INTERACTIVE MAPGEN

Team 64

OVERVIEW

- Introduction
- Related work
- Dataset/Platform
- Baseline
- Main Approach
- Evaluation Metric
- Results & Analysis
- Future work

Introduction

INTRODUCTION

We built an interface that allows users to generate fantasy maps directly from natural language descriptions. By combining the capabilities of LLMs (Grok) with open source map generation tools, we bridge the gap between intuitive user input and automated map creation. Our project simplifies the traditional complex map-generation process into a streamlined "**text-to-map**" workflow. This enables writers, game developers, and hobbyists to quickly visualize imaginative worlds without needing technical expertise or manual design work.

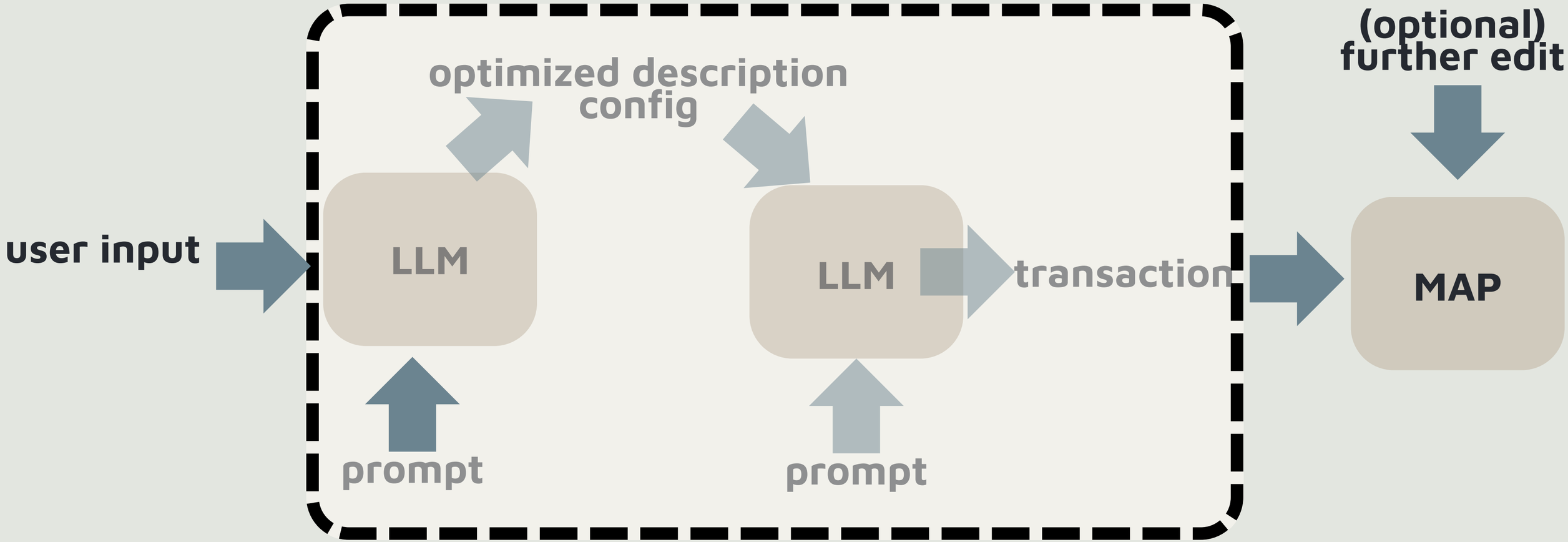
WHY IS THIS PROBLEM IMPORTANT?

- Target user: novelist, game developer
- Their needs:
  - customization
  - some sort of automization
  - logical terrain generation
  - Inspiration

WHY DO WE SPEND TIME WORKING ON THIS?

- generating maps by hand is slow and inefficient.
- extend existing tools features
  - semantic bridge between the visual geography and user intent.
  - improve UI for user experience

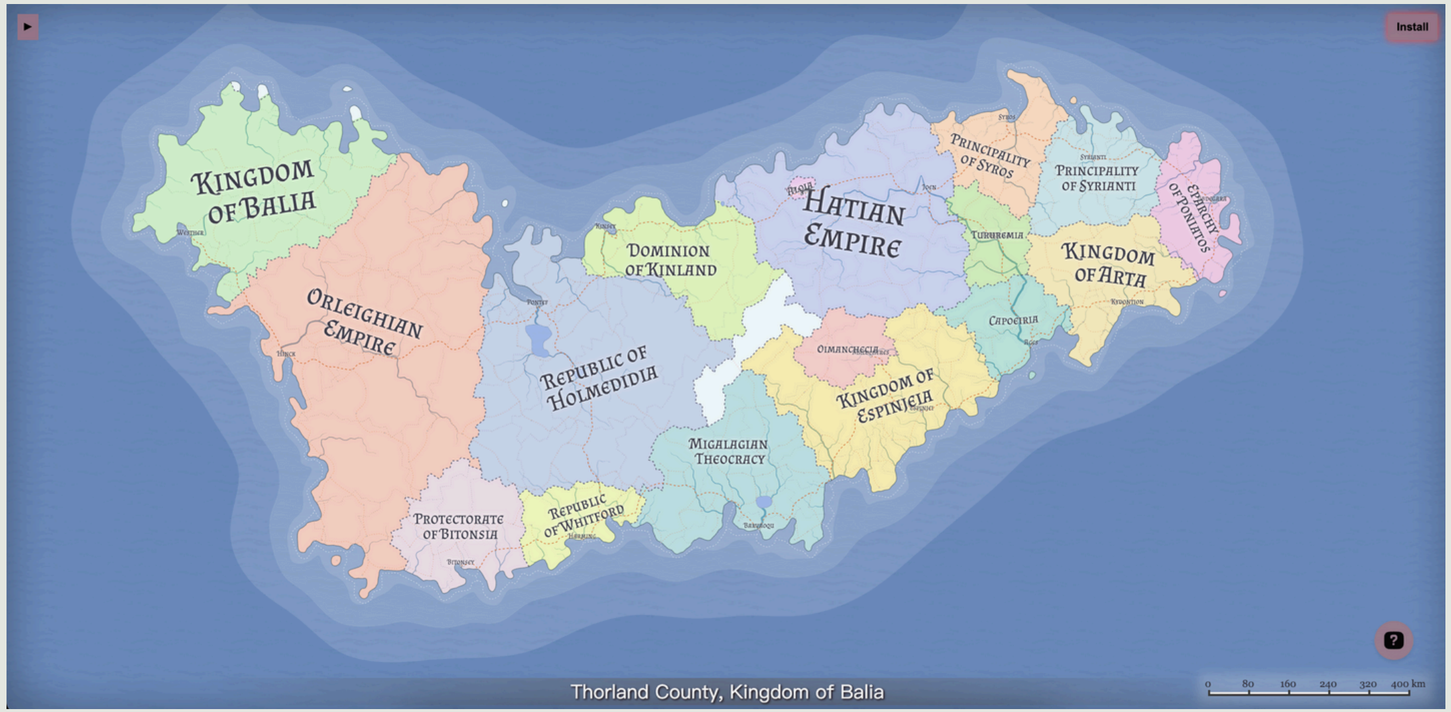
PIPELINE


optimized description
config
user input
LLM
LLM
transaction
(optional)
further edit
MAP
prompt
prompt
our work

