Symulacja ruchu ludzi w centrum handlowym

Paweł Kłeczek
pkleczek@student.agh.edu.pl

Kajetan Rzepecki
kajetan.rzepecki+agh@gmail.com

19 grudnia 2012

Streszczenie

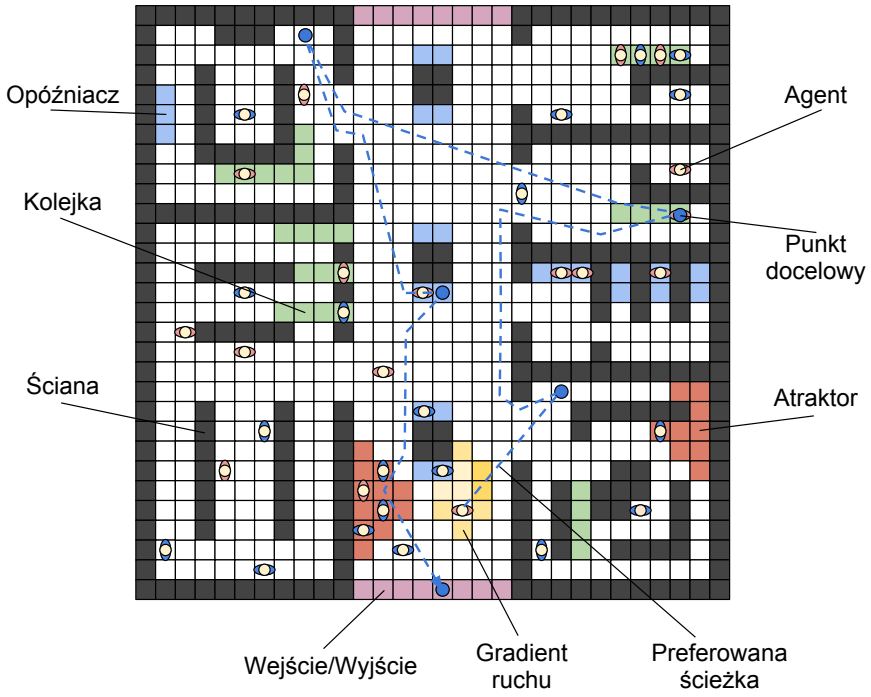
Praca i związany z nią projekt dotyczą symulowania ruchu ludzi w centrum handlowym z wykorzystaniem modelu *Social Distances* na dwuwymiarowej siatce.

Spis treści

1 Wprowadzenie

Celem wykonywanego projektu jest stworzenie modelu oraz symulacja ruchu ludzi w centrum handlowym w oparciu między innymi o model *Social Distances* ([Wąs, Gudowski, Matuszyk 2006, Karakayali 2009]).

Modelowanie ruchu dużych grup ludzi w środowisku centrum handlowego jest problemem złożonym i wymaga wykorzystania równie złożonych algorytmów celem dokładnego przybliżenia rzeczywistych zachowań. Dobrym podejściem jest dekompozycja problemu modelowania złożonego zjawiska na mniejsze, łatwiejsze do rozwiązania podproblemy, którymi zajmują się osobne, dobrze zdefiniowane i wyspecjalizowane algorytmy, i ponowne połączenie wyników ich działania w spójną całość - metoda *Divide and Conquer*.



Rysunek 1: Przykładowa dekompozycja problemu modelowania ruchu ludzi w centrum handlowym.

W zależności od domeny rozwiązywanego problemu dekompozycja może zachodzić ze względu na wiele czynników i dotyczyć różnych aspektów problemu - np. podział wejściowego zbioru danych na dwa mniejsze podzbiory w algorytmie *Quick sort*, czy podział modelu ruchu ludzi na elementarne zachowania, jak *kolejkowanie*, *atrakcja* lub *oczekiwanie*.

Na rysunku 1 została przedstawiona przykładowa dekompozycja problemu poruszanego w tym dokumencie. Wyszczególniono w niej podział na globalną, preferowaną ścieżkę wiodącą poruszającego się po centrum handlowym agenta do obranych przez niego celów i lokalny ruch zgodny z gradientem ruchu obliczonym na podstawie jego otoczenia. Dodatkowo zastosowano podział na specjalne strefy odpowiedzialne za modelowanie elementarnych zachowań ludzi w centrach handlowych, takie jak strefy kolejek, czekania i gromadzenia się, które realizowane są za pomocą innych modeli ruchu.

1.1 State of the art

Zagadnienia modelowania ruchu ludzi są od długiego czasu postrzegane jako istotne z punktu widzenia usługowego. Już w latach 80 ubiegłego wieku Aloys Borgers i Harry Timmermans zajmowali się modelowaniem ruchu pieszych w środowisku centrum handlowego w oparciu o modele probabilistyczne ([Borgers, Timmermans 1986]). Na przestrzeni lat modele ruchu pieszych ewoluowały w wielu kierunkach, od dynamiki płnów ([Helbing 1992]) przez automaty komórkowe ([Blue, Adler 2001]) i algorytmy genetyczne ([Kitazawa, Batty 2004]) aż do modeli opartych o systemy wieloagentowe ([Rauh, Schenk, Schrődl 2011]).

Cellular automata microsimulation for modeling bi-directional pedestrian walkways ([Blue, Adler 2001]) skupia się na modelowaniu dwukierunkowego ruchu ludzi z wykorzystaniem automatów komórkowych. Praca ta dowodzi, że stosunkowo nieskomplikowany automat komórkowy z małą liczbą reguł jest zdolny do symulowania skomplikowanych zachowań, które dobrze oddają rzeczywistość. Zaproponowany model i algorytm **CA-ped** pozwalają symulować ruch ludzi w dwóch przeciwnych kierunkach na dwóch oddzielnych jak i mieszanych pasach ruchu, a także dynamicznych, wieloliniowych (*dynamic multi-lane, DML*) pasach ruchu.

Algorytm **CA-ped** wyróżnia dwie fazy ruchu, z których każda charakteryzuje się trzema prostymi zasadami:

1. Zmiana pasa ruchu
    - Eliminacja konfliktów - dwóch przechodniów nie może znajdować się na jednej komórce, w przypadku konfliktu wolna komórka oddzielająca dwóch przechodniów jest przyznawana jednemu z nich z równym prawdopodobieństwiem.
    - Identyfikacja wolnych przestrzeni - wybierany jest ten pas ruchu, który cechuje największa liczba wolnych komórek, co implikuje bezproblemowy ruch w przyszłości.
    - Zmiana pasa - każdy przechodzień jest poruszany na jeden z dwóch sąsiednich pasów, lub pozostaje na obecnym pasie ruchu.
2. Ruch do przodu
    - Obliczanie szybkości ruchu - dla każdego przechodnia obliczana jest jego szybkość uzależniona od istnienia, bądź nie, wolnych komórek w jego otoczeniu.
    - Wymijanie - jeśli bezpośrednio przed przechodniem w niedużej odległości istnieje przeciwstawny przechodzień, to z zadanym prawdopodobieństwiem następuje wyminięcie obu przechodniów.
    - Ruch - każdy przechodzień jest przemieszczany do przodu zgodnie z szybkością jego ruchu.

