

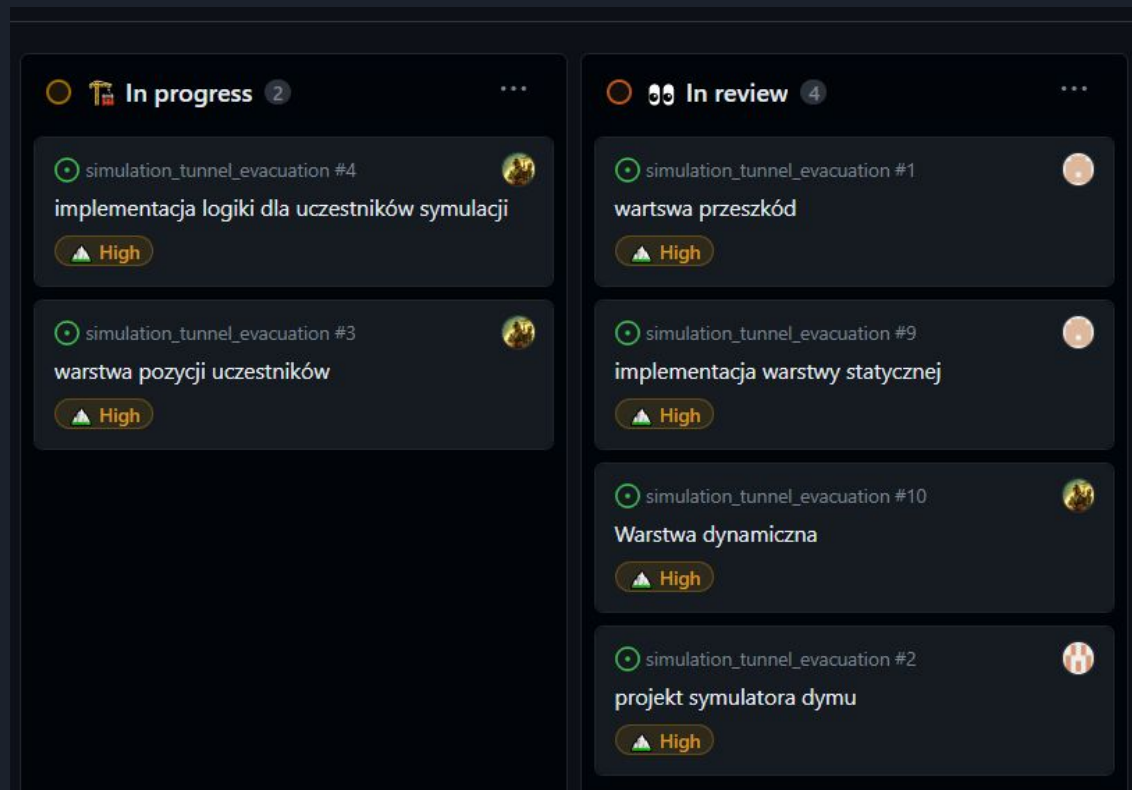
Symulacja ewakuacji z tunelu drogowego



Karol Błaszczak
Łukasz Chmielewski
Jakub Banach

Postęp prac

- warstwa dynamiczna
- warstwa pozycji
- model FDS





Projekt symulatora dymu i pożaru

Stworzony w oparciu o automat komórkowy dodając kolejną warstwę dynamiczną, reprezentującą rozprzestrzenianie się ognia oraz dymu.



Projekt symulatora dymu i pożaru

Stworzony w oparciu o automat komórkowy dodając kolejną warstwę dynamiczną, reprezentującą rozprzestrzenianie się ognia oraz dymu.

```
def calculateFireAndSmoke(self):
    for x in range(1, BOARD_WIDTH - 1):
        for y in range(1, BOARD_HEIGHT - 1):
            fire_value = self.board[x][y].getFireValue()
            smoke_value = self.board[x][y].getSmokeValue()

            if fire_value > 0:
                for i in range(-1, 2):
                    for j in range(-1, 2):
                        if not (i == 0 and j == 0):
                            neighbor_x = x + i
                            neighbor_y = y + j
                            if 0 <= neighbor_x < BOARD_WIDTH and 0 <= neighbor_y < BOARD_HEIGHT:
                                spread_probability = 0.8
                                if random.random() < spread_probability:
                                    self.board[neighbor_x][neighbor_y].setFireValue(fire_value * 0.8)

            if smoke_value > 0:
                self.board[x][y - 1].setSmokeValue(smoke_value * 0.7)

                for i in range(-1, 2):
                    neighbor_x = x + i
                    neighbor_y = y
                    if 0 <= neighbor_x < BOARD_WIDTH and 0 <= neighbor_y < BOARD_HEIGHT:
                        self.board[neighbor_x][neighbor_y].setSmokeValue(smoke_value * 0.5)

            if fire_value == 0 and smoke_value == 0:
                source_probability = 0.01
                if random.random() < source_probability:
                    intensity = random.randint(50, 100)
                    density = random.randint(20, 50)
                    self.board[x][y].setFireValue(intensity)
                    self.board[x][y].setSmokeValue(density)
```

```
def initializeFire(self):
    for _ in range(random.randint(5, 10)):
        x, y = random.randint(1, BOARD_WIDTH - 2), random.randint(1, BOARD_HEIGHT - 2)
        intensity = random.randint(50, 100)
        for i in range(-1, 2):
            for j in range(-1, 2):
                neighbor_x = x + i
                neighbor_y = y + j
                if 0 <= neighbor_x < BOARD_WIDTH and 0 <= neighbor_y < BOARD_HEIGHT:
                    self.board[neighbor_x][neighbor_y].setFireValue(intensity)

def initializeSmoke(self):
    for x in range(1, BOARD_WIDTH - 1):
        for y in range(1, BOARD_HEIGHT - 1):
            density = random.randint(20, 50)
            self.board[x][y].setSmokeValue(density)
```

Kod

Wszystkie dane są dostępne w pliku data.py

```
for move_x, move_y in moves:
    if 0 <= move_x < BOARD_WIDTH and 0 <= move_y < BOARD_HEIGHT:
        static_value = self.board[move_x][move_y].getStaticValue()
        dynamic_value = self.board[move_x][move_y].getDynamicValue()
        # TODO:
        # current_value
        # dynamic_value
        # define N, alfa, beta
        # isObstacle(), isOtherMan()
        # wzor = N * current_value * math.exp(alfa*dynamic_value) * math.exp(beta*static_value)
        if not self.board[move_x][move_y].isObstacle() and bestMove > static_value:
            bestMove = static_value
            bestPosition = (move_x, move_y)
            new_dynamic = dynamic_value

# UPGRADE dynamic_value
best_x, best_y = bestPosition
self.board[best_x][best_y].setLayerVal(LayerType.DYNAMIC, new_dynamic+1)
```

$$p_{ij} = NM_{ij} \exp(aD_{ij}) \exp(\beta S_{ij}) (1 - n_{ij}) d_{ij}$$

Co jest dalej do zrobienia

- Logika
- Wstępna wersja dokumentacji całej pracy
- Symulator dymu
- Warstwa przeszkód dostosowana do tunelu

• simulation_tunnel_evacuation #11

Podstawowa walidacja

 High

• simulation_tunnel_evacuation #5

porównanie czasów i gęstości zadymienia
(dodatkowe warunki)

 Low


• simulation_tunnel_evacuation #6 ***

podróżowanie w grupach

 Low

• simulation_tunnel_evacuation #7

Social Distancing

 Low

• simulation_tunnel_evacuation #8

dokumentacja

 Low