Related work

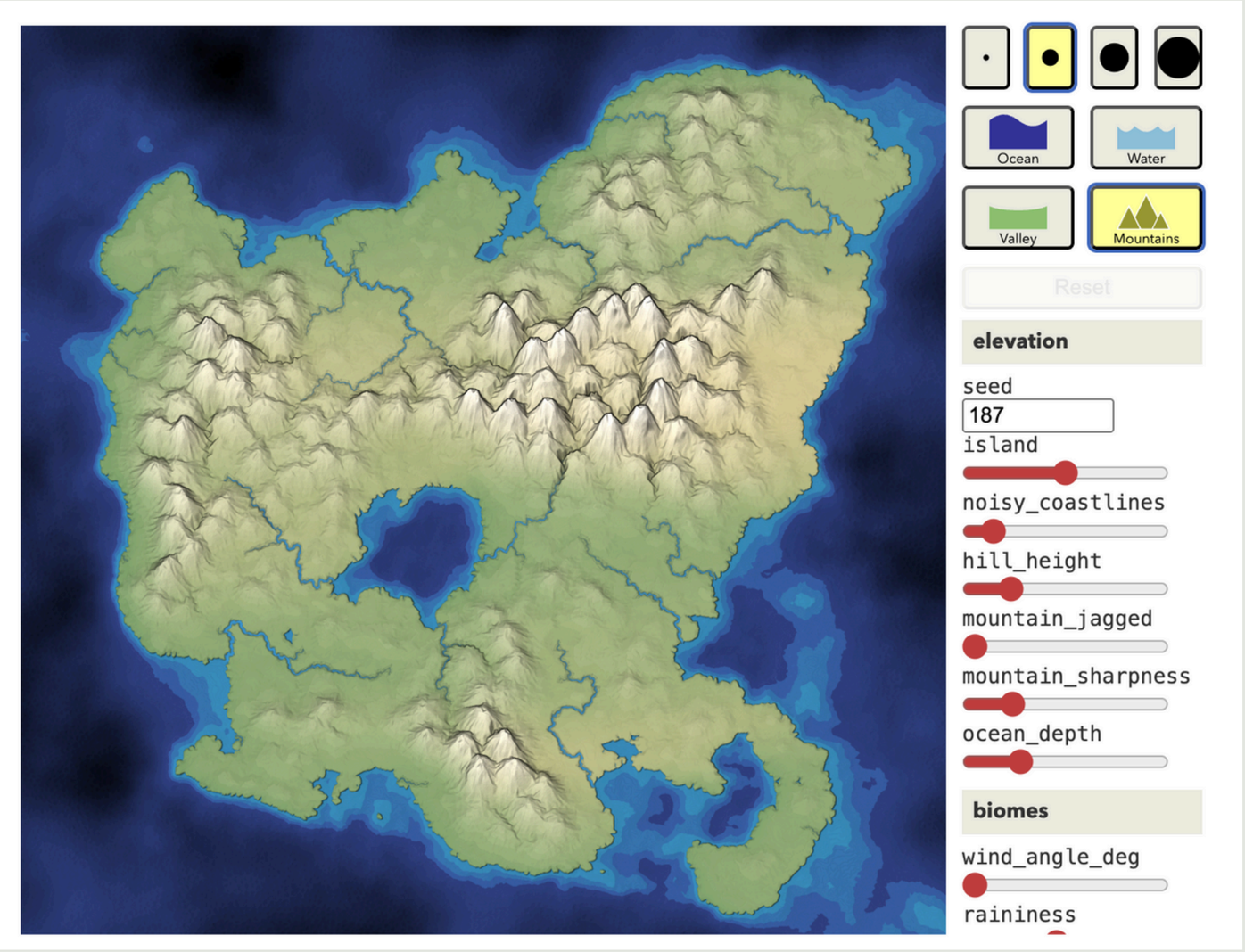
AZGAAR'S FANTASY MAP GENERATOR

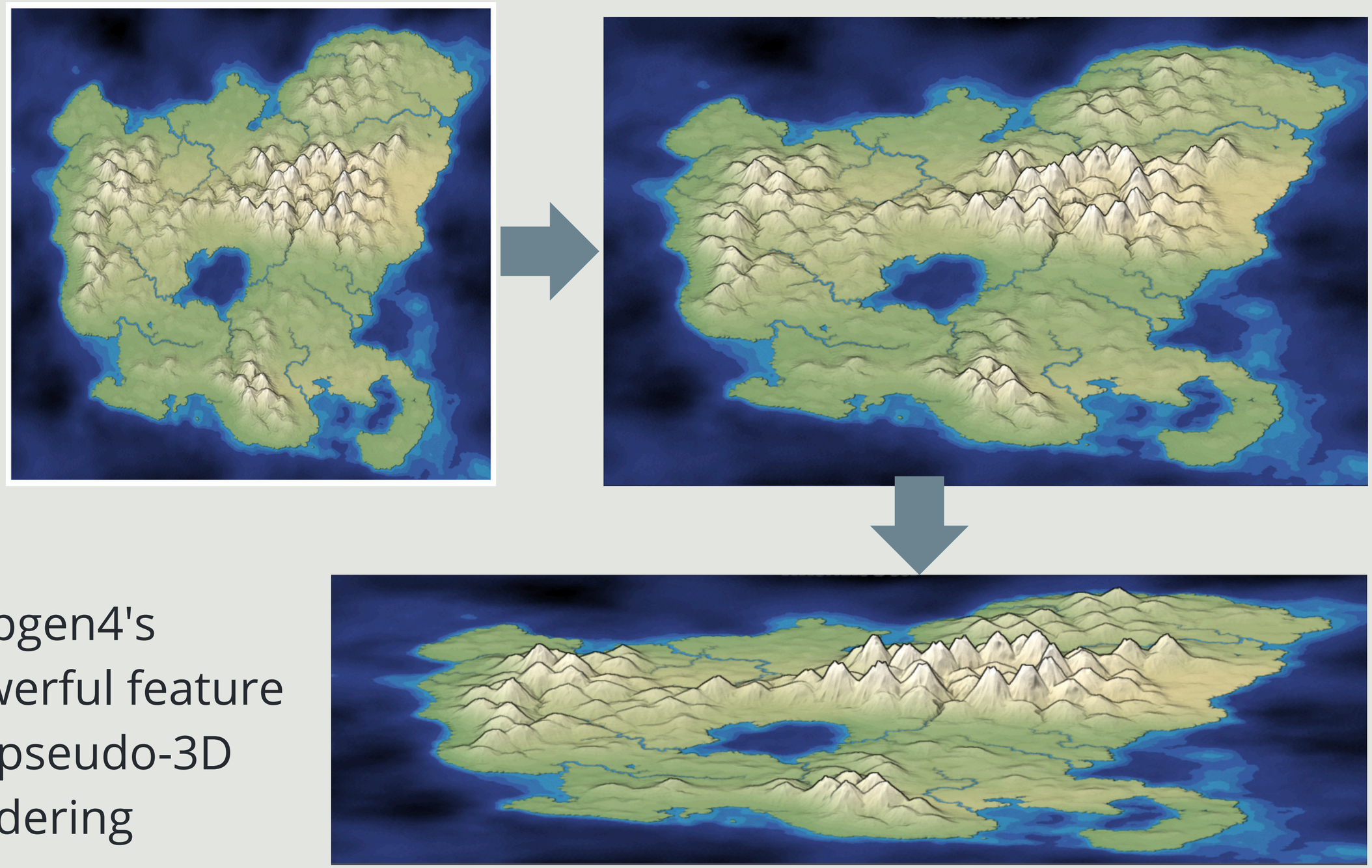
- open-source
- Automatic generation
- allow editing
- continent based
- 2D
- high randomness



MAPGEN4

- open-source
- Automatic generation
- allow editing
- terrain based
- 2.5D
- high randomness





- Mapgen4's powerful feature for pseudo-3D rendering

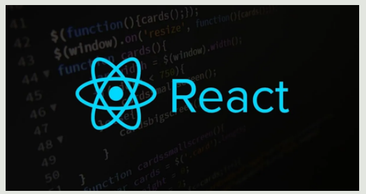
COMPARISON

	azgaar's map generator	Mapgen4	our work
Map control	Continent-level editing, 2D	Simple drawing interface, 2.5D	Hybrid approach: Natural language input + user editing, 2.5D
Customization	Political, cultural, climate	Editable elevation, Raininess, wind direction, etc.	Editable elevation, Raininess, wind direction, etc.
Use case focus	Worldbuilding & visualization	Terrain realism	Story-driven, fine-tuned map generation

Dataset/Platform

PLATFORM

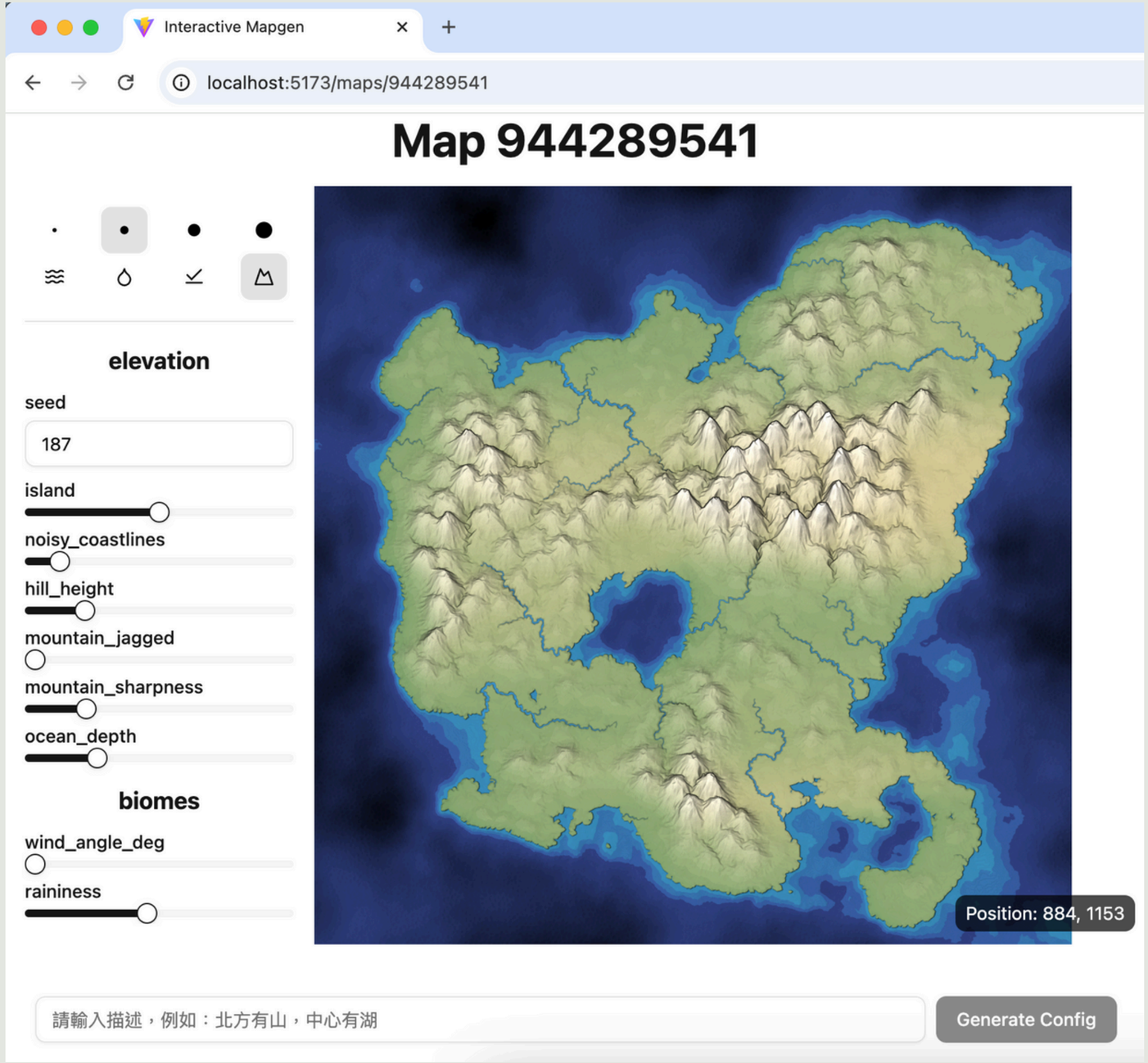
- xAI: grok-3-mini for api call
- mapgen4: designed by redblobgames
- react: modern framework for frontend



