6 czerwca 2019

PzL-Z1

1. Stanisław Wasik
2. Dawid Białek
3. Szymon Jędrzejewski
4. Maksymilian Słowiński

Projekt zespołowy - sprawozdanie końcowe

Symulator życia komórkowego

1. Wstęp

Charakterystyka zadania

Symulator życia komórkowego jest to oprogramowanie, będące aplikacją pośrednią między grą, a symulacją. Pozwala on na stworzenie i pośrednie kontrolowanie kolonii komórek, których jedynym zadanie jest przeżycie i rozmnażanie się.

Program ten umożliwia kontrolę komórek znajdujących się na planszy, umieszczanie pożywienia w środowisku oraz szeroko rozumianą manipulację organizmami w symulacji. Elementy żywe posiadają geny definiujące ich zachowanie. Na podstawie genów organizmy podejmują decyzje co do dalszych działań. Dodatkowo odpowiednie cechy organizmów umożliwiają im podejmowanie określonych akcji.

Gracz ma wpływ jedynie na środowisko oraz geny komórek. Nie można bezpośrednio wydawać poleceń organizmom. Rola gracza sprowadza się w głównej mierze do obserwacji przebiegu symulacji.

Niniejszy projekt jest w dużym stopniu realizacją automatu komórkowego. Jego celem jest obserwacja zmian zachodzących w organizmach pod wpływem funkcji matematycznych. Struktura projektu zapewnia szerokie możliwości rozbudowy o kolejne funkcje i zdolności organizmów, co było jednym z założeń początkowych tego przedsięwzięcia.

Przegląd dotychczasowych rozwiązań

Game Of Life [1]

Jest to jeden z pierwszych konceptów automatu komórkowego, wymyślony w roku 1970 przez Johna Conwaya.

Gra toczy się na planszy złożonej z kwadratów ułożonych w siatkę, co sprawia, że każda komórka ma stałą pozycję, a co za tym idzie: ośmiu sąsiadów. Każda komórka może znajdować się w jednym z dwóch stanów: żywa lub martwa. Stany komórek zmieniają się w czasie pod wpływem odgórnie zdefiniowanych reguł matematycznych biorących w ogólności pod uwagę liczbę sąsiadów o podanym stanie.

We wskazanym rozwiązaniu nie pojawia się gracz, jest on jedynie biernym obserwatorem symulacji. Może on ustalić jedynie początkowy układ komórek na planszy.

Cell Lab - aplikacja na system Android [2]

Jest to aplikacja symulująca życie drobnoustrojów w środowisku laboratoryjnym. Celem gry jest zaprojektowanie organizmu, który przetrwa w danym środowisku. Postęp w grze odblokowuje dodatkowe geny, które mogą nadać komórkom nowe właściwości.

Cell Lab zawiera również tryb eksperymentalny, w którym użytkownik może ustawić parametry środowiska i umieścić w nim zaprojektowane organizmy, aby zobaczyć ich rozwój. Dodatkowo można zmienić parametry środowiska, by zobaczyć jak organizmy dostosowują się do niego.

Gracz ma do dyspozycji narzędzia laboratoryjne, którymi zmienia środowisko lub manipuluje komórkami. Nieodłącznym elementem gry jest narzędzie tworzenia genomu komórek oraz możliwość umieszczania ich w środowisku.

Uzasadnienie podjęcia tematu

Temat ten został podjęty ze względu na chęć zrobienia czegoś innego niż kolejny web service oraz chęć spróbowania czegoś bardziej **ambitnego**, co można by podawać ciągłej ewolucji i możliwości dodawania coraz bardziej zaawansowanych technik, dzięki którym można by zaadaptować system jako symulacje do celów medycznych, farmakologicznych itp. Ponadto temat ten został wybrany ze względu na chęć napisania gry komputerowej.

Koncept projektu pojawił się po szerokim rozpoznaniu Game Of Life oraz po zajęciach z Programowania Współbieżnego, gdzie prowadzący barwnie opisywał możliwości implementacji Game Of Life jako aplikacji wielowątkowej. Ponadto pojawiła się opcja rozszerzenia wskazanej symulacji o dodatkowe funkcjonalności.

Symulator życia komórkowego miał w ogólności być rozszerzeniem Game Of Life o nowe możliwości. Dodatkowo sposób implementacji miał umożliwić sukcesywną rozbudowę systemu.

2. Cel i zakres pracy

Podstawowe funkcje systemu

Głównym zdanie tego oprogramowania jest dostarczenie rozrywki. Przy odpowiednim zmodyfikowaniu pozwala on na prowadzenie poważnej symulacji.

Podstawowymi funkcjami systemu są:

1. Prowadzenie symulacji życia komórek.
2. Możliwość modyfikacji genów komórek.
3. Możliwość umieszczania komórek w środowisku.
4. Możliwość tworzenia własnych komórek o określonym genotypie.
5. Możliwość umieszczania pożywienia w środowisku.
6. Obserwacja zachowań automatu komórkowego w zależności od określonych zmiennych (genów).

Prowadzona symulacja ogranicza się do zamkniętego środowiska, którego nie może opuścić żadna komórka. Wszystkie elementy symulacji znajdują się na planszy.

Uwzględniane i pomijane aspekty zagadnienia

Zagadnienia, które zostały uwzględnione względem tego systemu to wszelkie podstawowe funkcjonalności komórki powiązane z próbą przeżycia, takie jak poruszanie się, wyszukiwanie pożywienia, odżywianie się, czy też rozmnażanie. Dodatkowo aplikacja wyposażona jest w różne elementy pozwalające na sterowanie symulacją, takie jak interfejs graficzny pozwalający na zmianę różnych parametrów środowiska, w którym istnieją komórki, czy też samych komórek oraz różne narzędzia, które pozwalają wpływać w mniejszym lub większym stopniu na środowisko i komórki.

Elementów, które zostały pominięte to aspekty powiązane z sztuczną inteligencją, która miała by sterować komórkami.

Dodatkowo poruszające się komórki nie zachowują wielkości fizycznych, takich jak np. pęd. Nie zostało to ujęte w programie celem ułatwienia procesu programowania. Ponadto nie było to celem podstawowym tego projektu.

