

Inżynieria Oprogramowania

Projekt

Symulator GOPR

Wykonali:

Bartłomiej Gwóźdź

Bartłomiej Dziedzic

Mikołaj Filipski

Dawid Filipak

Opis:

Celem projektu jest stworzenie Symulatora ruchu turystów, zwierząt i pogody na potrzeby systemu wsparcia GOPR

Projekt polega na stworzeniu symulatora środowiskowego, którego celem jest generowanie realistycznych, dynamicznych danych dotyczących ruchu turystów, migracji zwierząt oraz zmian warunków pogodowych w górskim terenie. Symulator został opracowany z myślą o wsparciu testów systemu support gopr.

Główne cele projektu:

- **Dostarczanie danych testowych** symulujących warunki panujące w górach, w tym ruch turystyczny, pojawianie się dzikich zwierząt oraz zmienne warunki atmosferyczne.
- **Testowania algorytmów** w systemie gopr support, w tym symulacja sytuacji kryzysowych, nagłych załamań pogody.
- **Zwiększenie realizmu testów systemu GOPR**, bez konieczności przeprowadzania ich w warunkach rzeczywistych.

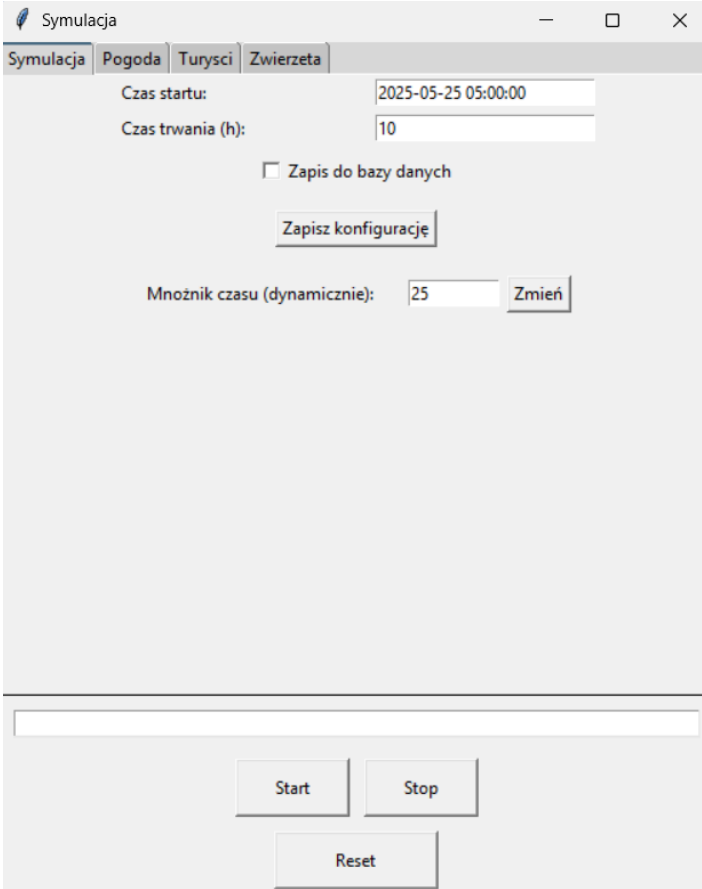
Zakres funkcjonalny symulatora:

- **Ruch turystów:** Modelowanie szlaków turystycznych, zmiennych natężeń ruchu w zależności od pory dnia, pogody i sezonu.
- **Ruch zwierząt:** Symulacja obecności i przemieszczania się dzikich zwierząt w różnych obszarach, z uwzględnieniem przechodzenia przez górskie szlaki.
- **Warunki pogodowe:** Generowanie dynamicznej pogody, zmieniającej się w czasie i przestrzeni, na podstawie danych wprowadzanych przez użytkownika.

Technologia:

Symulator został zaimplementowany z wykorzystaniem języka Python. Wyniki mogą być eksportowane w formacie zgodnym z wymaganiami systemu support (JSON), a także zapisywane do bazy danych.

Opis GUI i Funkcjonalności:



The screenshot shows a software window titled "Symulacja". It features a tabbed interface with four tabs: "Symulacja", "Pogoda", "Turysci", and "Zwierzeta". The "Symulacja" tab is currently selected. Within this tab, there are two text input fields: "Czas startu:" containing the date and time "2025-05-25 05:00:00", and "Czas trwania (h):" containing the number "10". Below these fields is an unchecked checkbox labeled "Zapis do bazy danych". Underneath the checkbox is a button labeled "Zapisz konfigurację". At the bottom of the main content area, there is a text input field for "Mnożnik czasu (dynamicznie):" with the value "25", and a button labeled "Zmień" to its right. At the very bottom of the window, there are three buttons: "Start", "Stop", and "Reset".

