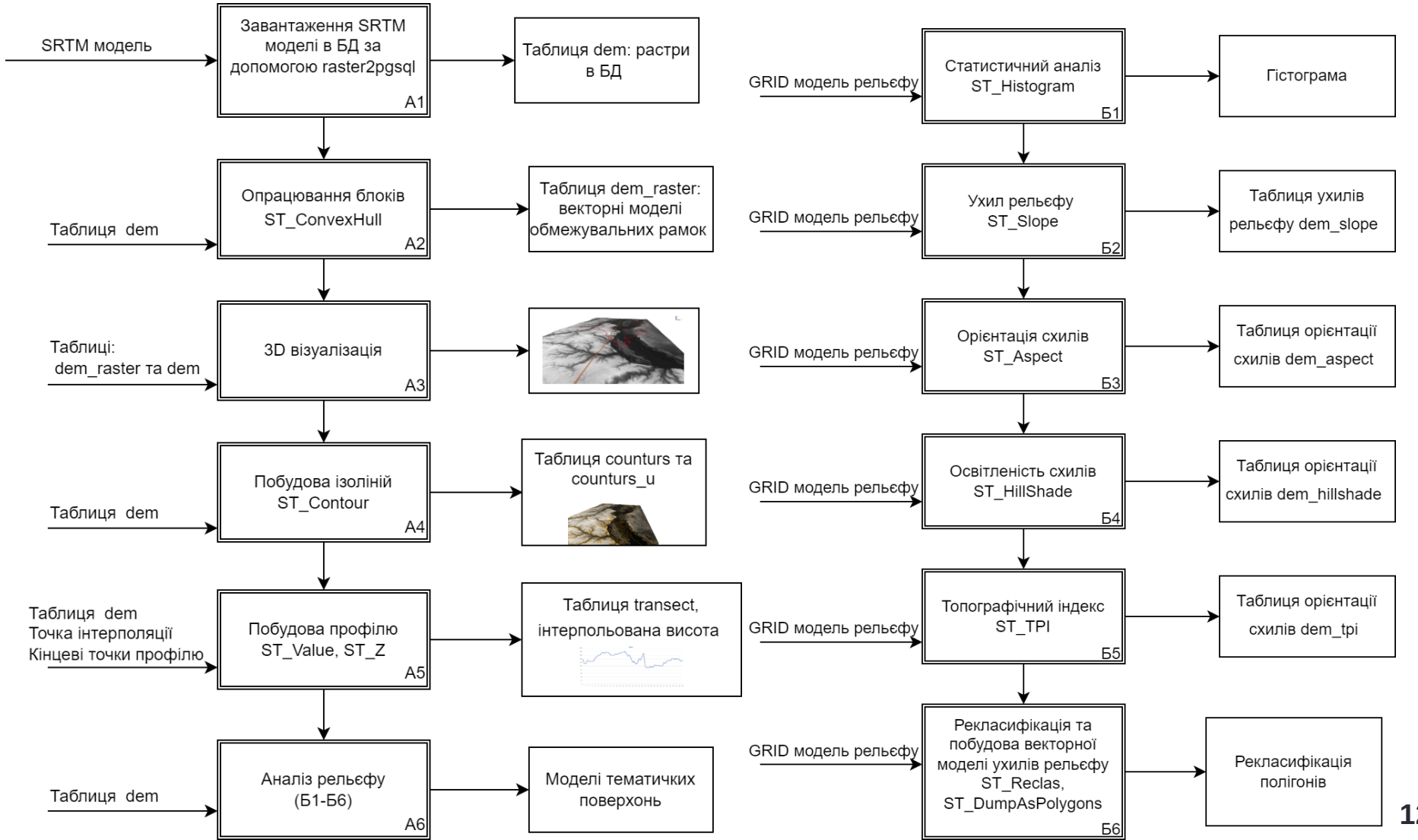
Створення TIN-моделі:



Структурно-функціональна схема ГІС моделювання рельєфу з використанням БГД

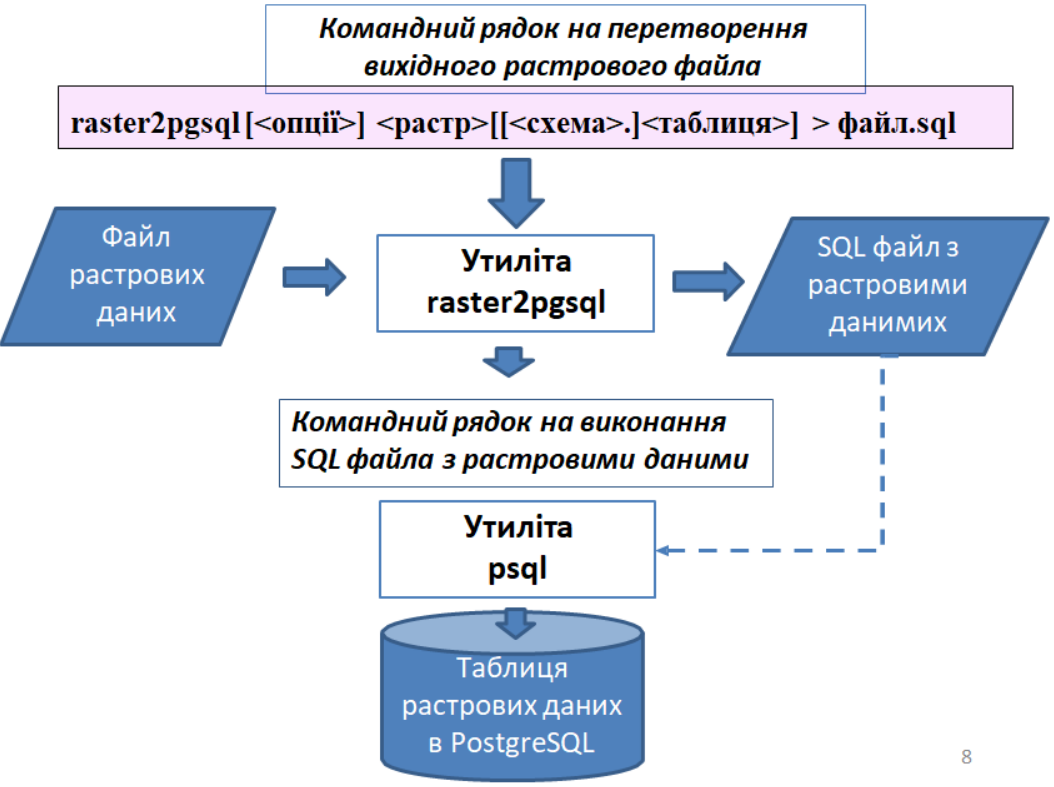


Технологічна схема дослідження використання GRID-моделі в PostGIS



Завантаження GRID-моделі в PostgreSQL/PostGIS та запит на створення рамок блоків растру

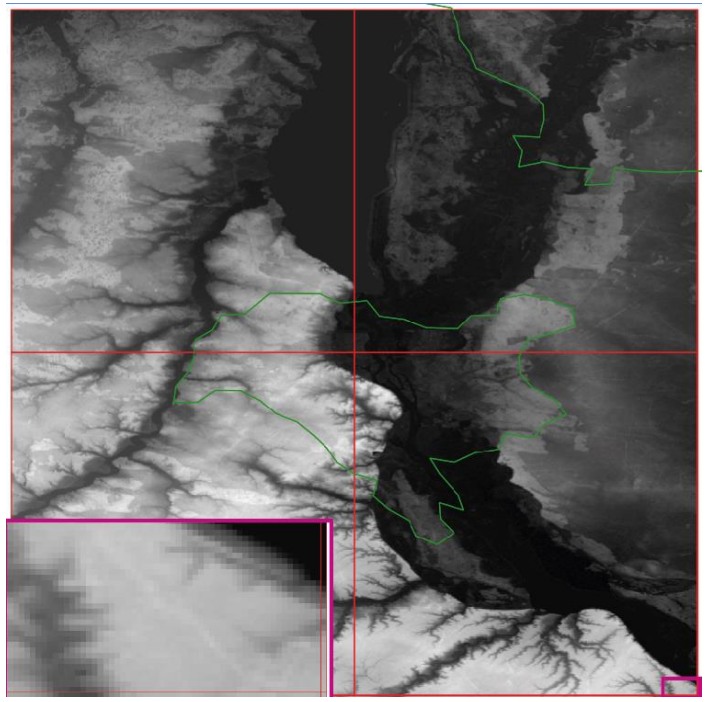
Схема завантаження растру:



8

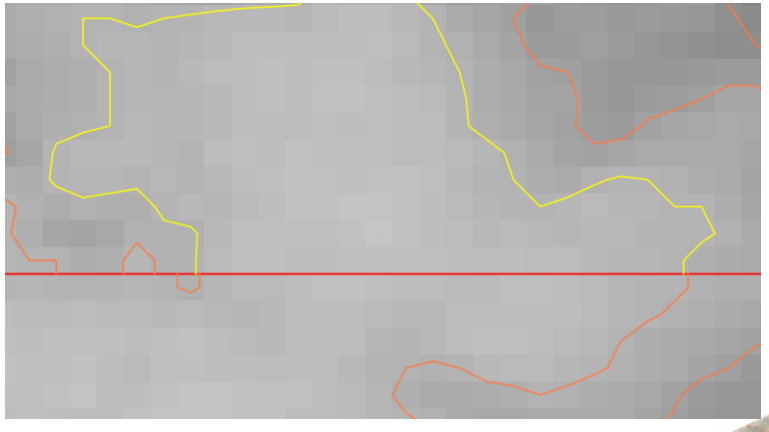
Запит на створення рамок блоків растру:

```
CREATE OR REPLACE VIEW
dem_rasters AS SELECT rid,
ST_ConvexHull(rast) AS geom
FROM dem;
```

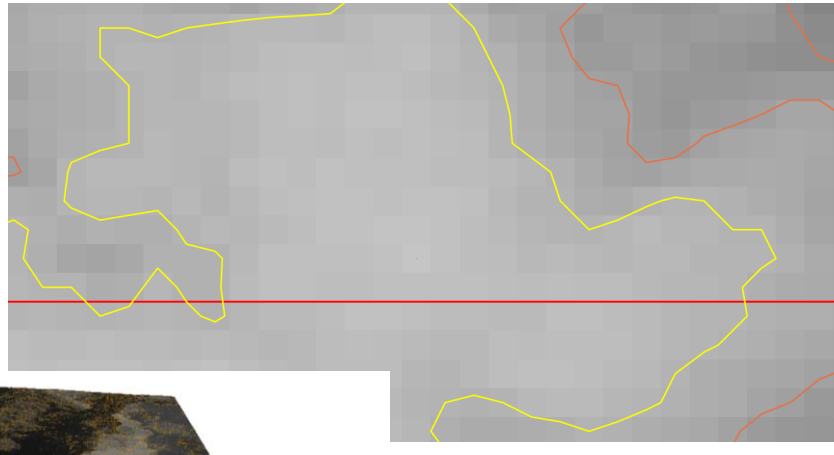


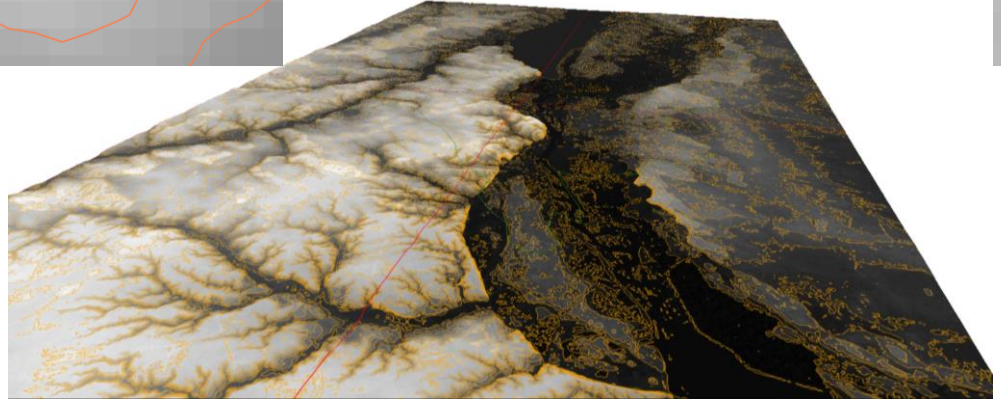
Запиту на побудову ізоліній та 3D візуалізація растру

```
CREATE TABLE contours AS
SELECT (ST_Contour(
 rast,
 bandnumber => 1,
 level_interval => 10)).*
 FROM dem;
```