Pedestrian behaviour modelling - An application to Retail Movements using a Genetic Algorithm ([Kitazawa, Batty 2004]) identifikuje problemy i nieścisłości wynikające ze stosowania modeli opartych o najkrótsze ścieżki (*shortest-paths*) i maksymalizację przydatności (*utility-maximization*) do symulowania ruchu ludzi w centrum handlowym. Specjalnie zaprojektowany algorytm genetyczny wykorzystuje informacje o optymalnych ścieżkach i mapę routingu i atrybutów konkretnych miejsc centrum handlowego do znalezienia mniej optymalnych ścieżek prowadzących agentów do wybranych przez nich sklepów, które lepiej modelują zachowanie ludzi w środowisku centrum handlowego.

Na podstawie społecznych i ekonomicznych atrybutów poszczególnych miejsc centrum handlowego i atrybutów ludzi przebywających w centrum handlowym, takich jak szybkość, wiek,

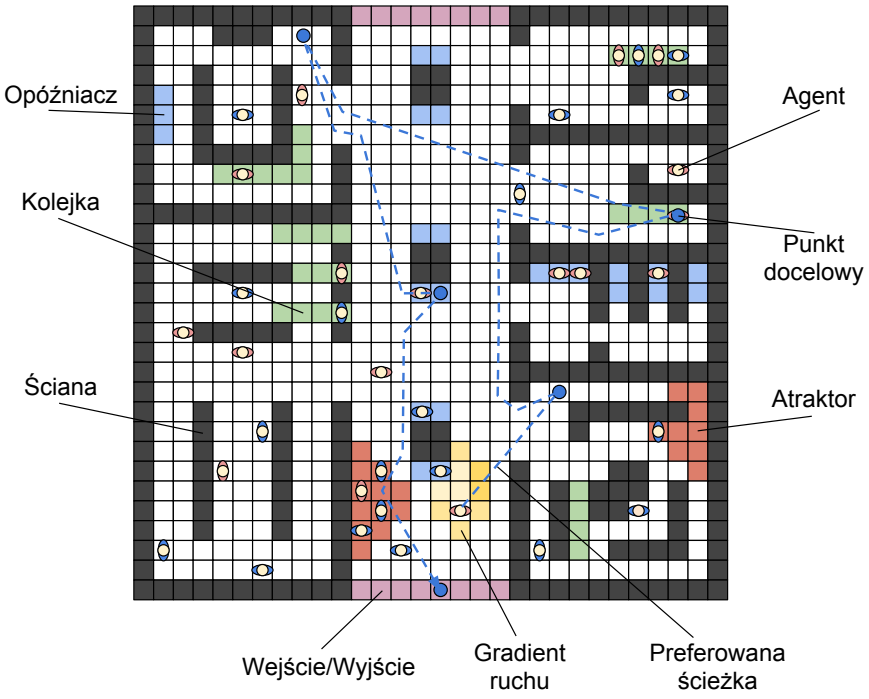
płeć i przychód, oraz korzystając z mapy routing algorytm wyznacza listę miejsc zainteresowania, które kupujący planują odwiedzić. Następnie z użyciem algorytmu genetycznego wyznaczane są ścieżki obejmujące wszystkie miejsca zainteresowania konkretnego kupującego, które zostają wykorzystane w dalszej części symulacji. Istotnym jest wyszczególnienie fazy ruchu lokalnego, gdzie zachodzą dodatkowe zjawiska, których nie obejmuje zasięg zastosowanego algorytmu genetycznego. Wspomniane zjawiska to *omijanie przeszkód* (*collision avoidance*) występujących w lokalnym otoczeniu poruszającego się człowika oraz *grupowanie się* (*flocking*), czyli skomplikowane zjawisko przyciągania kupującego do grup innych kupujących zgodnie z jego socjologicznymi preferencjami.

The Simulated Consumer - an Agent-based approach to Shopping Behaviour ([Rauh, Schenk, Schrődl 2011]) identyfikuje istnienie dużej liczby różnych schematów zachowań i szeroki zakres problemu modelowania ruchu ludzi w centrum handlowym proponując podejście agentowe jako dostatecznie elastyczną metodę modelowania skomplikowanej dynamiki kupujących. W oparciu o skorelowane w niewielkim stopniu dane pochodzące z północy Szwecji i południa Niemiec autorzy definiują agentowy model wyboru miejsc zainteresowania, który następnie z powodzeniem stosują do symulowania zachowania kupujących w dwóch różnych sektorach - spożywczym i odzieżowym, jednocześnie potwierdzają elastyczność zaproponowanego modelu.

Więcej interesujących publikacji związanych z tematyką tego projektu zawarto w secji 6.

2 Model centrum handlowego

Zgodnie z metodą *Divide and Conquer* zaproponowaną we wprowadzeniu, zdecydowano się na dekompozycję problemu modelowania ruchu ludzi w centrum handlowym na elementarne, abstrakcyjne zachowania oraz, ze względu na cele poszczególnych agentów i sposoby ich osiągania, na globalne i lokalne planowanie trasy podróży, co zawarto na rysunku 2.

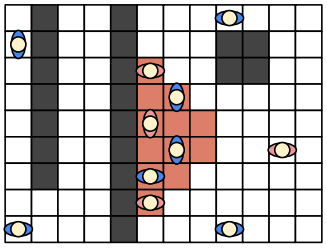


Rysunek 2: Zastosowana dekompozycja problemu.

Model centrum handlowego przewiduje istnienie specjalnych stref, wewnątrz których algorytmy odpowiedzialne za poruszanie agentów są modyfikowane lub zastępowane celem modelowania dobrze zdefiniowanych elementarnych zachowań, takich jak *kolejkowanie*, czy *grupowanie się*. W wyniku obserwacji stwierdzono istnienie czterech rodzajów stref specjalnych - strefy przyciągania uwagi (atraktory), kolejki, przejścia i miejsca oczekiwania (opóźniacze).

2.1 Atraktory

Atraktor jest specjalną strefą przyciągającą uwagę agentów poruszających się po centrum handlowym. Atraktor ma za zadanie modelować obecność przedmiotu lub zjawiska, które przykuwa uwagę ludzi na terenie centrum handlowego, a wynikiem jego działania jest spontaniczne powstawanie skupisk ludzi - *grupowanie się*.

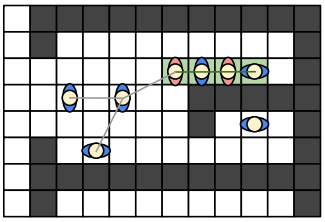


Rysunek 3: Grupowanie się agentów w obrębie atraktora.

Atraktory różnią się między sobą typami - atraktory różnych typów przyciągają inne rodzaje agentów, co modeluje różnice w preferencjach ludzi przebywających w centrum handlowym.