OUR WORK

- inherit all the features from mapgen4
- Integrate LLM
  - Sometimes we’re just too lazy to generate map manually
  - helps spark creativity by generating layout ideas users wouldn’t have thought of
- Modern framework--React
  - Built with React + TypeScript for extensibility
  - Compared to Mapgen4's raw HTML/Canvas-based approach, we gain modularity and UI flexibility

PLATFORM



running locally at port 5173

- position hint
- input field

DATASET

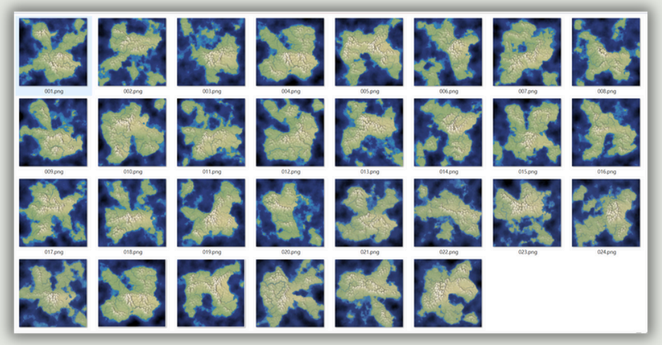
- We design our own dataset for testing/evaluation
- Our data includes:
  - user input
    - Ex: Mountains cover the entire west, while oceans form in the center and southeast
  - initial map (image)
  - generated result (image)
- Size of dataset: 30

DATASET

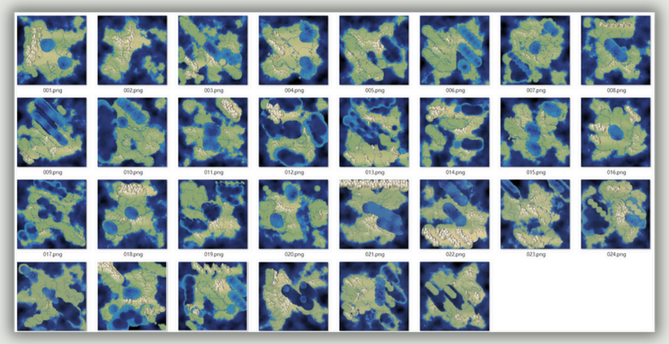
- Dataset design strategy (total 30 data)
  - 5 different terrain
    - ocean, lake, valley, hill, mountain
  - 9 different directions
    - N, E, S, W, center, NE, SE, SW, NW
  - Focused on balancing clarity and coverage — ensuring that every terrain type appears in multiple spatial contexts

DATASET

Original



Generated

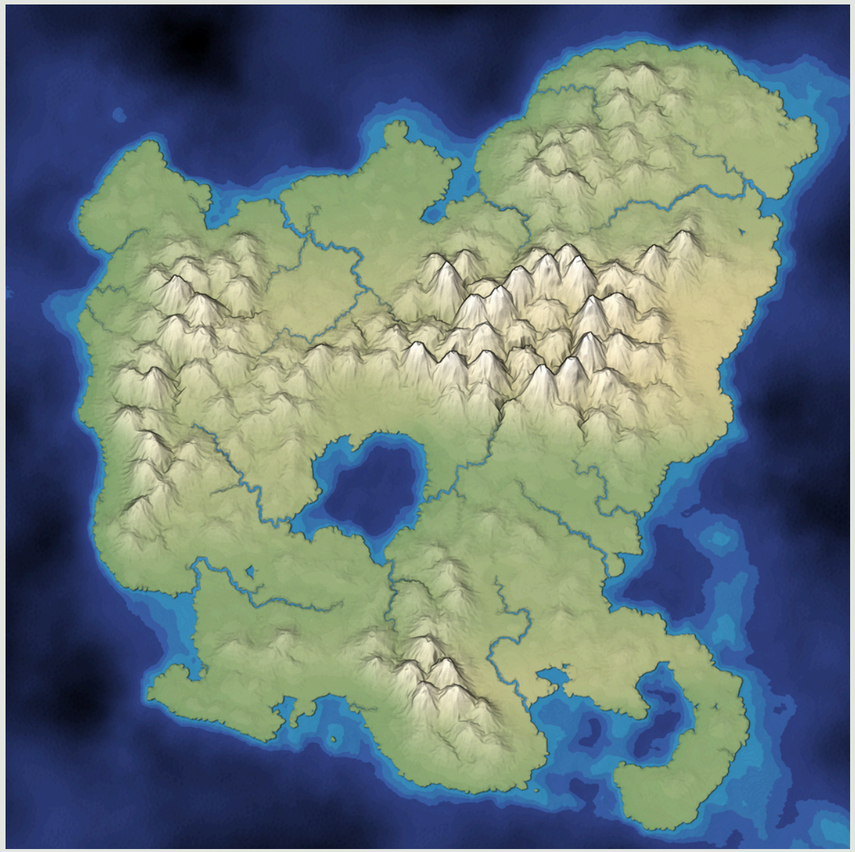


User input

Baseline

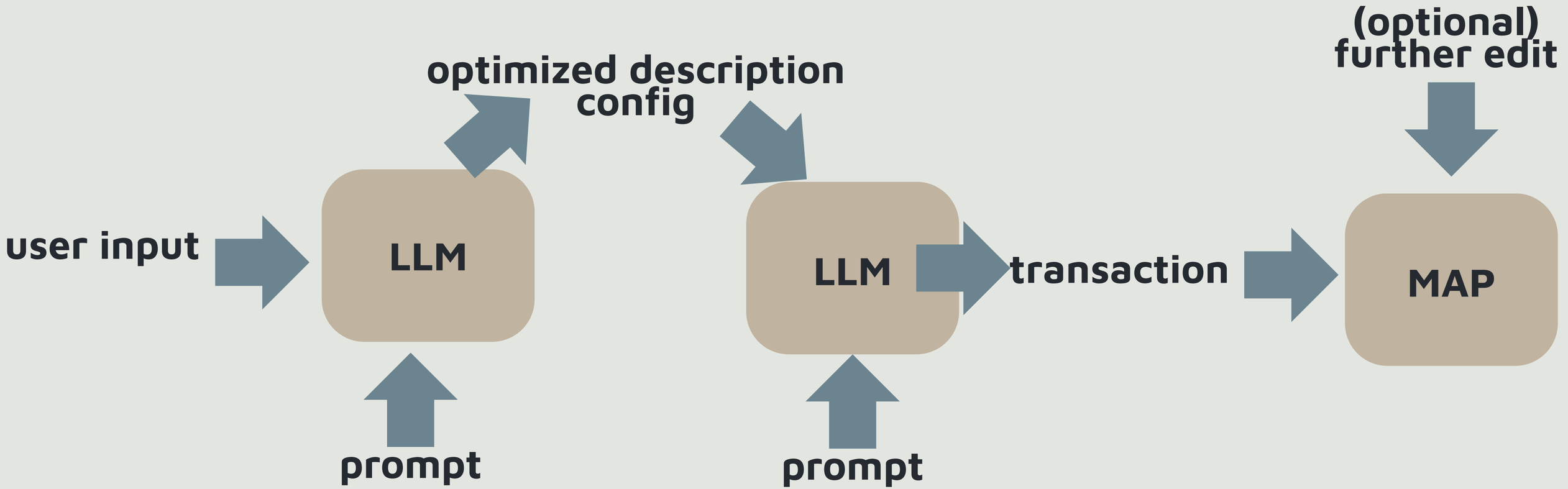
MAPGEN4

- We compare the default output of Mapgen4 with our prompt-generated map to highlight differences in layout precision