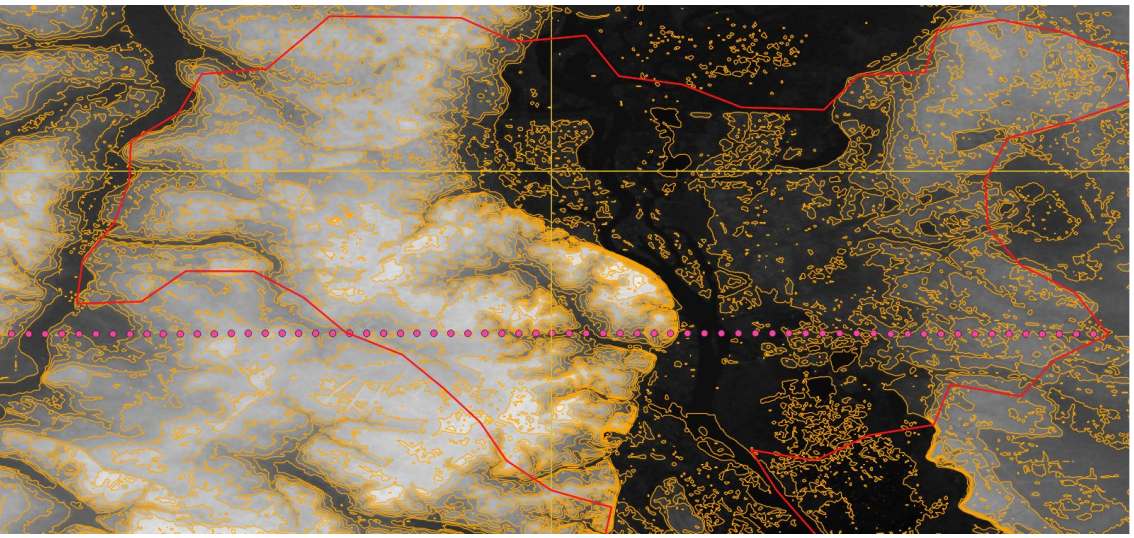
```
CREATE TABLE contours_u AS
SELECT (ST_Contour(
ST_Union(rast),
bandnumber => 1,
level_interval => 10)).*
FROM dem;
```

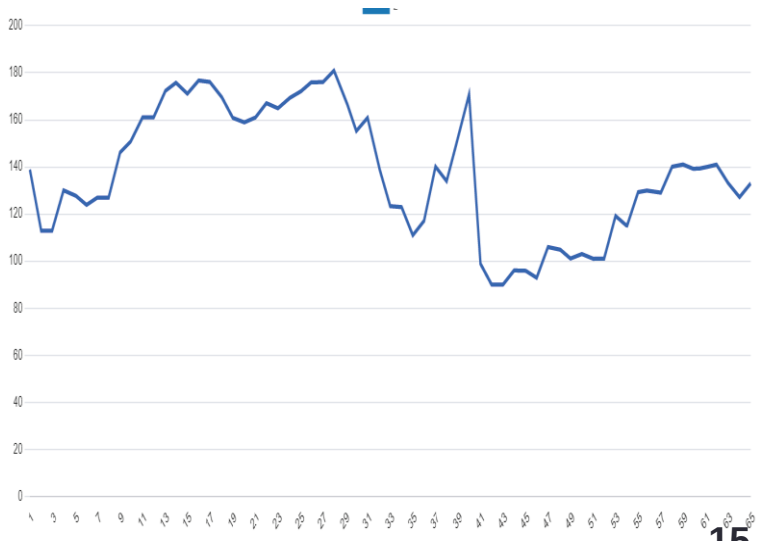




Створення профілю рельєфу з дослідженням методів інтерполяції

```
CREATE TABLE transect AS
WITH transect AS (
SELECT ST_Segmentize(ST_GeogFromText('LINESTRING(30.20 50.41, 30.80 50.41)'), 1000)::geometry AS geom ), rast AS (
SELECT ST_Union(dem.rast) AS rast FROM dem JOIN transect ON ST_Intersects(transect.geom, ST_ConvexHull(dem.rast)) ), z AS (
SELECT (ST_DumpPoints(ST_SetZ(rast.rast, transect.geom, resample => 'bilinear'))).* FROM rast CROSS JOIN transect )
SELECT round(ST_Z(geom)) AS z, geom FROM z;
```





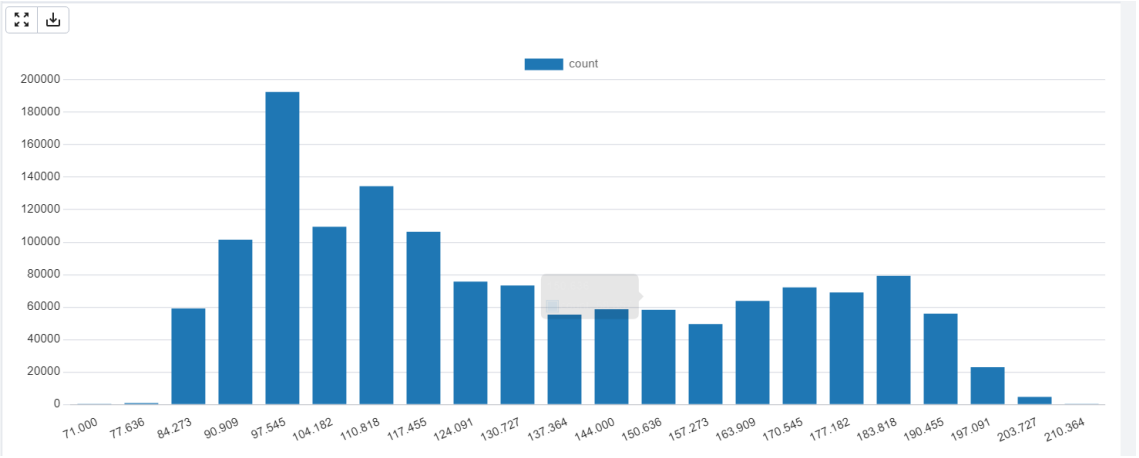
Узагальнення виконання запитів опрацювання GRID-моделі в PostgreSQL/PostGIS

Назва таблиці	Назва основної функції	Назва додаткових функцій	Число створених елементів моделі	Час виконання
dem_rasters	ST_ConvexHull	відсутня	9	53 мсек
counturs	ST_Contour	відсутня	17401	896 мсек
counturs_u	ST_Contour	ST_Union	16782	1 сек 238 мсек
pt (найближчий сусід)	ST_Value	ST_GeomFromText, ST_Intersects, ST_ConvexHull	1	52 мсек
pt (білінійна інтерполяція)	ST_Value	ST_GeomFromText, ST_Intersects ST_ConvexHull	1	93 мсек
transect	ST_Z	ST_Segmentize, ST_GeogFromText, ST_Union, ST_Intersects, ST_ConvexHull, ST_DumpPoints	65	284 мсек

Висновок: Наведені узагальнені дані засвідчують, що функції, які реалізовані для опрацювання GRID-моделі в PostGIS є ефективними. Найскладніший запит виконався трохи більше ніж за одну секунду, при цьому растр містив 1 442 401 пікселів, що свідчить про високий рівень продуктивності та оптимізацію використаних функцій.

Типовий запит на побудови гістограми висот рельєфу за GRID

```
WITH hist AS ( SELECT
(ST_Histogram(ST_Union(rast),1)) .*
from dem
WHERE filename='N50E030.hgt')
SELECT
round(min::numeric,3) AS min,
round(max::numeric,3) AS max, count,
round(percent::numeric,2) AS percent
FROM hist
ORDER BY min;
```

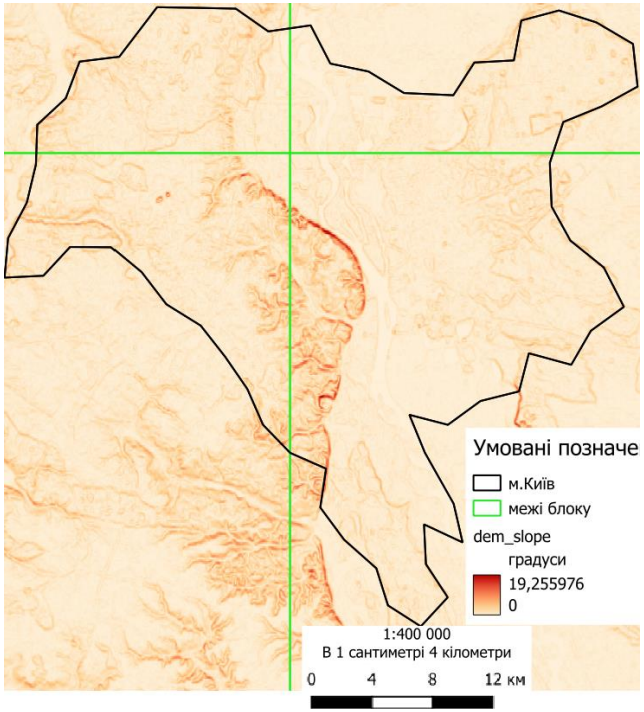


	min numeric	max numeric	count bigint	percent numeric
1	71.000	77.636	1	0.00
2	77.636	84.273	818	0.00
3	84.273	90.909	59153	0.04
4	90.909	97.545	101544	0.07
5	97.545	104.182	192505	0.13
6	104.182	110.818	109415	0.08
7	110.818	117.455	134449	0.09
8	117.455	124.091	106345	0.07
9	124.091	130.727	75645	0.05
10	130.727	137.364	73242	0.05
11	137.364	144.000	55364	0.04
12	144.000	150.636	58605	0.04
13	150.636	157.273	58357	0.04
14	157.273	163.909	49453	0.03
15	163.909	170.545	63748	0.04
16	170.545	177.182	72066	0.05
17	177.182	183.818	68950	0.05
18	183.818	190.455	79213	0.05
19	190.455	197.091	55862	0.04
20	197.091	203.727	22924	0.02
21	203.727	210.364	4641	0.00
22	210.364	217.000	101	0.00

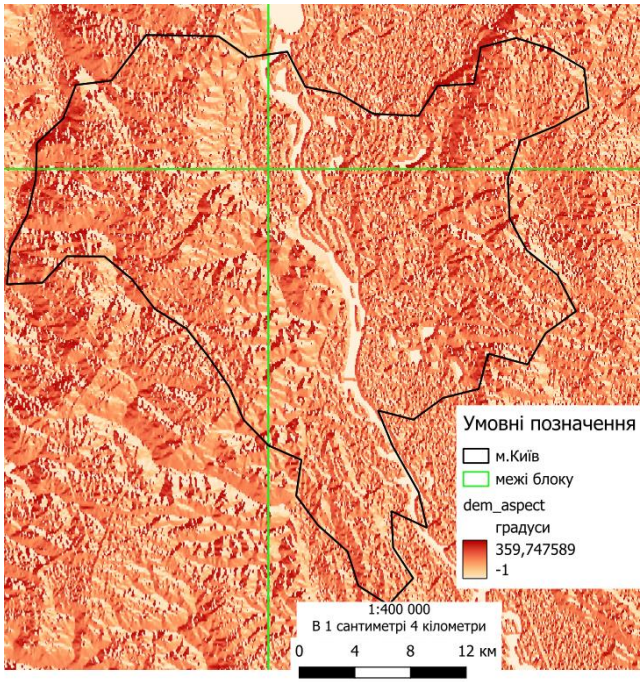
Total rows: 22 of 22 Query complete 00:00:00.885