Udział członków zespołu w realizacji zadań

1. **Stanisław Wasik**
    a. **Nadzór nad projektem**
    b. Środowisko
    c. Manipulacja komórkami
    d. Manipulacja planszą: zoom, przemieszczenie widoku
    e. Wpływ komórek na środowisko
    f. Zapis/odczyt z pliku
2. **Maksymilian Słowiński**
    a. Poruszanie się komórek

    b. Wykrywanie kolizje
    c. Rozmnażanie
    d. Wymiana genów
    e. Śledzenie innych komórek i wykrywanie pożywienia
3. **Dawid Białek**
    a. Odżywianie
    b. Umieszczanie pożywienia na planszy
    c. Rozwój
    d. Mutacje genów
    e. Walka
4. **Szymon Jędrzejewski**
    a. Panel modyfikacji środowiska
    b. Panel podglądu komórki
    c. Panel modyfikacji komórki
    d. Panel wyboru nowej komórki

3. Metody konstruowania systemu

Metoda pracy

Do prowadzenia projektu zostały użyte metody zwinne. Każdy członek zespołu odpowiedzialny był za wcześniej wyznaczone funkcjonalności. Poszczególne zadania zostały wydzielone na określony okres czasu. Kontrola wykonania zadań odbywała się co 2 tygodnie.

Praca nad projektem odbywała się zdalnie. Wszystkie zmiany były umieszczane na wspólnym repozytorium. W przypadku problemów z modułami wytworzonymi przez innego programistę, kontaktowano się z nim w celu ustalenia przyczyny problemu. Przy wystąpieniu poważniejszych błędów prace nad problemem były prowadzone przez więcej niż jednego programistę.

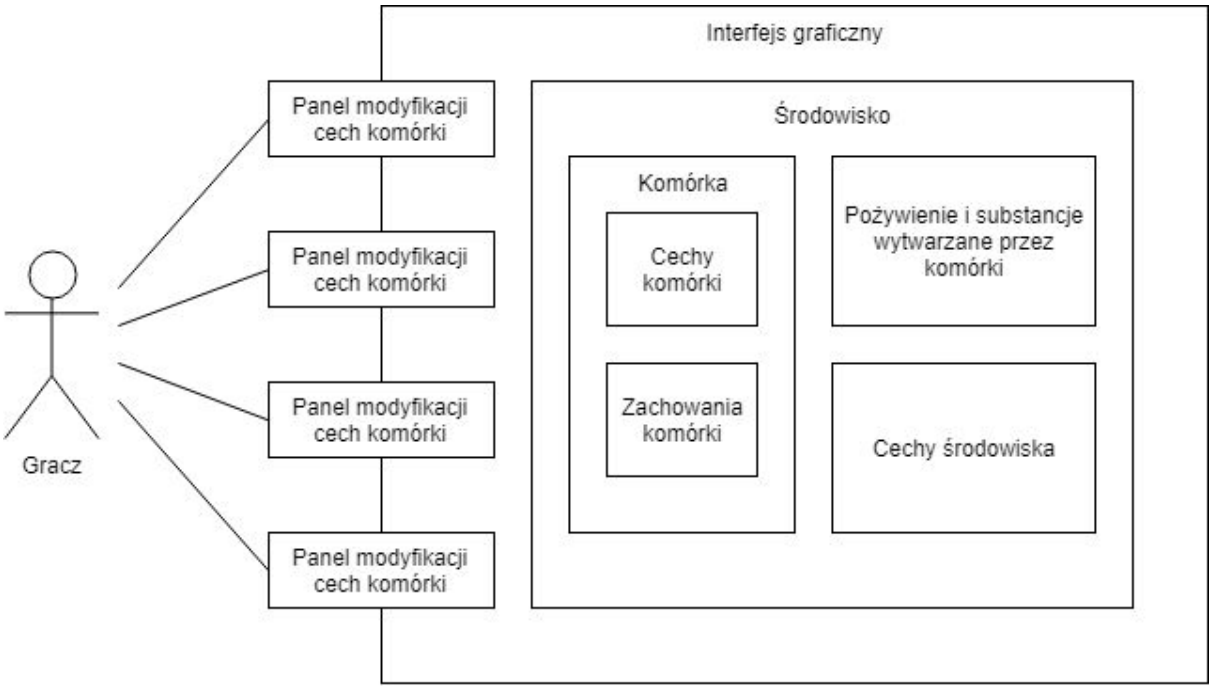
Projekt zarządzany był za pomocą Kanban Board na platformie Trello, która umożliwia np. informowanie członków zespołu o przydzieleniu zadania poprzez e-mail.

Metody modelowania

Modelowanie elementów projektu odbyło się przed przystąpieniem do prac programistycznych. W celu określenia poszczególnych elementów symulacji sporządzono listę właściwości i cech komórek oraz właściwości środowiska symulacji. Dodatkowo rozpatrzono możliwe ścieżki rozwoju projektu. Rozważono także elementy, które docelowo nie miały być zrealizowane. Miało to na celu sprawdzenia możliwości rozbudowy oraz ulepszania aplikacji. Przy modelowaniu zrezygnowano z tworzenia wielu diagramów UML ze względu na duży narzut pracy. Budowa podstawy aplikacji odbyła się według standardowego wzorca gry komputerowej. Wnętrze wzorca zostało następnie wypełnione planowanymi elementami. Ilość elementów nie była duża, zatem spis funkcji komórek oraz środowiska w zupełności wystarczył do podjęcia prac nad projektem. Dodatkowe funkcjonalności zostały dodane podczas prac jedynie ze względu na ciekawy pomysł danego członka zespołu.

Przewidziana platforma aplikacji to Windows ze względu na użyte funkcje, które pochodzą z API systemu Windows. Aplikacja nie przewiduje ponadto żadnej ochrony danych. Informacje zapisywane na dysku są w postaci tekstu, co w domyśle ma umożliwić modyfikację zapisów symulacji przez użytkownika.

W początkowej fazie budowy projektu stworzony został schemat poglądowy budowy aplikacji. Został on przedstawiony poniżej.



Środki implementacji

Do implementacji projektu użyty został język C++. Wersja języka to C++11. Użyto kompilatora dostarczonego z Visual Studio 2017. Kompilacje przeprowadzono dla architektury (x64). Aplikacja została skompilowana jako program desktopowy.

Jako IDE zostało wybrane Visual Studio 2017 ze względu na wysoki poziom wsparcia dla języka C++ oraz powszechną dokumentację środowiska jak i pojawiających się błędów. Dodatkowo wymienione zintegrowane środowisko programistyczne ma możliwość integracji z lokalnym repozytorium Git, dzięki czemu wysyłanie i pobieranie nowych wpisów oraz inne operacje są ułatwione.

Podstawowa biblioteka graficzna użyta do implementacji projektu to SFML w wersji 2.5.1. Biblioteka ta udostępnia przejrzysty interfejs oraz obszerną dokumentację. Ponadto oparta jest o OpenGL, co zapewnia optymalną wydajność. Biblioteka SFML została stworzona aby ułatwić rozwój gier i aplikacji multimedialnych. Składa się z pięciu modułów: systemu, okna, grafiki, dźwięku i sieci. Wspiera popularne systemy operacyjne, takie jak: Windows, Linux, macOS. Dodatkowo planowane jest rozszerzenie listy o systemy mobilne, czyli system Android oraz iOS. Biblioteka zapewnia również wsparcie dla innych języków programowania niż C++ oraz .Net, takich jak: Java, Ruby, Python oraz Go [3].