Symulator ma możliwość ustawienia daty rozpoczęcia symulacji, czasu trwania symulacji w godzinach.

Posiada opcję włączenia zapisu generowanych danych do bazy danych (konfiguracja połączenia z bazą w pliku DBConnector.py), a także zmiany prędkości generacji za pomocą mnożnika (podstawowo 10 sekund).

Przycisk Start uruchamia symulację, przycisk stop wstrzymuje generowanie danych, reset powoduje wyczyszczenie ustawień.

Symulacja

Symulacja Pogoda Turysci Zwierzeta

Minuta:

Detektory (np. 1,2,3):

Temperatura:

Wiatr:

Mgła:

Deszcz:

Min: 1 | Detektor: 1,2,3,4 | Temperatura: 20.0 | Wiatr: 5.0 | Mgła: 0.0 | Deszcz: 0.0

Min: 23 | Detektor: 2 | Temperatura: -10.0 | Wiatr: 90.0 | Mgła: 9.0 | Deszcz: 95.0

Generowanie pogody odbywa się za pomocą zdarzeń ustawianych przez użytkownika. Zdarzenia posiadają informacje o pogodzie, na których detektorach ma zostać ona zarejestrowana, oraz w której minucie symulacji. Przyciski “Ładna pogoda” i ”Brzydka pogoda” pozwalają na szybkie wstawienie przykładowych wartości.

Symulacja

—

□

×

Symulacja

Pogoda

Turysci

Zwierzęta

Natężenie na ścieżkach

ID: 1

Zapisz

ID: 5

Zapisz

ID: 7

Zapisz

Odśwież listę ścieżek

Start

Stop

Reset

W zakładce turyści mamy możliwość ustawienia częstotliwości wchodzenia turystów na ścieżkę (0 - brak możliwości pojawienia się, 100 - turysta pojawia się co każdą iterację)

Symulacja

Symulacja

Pogoda

Turyści

Zwierzęta

Typ	Start Long.	Start Lat.	Trasa(przez którą zwierze ma przejść)	Akcje
jelen	300	300	0	Usuń
jelen	300	320	0	Usuń
niedzwiedz	400	700	1	Usuń
lis			0	Usuń
lis			0	Usuń

Dodaj zwierzę

Zapisz zwierzęta

Start

Stop

Reset

W przypadku zwierząt mamy możliwość określenia ich rodzaju, miejsca startowego, oraz opcjonalnie ścieżki przez którą mają przejść

Dane wejściowe symulatora:

Symulator korzysta z danych zapisanych w pliku `map_sample.json`, który zawiera wszystkie niezbędne informacje do przeprowadzenia symulacji środowiska górskiego. Plik ten pełni rolę mapy konfiguracyjnej i jest zgodny ze strukturą określoną w dokumentacji projektu **SUPPORT GOPR**.

Struktura danych zawartych w `map_sample.json`:

- **Routes** – informacje o szlakach turystycznych, ich przebiegu oraz wejściach; stanowią podstawę do modelowania ruchu turystów.
- **Detektory** – lokalizacje punktów pomiarowych pogody umożliwiające ich identyfikację.
- **Stacje BTS** – lokalizacje stacji bazowych sieci komórkowej.
- **Miejsca specjalne** – wybrane lokalizacje o szczególnym znaczeniu, takie jak punkty widokowe, zagrożenia terenowe czy obszary ochrony przyrody. Wpływają na symulację turystów.
- **Wymiary mapy** – informacje o rozmiarze i granicach obszaru objętego symulacją, niezbędne do poprawnego pozycjonowania obiektów.

Pliki json używane w symulacji:

`config.json` - zapisane są w nim ustawienia symulatora

`animal_locations.json` - przechowuje informacje o zwierzętach

`tourist_location.json` - przechowuje informacje o turystach

`weather_events.json` - zapisuje wszystkie ustawienia pogody użytkownika

`weather_station.json` - zapisuje aktualne informacje o detektorach i pogodzie

`map_sample.json` - z tego pliku wczytywane są podstawowe informacje konfigurujące mapę

Pliki projektu:

main.py - główny plik programu, uruchamia całą aplikację i gui

simulation.py - plik odpowiedzialny za logikę symulacji oraz przycisków gui

DBConnector.py - plik konfiguracji bazy danych

simulationdb.py - zawiera zapytania do bazy danych

routes.py - służy do generowania pliku map_sample.json

animals.py - posiada całą logikę zwierząt

tourists.py - posiada całą logikę ruchu turystów

weather.py - posiada logikę generowania pogody

weather_events.py - obsługa zdarzeń pogody

wyświetlacz.py - generuje prostą mapę (map.html) służącą do wizualnego sprawdzenia poprawności działania symulacji

Działanie symulacji:

Symulacja wykonuje okresowo pętle (domyślnie co 10 sekund) które można przyspieszyć mnożnikiem, w trakcie każdej pętli wykonywana jest logika symulacji pozwalająca np. Na pojawienie się nowego turysty na wejściu czy ruch innych.