Типовий запит на створення поверхні ухилів та орієнтації схилів

```
CREATE TABLE dem_slope AS
SELECT rid, ST_Slope(rast::raster,
'1'::int,'32BF'::text,'DEGREES'::text,'
111120'::double precision) AS rast
FROM dem
WHERE filename='N50E030.hgt'
```



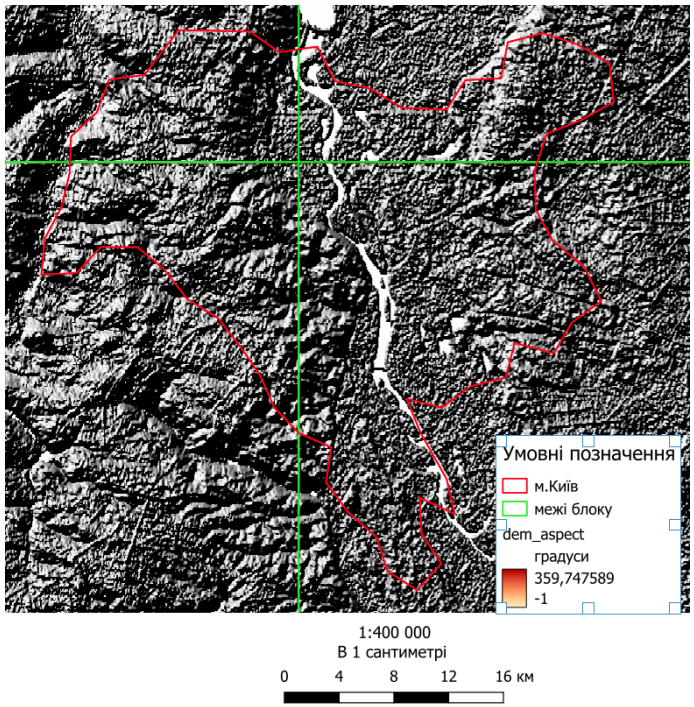

```
CREATE TABLE dem_aspect AS
SELECT rid,
ST_Aspect(rast::raster,'1'::int,'32BF'
::text,'DEGREES'::text,'1'::boolean)
AS rast  FROM dem
WHERE filename='N50E030.hgt'
```

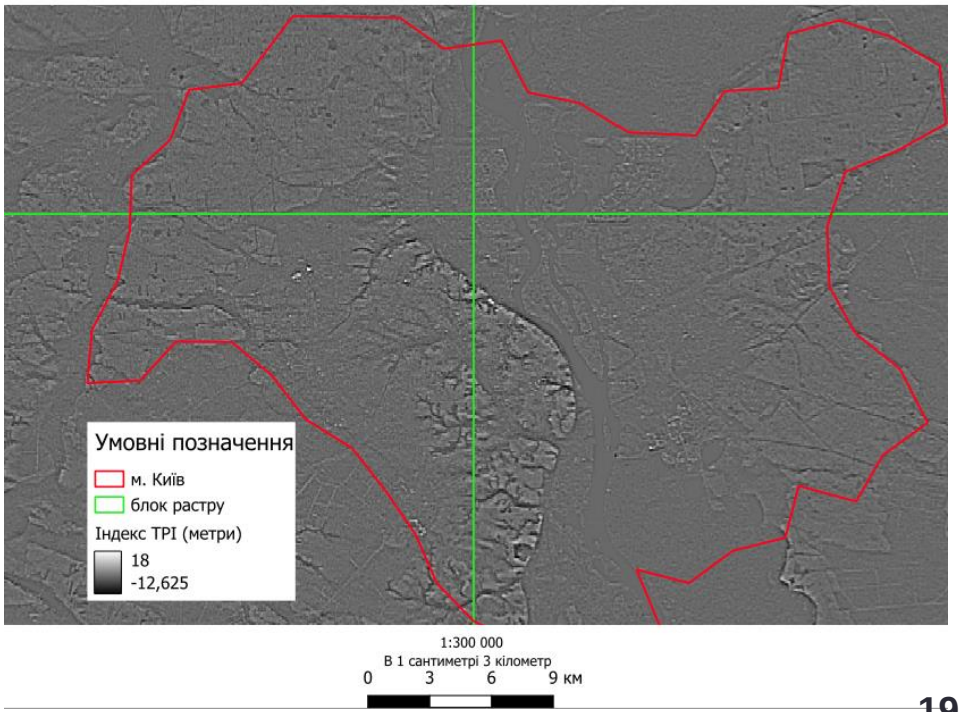


Типові запити для побудови освітленості схилів та індексу TPI

```
CREATE TABLE dem_hillshade AS
SELECT rid ST_HillShade(rast::raster,
1, '32BF'::text, 315, 45, 255, 1.0,
FALSE), AS rast
FROM dem
WHERE filename='N50E030.hgt'
```

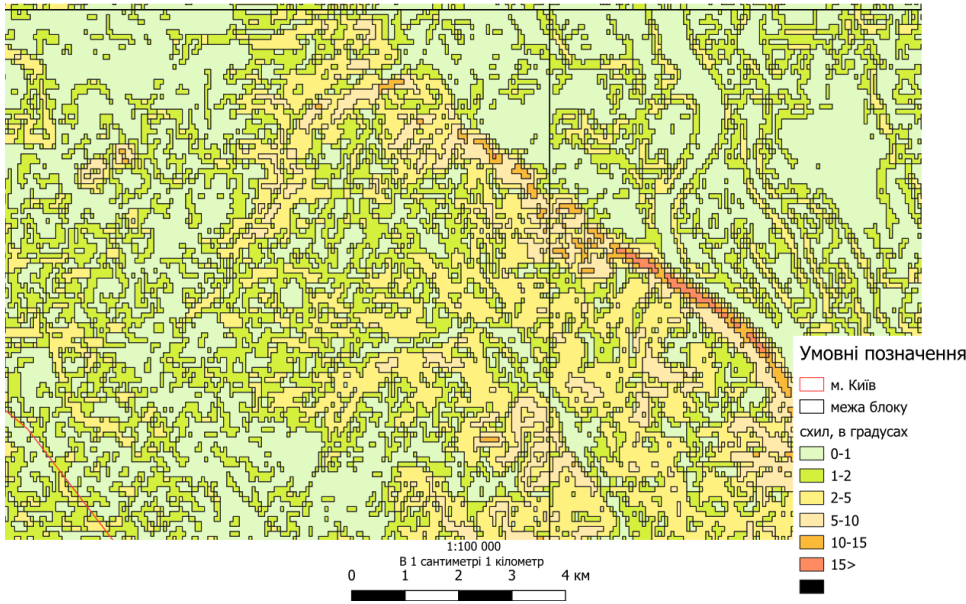
```
CREATE TABLE dem_tpi AS
SELECT rid, ST_TPI (a.rast, 1)
AS rast
FROM dem AS a
WHERE filename='N50E030.hgt'
```

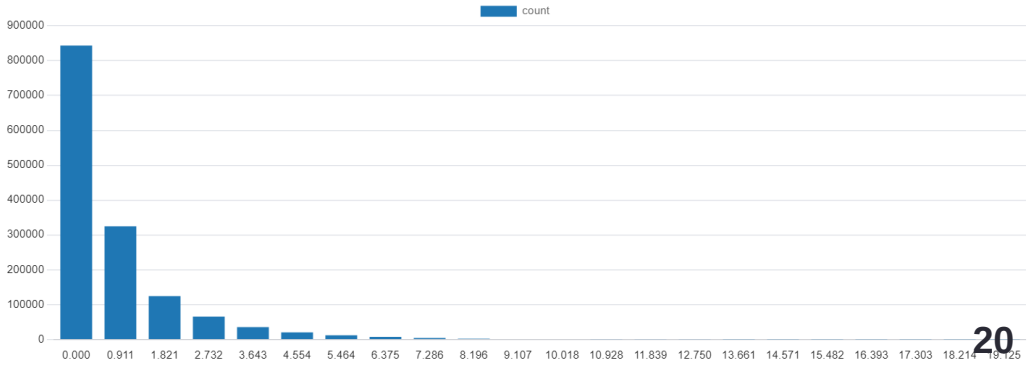




Автоматична векторизація рекласифікованої GRID моделі ухилу поверхні рельєфу

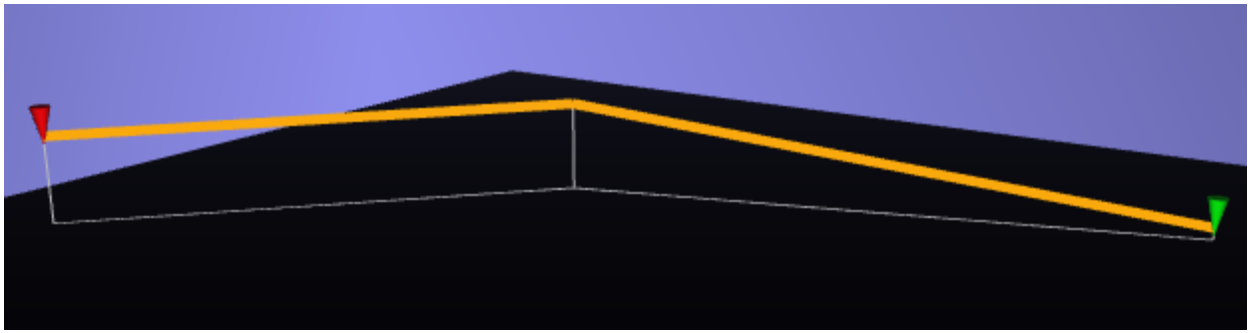
```
CREATE TABLE dem_reclassmpoly AS
SELECT val, geom AS geom
FROM (
  SELECT poly.*
  FROM dem, LATERAL
  ST_DumpAsPolygons(
    ST_Reclass(
      (ST_SLOPE(rast::raster, '1'::int,
'32BF'::text, 'DEGREES'::text,
'111120'::double precision))::raster,
      '1'::int, '0-1):1, 1-2):2, 2-5):3, 5-10):4,
10-15):5, [15-90):6'::text, '32BF'::text
    )
  ) AS poly
  WHERE dem.filename =
'N50E030.hgt') AS foo;
```



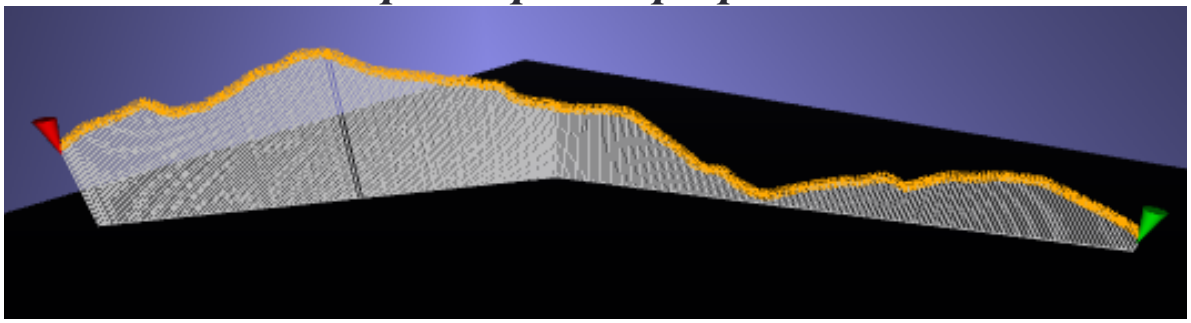


Типи моделей побудови 3D ліній на GRID-моделі

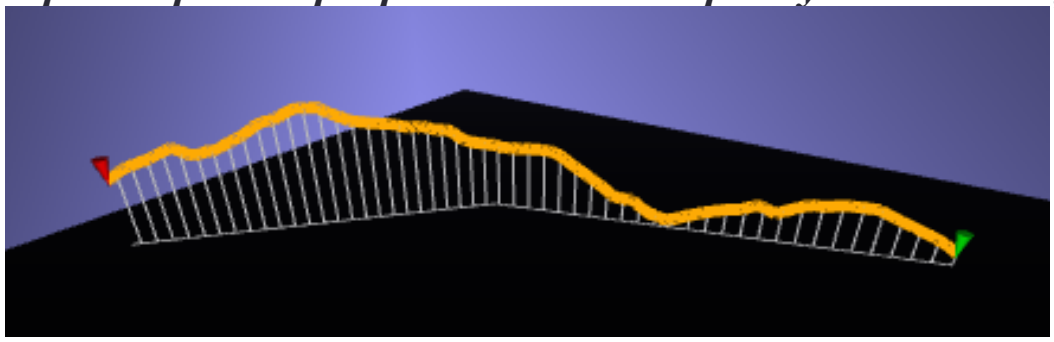
Моделювання з просторовим розрізненням вхідної 2D геометрії:



Моделювання з просторовим розрізненням GRID:

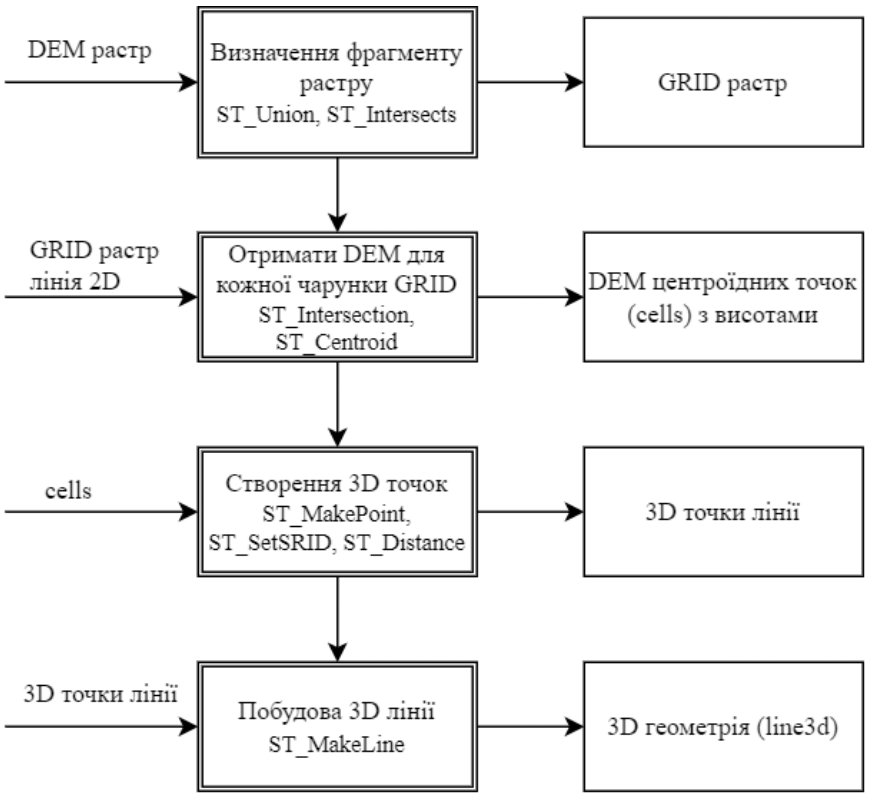


Моделювання з просторовим розрізненням інтервалу сегментації вхідної 2D лінії:



Функція побудови 3D ліній з розрізненням GRID-моделі

Схема алгоритму функції



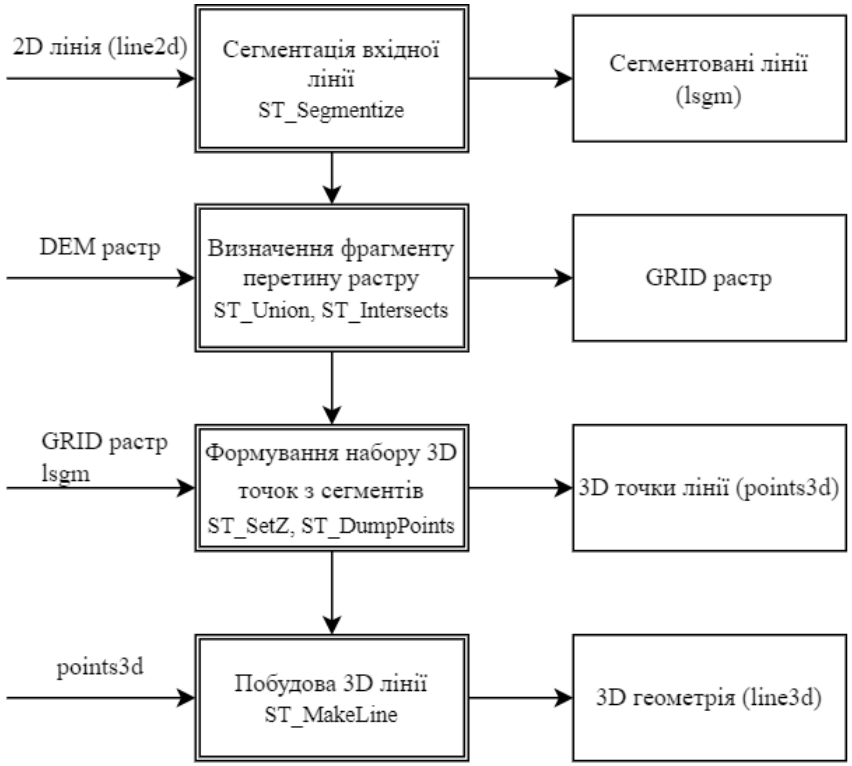
Текст SQL-функції:

```
CREATE OR REPLACE FUNCTION _line3d(line2d geometry)
RETURNS geometry AS
$BODY$
DECLARE
line3d geometry;
BEGIN
WITH grid AS
-- визначення фрагменту растру, перетинається з вхідною лінією
(SELECT ST_Union(dem.rast) AS rast  FROM dem
WHERE ST_Intersects(line2d, ST_ConvexHull(dem.rast))),
cells AS
-- отримати DEM для кожної чарунки GRID, що перетинається з лінією
(SELECT ST_Centroid((ST_Intersection(grid.rast, line2d)).geom) AS geom,
(ST_Intersection(grid.rast, line2d)).val AS val
FROM grid
WHERE ST_Intersects(grid.rast, line2d)),
-- створення 3D точок для перетину лінії з чарунками GRID
points3d AS (SELECT ST_SetSRID(ST_MakePoint(ST_X(cells.geom),
ST_Y(cells.geom), val), 4326) AS geom FROM cells
ORDER BY ST_Distance(ST_StartPoint(line2d), cells.geom))
-- побудова 3D лінії для 3D точок та повернення результату
SELECT ST_MakeLine(geom) INTO line3d FROM points3d;
 RETURN line3d;END;
                $BODY$
  LANGUAGE plpgsql VOLATILE STRICT
  COST 100;
```

Висновок: Функція забезпечує високу точність, але час обчислення є надто великим. Такий значний час обчислень пов'язаний із процедурою пошуку точок перетину вихідної лінії з прямокутниками пікселів GRID-моделі рельєфу. Прямокутники створюються для всього растру без просторового індексування, що потребує перебирати сотні тисяч пікселів, навіть для короткої лінії.

Функція побудови 3D ліній з розрізненням інтервалу сегментації вхідної 2D лінії

Схема алгоритму функції



Висновок: Функція ***_line3ds(line2d geometry, sgm_length float)*** виконується за 230–670 мс на лінію, що значно швидше порівняно з ***_line3d(line2d geometry)*** (~1 хв. 22 сек), завдяки прямій інтерполяції координат z без аналізу перетинів з прямокутниками пікселів GRID

Текст SQL-функції:

```
CREATE OR REPLACE FUNCTION _line3ds(line2d geometry,sgm_length
float)
  RETURNS geometry AS
$BODY$
DECLARE
line3d geometry;
          BEGIN
-- сегментація вхідної лінії з максимальною заданою довжиною сегмента
WITH lsgm AS (
 SELECT ST_Segmentize(line2d::geography,sgm_length)::geometry AS
geom),
-- визначення фрагменту dem, що перетинається з лінією
grid AS (
 SELECT ST_Union(dem.rast) AS rast
  FROM dem
  JOIN lsgm
  ON ST_Intersects(lsgm.geom, ST_ConvexHull(dem.rast))),
-- формування набору 3D точок із точок сегментів лінії та Z растру
points3d AS (
  SELECT (ST_DumpPoints(ST_SetZ(grid.rast, lsgm.geom, resample =>
'bilinear'))).*
 FROM grid
  CROSS JOIN lsgm)
-- формування 3d лінії
SELECT ST_MakeLine(points3d.geom) INTO line3d FROM points3d;
    RETURN line3d;
END;
          $BODY$
  LANGUAGE plpgsql VOLATILE STRICT
  COST 100;
```

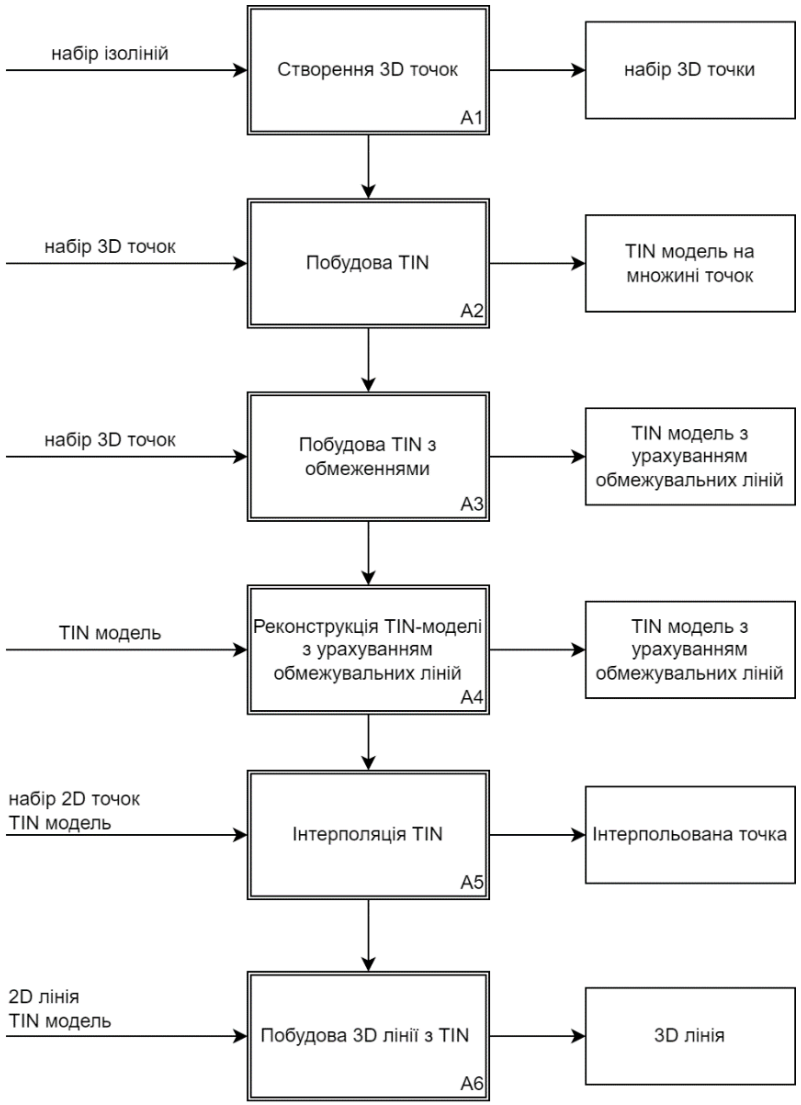
Функція обчислення довжини 3D лінії з просторовим розрізненням інтервалу сегментації вхідної 2D лінії

```
CREATE OR REPLACE FUNCTION _line3dslg(line2d geometry,sgm_length
float)
  RETURNS double precision AS
$BODY$
DECLARE
line3d geometry;
lg3d double precision;
          BEGIN
-- сегментація вхідної лінії з максимальною заданою довжиною сегмента
WITH lsgm AS (
 SELECT ST_Segmentize(line2d::geography,sgm_length)::geometry AS geom),
-- визначення фрагменту dem, що перетинається з лінією
grid AS (
 SELECT ST_Union(dem.rast) AS rast
  FROM dem
  JOIN lsgm
  ON ST_Intersects(lsgm.geom, ST_ConvexHull(dem.rast))),
-- формування набору 3D точок із точок сегметів лінії та Z растру
points3d AS (
  SELECT (ST_DumpPoints(ST_SetZ(grid.rast, lsgm.geom, resample =>
'bilinear'))).*
 FROM grid
  CROSS JOIN lsgm)
-- формування 3d лінії
SELECT ST_MakeLine(points3d.geom) INTO line3d FROM points3d;
-- обчислення довжини на еліпсоїді
 lg3d=ST_LengthSpheroid(line3d,
'SPHEROID["WGS 84",6378137,298.257223563]');
   RETURN lg3d;
END;
            $BODY$
  LANGUAGE plpgsql VOLATILE STRICT
  COST 100;
```

Висновок: Результати обчислень довжин ділянок вулиць для міста Києва показали, що різниця між довжинами 2D і 3D ліній, визначених на еліпсоїді, є незначною, до декількох метрів. Водночас, довжини на еліпсоїді та в проекції Меркатора суттєво відрізняються — до сотень і тисяч метрів.