Do implementacji interfejsu użytkownika początkowo użyta miała zostać podstawowa wersja biblioteki SFML. Koniecznym było dodanie własnych klas implementujących elementy interfejsu. Jednakże znaleziona i użyta została biblioteka TGUI, która jest nakładką na SFML i umożliwia szybkie tworzenie GUI.

TGUI jest międzyplatformową biblioteką GUI przeznaczoną dla C++ i SFML. Gui jest proste w użytkowaniu. Przy użyciu kilku linii kodu użytkownik jest w stanie stworzyć w pełni

funkcjonujące pole tekstowe na ekranie. Widgety mogą zostać stworzone przy użyciu kolorów, a nawet obrazów, co pozwala na duże dostosowanie interfejsu użytkownika. TGUI posiada swój własny Gui Builder, który pozwala na jeszcz prostsze projektowanie interfejsu. TGUI działa na wszystkich platformach wspierających SFML. Pozwala to na używanie biblioteki na systemach takich jak Windows, Linux, Mac OS X, FreeBSD oraz testowo na Raspberry Pi, Android, a także iOS [4][5].

Biblioteka TGUI posiada unikalny system wywołań zwrotnych. Został on napisany specjalnie dla TGUI, a jego celem było możliwie jak najprostsze użytkowanie. Użytkownik nie musi wywoływać samodzielnie std::bind przy przekazywaniu parametrów oraz może zostawić parametry niezwiązane, których widgety mogą używać do przekazywania informacji [4][5].

W implementacji użyto wzorce projektowe celem ujednolicenia kodu i zapewnienia spójnej organizacji danych. Głównym wzorcem był Singleton.

Repozytorium Git

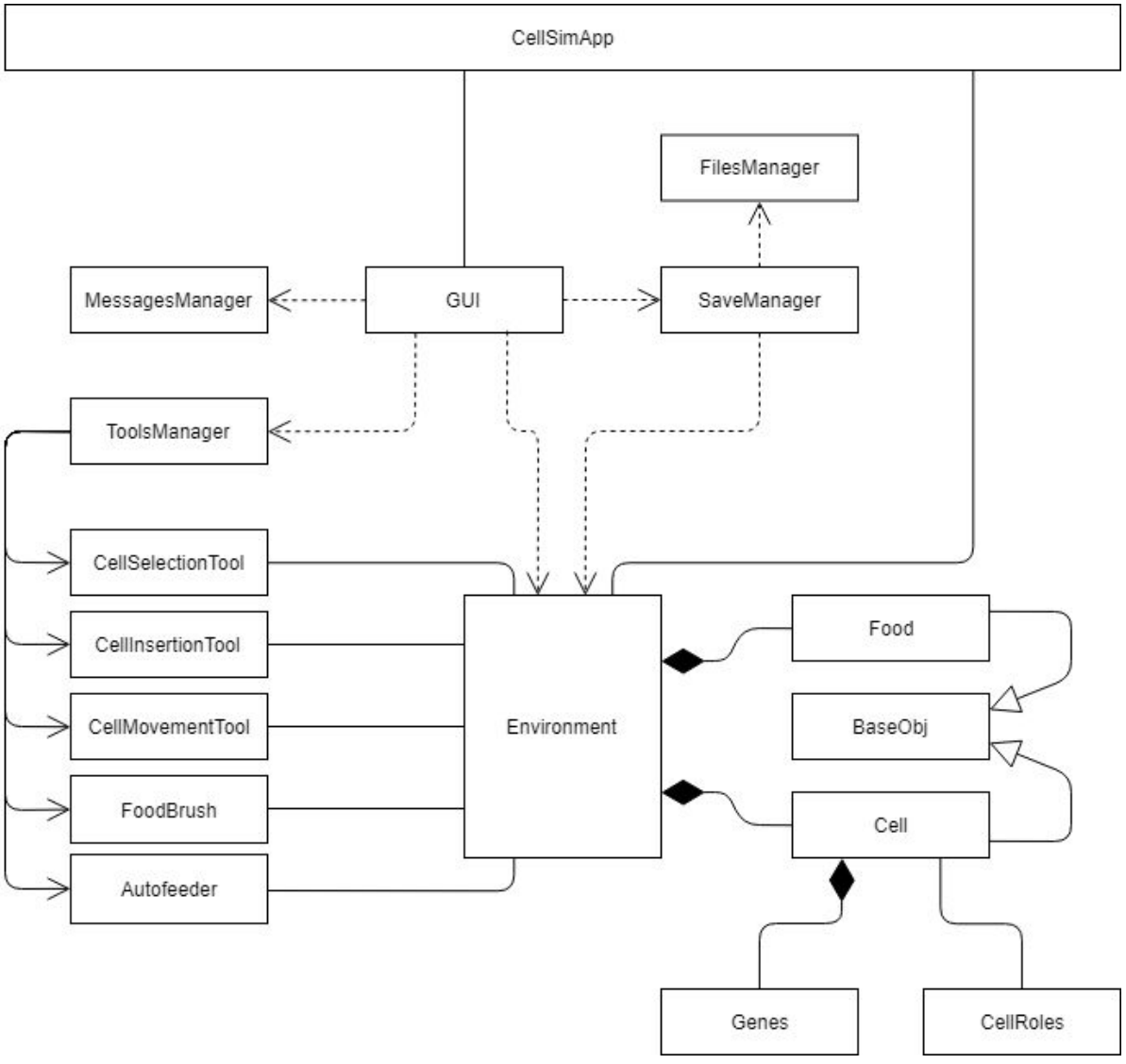
Pełny kod źródłowy tego projektu został umieszczony na repozytorium Git w serwisie GitHub. Link do repozytorium znajduje się poniżej.

https://github.com/st-wasik/CellSimulator

4. Budowa systemu

Ogólna architektura

Poniższy schemat przedstawia ogólną strukturę systemu. W większości przypadków klasy reprezentowane są przez wzorzec Singleton celem ułatwienia dostępu do zawartości obiektu oraz zapewnienia kontroli nad ilością instancji.



Opis modułów - struktura, funkcje

Większość modułów oparta jest o wzorzec Singleton, co zapewnia utworzenie tylko jednej instancji klasy.

Klasa główna (CellSimApp)

Jest to główny moduł programu obsługujący zdarzenia oraz pętlę główną. W module tym znajdują się następujące funkcjonalności dotyczące obsługi aplikacji:

- Konfiguracja używanych zasobów, w tym czcionka używana do wyświetlania informacji,
- Obsługa okna aplikacji,
- Obsługa przemieszczania widoku,
- Obsługa przybliżania widoku gry,
- Obsługa i konwersja współrzędnych myszy,
- Obsługa skrótów klawiszowych,
- Obsługa kodu interfejsu graficznego,
- Obsługa środowiska symulacji zawierającego komórki.