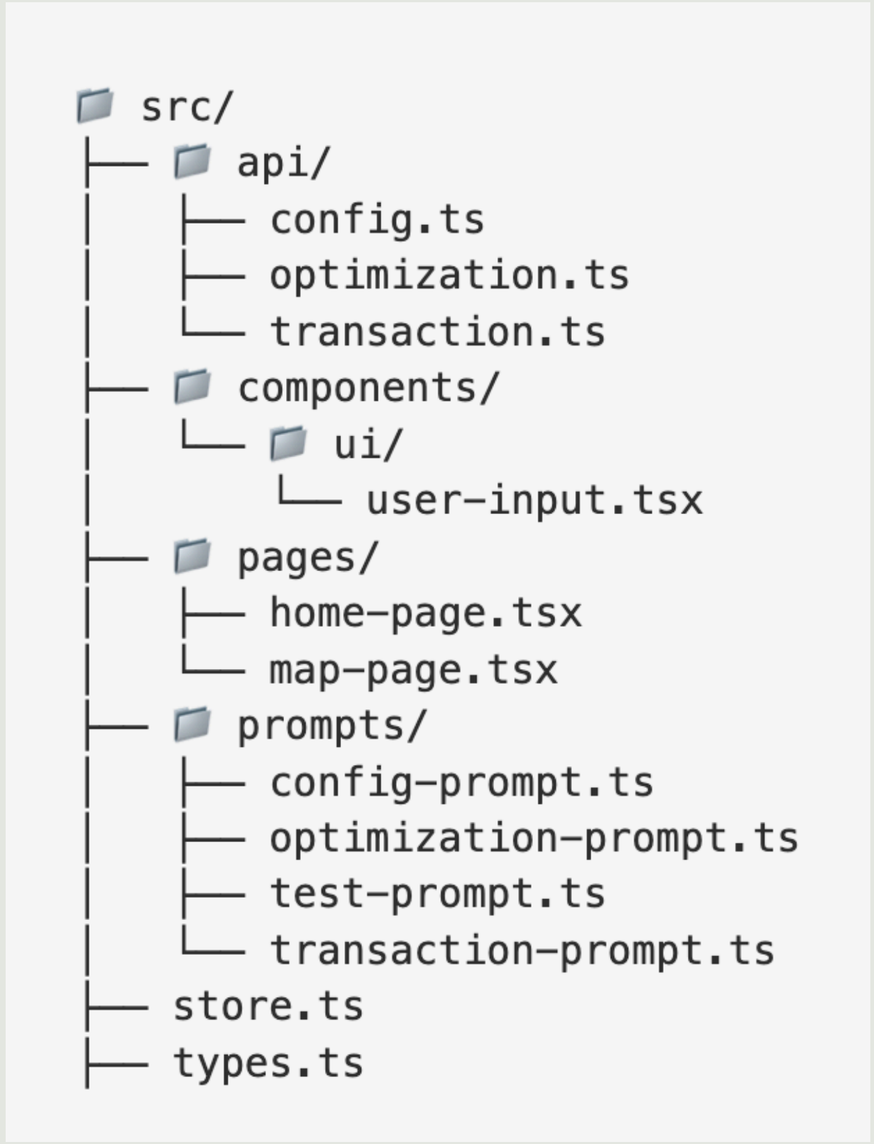


Main approach

METHODOLOGY



- Project structure



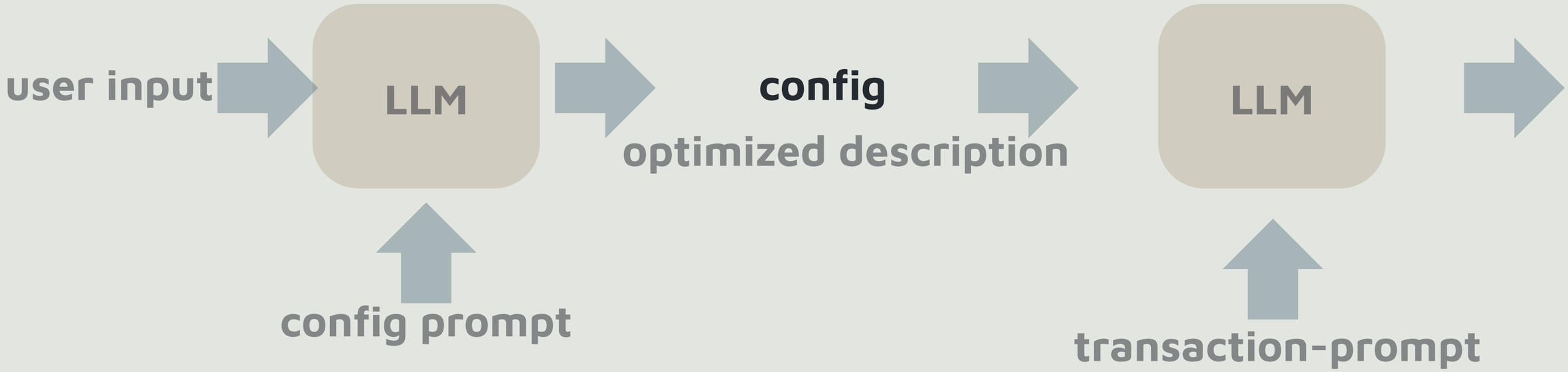
- api call
- input interface
- entry point/map page
- prompts
- global store/type definition

TYPE DEFINITION

- Config
- Transaction

```
export type TPoint = [number, number]
export type TScale = "xs" | "s" | "m" | "l" | "xl"
export type TBrush = "mountain" | "valley" | "water" | "ocean"
export type TTerrain = TBrush | "hill" | "lake"
export type TLocation = "east" | "west" | "north" | "south" | "northeast" | "northwest" |
"southeast" | "southwest" | "center"

export interface IConfig extends Record<TTerrain, TLocation[]> {
  selection: [TPoint, TPoint]
}

export interface ITransaction {
  type: TBrush
  position: TPoint
  scale: TScale
}
```

CONFIG

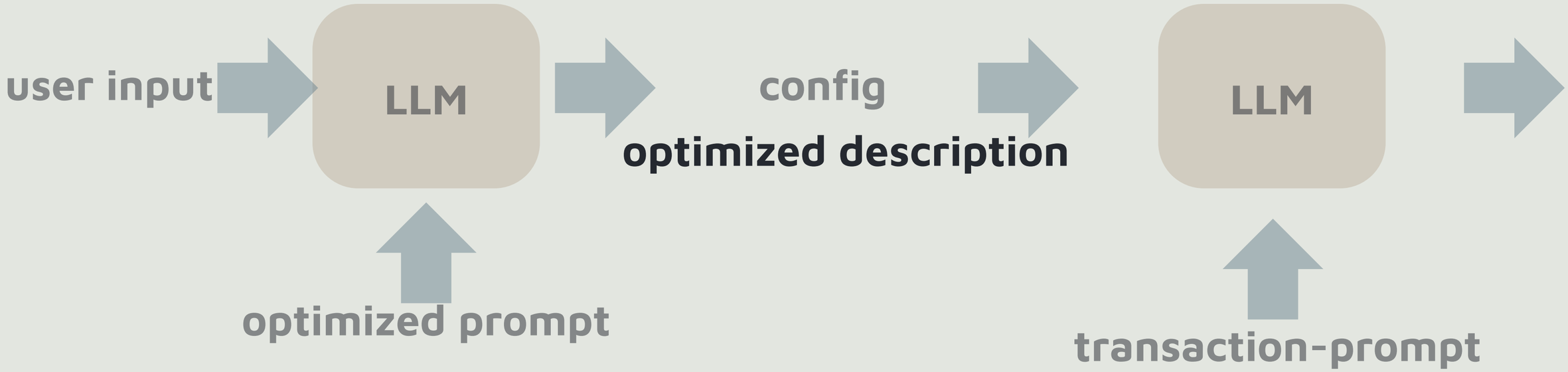

user input
LLM
config
optimized description
LLM
config prompt
transaction-prompt

CONFIG

- translates user inputs into clear, structured data
- use 8-direction + center for layout
- drawback?
  - lacking precision

```
"selection": [
  [0, 0],
  [2048, 2048]
],
"mountain": ["north", "northwest", "northeast"],
"hill": ["south"],
"valley": ["center"],
"lake": [],
"water": [],
"ocean": ["south"]
```

OPTIMIZED DESCRIPTION

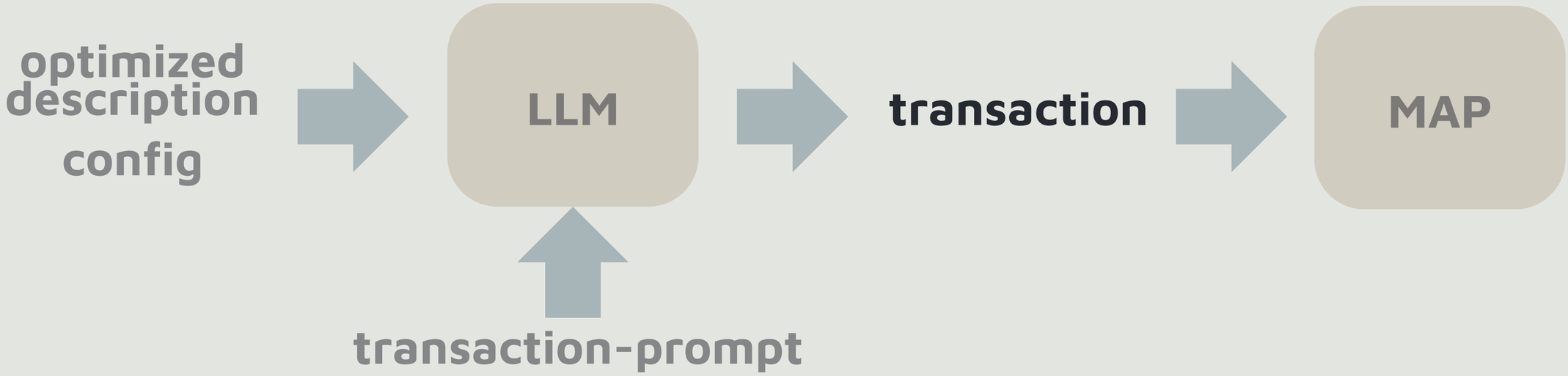


OPTIMIZED DESCRIPTION

- User input:
  - 有兩座島嶼互不相連，北方的島有山脈，南方的島有湖在中間
- Leverage LLM to infer **implicit terrain features**
  - Island separation → ocean
  - Lake → base plains first, then overlay lake region

```
Optimized instruction:  1. 北方島嶼基礎平原                    user-input.tsx:31
- 地形：valley
- 範圍：x ∈ [500, 1500], y ∈ [100, 600]
- 筆刷：m 搭配 s
- 說明：鋪設不規則形狀的平原作為北方島嶼基礎，偏向東側延伸以模擬自然地形起伏

2. 南方島嶼基礎平原
- 地形：valley
- 範圍：x ∈ [500, 1500], y ∈ [1400, 1900]
- 筆刷：m 搭配 s
- 說明：鋪設不規則形狀的平原作為南方島嶼基礎，偏向西側凹陷以確保自然過渡到海洋

3. 海洋分隔與周邊
- 地形：ocean
- 範圍：中間分隔 x ∈ [500, 1500], y ∈ [700, 1300]；並在兩島周邊補充，如 x
∈ [0, 500] 和 x ∈ [1500, 2000] 全 y，以及 y ∈ [0, 100] 和 y ∈ [1900,
2000] 全 x
- 筆刷：l 為主，邊緣使用 m（模擬波浪緩衝）
- 說明：使用不對稱曲線環繞兩島嶼，形成互不相聯的海洋，東側較寬以增加隨機性，並
在海洋與島嶼交界處補上 valley 作為過渡帶

4. 北方島嶼山脈
- 地形：mountain
- 範圍：x ∈ [700, 1300], y ∈ [200, 400]
- 筆刷：s 為主，邊緣使用 xs（模擬陡峭坡度減緩）
- 說明：在北方島嶼的基礎平原上添加高聳山脈，形狀偏向西北方向延伸，避免對稱，並
在其周圍補充 valley 作為自然緩衝

5. 南方島嶼中央湖泊
- 地形：water
- 範圍：x ∈ [900, 1100], y ∈ [1600, 1800]
- 筆刷：m 搭配 s
- 說明：位於南方島嶼的中央，繪製不規則湖泊形狀偏向東南方向，以增加隨機性，並在
湖泊周圍補充 valley 作為過渡帶
```