Generowanie turysty

W każdej iteracji pętli run, dla każdego wejścia (entrances) sprawdzana jest szansa (spawn_chance).

Jeśli warunek zostanie spełniony:

- Tworzony jest nowy turysta (Tourist), przypisany do danej trasy.
- Otrzymuje losowy numer telefonu (+48...).
- Jego początkowe współrzędne to pierwszy punkt trasy wejścia.
- Jeśli zapis do bazy jest aktywny (save_to_db), dane turysty są wstawiane do bazy.

Symulowanie ruchu turysty

Turysta posiada:

- aktualną pozycję (last_location),
- indeks aktualnego punktu trasy (current_point_index),
- kierunek poruszania się (przód / tył),
- prędkość (base_speed / current_speed),
- flagi stanu (czy się porusza, czy jest zgubiony itp.).

Klasa ActorsLocationSimulator wywołuje metody start_updating_tourist_locations, które aktualizują pozycję turysty.

Losowe zdarzenia wpływające na turystę:

Zgubienie drogi (should_get_lost_or_find_way):

- 1/500 szansy na zgubienie się,
- Jeśli już jest zgubiony, 1/100 szansy na odnalezienie trasy.
- Kontuzje (should_get_injured):
- 1/1000 szansy na nagle zatrzymanie się (turysta przestaje się poruszać).

Zatrzymanie przy specjalnych miejscach (check_special_places):

Jeśli turysta znajdzie się w promieniu specjalnego miejsca – 1/10 szansy na zatrzymanie się.

1/10 szansy na wznowienie ruchu, jeśli wcześniej się zatrzymał.

Zmiana prędkości (update_speed):

10% szansy co iterację na lekką fluktuację prędkości w zakresie ± 0.1 .

Dla każdego turysty tworzony jest wpis z aktualną lokalizacją (create_location).

Zapisywana jest w pliku tourist_location.json.

Opis systemu pogody

Dane pogodowe są przechowywane w pliku `weather_events.json`. Każde zdarzenie pogodowe (tzw. "weather event") zawiera:

- `minute` – moment wystąpienia (liczba minut od startu),
- `temperature, wind, fog, rain` – dane pogodowe,
- `detectors` – lista detektorów, których dotyczy pogoda (z współrzędnymi),
- `added` – znacznik czasu, kiedy zdarzenie zostało dodane.

Dodawanie zdarzenia pogodowego

Funkcja (`add_weather_event`) dodaje zdarzenie pogodowe na podstawie podanych detektorów. Każdy detektor ID musi istnieć w pliku mapy (`map_sample.json`), inaczej zdarzenie nie zostanie zapisane.

Pobieranie pogody dla detektora

Znajduje zdarzenia pogodowe związane z danym detektorem. Następnie jeśli są dwa wydarzenia (przed i po danej minucie) interpoluje między nimi (`smooth_weather_transition`). Jeśli jest tylko jedno wcześniejsze/późniejsze używa go z losową fluktuacją (`apply_variation`). Jeśli brak danych: zwraca wartości domyślne.

Interpolacja pogody

`smooth_weather_transition` płynnie wylicza wartości pogodowe między dwoma zdarzeniami:

$$\text{value} = a + (b - a) * \text{ratio}$$

Wartości są dodatkowo lekko zaburzane ($\pm 3\%$) przez `apply_variation`, by uniknąć nadmiernej sztywności.

generowanie tras zwierząt

z gui są pobierane dane zwierząt i na jej podstawie są generowane trasy przez które będą przechodzić zwierzęta. Trasa jest generowana jako listę punktów z znacznikami czasu, zaczynając od ustalonej bazowej lokacji, lub losowej jeżeli pusta. punkty są generowane w zasięgu 20 px od poprzedniego. Jeżeli została wybrana ścieżka przez które zwierzę ma przejść, zwierze będzie zmierzało w stronę ścieżki i przez pewien czas generować punkty wzdłuż jej.

symulowanie ruchu zwierząt

zwierzę w czasie symulacji będzie podążać za punktami swojej trasy , a jeżeli czas symulacji jest między 2 punktami, lokacja zwierzęcia jest obliczana przez interpolację liniową.