Środowisko (Environment)

Moduł ten obsługuje symulację jako całość. Przeprowadzane w nim są wszystkie obliczenia niezbędne do prowadzenia gry. Przechowywane są w nim następujące dane:

- Temperatura,
- Promieniowanie,
- Liczba żywych komórek,
- Liczba pożywienia na planszy,
- Zbiór żywych komórek,
- Zbiór pożywienia.

Moduł Environment odpowiedzialny jest za wyznaczanie zachowań komórek oraz kontroli pożywienia. Standardowa procedura aktualizacji przebiega następująco:

- Aktualizacja narzędzi,
- Umieszczanie nowych komórek na planszy,
- Aktualizacja komórek na podstawie posiadanych przez nie roli, np. przemieszczanie,
- Usuwanie komórek z planszy,
- Rysowanie elementów graficznych.

Ponadto moduł ten udostępnia dodatkowe narzędzia:

- narzędzie umożliwiające uzyskanie uchwytu do komórki znajdującej się na wskazanej pozycji,
- narzędzie sprawdzania, czy wskazana komórka znajduje się wewnątrz środowiska symulacji,
- narzędzie serializacji planszy, służące do zapisania stanu symulacji do pliku.

Narzędzia symulacji

Narzędzia umożliwiają kontrolę symulacji, w tym komórek i pożywienia. Dostępne są następujące narzędzia:

Narzędzie przemieszczania widoku

- umożliwia przemieszczanie widoku po planszy, celem lepszej obserwacji,
- umożliwia zmianę przybliżenia widoku.

Narzędzie zaznaczania komórek

- pozwala wybrać i śledzić już istniejącą komórkę,
- po wybraniu komórki wyświetla aktualny obiekt, za którym komórka podąża,
- udostępnia parametry wybranej komórki do interfejsu graficznego.

Narzędzie przesuwania komórek

- po wybraniu przez użytkownika konkretnej komórki kursorem myszy, narzędzie pozwala na przemieszczenie obiektu w inne miejsce.

Narzędzie wstawiania komórek

- pozwala ręcznie umieścić komórkę o odpowiednich parametrach na planszy,
- parametry utworzonej komórki pobierane są z interfejsu graficznego.

Narzędzie umieszczania pożywienia

- pozwala na ręczne umieszczanie pożywienia na planszy,
- narzędzie ma postać okręgu o średnicy, która może zostać zmodyfikowana przez użytkownika z poziomu interfejsu graficznego,
- możliwa jest regulacja poziomu twardości pędzla, im pędzel twardszy, tym więcej pożywienie pojawia się na planszy w takiej samej jednostce czasu.

Narzędzie automatycznego umieszczania pożywienia

- odpowiada za automatyczne rozmieszczanie pożywienia na planszy,
- można zmieniać maksymalną ilość pożywienia na planszy oraz szybkość tworzenia nowego pożywienia za pomocą modyfikacji wartości odpowiednich zmiennych.

GUI

GUI zostało zaimplementowane przy pomocy biblioteki rozszerzającej funkcjonalność SFML. Pozycjonowanie elementów jest statyczne, co narzuca rozdzielczość ekranu 1920x1080. Wszystkie elementy GUI korzystają z czcionki systemowej (jest przystosowane do uruchomienia bez pobierania dodatkowych plików). Dzięki komunikacji pomiędzy elementami GUI a aplikacją za pomocą obsługi sygnałów, wartości obecne w interfejsie aktualizowane są tylko, gdy zajdzie zmiana. Obsługa sygnałów nie koliduje z procesami zachodzącymi wewnątrz aplikacji.

Panel komunikatów

Panel ten został utworzony by poinformować użytkownika o ważnych zdarzeniach z pracy aplikacji. Wyświetla on komunikaty o błędach pochodzące np. z pól tekstowych GUI. Ponadto informuje użytkownika o stanie symulacji, np. o postępie zapisu danych do pliku. Wiadomości wyświetlane są na tle środowiska symulacji.

Panel modyfikacji środowiska

Panel ten umożliwia kontrolę warunków panujących w środowisku. Posiada on następujące elementy:

1. **Menu** - element zarządzający symulacją, posiadający następujące elementy:
    a. **new** - tworzy nową, pustą planszę,
    b. **new random** - tworzy nową, losową planszę,
    c. **clear** - czyści planszę,
    d. **save** - zapisuje aktualny stan planszy,
    e. **load** - wczytuje jeden z zapisanych stanów planszy,
    f. **exit** - kończy działanie aplikacji.
2. **Elementy modyfikujące temperaturę środowiska** - zestaw elementów wyświetlający aktualną temperaturę środowiska oraz umożliwiający modyfikację temperatury za pomocą suwaka lub wprowadzania wartości w okno tekstowe. Modyfikacja wartości temperatury jest wykonywana za pomocą sygnału wysyłanego przez suwak. W przypadku wprowadzenia wartości w okno tekstowe zatwierdzenie zmiany skutkuje w zmianie wartości suwaka, co pozwala na zabezpieczenie przed wielokrotną modyfikacją zmiennej na tą samą wartość.
3. **Elementy modyfikujące promieniowanie środowiska** - zestaw elementów wyświetlający aktualne promieniowanie środowiska oraz umożliwiający modyfikację promieniowania za pomocą suwaka lub wprowadzania wartości w okno tekstowe. Modyfikacja wartości promieniowania jest wykonywana za pomocą sygnału wysyłanego przez suwak. W przypadku wprowadzenia wartości w okno tekstowe zatwierdzenie zmiany skutkuje w zmianie wartości suwaka, co pozwala na zabezpieczenie przed wielokrotną modyfikacją zmiennej na tą samą wartość.
4. **Checkbox uruchamiający automatyczne karmienie** - element GUI aktywujący narzędzie automatycznego umieszczania pożywienia w środowisku oraz elementy pozwalające na modyfikację jego parametrów.
5. **Elementy modyfikujące opcje automatycznego karmienia** - zestaw elementów modyfikujący parametry narzędzia automatycznego umieszczania pożywienia w środowisku w zakresie limitu pożywienia oraz częstotliwości pojawiania się nowego pożywienia.
6. **Wybór narzędzia** - zbiór przycisków pozwalających na przełączanie się pomiędzy różnymi narzędziami. Zawarte narzędzia:
    a. podgląd komórki,
    b. tworzenie komórki,
    c. modyfikacja komórki,
    d. wstawianie komórek,
    e. manualne umieszczanie pożywienia.