2.2 Kolejki

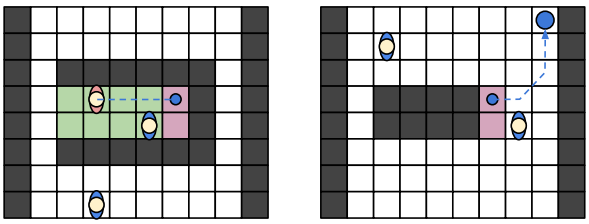
Drugim istotnym z punktu widzenia modelu centrum handlowego typem strefy specjalnej jest strefa kolejki. Zadaniem kolejki jest modelowanie zjawiska *ścisłego kolejkowania się* ludzi, na przykład przy kasie sklepowej, lub na stopniach eskalatora, co nie wynika z ogólnego modelu ruchu.



Rysunek 4: Kolejkowanie się ludzi przy kasie sklepowej z zaznaczoną kolejką ścisłą.

2.3 Przejścia/wejścia/wyjścia

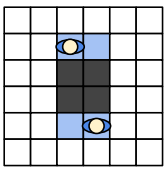
Strefy przejść podobnie jak koleki modyfikują preferencje odległościowe agentów i prowądzą do zmniejszenia odległości między nimi, co nie wynika z ogólnego modelu ruchu. Przejścia, jak sama nazwa wskazuje, modelują wszelkiego rodzaju wejścia, wyjścia i przejścia prowadzące między piętrami centrum handlowego.



Rysunek 5: Poruszanie się agenta po eskalatorze pomiędzy piętrami centrum handlowego.

2.4 Miejsca oczekiwania (Opóźniacze)

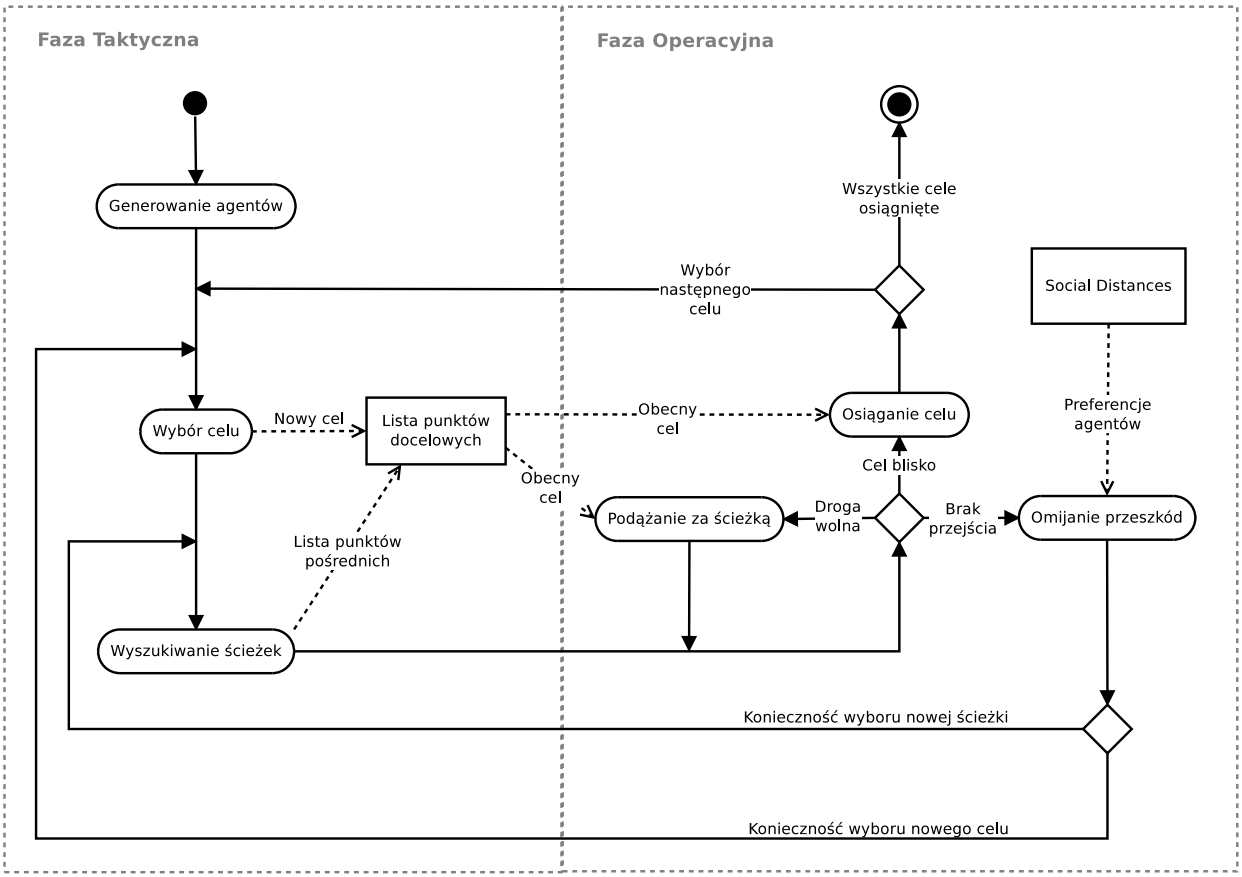
Ostatnim istotnym typem strefy specjalnej jest miejsce oczekiwania, modelujące wszelkiego rodzaju miejsca spędzania czasu w bezruchu.



Rysunek 6: Agenci oczekujący końca świata.

3 Model ruchu ludzi

W zastosowanym algorytmie można wyszczególnieć dwie główne, wzajemnie od siebie zależne fazy - fazę taktyczną oraz fazę operacyjną, których interakcję przestawiono na poniższym, uproszczonym diagramie.

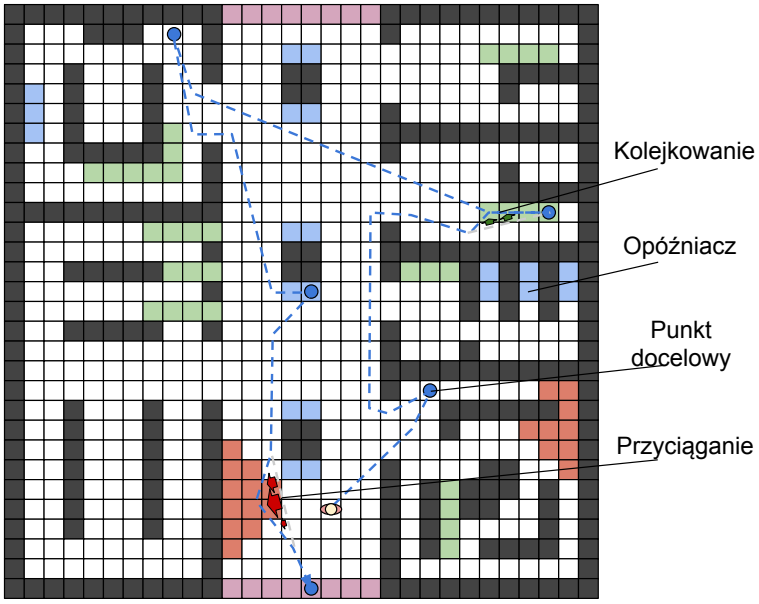


Rysunek 7: Diagram aktywności agentów.