Назва вулиці	Довжина вулиць, м			Різниця довжини	
	2D, на еліпсоїді	3D, на еліпсоїді	2D, на проєкції	на еліпсоїді	порівняно з L2D на проєкції
"Круглоуніверситетська"	609.278	611.853	954.811	2.575	342.958
"Велика Васильківська"	3704.338	3706.724	5803.161	2.386	2096.437
"Тараса Шевченка бульвар"	1877.911	1880.295	2942.979	2.384	1062.684

Технологічна схема дослідження використання TIN-моделі в PostGIS



A1) отримання 3D точок на основі набору ізоліній;

A2) побудова TIN-моделі без обмежень для набору 3D точок з використанням функції PostGIS ST_DelaunayTriangles;

A3) побудова TIN-моделі з врахуванням обмежувальних ліній з використанням функції PostGIS ST_ConstrainedDelaunayTriangles;

A4) розробка і використання функції для реконструкції TIN-моделі Делоне за набором обмежувальних ліній;

A5) реалізація функції інтерполяції висоти довільної 2D точки з використанням TIN-моделі

A6) використання TIN-моделі для підняття 2D лінії до 3D, враховуючи рельєф.

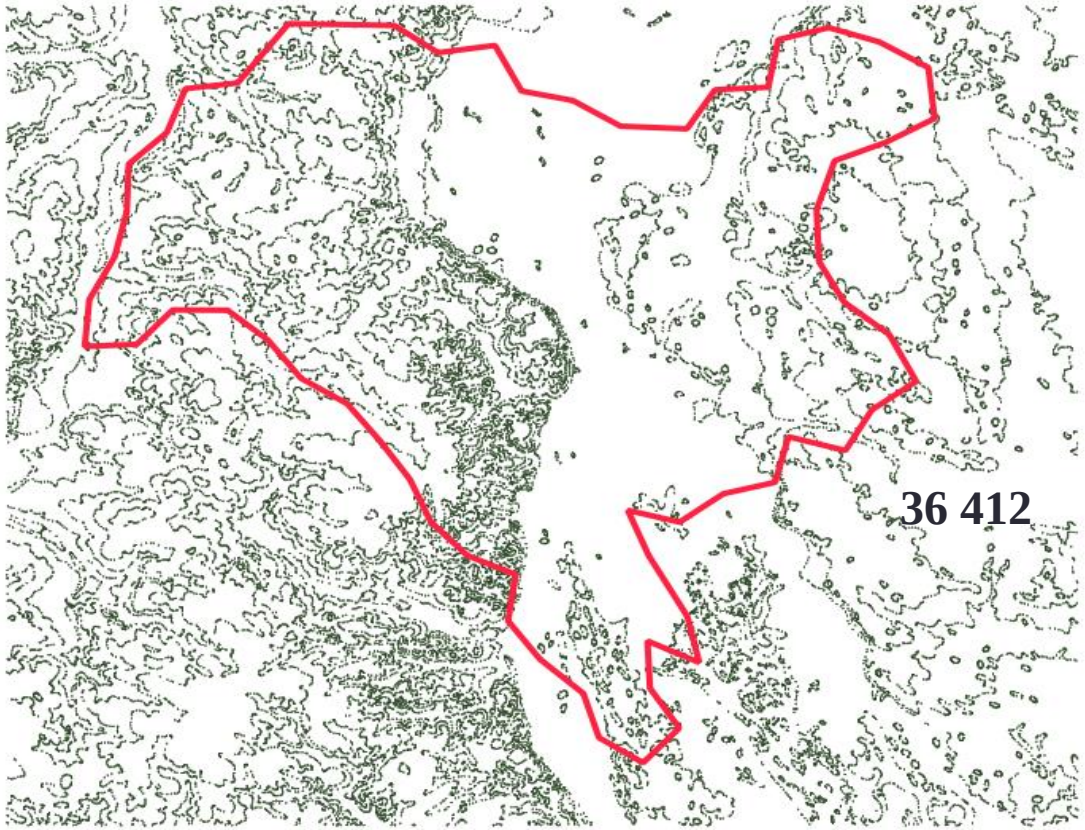
Формування набору 3D точок

Створення таблиці 3D точок:

```
CREATE TABLE pts3d(
    gid serial PRIMARY KEY,
    geom geometry(POINTZ,4326) );
```

Заповнення таблиці pts3d на основі 2D точок з атрибутом висоти:

```
INSERT INTO pts3d (gid, geom)
SELECT gid, ST_Force3DZ(p.geom, p.hg)
FROM point_kyiv AS p;
```



Запит тестування:

```
SELECT st_astext(geom) FROM pts3d LIMIT 4;
```

Результат:_____

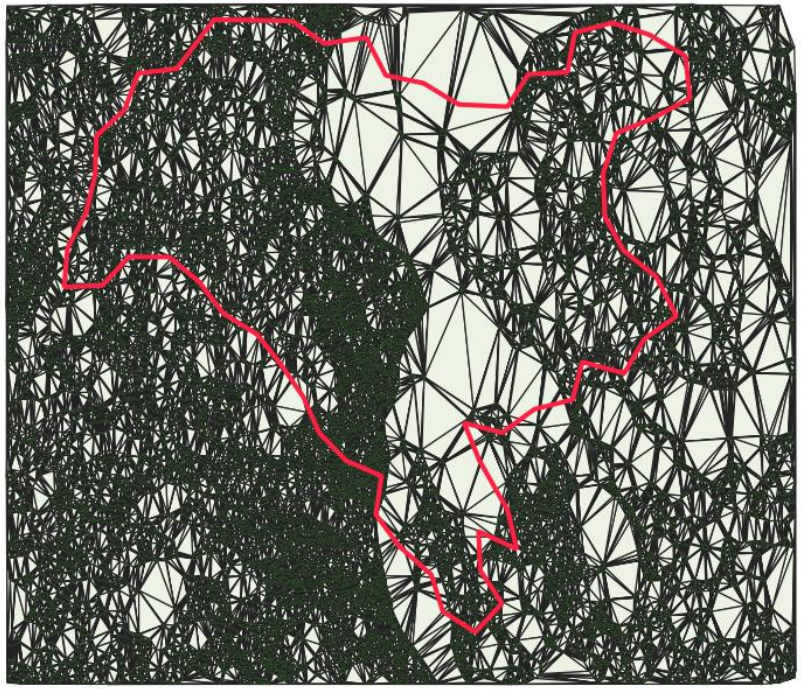
"POINT Z (30.742868999999992 50.595043000000004 120)"
"POINT Z (30.742466 50.594674999999995 120)"
"POINT Z (30.741374000000004 50.594024000000005 120)"
"POINT Z (30.74094 50.593279 120)"

Запити формування TIN-моделі та вивід окремих трикутників

Запит на побудову тріангуляції Делоне:

```
CREATE TABLE tin_dln AS
WITH pts AS
(
SELECT geom AS pt
FROM pts3d
)
SELECT ST_DelaunayTriangles(ST_Union(pt::geometry), 0.0, 0) AS
the_geom
FROM pts;
```

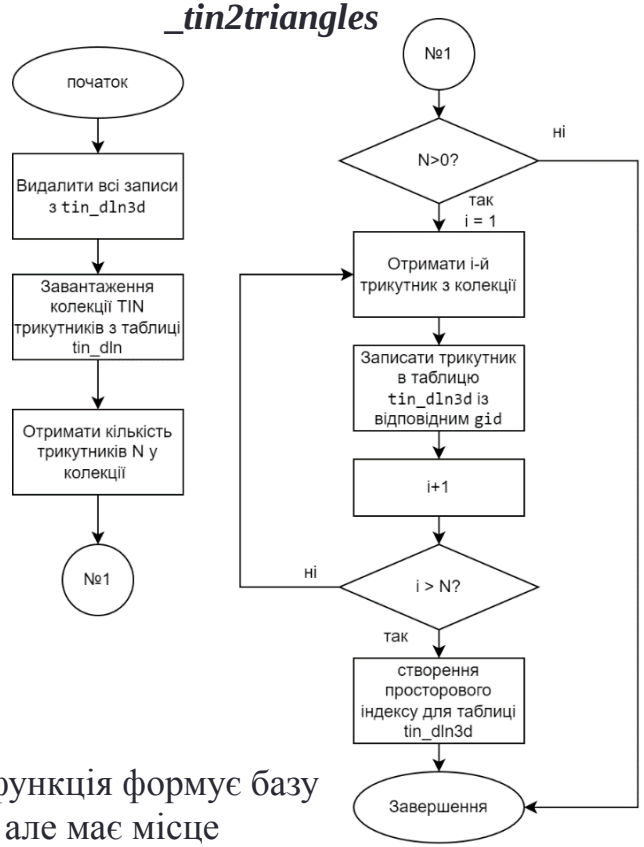
Висновок запиту: створено 71 940 трикутників за 808 msec, але вся TIN знаходиться в єдиному записі.