Panel podglądu komórki

Panel, którego zadaniem jest wyświetlanie statystyk oraz wartości genów aktualnie zaznaczonej komórki. Panel składa się z 8 zestawów widgetów. Są to:

1. **Rozmiar** - wyświetla aktualny rozmiar komórki oraz wartości minimalną i maksymalną dla genu.

2. **Prędkość** - wyświetla aktualną prędkość komórki oraz wartości minimalną i maksymalną dla genu.
3. **Wiek** - wyświetla aktualny wiek komórki oraz wartości minimalną i maksymalną dla genu.
4. **Płodność** - wyświetla aktualny współczynnik płodności komórki oraz wartości minimalną i maksymalną dla genu.
5. **Agresja** - wyświetla aktualny współczynnik agresji komórki oraz wartości minimalną i maksymalną dla genu.
6. **Nasycenie** - wyświetla aktualny poziom nasycenia komórki wartości minimalną i maksymalną dla genu.
7. **Próg podziału** - wyświetla próg podziału na mniejsze komórki dla zaznaczonej komórki.
8. **Zasięg widzenia** - wyświetla zasięg widzenia pożywienia oraz innych komórek dla zaznaczonej komórki.

Panel zawiera również podgląd wyglądu zaznaczonej komórki w wersji powiększonej.

Panel tworzenia komórki

Panel, który umożliwia tworzenie nowych komórek. Jest on podzielony na 4 części:

1. **Wprowadzanie wartości genów nowej komórki** - pozwala na wprowadzenie wartości dla następujących genów:
    a. rozmiar,
    b. prędkość,
    c. wiek,
    d. agresja,
    e. nasycenie (poziom pożywienia),
    f. próg podziału,
    g. zasięg widzenia.
2. **Wybór typu komórki** - pozwala na wybór jednego z trzech typów komórek. Są to:
    a. mięsożerna,
    b. wszystkożerna,
    c. roślinożerna.
3. **Nadanie nazwy** - pozwala na nadanie nazwy tworzonej komórce, która będzie wyświetlana podczas zaznaczenia komórki na planszy. Dodatkowo po zapisaniu komórki będzie można ją znaleźć w panelu wstawiania komórek pod wskazaną nazwą.
4. **Opcja wstawieniu i zapisu** - pozwala na wstawienie na plansze komórki lub zapisanie jej w pamięci, w celu późniejszego użycia.

Panel modyfikacji komórki

Panel pozwalający na modyfikację zakresu genów dla obecnie zaznaczonej komórki. Składa się on z 4 części:

1. **Wprowadzenie nowych wartości genów komórki** - pozwala na wprowadzenie nowych wartości genów. W przypadku pozostawienia pola pustym, jest ono ignorowane w procesie modyfikacji. Geny które można zmodyfikować to:
    a. rozmiar,

    b. prędkość,
    c. wiek,
    d. agresja,
    e. nasycenie (poziom pożywienia),
    f. próg podziału,
    g. zasięg widzenia.
2. **Zmiana typu komórki** - pozwala na zmianę typu pożywienia, którym żywi się zaznaczona komórka. Można wybrać spośród:
    a. mięsożerności,
    b. wszystkożerności,
    c. roślinożerności.
3. **Zmiana nazwy** - pozwala na modyfikację nazwy, która jest wyświetlana nad komórką po jej zaznaczeniu.
4. **Potwierdzenie modyfikacji** - przycisk służący do potwierdzenia zakończenia procesu modyfikacji i wprowadzenia zmiany w komórce.

Panel wstawiania komórek

Panel służący do wstawiania predefiniowanych oraz wcześniej zapisanych w pamięci komórek. Składa się z 2 części:

1. **Podgląd genów komórki** - pozwala na podejrzenie wartości genów aktualnie wybranej z listy komórki. Wyświetlane są następujące informacje:
    a. rozmiar,
    b. prędkość,
    c. wiek,
    d. agresja,
    e. nasycenie (poziom pożywienia),
    f. próg podziału,
    g. zasięg widzenia.
2. **Lista komórek** - lista, na której znajdują się zapisane w pamięci komórki. Pojedyncze kliknięcie na element listy wybiera ją do podglądu, a podwójne - podpina pod kursor w celu wstawienia na plansze

Panel manualnego umieszczania pożywienia

Panel aktywuje tryb manualnego umieszczania pożywienia. Narzędzie działa na zasadzie pędzla. Można wyróżnić 2 zestawy widgetów:

1. **Zestaw modyfikacji promieniem** - pozwala na modyfikację promienia pędzla umieszczającego pożywienie,
2. **Zestaw modyfikacji twardości** - pozwala na modyfikację ilości pożywienia, która zostanie wprowadzona po pojedynczym kliknięciu lewym przyciskiem myszy.

Komórka (Cell)

- jest podstawowym elementem symulacji, reprezentuje organizmy żywe,
- przechowuje zestaw genów przyporządkowany tylko danej komórce,
- przechowuje elementy graficzne reprezentujące komórkę,

- przechowuje zestaw uchwytów do funkcji reprezentujących zasady gry - role komórek.

Pożywienie (Food)

- jest podstawowym elementem pożywienia w symulacji,
- przechowuje elementy graficzne reprezentujące pożywienie,
- posiada funkcje odpowiedzialne za “wzrost” pożywienia na planszy.

Role komórek (Cell Roles)

Role opisują funkcje spełniane przez komórkę. Każda z komórek posiada zestaw ról modyfikujących jej parametry. Niektóre komórki posiadają mniej ról, inne więcej - wszystko w zależności od typu komórki, tj. komórki “roślinożerne” nie posiadają roli odpowiedzialnych za namierzanie jak i atakowanie innych organizmów itd.

Role zostały umieszczone w osobnej klasie - innej niż Cell, celem łatwiejszego dodawania nowych ról, jak i zapewnienia przejrzystości opisywanych zachowań.

Każda z funkcji została specjalnie dostosowana oraz zoptymalizowana za pomocą testów i obserwacji.

Opis działania środowiska

Na środowisko składa się plansza wraz z cechami takimi jak temperatura i promieniowanie.

Plansza

Plansza jest prostokątem o zadanych rozmiarach. Wszystkie działania symulacyjne odbywają się wewnątrz planszy. Żaden element symulacji nie może opuścić planszy, co zapewnione jest przez odpowiednie algorytmy detekcji kolizji.

Temperatura

Temperatura odpowiada za żwawość komórek. Przy zmianie temperatury komórki zmieniają swoją prędkość linowo. Gdy temperatura jest ustawiona na możliwe minimum komórki zaprzestają się przemieszczać.