TRANSACTION



TRANSACTION

- An example of transaction
  - transaction consist of type, position, scale

```
{ "type":"valley","position":[800,1250],"scale":"xs" },
{ "type":"valley","position":[950,850],"scale":"s" },
{ "type":"mountain","position":[350,850],"scale":"s" },
{ "type":"mountain","position":[400,900],"scale":"s" },
```

- Where can the transaction be applied?
  - calling the Painting.paintAt function provided by mapgen4

```
transactions.forEach(({ type, position: [x, y], scale }) => {
  Painting.paintAt(brushes[type], x / 2048, y / 2048, scales[scale], 100)
})
```

PROMPT ENGINEERING

Rendering Techniques

```
### 繪製技巧：
- 越靠近地形中心，使用越大的筆刷。越靠近地形邊界，使用越小的筆刷。
- 接著分析文字描述，先畫water和ocean，然後再畫其他地形。
- ocean可以使用中等的筆刷(m、s)，其他地形如mountain、valley、water等則使用較小的筆刷(s, m)。
- 最外面的邊界地形應使用最小的筆刷(xs)，並且離邊界至少 300 像素。
- 要畫比較複雜的地形（如lake、hill），你應該盡量使用 **多筆** transaction 組合、較小的筆刷來模擬
- 在島中心以非均勻方式擺放 mountain，並引入微隨機擾動
- 每座山腳周圍產生 valley 作為地形緩衝過渡
- 中央適度嵌入一小片 water 模擬封閉湖泊
- 位置與比例都避免對稱與人工感
- **當需要在「周圍」分布地形時，必須覆蓋所有八個方向（東南西北及四個斜向），而非僅在四個主要方向**
```

Giving example

```
以下是幾個參考輸出範例：
範例 1（input: 「中央是湖，周圍是山」）：

1. 中央湖泊
- 地形：water
- 範圍：x ∈ [950, 1100], y ∈ [950, 1100]
- 筆刷：m 搭配 s
- 說明：使用不規則形狀排列，形成封閉湖泊，形狀偏向東南或西北方以避免完美對稱

2. 湖外過渡平原
- 地形：valley
- 範圍：環繞湖泊的360度區域，x ∈ [800, 1250], y ∈ [800, 1250]
- 筆刷：m 搭配 s
- 說明：全方位環繞湖泊形成平原帶，不只東西南北四個方向，務必在所有方位都有平原作為過渡，厚度不均勻
```

PROMPT ENGINEERING

Make it close to real world

```
### 其他注意事項：
- 輸出的位置要有隨機性，不要看起來太過工整（太規律）。
- 建立湖泊時，請確保其四周完全被陸地（如valley、hill）包圍，**不得直接接觸海洋或地圖邊界**。
- 丘陵必須置於陸地上，常見於平原與山地之間或內陸地區。請勿將丘陵單獨放置在海洋中或遠離其他陸地。
- 特殊類型：
  - **lake**：請使用一組 water transaction 表示，可能搭配 ocean，但務必被陸地包圍。
  - **hill**：表示為一組小型的 mountain、valley、water組合，模擬較溫和地勢起伏。
- 請模擬真實世界地形邏輯，避免呈現人工網格狀或不自然排列。地形過渡應合乎常理，例如：山→丘陵→平原。
- 請僅對 config 中有設定的地形項目進行生成，略過空陣列。
```

Strict Format

```
---
型別定義：
type TPoint = [number, number]
type TScale = xs, s, m, l, xl
type TBrush = mountain, valley, water, ocean

interface ITransaction {
  type: TBrush
  position: TPoint
  scale: TScale
}
---
```

Evaluation Metric

SEMANTIC ALIGNMENT EVALUATION

Region split

```python
def _define_spatial_regions(self) -> Dict[str, Tuple[slice, slice]]:
    h, w = self.image_size
    return {
        'north': (slice(0, h//3), slice(0, w)),
        'south': (slice(2*h//3, h), slice(0, w)),
        'east': (slice(0, h), slice(2*w//3, w)),
        'west': (slice(0, h), slice(0, w//3)),
        'center': (slice(h//3, 2*h//3), slice(w//3, 2*w//3)),
        'northeast': (slice(0, h//2), slice(w//2, w)),
        'northwest': (slice(0, h//2), slice(0, w//2)),
        'southeast': (slice(h//2, h), slice(w//2, w)),
        'southwest': (slice(h//2, h), slice(0, w//2)),
        'northern': (slice(0, h//2), slice(0, w)),
        'southern': (slice(h//2, h), slice(0, w)),
        'eastern': (slice(0, h), slice(w//2, w)),
        'western': (slice(0, h), slice(0, w//2)),
        'entire_west': (slice(0, h), slice(0, w//2)),
        'entire_east': (slice(0, h), slice(w//2, w)),
        'entire_north': (slice(0, h//2), slice(0, w)),
        'entire_south': (slice(h//2, h), slice(0, w)),
    }
```

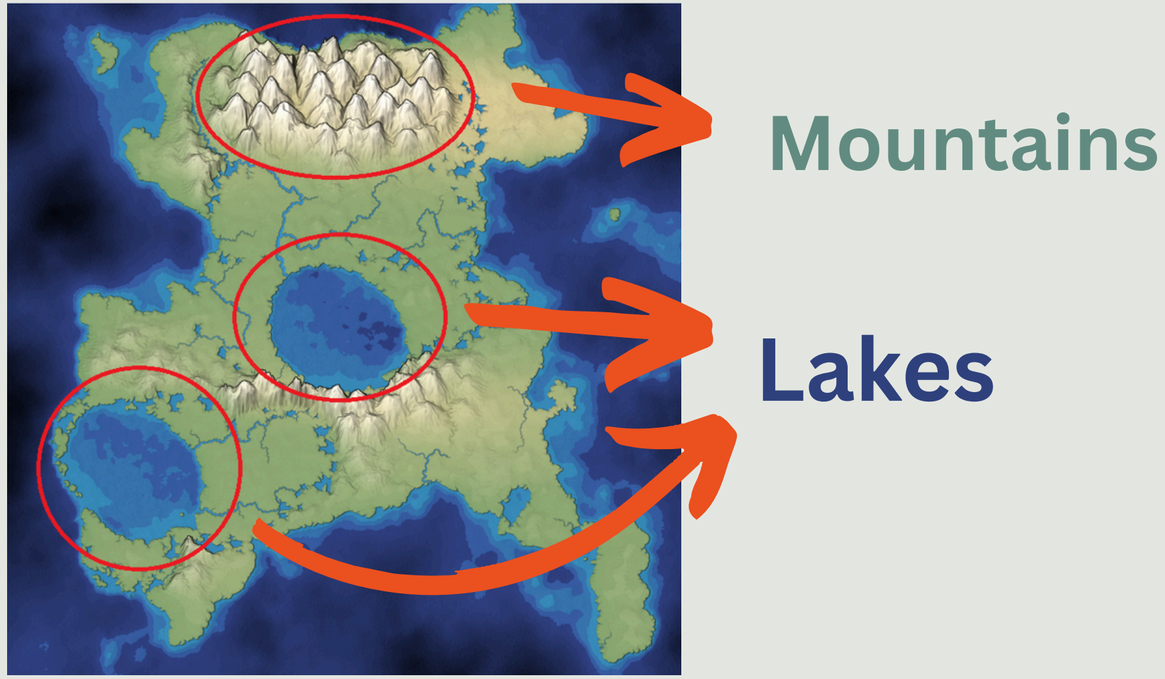
Text Parsing

```python
class TextParser:
    def __init__(self):
        self.direction_patterns = {
            'north': r'\b(north|northern|top|upper)\b',
            'south': r'\b(south|southern|bottom|lower)\b',
            'east': r'\b(east|eastern|right)\b',
            'west': r'\b(west|western|left)\b',
            'center': r'\b(center|central|middle)\b',
            'northeast': r'\b(northeast|north-east|upper right)\b',
            'northwest': r'\b(northwest|north-west|upper left)\b',
            'southeast': r'\b(southeast|south-east|lower right)\b',
            'southwest': r'\b(southwest|south-west|lower left)\b',
            'entire_west': r'\b(entire west|whole west|throughout west)\b',
            'entire_east': r'\b(entire east|whole east|throughout east)\b',
            'entire_north': r'\b(entire north|whole north|throughout north)\b',
            'entire_south': r'\b(entire south|whole south|throughout south)\b',
        }

        self.terrain_patterns = {
            'mountains': r'\b(mountain|mountains|mountainous|peak|peaks|hill|hills|highland|range)\b',
            'seas': r'\b(sea|seas|ocean|oceans)\b',
            'lakes': r'\b(lake|lakes|pond|ponds)\b',
            'rivers': r'\b(river|rivers|stream|streams|waterway|waterways)\b',
        }
```