Розроблення таблиці вивідуй трикутників:

```
CREATE TABLE tin_dln3d(
   gid serial PRIMARY KEY,
   geom geometry(POLYGONZ,4326) );
```

Схема алгоритму функції *_tin2triangles*



Оцінка алгоритму: функція формує базу окремих трикутників але має місце великих затрат часу

Типові запити для формування TIN моделей з обмеженнями

запит типу PLT:

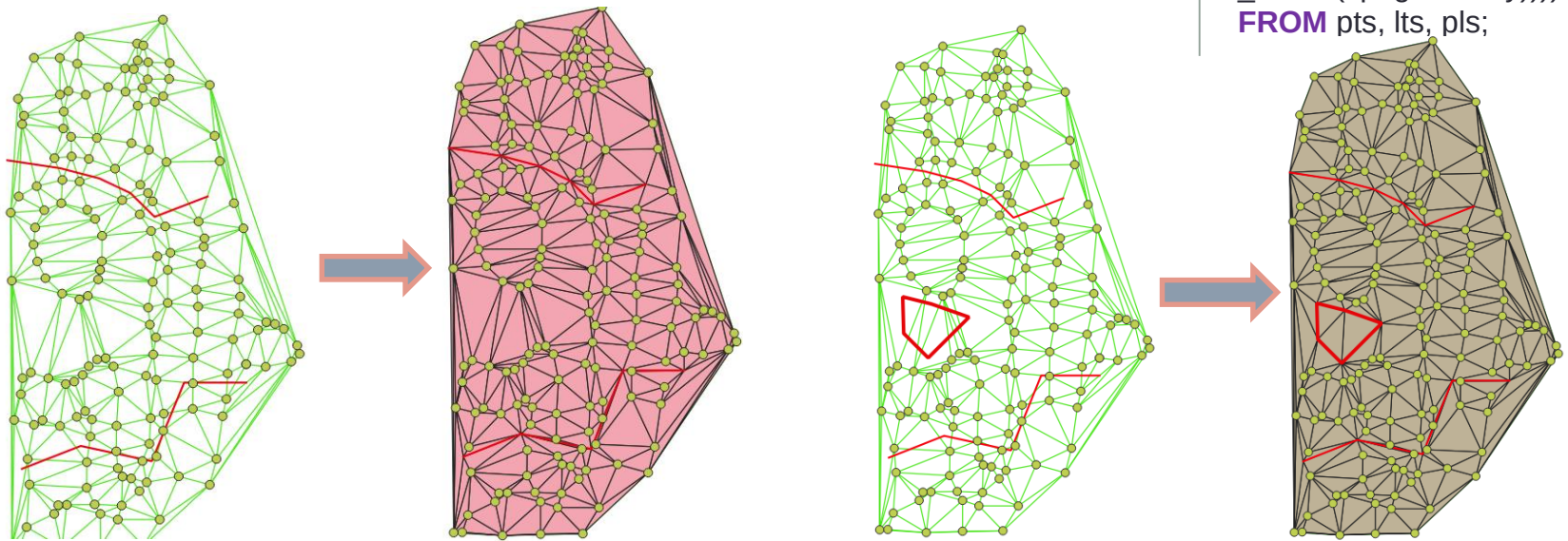
```
CREATE TABLE tin_dlc AS
WITH pts AS
(SELECT geom AS pt FROM points),
lts AS
(SELECT geom AS cl FROM constr_line)
SELECT
ST_ConstrainedDelaunayTriangles(ST_Colle
ct(ST_Union(pt::geometry),
ST_Union(cl::geometry))) AS geom
FROM pts, lts;
```

запит типу PPT :

```
CREATE TABLE tin_dlc AS
WITH pts AS
(SELECT geom AS pts FROM points),
pls AS
(SELECT geom AS cp FROM
constr_polygon)
SELECT
ST_ConstrainedDelaunayTriangles(ST_
Collect(ST_Union(pt::geometry),
ST_Collect(ST_Union(cp::geometry)))
AS geom
FROM pts, pls;
```

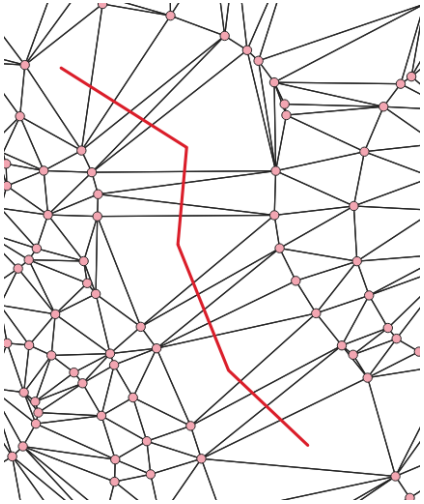
запит типу PLPT:

```
CREATE TABLE tin_dlc AS
WITH pts AS
(SELECT geom AS pts FROM points),
lts AS
(SELECT geom AS cl FROM
constr_line),
pls AS
(SELECT geom AS cp FROM
constr_polygon)
SELECT
ST_ConstrainedDelaunayTriangles(ST
_Collect(ST_Union(pt::geometry),
ST_Collect(ST_Union(cl::geometry),ST
_Union(cp::geometry)))) AS geom
FROM pts, lts, pls;
```

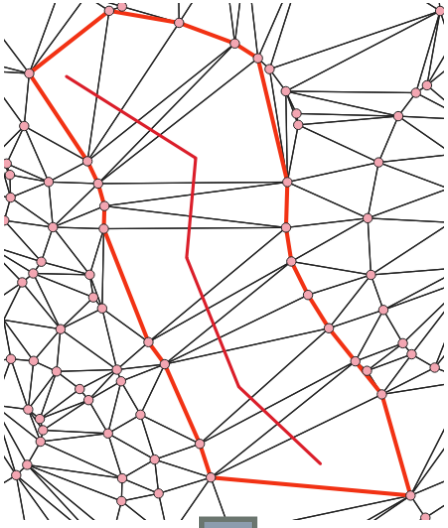


Реалізація функції реконструкції TIN-моделі за набором структурних ліній

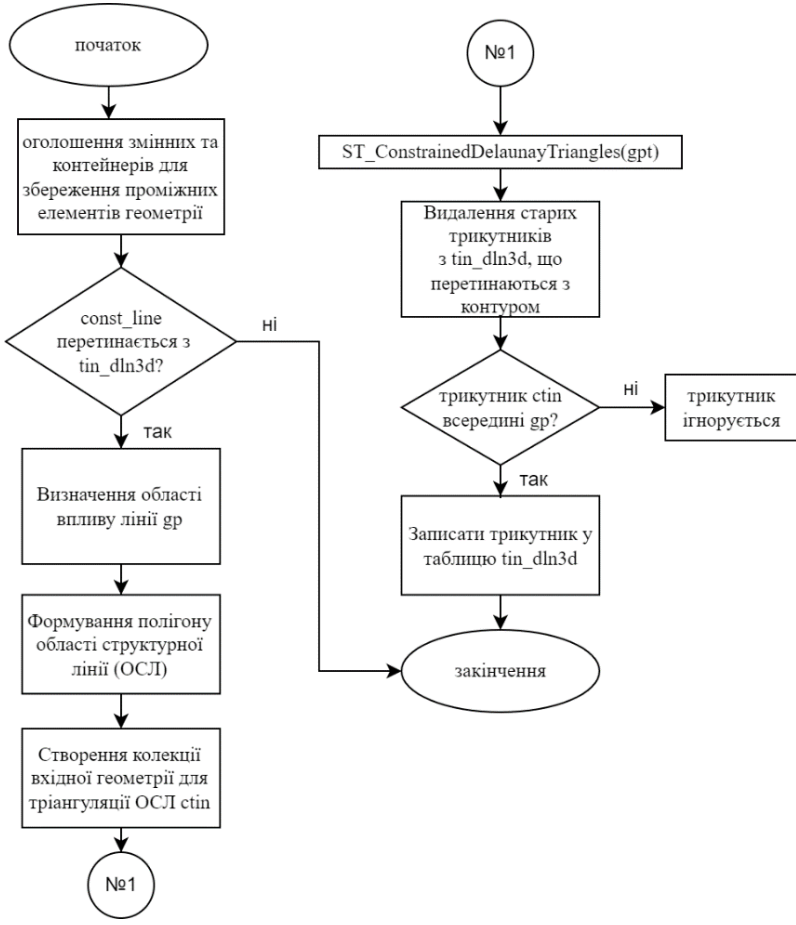
Структурна лінія:

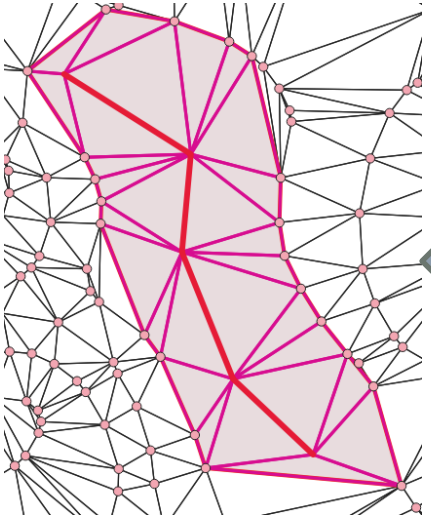


Визначення полігону області реконструкції та її вилучення :

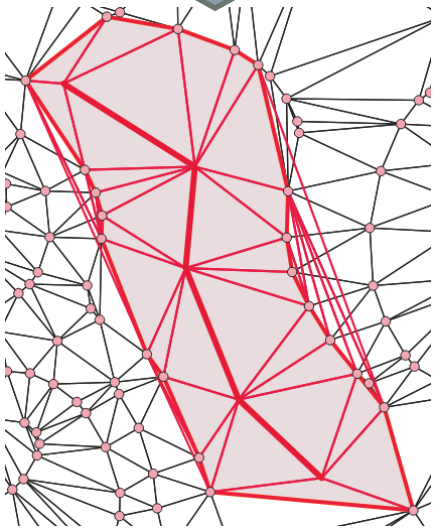


Узагальнена схема алгоритму функції *_rectin_cl()*





Обрізання зайвих трикутників та вставлення в TIN:



Тріангуляція області реконструкції:

Функція формування набору трикутників 3DTIN моделі рельєфу та її реконструкції

Функції _tin2triangles перетворює колекцію трикутників типу TIN в окремі трикутники

```
CREATE OR REPLACE FUNCTION _tin2triangles()
  RETURNS text AS
$BODY$
DECLARE
k record;
coltin geometry;
BEGIN
delete from tin_dln3d;
coltin = (SELECT geom FROM tin_dln);
FOR k in 1..ST_NumGeometries(coltin)
LOOP
insert into tin_dln3d(gid,geom)
            values (k,ST_GeometryN(coltin,k));
END LOOP;
CREATE INDEX tin_dln3d_geom_idx ON
            tin_dln3d USING GIST (geom);
return 'Done';
END;
             $BODY$
  LANGUAGE plpgsql VOLATILE STRICT
  COST 100;
```

Функція _rectin_cl() призначена для реконструкції TIN за обмеженнями

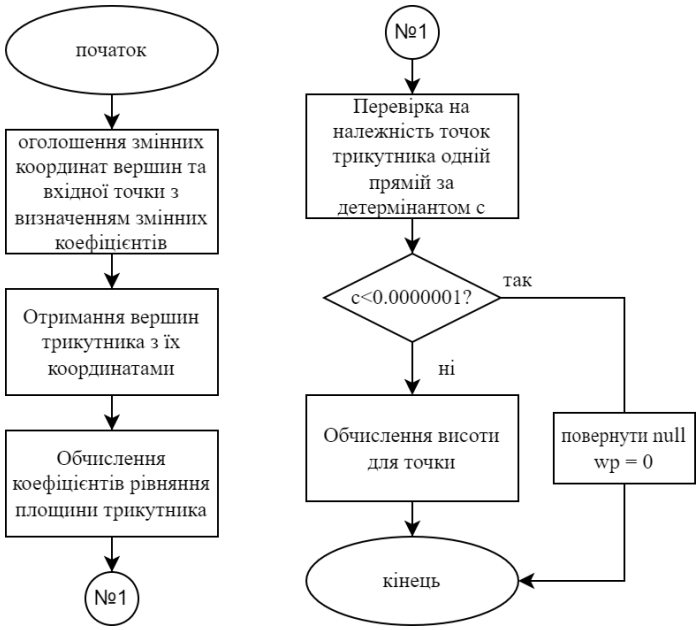
```
CREATE OR REPLACE FUNCTION _rectin_cl()
  RETURNS text AS
$BODY$
DECLARE
j record;
k record;
gpt geometry;
ctin geometry;
ontr geometry;
BEGIN
FOR j IN (SELECT gid,geom FROM constr_line)
  LOOP
gp=(SELECT ST_Union(geom) FROM tin_dln3d AS a
      WHERE ST_Intersects(j.geom, a.geom));
gpt=ST_Collect(j.geom,gp);
ctin=ST_ConstrainedDelaunayTriangles(gpt);
DELETE FROM tin_dln3d AS a
WHERE ST_Intersects(j.geom, a.geom);
FOR k in 1..ST_NumGeometries(ctin)
LOOP
ontr=ST_GeometryN(ctin,k);
IF ST_Within(ontr,gp) THEN
  insert into tin_dln3d (geom) values (ST_GeometryN(ctin,k);
END IF;
END LOOP;
END LOOP;
return 'Done';
            END;
            $BODY$
  LANGUAGE plpgsql VOLATILE STRICT
  COST 100;
```