Promieniowanie

Promieniowanie wpływa na mutacje genów komórek. Im większa ilość promieniowania w środowisku tym większa szansa na wystąpienie dowolnej mutacji. Przy wysokim poziomie promieniowania mutacje genów sprawiają, że niektóre komórki nie spełniają założeń co do wielkości określonych w genach, przez co te umierają. Nie było to celowo zaimplementowane, jednakże jest to naturalnym następstwem występowania zmian w genach.

Opis zachowań komórek

Zachowanie komórek opisane jest przez tzw. role komórek. Każda komórka posiada pewien zestaw uchwytów do ról, które definiują zasady życia danej komórki. Na modyfikację właściwości komórki oraz jej zachowania mają wpływ geny oraz warunki środowiska.

Geny

Geny mają za zadanie opisywać cechy charakterystyczne dla danej komórki. Na ich podstawie określane jest jej zachowanie oraz sposób reagowania na środowisko. Mogą być wymieniane podczas rozmnażania. Dla komórek generowanych podczas tworzenia planszy wartości genów są przydzielane losowo, natomiast podczas ich ręcznego dodawania możliwe jest manualne ustalenie wartości przez użytkownika. Geny określają np.: maksymalną prędkość komórki, poziom agresji względem innych komórek, zasięg dostrzegania innych obiektów, maksymalną ilość spożywanego pożywienia, maksymalny rozmiar, wiek, typ oraz poziom metabolizmu.

Poruszanie

Poruszanie polega na płynnej zmianie pozycji na planszy w linii prostej, skierowanej w kierunku “patrzenia” komórki oraz zmianie kierunku “patrzenia”, czyli możliwości skręcania. Ustawienie temperatury ma wpływ na prędkość przemieszczania się komórki. Im jest ona większa, tym komórki są szybsze, a zmniejszając temperaturę także zmniejszamy prędkość komórek, aż do momentu w którym przestają się ruszać.

Rozmnażanie

Rozmnażanie daje możliwość komórkom na stworzenia potomstwa na dwa różne sposoby:

- poprzez mitozę (rozdzielenie się komórki na dwie identyczne kopie),
- poprzez rozmnażanie płciowe (nowa komórka tworzy się poprzez spotkanie się dwóch innych komórek).

Rozmnażanie poprzez mitozę następuję wtedy gdy komórka jest w pełni nasycona i urośnie do maksymalnych rozmiarów. Rozmnażanie płciowe następuje, gdy dwie komórki mają ochotę na kopulację, występuje między nimi kolizja oraz są one tego samego typu.

Wymiana genów

Wymiana genów zachodzi przy rozmnażaniu płciowym i pozwala na przekazanie części genów rodzica do potomka. Geny potomka są liczone poprzez średnia aretmetyczną genów rodziców. Przy wymianie genów istnieje też szansa na mutację genów i jest ona linowo powiązana z poziomem promieniowania. Im większe promieniowanie tym większa szansa na mutacje, aż do 100% szansy.

Wykrywanie pożywienia i innych komórek

Wykrywanie polega na znalezieniu pozycji obiektu, który jest w zasięgu widzenia komórki i jest najbliższym obiektem. Aby skrócić przeszukiwanie oraz zmniejszyć

zapotrzebowanie na zasoby zostały zastosowane niewidzialne sektory. Komórka przeszukuje tylko najbliższe sektory, aby nie robić tego dla całej planszy.

Odżywianie

Odżywianie polega na spożywaniu pożywienia, które znajduje się wystarczająco blisko komórki. Zwiększa to poziom nasycenia danej komórki. Komórka nie będzie się odżywiać, jeżeli jest już nasycona.

Rozwój

Rozwój komórki polega na zdolności komórki do powiększania swojego rozmiaru. Komórka rośnie, gdy jest nasycona, a gdy głoduje zaczyna się ona zmniejszać.

Mutacje genów

Mutacje genów postępują zgodnie z ilością promieniowania w środowisku. Promieniowania określa procentową szansę na mutacje. Mutacja może występować w trakcie życia komórki jak i także przy rozmnażaniu płciowym. Mutacja zmienia w losowy sposób wartości genów danej komórki.

Walka

Dwie komórki mogą ze sobą walczyć. Wynik jednej potyczki jest zależny od: wielkości komórek, poziomu agresji oraz czynnika losowego. W zwycięskiej komórce poziom głodu ulega zmniejszeniu, natomiast przegrana komórka ulega pomniejszeniu.

Wpływ na środowisko

Wpływ występuje wtedy, gdy komórka umiera. Pozostawia ona po sobie pożywienie, które inne komórki mogą spożyć. Ponadto komórki specjalnego typu pozostawiają na planszy pożywienie w trakcie swojego życia.

Wygląd komórek

Wygląd komórek ustalany jest na podstawie genów oraz stanu środowiska. Na wygląd zewnętrzny mają wpływ następujące elementy:

- Agresja komórki definiuje jej kolor. Czerwona komórka jest agresywna, niebieska pasywna, zaś zielona jest komórką o średniej agresywności.
- Maksymalny poziom nasycenia komórki definiuje rozmiar okręgu będącego obwódką komórki.
- Kolor obwódki zależny jest od maksymalnej szybkości komórki. Dla komórek mogących rozwijać duże szybkości jest to kolor żółty. Dla powolnych komórek jest to półprzezroczysty kolor żółty.
- Na znak wewnątrz komórki wpływ ma typ komórki. Komórka “mięsożerna” posiada wewnątrz trójkąt, komórka “roślinożerna” kwadrat, zaś komórka “wszystkożerna” siedmiokąt.
- W wyższych temperaturach komórki uzyskują dodatkowo barwę czerwoną.
- W niskich temperaturach komórki uzyskują barwę niebieską.

- W warunkach wysokiego promieniowania komórki uzyskują barwę zieloną.

Tekstura jest nakładana na komórki w całkowicie losowy sposób, zatem nie wskazuje wartości żadnego genu.

Zapis do pliku - Odczyt z pliku

By umożliwić przenoszenie symulacji oraz modyfikowanie jej parametrów w różnych momentach, dodany został moduł serializujący zawartość środowiska, w tym komórek i pożywienia.

Przygotowany moduł pozwala odczytywać pliki starszych wersji symulacji, nawet po dodaniu nowych cech do symulacji. Wszystkie zapisy przechowywane są w formie tekstowej, co umożliwia ich modyfikowanie i dostosowywanie do własnych założeń symulacyjnych.

Klasa odpowiedzialna za zapis i odczyt wykorzystuje metodę serializującą całe środowisko symulacji wraz z zawartością. Następnie łańcuch znaków przekazywany jest do klasy odpowiedzialnej za zapis do pliku. Wczytanie symulacji z pliku następuje w analogiczny sposób.

Testy - weryfikacja poprawności i ocena efektywności

Testy były prowadzone w trybie DEBUG w celu śledzenia parametrów komórek. Po stwierdzeniu występowania przepełnienia stosu podczas odczytu planszy z pliku testy zaczęto prowadzić w trybie RELEASE, co także znacznie przyspieszyło działanie aplikacji.