Algorytm rozpoczyna działanie od wygenerowania agenta na podstawie wcześniej zdefiniowanych archetypów. Dla każdego agenta wybierana jest wstępna lista miejsc docelowych, które zostaną przez niego odwiedzone w czasie działania symulacji, oraz obliczana jest optymalna ścieżka łącząca wybrane w poprzednim kroku miejsca docelowe. Algorytm następnie modyfikuje ścieżkę w oparciu o mapę rozkładu stref specjalnych centrum handlowego by lepiej modelować faktyczne zamiary danego agenta. Faza taktyczna kończy się wybraniem niewielkiej ilości punktów pośrednich leżących na wyznaczonej ścieżce.

Po wygenerowaniu niezbędnych danych taktycznych dla każdego agenta algorytm przechodzi do fazy operacyjnej, która odpowiada za właściwe przemieszczanie agentów. Faza ta zachodzi w lokalnym otoczeniu każdego agenta i odpowiada za zachowania takie jak omijanie przeszkód, grupowanie się i podążanie za obraną ścieżką. Algorytm na podstawie bezpośredniego otoczenia agenta oraz metadanych dotyczących obecnego celu jego podróży podejmuje decyzje o możliwości wykonania ruchu, lub w przypadku skrajnym o modyfikacji wybranej ścieżki prowadzącej do celu, czy nawet zmianie aktualnego celu podróży. W przypadku osiągnięcia miejsca docelowego algorytm przechodzi do rozpatrywania następnego miejsca docelowego, lub w tryb "błądzenia", gdy osiągnięto już wszystkie wyznaczone cele.

3.1 Faza taktyczna



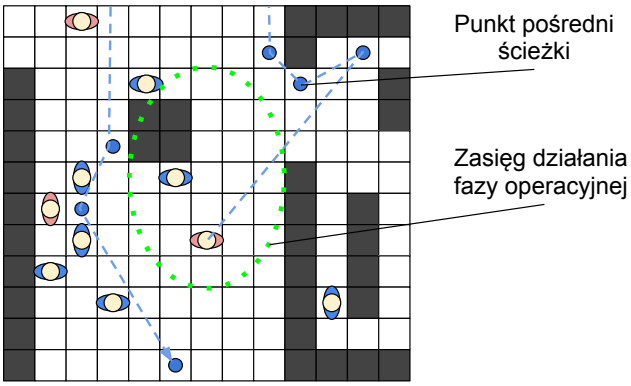
Rysunek 8: Zakres operacji taktycznej części modelu ruchu.

Faza taktyczna zachodzi globalnie dla każdego agenta bez uwzględnienia jego lokalnego otoczenia, innych agentów, czy fizycznych właściwości centrum handlowego - nie jest istotnym, czy dany korytarz został zablokowany przez grupę ludzi i nie umożliwia przejścia. Faza ta modeluje abstrakcyjne zamiary agenta i jej celem jest przede wszystkim wybór listy miejsc docelowych oraz wyznaczenie dróg do nich prowadzących.

Osiągnięto to dzięki algorytmowi znajdowania ścieżek A* oraz mapie rozkładu stref specjalnych centrum handlowego, która jest wykorzystywana jako dodatkowa heurystyka A* - specjalne strefy centrum handlowego modyfikują estymowaną ocenę danego punktu wyznaczanej ścieżki nadając jej pożądany kształt i prowadząc ją w odpowiedni sposób do celu podróży.

Przy wyznaczaniu ścieżek pod uwagę brane są atraktory, kolejki i przejścia, które algorytm stara się uwzględnić za pomocą opisanej heurystyki.

3.2 Faza operacyjna



Rysunek 9: Zakres działania operacyjnej części modelu ruchu.

Faza operacyjna zachodzi w lokalnym otoczeniu każdego agenta, a jej celem jest wykonanie właściwego ruchu agenta. Faza ta jest odpowiedzialna za unikanie kolizji i omijanie przeszkód. Pod uwagę brani są inni agenci oraz metadane dotyczące drogi prowadzącej do aktualnego celu podróży wygenerowane w taktyczniej fazie działania algorytmu. W implementacji fazy operacyjnej wykorzystano zmodyfikowane algorytmy Ped-4 i Social-Force.

Oryginalny **Ped-4** został zaproponowany w pracy *Modeling Four Directional Pedestrian Movements* ([Blue, Adler 2000]). Ponieważ prowadzi on do samoorganizowania się pieszych w kolumny, w symulacji galerii używany jest w odniesieniu do przestrzeni, po których ruch odbywa się w sposób uporządkowany (tj. do korytarzy, alejek oraz niewielkich skrzyżowań).

Algorytm składa się z następujących etapów:

1. **Dostosowanie kierunku ruchu** - gdy kąt między kierunkiem ruchu, a kierunkiem do celu przekroczy wartość graniczną, pieszy skręca w stronę celu.

2. **Zmiana pasa ruchu** - obliczana jest ilość wolnych pól dla obecnie zajmowanego pasa i dla pasów sąsiednich. Pieszy zajmuje ten pas, który na chwilę obecną umożliwia mu przesunięcie się o największą ilość pól do przodu. Przy wyborze preferowany jest dotychczasowy pas. W przypadku, gdy obecnie zajmowany pas nie jest "czysty" (tzn. z naprzeciwka idzie inny pieszy), aby uniknąć kolizji pieszy stara się "schować" za inną osobą idącą w tym samym co on kierunku.

3. **Krok naprzód** - w przypadku, gdy pole bezpośrednio przed pieszym (zgodnie z kierunkiem ruchu) jest wolne - próba zajęcia go. Jeśli nie jest to możliwe - próba dokonania zamiany miejsca z innym pieszym. Wymiana odzwierciedla sytuację, gdy piesi "skręcają tułowie", by "przecisnąć się" obok siebie. Etap *krok naprzód* powtarzany jest wielokrotnie, dopóki nie skończą się "punkty ruchu" pieszego.

Poniżej zamieszczono preferowaną kolejność dokonywania zamian:

1. między pieszymi idącymi w przeciwnych kierunkach na tym samym pasie
2. między pieszymi idącymi w przeciwnych kierunkach na sąsiednich pasach (na polach po przekątnej)
3. między pieszymi idącymi w kierunkach prostopadłych względem siebie (na polach po przekątnej)
4. między pieszymi idącymi w kierunkach prostopadłych względem siebie (jeden "tarasuje" drugiemu drogę)

Pieszy może wybrać daną możliwość tylko wtedy, gdy jego "partner" do zamiany nie wykonał jeszcze swojego ruchu. Jeśli dana wymiana nie dojdzie do skutku, próbowana jest kolejna opcja z listy (znajduje to potwierdzenie w obserwacjach - w sytuacji dużego tłoku ludzie chętniej dokonują takich manewrów).