Реалізація функції інтерполяції висоти довільної 2D точки за 3DTIN моделлю рельєфу

Функція _point_tin_value (triangle3d, point2d) повертає значення висоти для точки point2d за трикутником triangle3d поверхні TIN

```
CREATE OR REPLACE FUNCTION _point_tin_value
(triangle3d geometry, point2d geometry)
RETURNS double precision AS
$BODY$
DECLARE
x1 double precision;
y1 double precision;
w1 double precision;
x2 double precision;
y2 double precision;
w2 double precision;
x3 double precision;
y3 double precision;
w3 double precision;
xp double precision;
yp double precision;
tolerance double precision = 0.0000001;
a double precision;
b double precision;
c double precision;
pts geometry; -- контейнер мультиточки для вершин трикутника
p1 geometry; -- контейнери для точок вершин трикутника
p2 geometry;
p3 geometry;
BEGIN
-- отримання (витягування) точок та їх координат із вхідної геометрії
pts = ST_Points(triangle3d);
p1 = ST_GeometryN(pts, 1);
p2 = ST_GeometryN(pts, 2);
p3 = ST_GeometryN(pts, 3);
x1 = ST_X(p1);
y1 = ST_Y(p1);
w1 = ST_Z(p1);
x2 = ST_X(p2);
y2 = ST_Y(p2);
w2 = ST_Z(p2);
x3 = ST_X(p3);
y3 = ST_Y(p3);
w3 = ST_Z(p3);
xp = ST_X(point2d);
yp = ST_Y(point2d);
-- обчислення детермінантів для рівняння площини трикутника
a := ((y2-y1) * (w3-w1) - (w2-w1) * (y3-y1));
b := ((w2-w1) * (x3-x1) - (x2-x1) * (w3-w1));
c := ((x2-x1) * (y3-y1) - (y2-y1) * (x3-x1));
-- перевірка щодо належності точок трикутника одній прямій
if abs(c) < tolerance THEN return null;
end if;
-- обчислення та повернення висоти для вхідної точки
return (w1 - (((a * (xp-x1))+(b * (yp-y1)))/c));
END;
$BODY$
LANGUAGE plpgsql VOLATILE STRICT
  COST 100;
```

Загальна схема алгоритму функції **_point_tin_value (triangle3d, point2d)**



Запит на тестування функції:
_point_tin_value(triangle3d, point2d):

```
SELECT _point_tin_value(
'TIN Z(
((1000.0 1000.0 100.0,2000.0 1800.0 500.0,
2000 1000.0 100.0,1000.0 1000.0
100.0)))'::geometry,
'POINT(1250 1100)'::geometry);
```

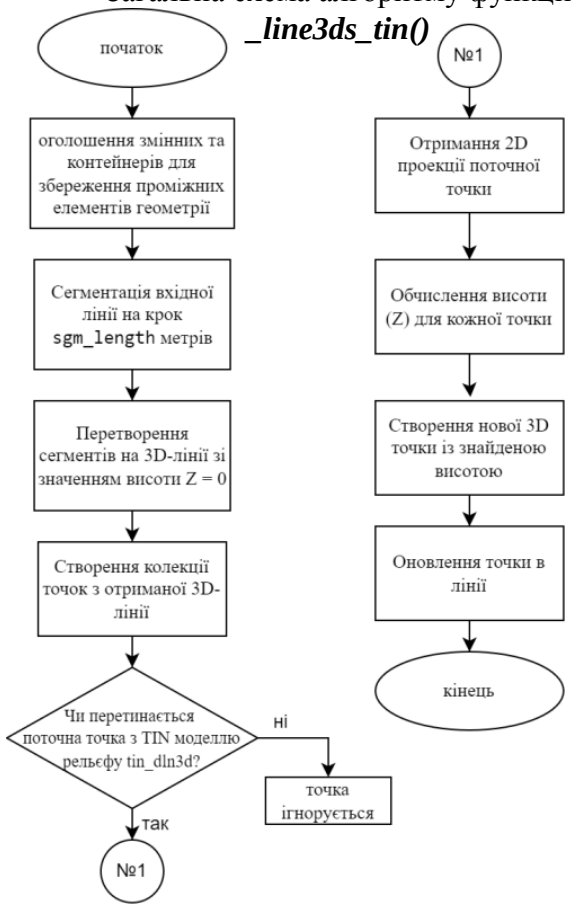
Результат: 150

Реалізація функції побудови 3D ліній з використанням 3DTIN моделлю рельєфу

Функція: _line3ds_tin(line2d geometry, sgm_length float) повертає 3D геометрію, сформовану для полілінії line2d з точками її сегментації.

```
CREATE OR REPLACE FUNCTION _line3ds_tin(line2d geometry, sgm_length
float)
  RETURNS geometry AS
$BODY$
DECLARE
lsgm geometry;
lsgm3d geometry;
pts geometry;
line3d geometry;
k record;
p2d geometry;
p3d geometry;
zp double precision;
BEGIN
lsgm = ST_Segmentize(line2d::geography,sgm_length)::geometry;
lsgm3d = ST_Force3D(lsgm, 0.0);
pts = ST_Points(lsgm3d); line3d = ST_LineFromMultiPoint(pts);
FOR k in 1..ST_NumGeometries(pts)
LOOP
p2d = ST_Force2D(ST_GeometryN(pts,k));
zp=(SELECT _point_tin_value(tin_dln3d.geom, p2d) FROM tin_dln3d
WHERE ST_Intersects(tin_dln3d.geom,p2d));
p3d = ST_Force3D(p2d,zp);
line3d = ST_SetPoint(line3d,k-1,p3d);
END LOOP;
RETURN line3d;
END;
$BODY$
  LANGUAGE plpgsql VOLATILE STRICT
  COST 100;
```

Загальна схема алгоритму функції ***_line3ds_tin()***



Запит обрахунку бульвару Тараса Шевченка:

```
SELECT ST_AsText (_line3ds_tin(str_test.geom, 50.0))
FROM str_test WHERE str_test.gid = 1;
```

Результат:

Successfully run. Total query runtime: 196 msec

"LINESTRING Z (30.4944945 50.4463704 140,30.4950213 50.4462878 140)"

Порівняння висоти точок і довжин вулиць отриманих за різних моделями рельєфу

Порівняння висоти 3D точок осьових ліній бульвару Тараса Шевченка в Києві за GRID і TIN моделями

№	Координати точок				Відхилення Ztin - Zgrid
	Довгота	Широта	Zgrid, м	Ztin, м	
1	30.5171977	50.4426882	166.184	166.971	0,787
2	30.5172358	50.4426821	166.121	166.956	0,853
3	30.517305	50.442671	166.007	166.928	0,921
4	30.517657027912414	50.44261506621689	165.825	165.966	0,141
5	30.51800905499273	50.442559131371794	166.332	164.835	- 1,497
6	30.51836108124095	50.442503195464745	166.800	163.704	- 2,006
7	30.518713106657042	50.442447258495704	164.208	162.573	- 1,635
8	30.519065131241003	50.442391320046471	161.500	161.443	- 0,057
9	30.519417154992833	50.442335381371755	158.196	160.312	2,116
10	30.5197691779125	50.44227944121686	154.920	160.000	5,080
11	30.5201212	50.4422235	152.851	158.574	5,723

Порівняння загальних довжин осьових ліній вулиць тестового набору

№	Назва вулиці	Довжина вулиці на еліпсоїді, м			L_3Dgrid - L_3D tin
		L_2D	L_3D grid	L_3D tin	
1	Круглоуніверситетська	609.278	611.853	610.987	0.866
2	Велика Васильківська	3704.338	3706.724	3705.428	1.296
3	Тараса Шевченка бульвар	1877.911	1880.295	1879.143	1.153

Статистичний аналіз відхилення висот між GRID і TIN моделями:

- Мінімальне відхилення: 0,057 м
- Середнє абсолютне відхилення: 1,892 м
- Середнє квадратичне відхилення: 2,598 м

Основні причини відхилень:

1. Різні системи відліку висот (GRID: SRTM, TIN: ізолінії рельєфу).
2. Відмінності в просторовому розрізненні моделей.

Висновок:

1.Незважаючи на відмінності значень координат **z** між GRID і TIN моделями рельєфу, **різниця у довжинах 3D ліній** є незначною (від **0,866 до 1,296 м**).

2.Це підтверджує **коректність** розроблених функцій **_point_tin_value** та **_line3ds_tin**.

Загальні висновоки

У роботі на реальних наборах даних проведено обчислювальні експерименти щодо створення цифрових моделей рельєфу в середовищі СКБД PostgreSQL/PostGIS та встановлено що:

а) в PostGIS надаються ефективні засоби для підтримки і аналізу GRID моделей рельєфу на основі спеціального типу даних для растрових моделей та набору функцій побудови поверхонь морфологічних характеристик рельєфу;

б) базові функції PostGIS реалізують тріангуляцію Делоне та тріангуляцію Делене з обмеженнями з наданням результатів у форматі спеціального типу даних TIN як єдиної колекції трикутників, що є раціональним з точки зору реалізації вбудованих прикладних функцій тріангуляції.

2. Для підвищення ефективності використання базових функцій PostGIS моделювання рельєфу в прикладних задачах в роботі реалізовані прикладні SQL функції, які забезпечують:

а) побудову 3D моделей ліній та визначення їх довжини на поверхні з використанням GRID моделі рельєфу;

б) формування набору трикутників TIN моделі рельєфу з використанням базових функцій PostGIS для тріангуляції Делоне та тріангуляції з обмеженнями та побудовою просторового індексу для набору трикутників для оптимізації доступу до них в прикладних задачах моделювання 3D об’єктів та аналізу рельєфу;

в) реконструкцію існуючих TIN моделі рельєфу за набором геопросторових даних структурних ліній рельєфу;

г) визначення висоти довільної 2D точки на поверхні з використанням TIN моделі рельєфу;

д) побудову 3D моделей ліній на поверхні заданої TIN моделлю рельєфу.

3. Коректність запропонованих алгоритмів та реалізованих функцій підтверджено результатами обчислювальних експериментів з побудови TIN моделі рельєфу на дослідну територію міста Києва, моделювання 3D ліній ділянок осьових ліній тестового набору вулиць та визначення їх довжини.