SEMANTIC ALIGNMENT EVALUATION

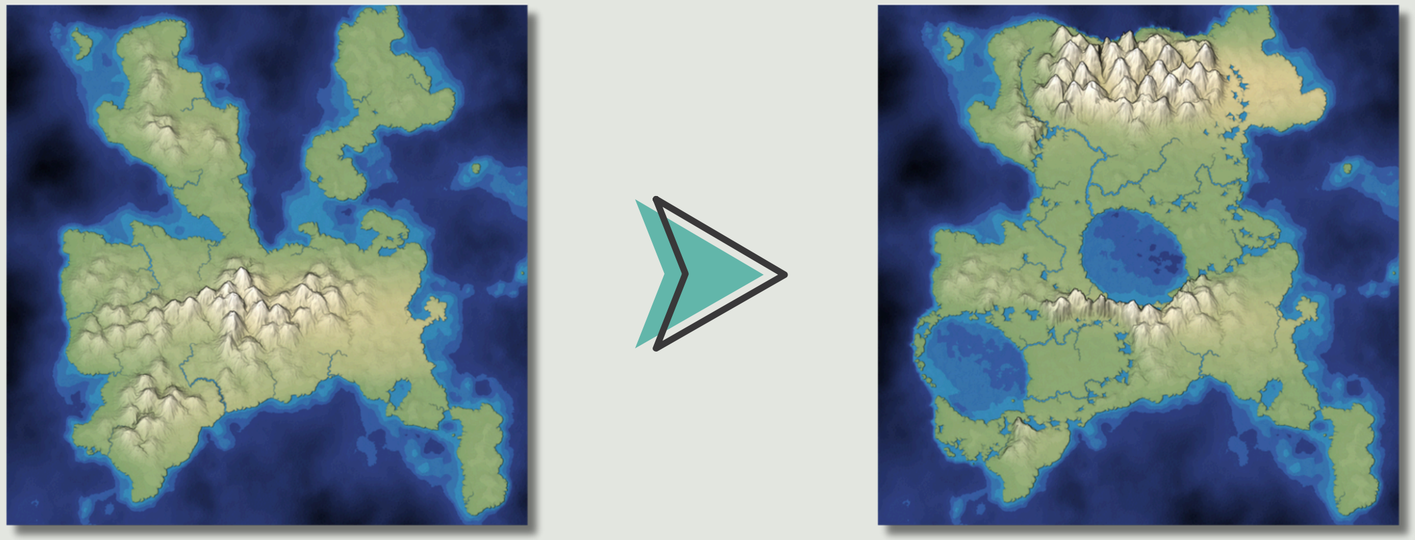
Terrain Detection

```python
def detect_mountains(image_region: np.ndarray) -> float:
    if len(image_region.shape) == 3:
        gray = cv2.cvtColor(image_region, cv2.COLOR_RGB2GRAY)
        hsv = cv2.cvtColor(image_region, cv2.COLOR_RGB2HSV)
        rgb = image_region
    else:
        gray = image_region
        hsv = None
        rgb = None
    white_peak_score = 0
    if rgb is not None:
        white_mask = ((rgb[:,:,0] >= 180) & (rgb[:,:,1] >= 180) & (rgb[:,:,2] >= 180))
        light_gray_mask = ((rgb[:,:,0] >= 150) & (rgb[:,:,1] >= 150) & (rgb[:,:,2] >= 150) &
                           (np.abs(rgb[:,:,0] - rgb[:,:,1]) <= 30) &
                           (np.abs(rgb[:,:,1] - rgb[:,:,2]) <= 30))

        peak_mask = white_mask | light_gray_mask
        white_peak_score = np.sum(peak_mask) / peak_mask.size

        if white_peak_score >= 0.15:
            white_peak_score = min(white_peak_score * 2.0, 1.0)

    edges = cv2.Canny(gray, 30, 120)
    edge_density = np.sum(edges > 0) / edges.size

    texture_score = np.std(gray) / 255.0
```



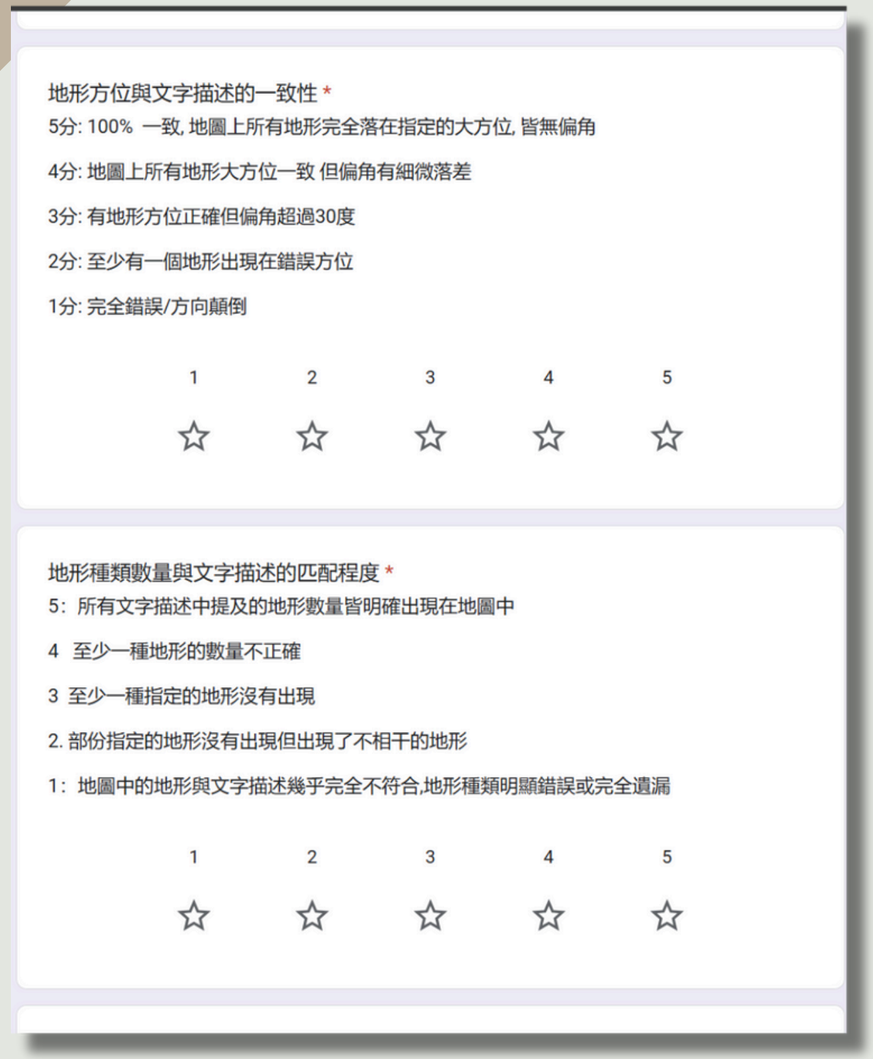
Generate lakes in the center and southwest; generate mountains in the north

SEMANTIC ALIGNMENT EVALUATION



```
ID: 018
描述: Generate lakes in the center and southwest; generate mountains in the north
生成圖片分數: 0.3105
原始圖片分數: 0.2485
較高者: generated
分數差距: 0.0620

生成圖片分析:
  lakes_center: 0.3133
  lakes_southwest: 0.4563
  mountains_north: 0.1621

原始圖片分析:
  lakes_center: 0.2791
  lakes_southwest: 0.4017
  mountains_north: 0.0648
```

USER STUDY



地形方位與文字描述的一致性 *

5分: 100% 一致, 地圖上所有地形完全落在指定的大方位, 皆無偏角

4分: 地圖上所有地形大方位一致 但偏角有細微落差

3分: 有地形方位正確但偏角超過30度

2分: 至少有一個地形出現在錯誤方位

1分: 完全錯誤/方向顛倒

1 2 3 4 5

☆ ☆ ☆ ☆ ☆

地形種類數量與文字描述的匹配程度 *

5: 所有文字描述中提及的地形數量皆明確出現在地圖中

4 至少一種地形的數量不正確

3 至少一種指定的地形沒有出現

2. 部份指定的地形沒有出現但出現了不相干的地形

1: 地圖中的地形與文字描述幾乎完全不符合,地形種類明顯錯誤或完全遺漏

1 2 3 4 5

☆ ☆ ☆ ☆ ☆

地形方位與文字描述	地形種類數量與文字	地形大小（深淺,高低）	地形高度分佈(梯度)	地形分佈與現實世界
4	3	5	4	4
4	3	4	3	4
4	4	3	4	3
3	4	4	4	3
5	3	4	3	3
5	4	4	5	4
4	4	3	5	4
4	4	3	4	3
4	5	4	4	3

Five Evaluation Dimensions

- Terrain Orientation Consistency
- Terrain Type and Quantity Matching
- Terrain Size Feature Matching
- Elevation Distribution Reasonableness
- Terrain Distribution Naturalness