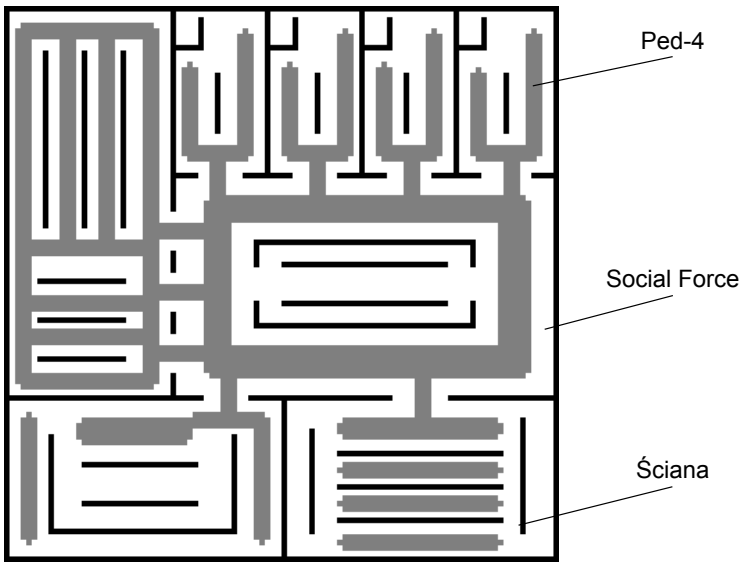
Algorytm **Social Force** opiera się na obserwacji, że każdy człowiek posiada wokół siebie (głównie przed sobą) tzw. *strefę komfortu*, w której niechętnie widzi osoby postronne. Po zsumowaniu stref komfortu wszystkich agentów otrzymuje się swoisty rozkład potencjału, na podstawie którego wyznaczany jest ich dalszy kierunek ruchu. Zasada, według której odbywa się ruch, jest prosta - dojść do celu po polach o jak najmniejszym potencjale. Jeśli w danym momencie brak jest dostępnych pól, pieszy "drepcze" w miejscu w oczekiwaniu na zwolnienie się któregoś z nich. Algorytm *Social Force* służy głównie do modelowania miejsc, w których ludzie "wałęsają się", jak place czy wnętrza sklepów.

4 Implementacja

W celu wykonania projektu posłużono się językiem **Java** oraz bibliotekami **Java Swing** oraz **Monte Media Library**. Poniżej zawarto opis implementacji wybranych algorytmów i obiektów symulacji.

4.1 Reprezentacja centrum handlowego

Centra handlowe reprezentowane są za pomocą map bitowych (plików *.bmp*), których piksele kodują odpowiadające im komórki siatki symulacji, co pozwala tworzyć nowe rozkłady pomieszczeń i stref spejcalnych w bardzo prosty i szybki sposób. Implementacja przewiduje istnienie dwóch map dla każdego centrum handlowego - mapę rozkładu pomieszczeń i algorytmów ruchu fazy operacyjnej (rysunek 10) oraz mapę rozkładu stref specjalnych centrum handlowego (rysunek 11).

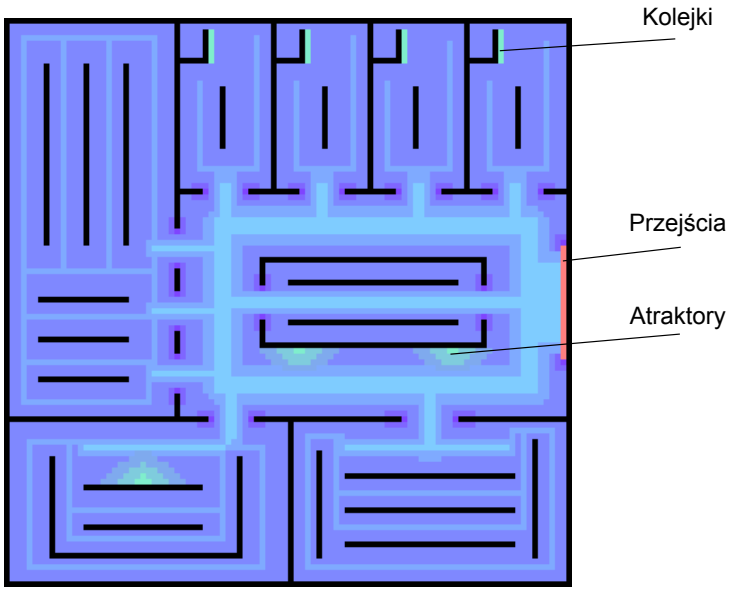


Rysunek 10: Przykładowy rozkład pomieszczeń i algorytmów ruchu małego centrum handlowego.

Mapa rozkładu pomieszczeń zawiera informacje o fizycznych właściwościach pomieszczeń modelowanego centrum handlowego - usytułowanie ścian i innych obiektów blokujących przejście. Dodatkowo zawarto na niej rozkład algorytmów ruchu wykorzystanych w fazie operacyjnej, co pozwala jednoznacznie określić położenie korytarzy, skrzyżowań i innych cech centrum handlowego związanych z ruchem agentów.

Wartości znaczące poszczególnych pixeli RGB mapy:

- $(0x00, *, *)$ - wartość zarezerwowana dla ścian i innych obiektów blokujących przejście.
- $(0x7F, *, *)$ - wartość determinująca wykorzystanie algorytmu Ped-4.
- $(0xFF, *, *)$ - wartość determinująca wykorzystanie algorytmu Social Force.



Rysunek 11: Przykładowy rozkład stref specjalnych małego centrum handlowego.

Mapa rozkładu stref specjalnych centrum handlowego zawiera informacje o funkcjonalnych cechach modelowanego centrum handlowego - rozmieszczenie miejsc przeznaczonych do czekania, kolejek przy kasach sklepowych, okien wystawowych i sieci routingu ułatwiającej przemieszczanie się w centrum handlowym. Informacje te wykorzystywane są przede wszystkim przez fazę taktyczną wykorzystanego algorytmu, gdzie są służą jako dodatkowa heurystyka algorytmu wyszukiwania ścieżek A*.

Wartości znaczące poszczególnych pixeli RGB mapy:

- $(0x7F, A, H)$ - wartość oznaczająca uogólniony atraktor o *atrakcyjności* A i *czasie wstrzymania* H.
- $(0xFF, *, *)$ - wartość oznaczająca przejścia/wejścia, które odpowiednio tworzą lub usuwają agentów z symulacji.

Z racji ograniczonego czasu i niższego priorytetu implementacja nie uwzględnia istnienia wielu pięter w centrum handlowym - symulacja ruchu ludzi wewnątrz wielopiętrowego centrum handlowego może być traktowana jako wiele symulacji ruchu ludzi wewnątrz jednopiętrowego centrum handlowego.