Testy komórek były prowadzone poprzez obserwacje odpowiednie dopasowanie środowiska, by móc zmaksymalizować występowanie sprawdzanego zachowania, tj. czy dana funkcja jest realizowana zgodnie z założeniami.

Testy narzędzi były wykonywane poprzez zwykłe używanie narzędzi i sprawdzanie czy wynik użycia jest zadowalający. Niektóre próby starały się przewidzieć możliwe sposoby wykorzystania narzędzi przez użytkowników, by dowiedzieć się, czy wszystko będzie nadal działać, pomimo nieodpowiednich zachowań użytkowników.

5. Użytkowanie

Sterowanie symulacją

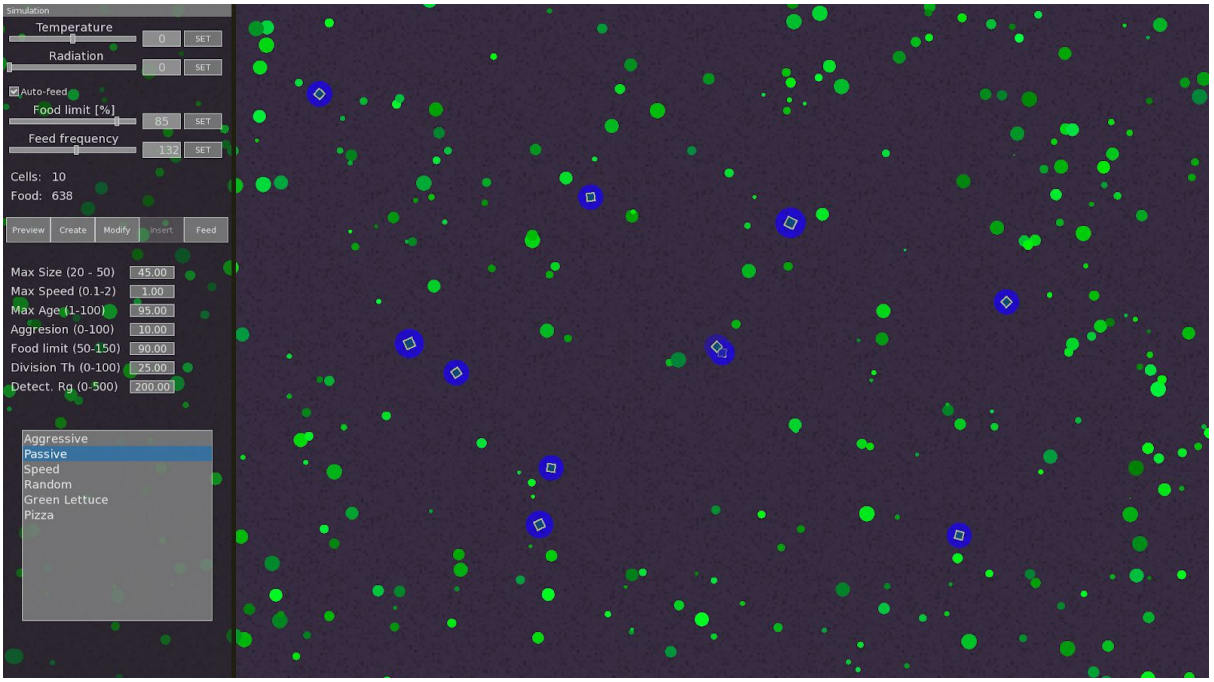
Sterowanie odbywa się głównie przy pomocy kursora myszy oraz klawiatury. Za pomocą kursora możliwy jest wybór opcji w menu, zaznaczanie obiektów na planszy oraz operowanie widokiem; a przy wykorzystaniu klawiatury użytkownik jest w stanie wpisać poszczególne wartości w pola znajdujące się w interfejsie graficznym, jak i wykorzystać niżej wymienione skróty klawiszowe.

Skróty klawiszowe

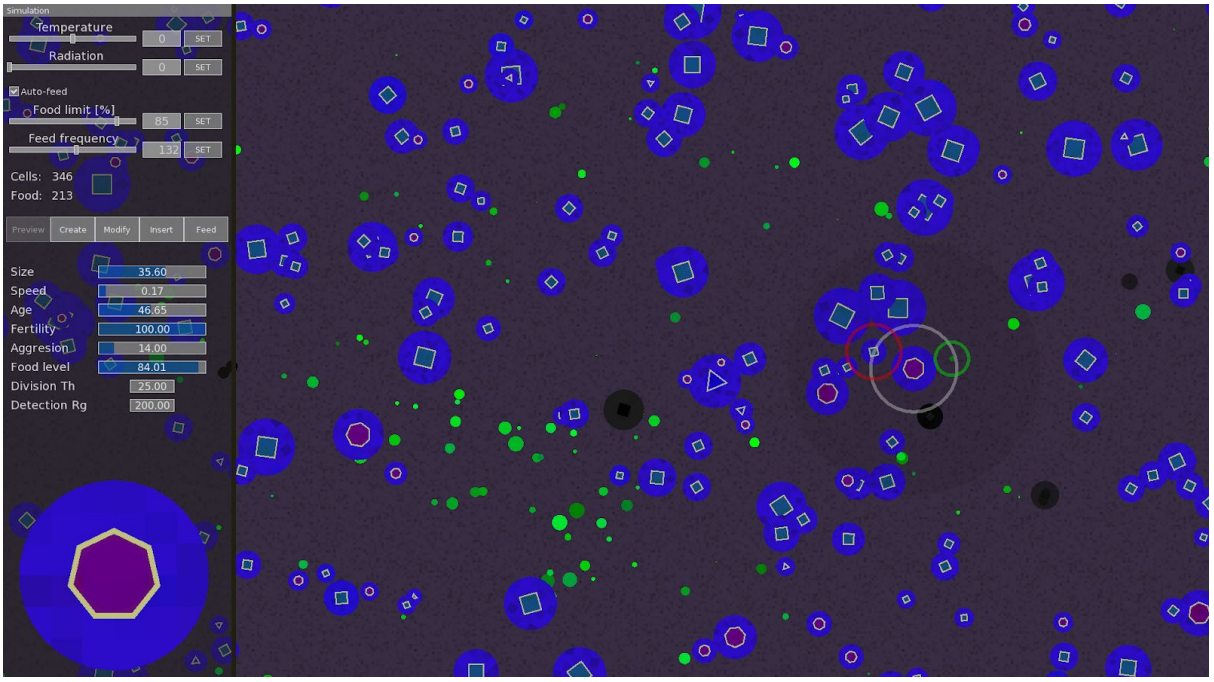
- Esc - wyjście z aplikacji,
- Kółko myszy - obsługa przybliżania i oddalania,
- Num+ - przybliżenie widoku,
- Num- - oddalenie widoku,
- Space - narzędzie zaznaczania: śledzenie zaznaczonej komórki,
- Enter - zatrzymanie i wznowienie symulacji,
- F5 - szybki zapis symulacji,
- F6 - zapis zaznaczonej komórki do pliku,
- F9 - szybkie wczytanie symulacji,
- F1 - narzędzie wstawiania: szybki wybór agresywnej komórki,
- F2 - narzędzie wstawiania: szybki wybór pasywnej komórki,
- F3 - narzędzie wstawiania: szybki wybór losowo generowanych komórek,
- F11 - narzędzie wstawiania: szybki wybór komórki specjalnej “sałata”,
- F12 - narzędzie wstawiania: szybki wybór komórki specjalnej “pizza”.
- Lewy Ctrl - zmiana trybu narzędzia - jeśli dostępna.