Results&Analysis

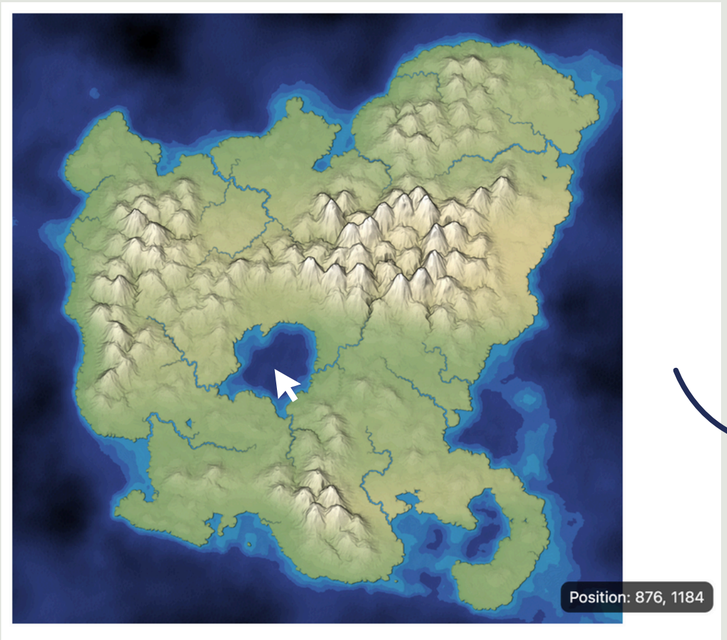
POSITION SPECIFICATION

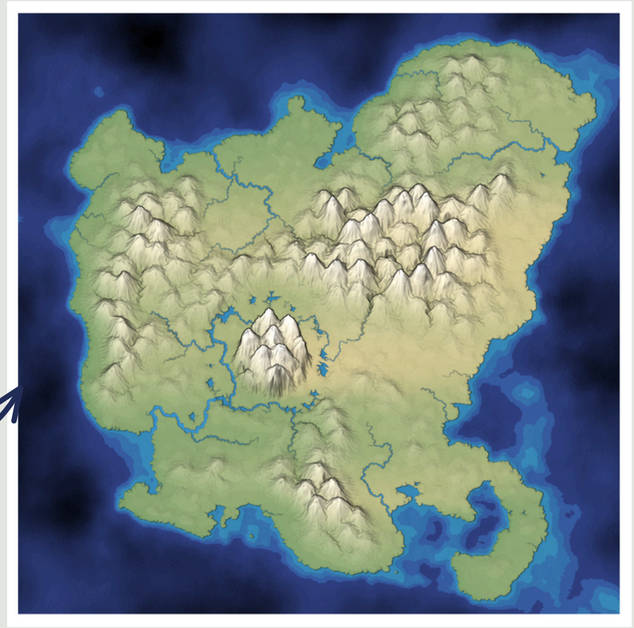
User input

```
User input:  generate mountains at the position [876, 1184]
```
user-input.tsx:26

Optimized description

```
Optimized instruction:  1. 山脈基礎平原
- 地形：valley
- 範圍：x ∈ [800, 950], y ∈ [1100, 1250]
- 筆刷：m 搭配 s
- 說明：以指定位置為中心擴展，營造平緩過渡區域，避免山脈過於突兀，形狀略為不規則

2. 山脈
- 地形：mountain
- 範圍：x ∈ [850, 900], y ∈ [1160, 1200]
- 筆刷：s 為主，邊緣使用 xs
- 說明：集中在指定位置 [876, 1184] 附近，形成自然隨機的山峰群，模擬真實地形起伏以增強視覺多樣性
```
user-input.tsx:31





THE LORD OF THE MAPS

An example story :

*In the western land of the **Shire**, the grey-robed wizard Gandalf discovered a long-lost artifact in the hobbit village of Bag End — the One Ring, forged by Sauron to dominate all other Rings of Power. Gandalf knew that only by casting it into the fires of Mount Doom in Mordor could its evil be destroyed forever.*

*Thus began his perilous journey to Mordor. Along the way, near the ruins of **Minas Tirith**, he encountered a young man named Aerys, a descendant of the fallen bloodline of the Dúnedain. Eager to reclaim his family's lost honor, Aerys joined Gandalf on the quest.*

Together they crossed the Misty Mountains, evading the patrols of wolves and orcs through snow-covered peaks and narrow passes. They entered the Elven realm of Lothlórien, where Queen Galadriel offered them rest and bestowed upon them a gift — a radiant Elven blade named Silverstar, capable of piercing the shadows.

*Their path next led through the **Dead Marshes**, haunted by ghostly apparitions and the spirits of the fallen. Gandalf, overcome by illusions, nearly succumbed to the marsh's grip, but Aerys courageously pulled him back from the brink.*

After traversing Ithilien and the desolate plains of Morannon, they finally reached Mount Doom beneath the shadow of Barad-dûr. At the fiery summit, the lingering will of Sauron appeared as a dark phantom, tempting Aerys to claim the Ring and restore his kingdom with its power. Struggling within, Aerys ultimately rejected the temptation and severed the connection to the darkness, allowing Gandalf to cast the Ring into the molten core.

*With the Ring destroyed, Barad-dûr collapsed, Sauron's power was broken, and the dark clouds over **Mordor** were swept away. Middle-earth was at last free.*

PROCEDURE

first input txt

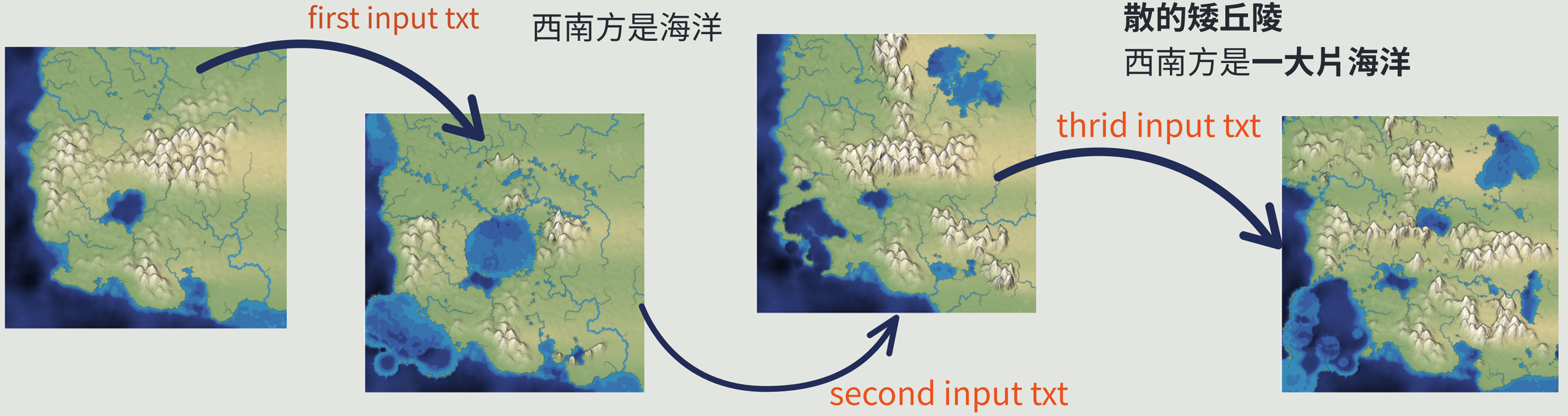
東北方是平原且**中心有一個大湖泊**
東南方是**高山**圍成一圈
中部的高山由**西部延伸到東部**
北方的高山**延伸到中部**
西北方是平原
西南方是海洋

second input txt

東北方是平原中心有一個**大湖泊和破碎的小平原**
東南方是**很多高山**圍成一圈
中部是一條**東西向**的高山
北方是一條**南北向**的高山
西北方是有平原和**丘陵**
西南方是海洋

third input txt

東北方是平原中心有一個大湖泊和破碎的小平原
東南方是多座高山圍成一圈
中部是一條很長東西向的高山
北方是是一條南北向的高山
西北方是有大面積平原和**50座分散的矮丘陵**
西南方是**一大片海洋**