4.2 Wybór puntków docelowych

Punkty docelowe reprezentują cele podróży agentów. Z racji mniejszego priorytetu i braku czasu algorytm wybiera zadaną ilość losowych miejsc znajdujących się we wnętrzu centrum handlowego, które następnie sortuje celem minimalizacji sumarycznej długości drogi je łączącej. Opisany sposób wyboru miejsc docelowych pozwala symulować wszystkie zachowania poruszających się ludzi na poziomie lokalnym (na przykład kolejkowanie się) i większość schematów zachowań tłumu na poziomie globalnym (z wyłączeniem bardziej abstrakcyjnych zachowań uwarunkowanych socjologicznie i kulturowo, takich jak różne stosunki ilościowe agentów danego typu w różnych częściach centrum handlowego).

4.3 Znajdowanie ścieżek

W celu wyznaczania ścieżek prowadzących do punktów docelowych wykorzystano algorytm **A***. A* jest algorytmem kompletnym, służącym do wyznaczania najkrótszej drogi łączącej dwa wierzchołki grafu, który w każdym przypadku jest w stanie znaleźć drogę, jeśli takowa istnieje. A* wykorzystuje heurystykę, która estymuje koszt ścieżki prowadzącej z danego wierzchołka do celu, starając się rozpatrywać tylko najbardziej obiecujące wierzchołki.

Algorytm przechowuje zbiory wierzchołków "otwartych" i "zamkniętych", czyli odpowiednio, wierzchołków, które są rozpatrywane i tych, które już zostały sprawdzone. Działanie algorytmu rozpoczyna się od dodania aktualnej pozycji agenta do zbioru wierzchołków otwartych. W każdym następnym kroku algorytm wykonuje iteracyjnie następujące operacje:

- Ze zbioru wierzchołków otwartych wybierany jest najbardziej obiecujący wierzchołek x, minimalizujący funkcję $f(x)$:
$$f(x) = g(x) + h(x)$$
gdzie $g(x)$ określa długość drogi prowadzącej z wierzchołka początkowego do wierzchołka x, a $h(x)$ określa przewidywaną przez heurystykę długość drogi łączącej wierzchołek x i cel podróży.
- W przypadku osiągnięcia punktu docelowego algorytm kończy się sukcesem.
- W przeciwnym przypadku wierzchołek x dodawany jest do zbioru wierzchołków zamiknięty ch, a sąsiadujące z nim wierzchołki (wierzchołki osiągalne w jednym kroku zaczynając w x), które nie należą do zbioru wierzchołków zamkniętych, dodawane są do zbioru wierzchołków otwartych.

Zależnie od zastosowanej heurystyki algorytm A* jest w stanie dostarczać optymalne ścieżki łączące dwa punkty lub suboptymalne ścieżki, które są obliczane szybciej. Ze względu na rozwiązywany problem i brak wymogu dostarczania optymalnych ścieżek w implementacji zastosowano złożoną, dopuszczalną (*ang. admissible*) heurystykę opisaną w następnej sekcji, którą dodatkowo obarczono wagą $\epsilon = 5$, co pozwala szybko generować ścieżki o pożądanym i koszcie nie przekraczającym ϵ razy kosztu ścieżki optymalnej.

4.4 Dewiacja ścieżek

Dewiacja ścieżek zachodzi podczas działania algorytmu wyszukiwania ścieżek opisanego w poprzedniej sekcji celem ułatwienia zachodzenia pewnych zachowań, takich jak poruszanie się w odpowiedni sposób po korytarzach centrum handlowego. Odpowiednia dewiacja ścieżek została osiągnięta przez modyfikację heurystyki zastosowanego algorytmu A* - każdy fragment mapy rozkładu stref specjalnych centrum handlowego posiada przypisaną mu wartość determinującą parametry **uogólnionego atraktora** - *atrakcję A* oraz *czas wstrzymania H*.

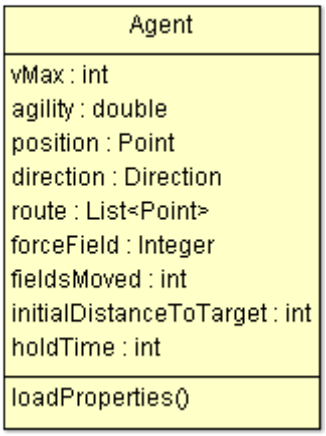
Atrakcja odpowiada za przyciąganie agentów do pewnych stref centrum handlowego i ma za zadanie przede wszystkim umożliwić realizację zdefiniowanych przy analizie problemu atraktorów. Jej wartość kodowana jest przez komponent G pixeli RGB mapy rozkładu stref specjalnych centrum handlowego, kodujących atraktory (komponent R równy $0x7F$). Wartość komponentu G należy podzielić przez $0x7F$ celem przeskalowania do przedziału $[0, 2)$. Heurystyka algorytmu A* wykorzystuje tę wartość, jako dodatkowy mnożnik estymowanego kosztu ścieżki, toteż wartość 0 oznacza najsilniejszą atrakcję, wartość 1 brak atrakcji natomiast wartość 2 oznacza odpychanie - algorytm wyszukiwania ścieżek będzie preferował inne drogi.

Czas wstrzymania odpowiada za opóźnianie agentów podczas wykonywania ruchu - agent osiągający komórkę siatki symulacji obarczoną czasem oczekiwania zostaje opóźniony na zadaną ilość czasu. Wartość czasu wstrzymania kodowana jest przez komponent B pixeli RGB mapy rozkładu stref specjalnych centrum handlowego, kodujących atraktory. Wartość tego komponentu traktowana jest jako ilość kroków symulacji, na które opóźniany agent zostanie wstrzymany.

Implementacja *uogólnionego atraktora* okazała się na tyle elastyczna, że wyeliminowała konieczność osobnej implementacji kolejek i opóźniaczy - kolejki realizowane są jako linie znacząco zwiększonej atrakcji o krótkim czasie wstrzymania, natomiast opóźniacze są punktami o znacząco zwiększonym czasie wstrzymania. Atraktory zostały zrealizowane jako gradienty zwiększającej się atrakcji i czasu wstrzymania - agenci odwiedzajacy atraktor spędzają w nim tym więcej czasu, im bliżej jego centrum się znajdują. Dodatkowym atutem zastosowanej implementacji jest możliwość routingu agentów w centrum handlowym. Wszystkie formy stref specjalnych wyrażonych przez modyfikację heurystyki algorytmu A* zostały zawarte na poglądowym rysunku 11.

4.5 Reprezentacja agentów

Na chwilę obecną agenci opisywani są przez parametry wpływające tylko i wyłącznie na ich ruch w fazie operacyjnej. Docelowo planowane jest rozszerzenie ich zachowania o preferencje dotyczące odwiedzanych miejsc, czasu spędzanego w galerii itp.