ДОДАТОК Б

ТЕКСТИ РОЗРОБЛЕНИХ ПРИКЛАДНИХ SQL ФУНКЦІЙ

Б.1 Функції побудови 3D ліній з використанням GRID моделей

Б.1.1 Функція побудови 3D лінії з просторовим розрізненням GRID моделі рельєфу

```
-- Функція: _line3d(line2d geometry)
-- повертає 3D геометрію, сформовану для полілінії
-- з точками перетину полілінії з чарунками GRID рельєфу,
-- що міститься в таблиці dem
        CREATE OR REPLACE FUNCTION _line3d(line2d geometry)
        RETURNS geometry AS
 $BODY$
 DECLARE
 line3d geometry;
        BEGIN
        WITH grid AS
 -- визначення фрагменту растру, перетинається з вхідною лінією
        (SELECT ST_Union(dem.rast) AS rast   FROM dem
        WHERE ST_Intersects(line2d, ST_ConvexHull(dem.rast))),
 cells AS
-- отримати DEM для кожної чарунки GRID, що перетинається з лінією
        (SELECT ST_Centroid((ST_Intersection(grid.rast, line2d)).geom) AS geom,
        (ST_Intersection(grid.rast, line2d)).val AS val
        FROM grid
        WHERE ST_Intersects(grid.rast, line2d)),
-- сворення 3D точок для перетину лінії з чарунками GRID
 points3d AS (SELECT ST_SetSRID(ST_MakePoint(ST_X(cells.geom), ST_Y(cells.geom),
 val), 4326) AS geom FROM cells
        ORDER BY ST_Distance(ST_StartPoint(line2d), cells.geom))
-- побудова 3D лінії для 3D точок та повернення результату
        SELECT ST_MakeLine(geom) INTO line3d FROM points3d;
         RETURN line3d;
 END;
 $BODY$
 LANGUAGE plpgsql VOLATILE STRICT
 COST 100;
-- DROP FUNCTION _line3d(geometry)
```

Б.1.2 Функція побудови 3D лінії з просторовим розрізненням інтервалу сегментації вхідної 2D лінії

```
-- Функція: _line3ds(line2d geometry, sgm_length float)
-- повертає 3D геометрію, сформовану для полілінії line2d
-- з точками її сегментації на крок sgm_length м та Z
-- за чарунками GRID рельєфу, що міститься в таблиці dem
CREATE OR REPLACE FUNCTION _line3ds(line2d geometry,sgm_length float)
  RETURNS geometry AS
$BODY$
DECLARE
line3d geometry;
        BEGIN
-- сегментація вхідної лінії з максимальною заданою довжиною сегмента
WITH lsgm AS (
 SELECT ST_Segmentize(line2d::geography,sgm_length)::geometry AS geom),
-- визначення фрагменту dem, що перетинається з лінією
grid AS (
 SELECT ST_Union(dem.rast) AS rast
  FROM dem
  JOIN lsgm
  ON ST_Intersects(lsgm.geom, ST_ConvexHull(dem.rast))),
-- формування набору 3D точок із точок сегментів лінії та Z растру
points3d AS (
  SELECT (ST_DumpPoints(ST_SetZ(grid.rast, lsgm.geom, resample => 'bilinear'))).*
 FROM grid
  CROSS JOIN lsgm)
-- формування 3d лінії
SELECT ST_MakeLine(points3d.geom) INTO line3d FROM points3d;
   RETURN line3d;
END;
        $BODY$
  LANGUAGE plpgsql VOLATILE STRICT
  COST 100;
```

Б.1.3 Функція обчислення довжини 3D лінії з просторовим розрізненням інтервалу сегментації вхідної 2D лінії

```
-- Функція: _line3dslg(line2d geometry, sgm_length float)
-- повертає довжину на еліпсоїді 3D лінії, сформованої для полілінії line2d
-- з точками її сегментації на крок sgm_length м та Z за чарунками GRID рельєфу,
-- що міститься в таблиці dem
CREATE OR REPLACE FUNCTION _line3dslg(line2d geometry,sgm_length float)
  RETURNS double precision AS
$BODY$
DECLARE
line3d geometry;
lg3d double precision;
        BEGIN
-- сегментація вхідної лінії з максимальною заданою довжиною сегмента
WITH lsgm AS (
 SELECT ST_Segmentize(line2d::geography,sgm_length)::geometry AS geom),
-- визначення фрагменту dem, що перетинається з лінією
grid AS (
 SELECT ST_Union(dem.rast) AS rast
  FROM dem
  JOIN lsgm
  ON ST_Intersects(lsgm.geom, ST_ConvexHull(dem.rast))),
-- формування набору 3D точок із точок сегметів лінії та Z растру
points3d AS (
  SELECT (ST_DumpPoints(ST_SetZ(grid.rast, lsgm.geom, resample => 'bilinear'))).*
 FROM grid
  CROSS JOIN lsgm)
-- формування 3d лінії
SELECT ST_MakeLine(points3d.geom) INTO line3d FROM points3d;
  lg3d=ST_LengthSpheroid(line3d,
        'SPHEROID["WGS 84",6378137,298.257223563]');
   RETURN lg3d;
END;
        $BODY$
  LANGUAGE plpgsql VOLATILE STRICT
  COST 100;
-- DROP FUNCTION _line3dslg(geometry,float)
```

Б.2 Функції побудови та використання TIN моделі рельєфу

Б.2.1 Функція формування набору трикутників 3DTIN моделі рельєфу

```
-- функції _tin2triangles перетворює колекцію трикутників типу TIN
-- тріангуляції Делоне із таблиці tin_dln в набір окремих
-- трикутників зі зберіганням їх полігонів в таблиці tin_dln3d
CREATE OR REPLACE FUNCTION _tin2triangles()
  RETURNS text AS
$BODY$
DECLARE
k record;
coltin geometry; -- колекція трикутників TIN
        BEGIN
delete from tin_dln3d;
-- читаємо колекцію полігонів трикутників TIN
coltin = (SELECT geom FROM tin_dln);
-- цикл за числом трикутників в TIN для їх запису в таблиці tin_dln3d
FOR k in 1..ST_NumGeometries(coltin)
LOOP
insert into tin_dln3d(gid,geom)
                        values (k,ST_GeometryN(coltin,k));
END LOOP;
-- створення просторового індексу для таблиці tin_dln3d
CREATE INDEX tin_dln3d_geom_idx ON
        tin_dln3d USING GIST (geom);
return 'Done';
        END;
        $BODY$
  LANGUAGE plpgsql VOLATILE STRICT
  COST 100;
-- DROP FUNCTION _tin2triangles()
```

Б.2.2 Функція реконструкції TIN моделі за структурними лініями

```
-- Функція _rectin_cl() призначена для реконструкції TIN за обмеженнями
 -- набору структурних ліній.
-- Вхідні дані: tin_dln3d – таблиця з набором трикутників TIN;
-- constr_line  - таблиця з набором структурних ліній.
-- Результат – оновлений набір трикутників TIN в tin_dln3d.
CREATE OR REPLACE FUNCTION _rectin_cl()
  RETURNS text AS
```

```
$BODY$
DECLARE
-- змінні циклів
j record;
k record;
-- контейнери для зберігання проміжних геометричних елементів
gp geometry; -- полігон області структурної лінії (ОСЛ)
gpt geometry; -- колекція полігону ОСЛ і структурної лінії
ctin geometry; -- колекція трикутників після тріангуляції gpt
ontr geometry; -- полігон поточного трикутника
        BEGIN
-- цикл для таблиці структурних ліній
FOR j IN (SELECT gid,geom FROM constr_line)
 LOOP
-- формування полігону області структурної лінії (ОСЛ) об'єднанням
-- трикутників, які перетинаються структурною лінією
 gp=(SELECT ST_Union(geom) FROM tin_dln3d AS a
        WHERE ST_Intersects(j.geom, a.geom));
-- формування колекції вхідної геометрії для тріангуляції ОСЛ
 gpt=ST_Collect(j.geom,gp);
-- тріангуляція з обмеженням колекції геометрії ОСЛ
 ctin=ST_ConstrainedDelaunayTriangles(gpt);
-- вилучення трикутників із tin_dln3d, які перетинаються структурною лінією
DELETE FROM tin_dln3d AS a
WHERE ST_Intersects(j.geom, a.geom);
-- розпаковка колекції трикутників TIN ОСЛ  та їх запис у таблицю tin_dln3d
FOR k in 1..ST_NumGeometries(ctin)
        LOOP
        ontr=ST_GeometryN(ctin,k);
--  записуємо лише трикутники, що лежать всередині ОСЛ
        IF ST_Within(ontr,gp) THEN
  insert into tin_dln3d (geom) values (ST_GeometryN(ctin,k);
        END IF;
        END LOOP;
END LOOP;
return 'Done';
        END;
        $BODY$
 LANGUAGE plpgsql VOLATILE STRICT
 COST 100;
```

Б.2.3 Функція інтерполяції висоти довільної 2D точки за 3DTIN моделлю рельєфу

```
-- Функція _point_tin_value (triangle3d, point2d)
-- повертає значення висоти для точки point2d
-- за трикутником triangle3d поверхні TIN
-- на основі визначення коєфіцієнтів рівняння площини трикутника
-- за 3D координатами його вершин
CREATE OR REPLACE FUNCTION _point_tin_value
(triangle3d geometry, point2d geometry)
RETURNS double precision AS
$BODY$
DECLARE
x1 double precision;
y1 double precision;
w1 double precision;
x2 double precision;
y2 double precision;
w2 double precision;
x3 double precision;
y3 double precision;
w3 double precision;
xp double precision;
yp double precision;
tolerance double precision = 0.0000001;
a double precision;
b double precision;
c double precision;
pts geometry; -- контейнер мультиточки для вершин трикутника
p1 geometry; -- контейнери для точок вершин трикутника
p2 geometry;
p3 geometry;
BEGIN
-- отримання (витягування) точок та їх координат із вхідної геометрії
pts = ST_Points(triangle3d);
p1 = ST_GeometryN(pts, 1);
p2 = ST_GeometryN(pts, 2);
p3 = ST_GeometryN(pts, 3);
x1 = ST_X(p1);
y1 = ST_Y(p1);
w1 = ST_Z(p1);
```

```
x2 = ST_X(p2);
y2 = ST_Y(p2);
w2 = ST_Z(p2);
x3 = ST_X(p3);
y3 = ST_Y(p3);
w3 = ST_Z(p3);
xp = ST_X(point2d);
yp = ST_Y(point2d);
-- обчислення детермінантів для рівняння площини трикутника
a := ((y2-y1) * (w3-w1) - (w2-w1) * (y3-y1));
b := ((w2-w1) * (x3-x1) - (x2-x1) * (w3-w1));
c := ((x2-x1) * (y3-y1) - (y2-y1) * (x3-x1));
-- перевірка щодо належності точок трикутника одній прямій
if abs(c) < tolerance THEN return null;
end if;
-- обчислення та повернення висоти для вхідної точки
return (w1 - (((a * (xp-x1))+(b * (yp-y1)))/c));
END;
$BODY$
LANGUAGE plpgsql VOLATILE STRICT
  COST 100;
-- DROP FUNCTION _point_tin_value(geometry,geometry)
```

Б.2.4 Функція побудови 3D лінії з використанням 3DTIN моделі рельєфу

```
-- Функція: _line3ds_tin(line2d geometry, sgm_length float)
-- повертає 3D геометрію, сформовану для полілінії line2d
-- з точками її сегментації на крок sgm_length метрів та Z
-- за 3D TIN рельєфу, що міститься в таблиці tin_dln3d в BLZ
CREATE OR REPLACE FUNCTION _line3ds_tin(line2d geometry, sgm_length float)
  RETURNS geometry AS
$BODY$
DECLARE
lsgm geometry; -- сегментована вхідна 2D лінії
lsgm3d geometry; -- сегментована лінія у форматі 3D лінії
pts geometry; -- 3D MULTIPOINT сегментів лінії без дублювання точок
line3d geometry; -- контейнер для змодельованої 3D лінії
k record; -- змінна циклу за числом точок в pts
p2d geometry; -- поточна 2D POINT
```

```
p3d geometry; -- поточна 3D POINT
zp double precision; -- висота поточної точки
BEGIN
-- сегментація вхідної лінії з максимальною заданою довжиною сегмента
lsgm = ST_Segmentize(line2d::geography,sgm_length)::geometry;
-- перетворення сегментованої 2D лінії у формати 3D лінії із z=0
lsgm3d = ST_Force3D(lsgm, 0.0);
-- створення колекції 3D MultiPoint із сегментів 3D лінії
pts = ST_Points(lsgm3d);
-- побудова коректного шаблону моделі 3D лінії типу LINESTRINGZ
line3d = ST_LineFromMultiPoint(pts);
-- цикл визначення висоти для точок 3D лінії
FOR k in 1..ST_NumGeometries(pts)
LOOP
p2d = ST_Force2D(ST_GeometryN(pts,k));
zp=(SELECT _point_tin_value(tin_dln3d.geom, p2d) FROM tin_dln3d
                        WHERE ST_Intersects(tin_dln3d.geom,p2d));
p3d = ST_Force3D(p2d,zp); -- формування поточної 3D точки
-- оновлення поточної точки в шаблоні 3D лінії типу LINESTRINGZ
line3d = ST_SetPoint(line3d,k-1,p3d);
END LOOP;
-- повернення побудованої 3D лінії
   RETURN line3d;
END;
        $BODY$
  LANGUAGE plpgsql VOLATILE STRICT
  COST 100;
-- DROP FUNCTION _line3ds_tin(geometry,float)
```