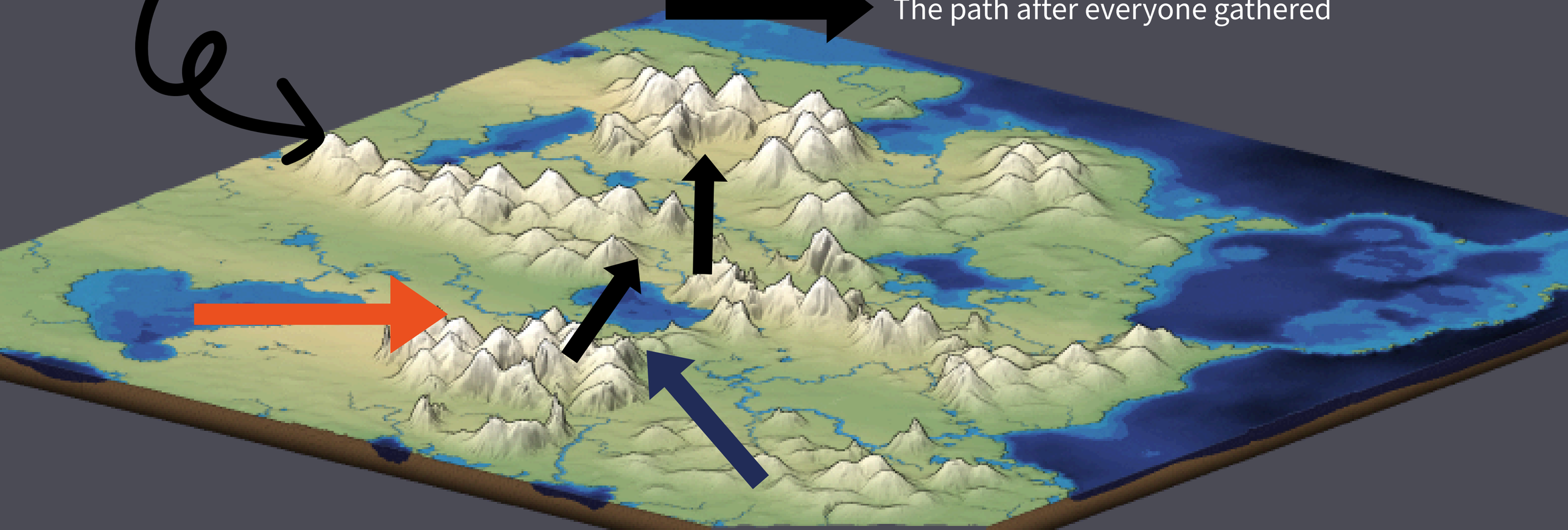
THE LORD OF THE MAPS

Map generated based on the story

The path taken by the Aerys

The path taken by Gandalf and the Hobbits

The path after everyone gathered



THE LORD OF THE MAPS

Minas Tirith

Dead Marshes

Mordor

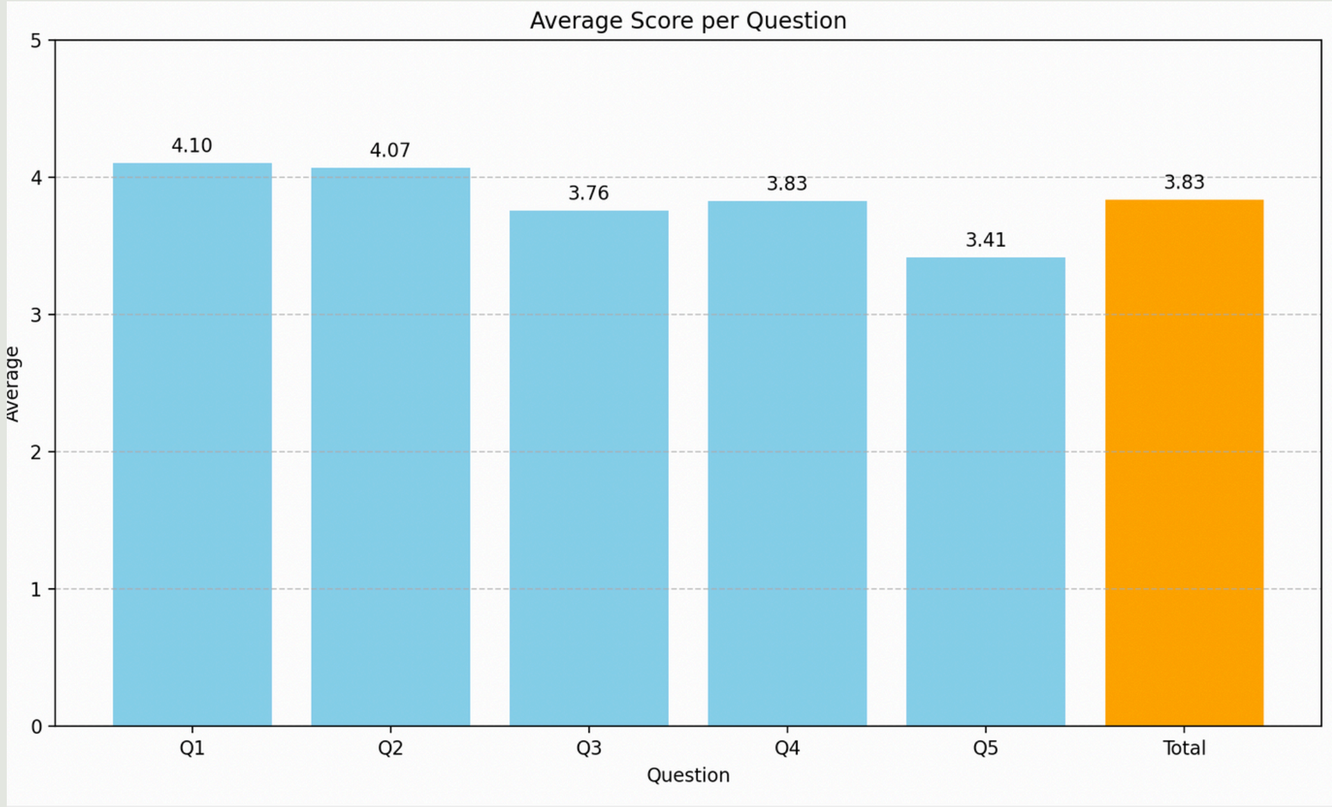
The far-left image shows Minas Tirith, a mountainside city. Next is the Dead Marshes, represented by a lake and surrounding peaks. The upper right shows Mordor, encircled by mountains. The lower right is the Shire, depicted with hills to reflect the Hobbits' home.

Shire

ANALYSIS--USER STUDY

- we designed 5 questions for user study, 30 participant in total

- **Terrain Orientation Consistency**
- **Terrain Type and Quantity Matching**
- **Terrain Size Feature Matching**
- **Elevation Distribution Reasonableness**
- **Terrain Distribution Naturalness**



ANALYSIS--USER STUDY

The strongest performance was observed in Q1 and Q2, indicating users found high alignment in position and quantity matching.

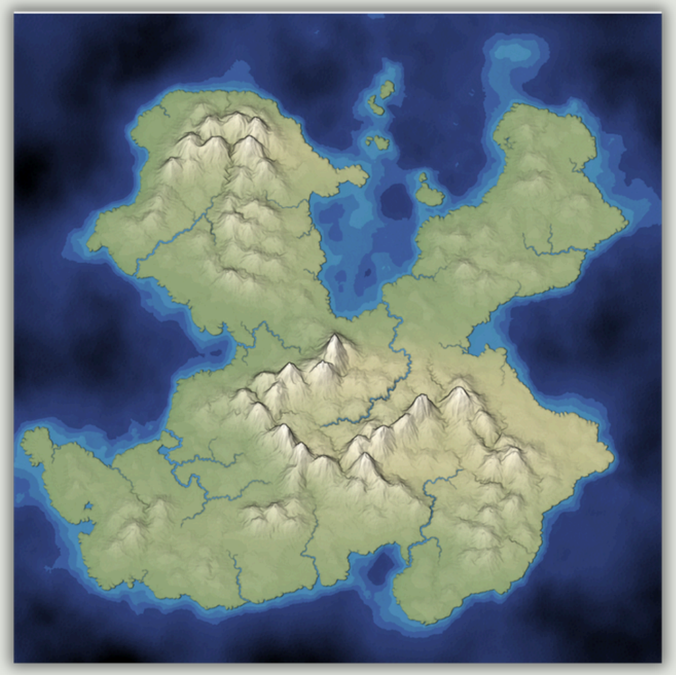
However, Q5 had a clear low average (3.41), indicating a challenge in mapping terrain distribution to real-world expectations.
This suggests the need for future refinements in simulating realistic geospatial distributions.
Further, a deeper exploration via qualitative feedback or targeted user interviews is advised to enhance interpretation fidelity and design accuracy.

ANALYSIS--SEMANTIC ALIGNMENT

```
總數量: 30
生成圖片較高: 27 (90.0%)
原始圖片較高: 3 (10.0%)
平均分數 - 生成: 0.3566
平均分數 - 原始: 0.2723
平均分數差距: 0.0863
詳細結果已保存至: geographic_evaluation_results.txt
```

Improve rate:90%

27/30



ID: 021
描述: The entire north is covered by mountains, while lakes form in the southeast and center
生成圖片分數: 0.2427
原始圖片分數: 0.0958
較高者: generated
分數差距: 0.1469

生成圖片分析:
  mountains_north: 0.1144
  lakes_center: 0.3710

原始圖片分析:
  mountains_north: 0.0086
  lakes_center: 0.1831

ID: 027
描述: A lake is located in the eastern part of the map, mountains are in the north, hills cover the southwest, a valley is in the center, and the west is ocean.
生成圖片分數: 0.2373
原始圖片分數: 0.2581
較高者: reference
分數差距: 0.0208

生成圖片分析:
  lakes_east: 0.2213
  mountains_north: 0.1611
  mountains_southwest: 0.0061
  seas_west: 0.5607

原始圖片分析:
  lakes_east: 0.3468
  mountains_north: 0.1351
  mountains_southwest: 0.0017
  seas_west: 0.5489

Incorrectly identified the sea as a lake!!

Future work

FINE-TUNING

- Leverage Reinforcement Learning (RL) to fine-tune the LLM.
- Having computed semantic alignment scores, we can utilize this data to adapt the model toward generating more contextually aligned and semantically accurate responses.

ADVANCED USER INTERFACE

- Selection area
  - Instead of typing the position of area that users want to modify, provide an interface that user can direct select, and automatically send this information to LLM

GITHUB

- github link
  - https://github.com/hansliao2015/AI-final-project

REFERENCE

- procedure map generation
  - https://github.com/redblobgames/mapgen4
  - https://azgaar.github.io/Fantasy-Map-Generator/
- prompt engineering
  - https://medium.com/@cch.chichieh/llm-%E5%90%84%E7%A8%AE%E6%8A%80%E5%B7%A7-prompt-engineering-%E6%8C%87%E5%8D%97-6ac4201a4cbe

CONTRIBUTION

- 112550087 廖漢軒：50%
  - coding (frontend, api, prompt engineering), evaluation metric, report
- 112550073 陳柏宗：20%
  - dataset generation, evaluation metric, report
- 112550103 游翔宇：20%
  - dataset generation, evaluation metric, report
- 613001013 駱巍文：10%
  - evaluation metric

Thank You

For your attention