Rysunek 12: Klasa agenta

Znaczenie poszczególnych parametrów:

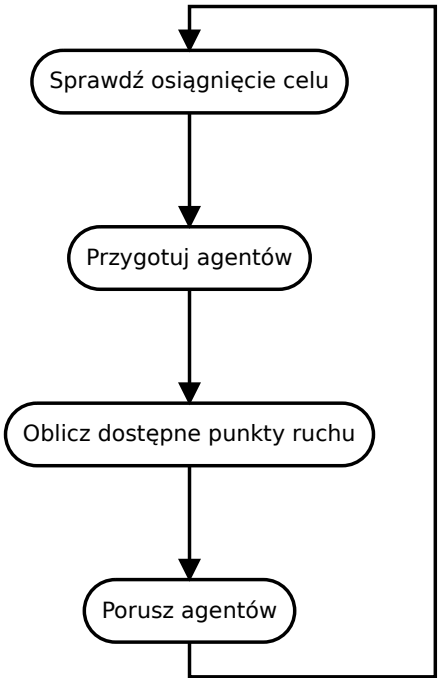
- **vMax** – maksymalna ilość punktów ruchu do wykorzystania w danej iteracji fazy operacyjnej
- **agility** – skłonność do dokonywania zamian z innymi podczas etapu poruszania agentów
- **position** – położenie agenta na planszy galerii
- **direction** – aktualny kierunek ruchu (dostępne kierunki są zgodne ze stronami świata)
- **route** – lista współrzędnych płytek-celów do odwiedzenia
- **forceField** – opis wartości potencjału pól wokół agenta (przy założeniu, że agent zwrócony jest w kierunku północnym)

- **fieldsMoved** – odległość (mierzona w płytkach) przebyta od ostatniego celu do danej chwili
- **initialDistanceToTarget** – teoretyczna minimalna odległość od poprzedniego do obecnego celu
- **holdTime** – pozostały czas oczekiwania (np. w kolejce)

Część spośród powyższych parametrów agenta inicjalizowana jest na podstawie plików zawierających opisy pewnych ogólnych charakterystyk ludzi, np. ”typ dynamicznyóznacza człowieka poruszającego się szybko i skłonnego do dokonywania zamian, ”typ emerytaóznacza stateczny krok i niechęć do zamian.

4.6 Ruch agentów i Ped-4

Faza operacyjna składa się z czterech wykonywanych w pętli etapów, co zilustrowano na poniższym schemacie:



Rysunek 13: Pętla fazy operacyjnej

Sprawdzanie osiągnięcia celu służy wyznaczeniu kolejnego celu, o ile poprzedni został osiągnięty. Cel można osiągnąć na dwa sposoby: wchodząc bezpośrednio na płytkę o zadanych współrzędnych bądź wchodząc na płytkę w sąsiedztwie celu (w tym przypadku prawdopodobieństwo uznania celu za osiągnięty zależy od odległości płytki od niego). Po wyznaczeniu nowego celu aktualizowane są parametry wykorzystywane przez algorytmy ruchu, jak początkowa (teoretyczna) minimalna odległość od celu.

Przygotowanie agentów polega na wywołaniu dla każdego z nich metody *prepare()* algorytmu przypisanego do płytki, na której agent aktualnie się znajduje.

Obliczanie dostępnych w danym obiegu fazy iteracyjnej punktów ruchu to przepisanie aktualnej prędkości maksymalnej każdego z agentów.

W etapie poruszania agentów agenci rozpatrywani są cyklicznie. W jednej iteracji (“kroku”) każdy z agentów może wykonać tylko jedną akcję, np. czkać bądź przemieścić się o jedno (!) pole. Dzięki takiemu rozwiązaniu żaden z agentów nie jest faworyzowany przy dostępie do pól.

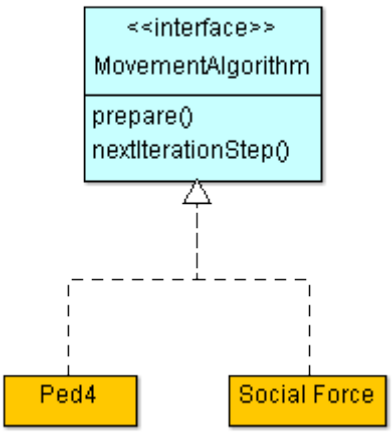
Algorithm 1 Faza operacyjna

```
procedure OPERATION_PHASE
    for all agent w galerii do                                  ▷ sprawdź osiągnięcie celu
        if agent osiągnął cel OR agent uznał, że osiągnął cel then
            pobierz kolejny cel z listy
            zapisz statystyki dotyczące dalszej drogi
        end if
    end for

    for all agent w galerii do                                  ▷ przygotuj agentów
        call Algorithm.prepare with agent
    end for

    for all agent w galerii do                                  ▷ oblicz punkty ruchu
        punkty_ruchu_agenta ← predkosc_agenta
    end for

    for krok od 1 do v_max do                                   ▷ porusz agentów
        for all agent w galerii do
            if agent jest wstrzymywany then
                zmniejsz pozostały czas wstrzymania
            else if agent nie wykorzystał jeszcze w tym kroku punktów ruchu then
                call Algorithm.nextIterationStep with agent
                call Feature.performAction with agent
                wykorzystaj punkt ruchu
            end if
        end for
    end for
end procedure
```

Metoda *performAction()*, wywoływana dla elementów galerii, ustawia parametry agenta związane z symulacją takich zdarzeń, jak np. oczekiwanie w kolejce.

Wszystkie algorytmy ruchu implementują interfejs *MovementAlgorithm*. Interfejs ten zawiera dwie metody wywoływane w fazie operacyjnej dla każdego agenta:

- *prepare()* - wywoływana jeden raz, na początku; służy ustawieniu agenta na odpowiedniej pozycji początkowej
- *nextIterationStep()* - wywoływana tyle razy, ile agent posiada punktów ruchu; obsługuje ruch "do przodu"; przy każdym jej wywołaniu agent przesuwa się co najwyżej o jedno pole (w sąsiedztwie Moore'a)



Rysunek 14: Interfejs *MovementAlgorithm* i jego realizacje

4.7 Social Force

Poniżej przedstawiono pseudokody metody odpowiedzialnej za ruch "do przodu"w algorytmie Social Force:

Funkcja wyznaczająca dostępne pola ogranicza możliwoś ruchu tak, aby agent albo dalej przemieszczał się po dotychczasowej trajektorii, albo skręcił w stronę celu.

Potencjał pola obliczany jest na podstawie siły oddziaływania innych ludzi (ujemna składowa) oraz odległości pola od celu (dodatnia składowa). W przypadku, gdy agent już dość długo zmierza do celu (tzn. przebyta przez niego droga znacząco przekracza minimalną drogę konieczną do osiągnięcia celu), lecz nie może go osiągnąć, przestaje zwracać taką uwagę na innych agentów – efekt ten został osiągnięty przez dodatkowe zwiększenie potencjału dostępnej płytki leżącej najbliżej celu.

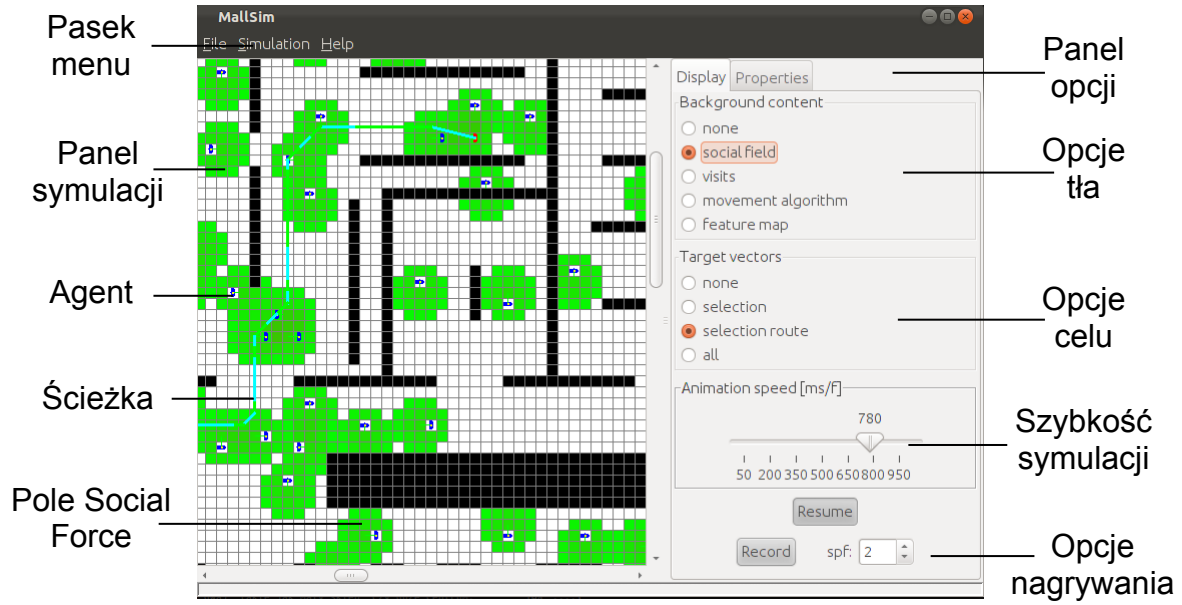
W przypadku, gdy brak jest dostępnych płytek, agent obraca się odrobinę w miejscu, by w następnym kroku zyskać nowe możliwości ruchu (ze względu na zmianę zbioru dostępnych płytek).

Algorithm 2 Ruch agenta w algorytmie Social Force

```
procedure nextIterationStep
    if brak możliwości ruchu then
        obróć agenta                          ▷ umożliwia przeszukanie nowych obszarów
    else
        cel ← wyznacz_pole_o_najwiekszym_potencjale()
        przesuń agenta na docelową płytkę
        odzwierciedl kierunek wykonanego ruchu poprzez zmianę zwrotu agenta
    end if
end procedure

function wyznacz_pole_o_najwiekszym_potencjale
    dostepne_pola ← wyznacz_dostepne_pola()
    for all dostępne pole do
        uwzględnij wartość pola potencjału innych ludzi i odległość od celu
    end for
    if agent jest zmęczony then
        zwiększ dodatkowo potencjał dostępnego pola najbliżej celu
    end if
    pozostaw pola o najwyższym potencjale
return pole leżące najbliżej celu
end function

function wyznacz_dostepne_pola
    pola_agenta ← pola w sąsiedztwie von Neumanna leżące w półkolu wyznaczonym przez
kierunek ruchu agenta
    pola_celu ← pola w sąsiedztwie von Neumanna leżące w półkolu wyznaczonym przez
kierunek do celu
return pola_agenta AND pola_celu                  ▷ część wspólna zbiorów
end function
```

4.8 Wizualizacja i pozyskiwanie danych

Wizualizację przebiegu symulacji wykonano w oparciu o bibliotekę **Java Swing** z racji na prostotę implementacji i zadowalające efekty wizualne, jakie ona dostarcza. Na rysunku 15 zaprezentowano wygląd głównego okna programu podczas przykładowej symulacji.



Rysunek 15: Główne okno programu.

Program udostępnia wiele konfigurowalnych opcji odpowiadających za działanie symulacji i jej wizualizacji dostępnych za pośrednictwem *panelu opcji*:

- **Opcje tła** - wybór danych wizualizowanych w tle symulacji; do wyboru jedna wartość z: *brak*, *gradient Social Force*, *rozkład odwiedzin danych komórek centrum handlowego*, *rozkład algorytmów ruchu*, *rozkład stref specjalnych centrum handlowego*.
- **Opcje celu** - opcje wizualizacji celu podróży agentów; do wyboru jedna wartość z: *brak*, *wektor najbliższego celu wybranego agenta*, *aktualna ścieżka wybranego agenta*, *wektory najbliższego celu wszystkich agentów*.
- **Szybkość symulacji** - pasek wyboru szybkości symulacji (ilość milisekund na krok symulacji); wartości z przedziału 50 do 1000 milisekund.
- **Opcje nagrywania** - opcje odpowiadające za nagrywanie przepiegu symulacji; obecnie dostępna jest tylko konfiguracja ilości kroków na klatkę filmu (*spf*).

...oraz *paska menu*:

- **Seed** - opcje odpowiadające za *ziarno* symulacji umożliwiające kontrolowanie powtarzalności wyników.

W celu ułatwienia pozyskiwania danych do późniejszej analizy program udostępnia zakładkę *Properties* generującą dynamicznie podgląd parametrów wybranego agenta oraz możliwość nagrania przebiegu symulacji za pomocą biblioteki **Monte Media Library** - przebieg symulacji zapisywany jest w głowym katalogu programu w formacie *.avi*.

5 Symulacja i analiza wyników

5.1 Kalibracja i walidacja parametrów symulacji

5.2 Uzyskane wyniki ilościowe i jakościowe

5.3 Wyniki symulacji a rzeczywistość

6 Literatura

[Wąs, Gudowski, Matuszyk 2006] - Social Distances Model of Pedestrian Dynamics

[Karakayali 2009] - Social Distance and affective orientation

[Blue, Adler 2000] - Modelling Four Directional Pedestriam Movements

[Blue, Adler 2001] - Cellular automata microsimulation for modeling bi-directional pedestrian walkways

[Bitgood, Dukes 2005] - Economy of Movement and Pedestrian Choice Point Behavior in Shopping Malls

[Borgers, Timmermans 1986] - A Model of Pedestrian Route Choice and Demand for Retail Facilities within Inner-City Shopping Areas

[Borgers, Timmermans 1986] - City centre entry points, store location, patterns and pedestrian route choice behaviour: a microlevel simulation model

[Borgers, Timmermans 2005] - Modelling pedestrian behaviour in downtown shopping areas

[Kitazawa, Batty 2004] - Pedestrian Behaviour Modelling - An Application to Retail Movements using a Genetic Algorithm

[Zacharias 2000] - Shopping behavior at Alexis-Nihon Plaza in Montreal

[Rauh, Schenk, Schrődl 2011] - The Simulated consumer - an agent-based approach to shopping behaviour

[Helbing 1992] - A Fluid Dynamic Model for the Movement of Pedestrians