Przykładowa sesja użytkowania

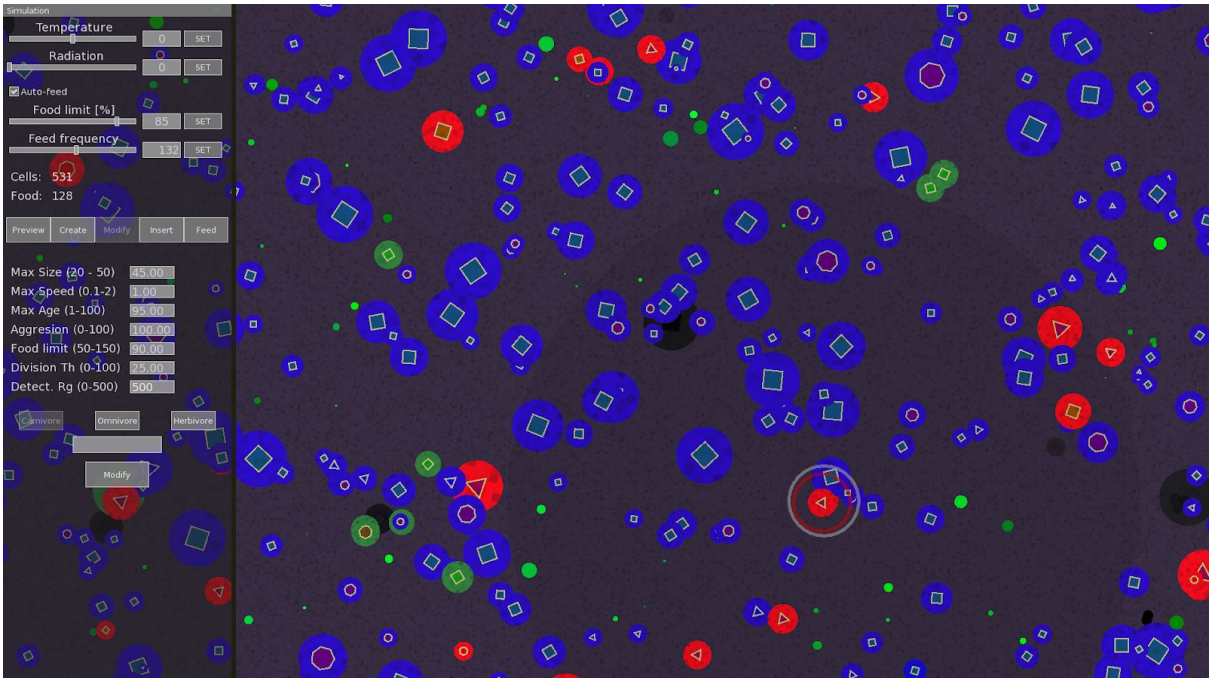
Obraz 1. Gracz wybiera początkowe ustawienia symulacji - wprowadza wymiary planszy.



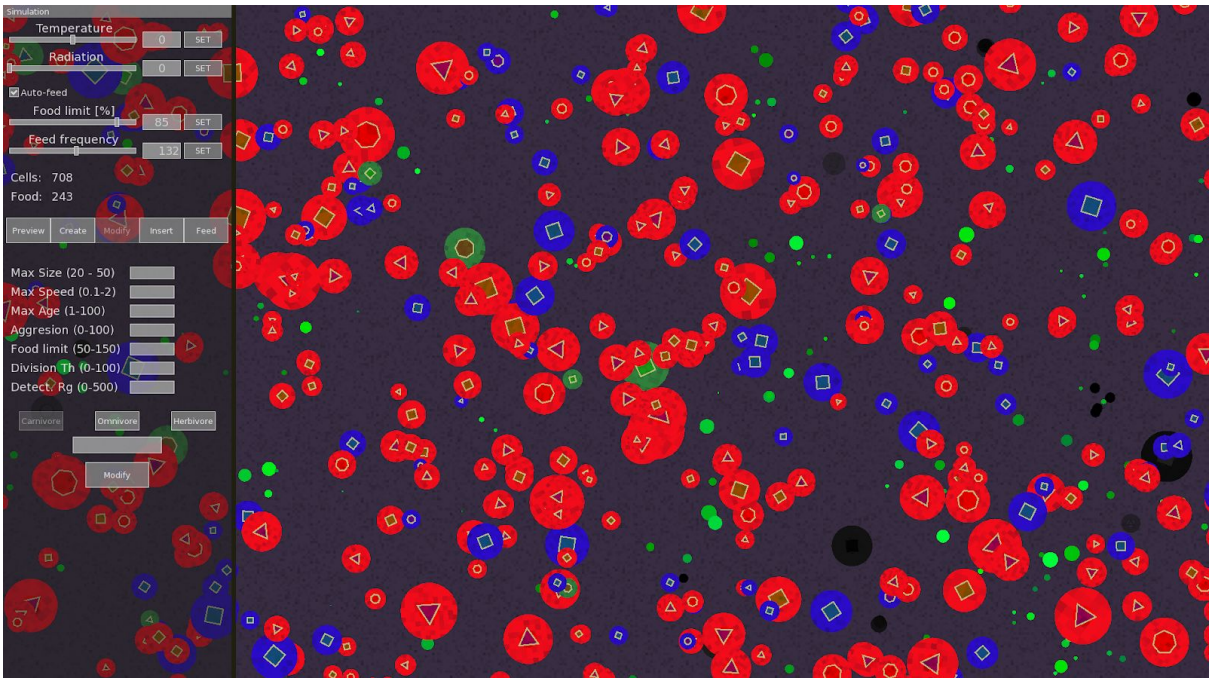
Obraz 2. Gracz uruchamia narzędzie automatycznego karmienia komórek oraz wstawia do środowiska kilka pasywnych komórek.



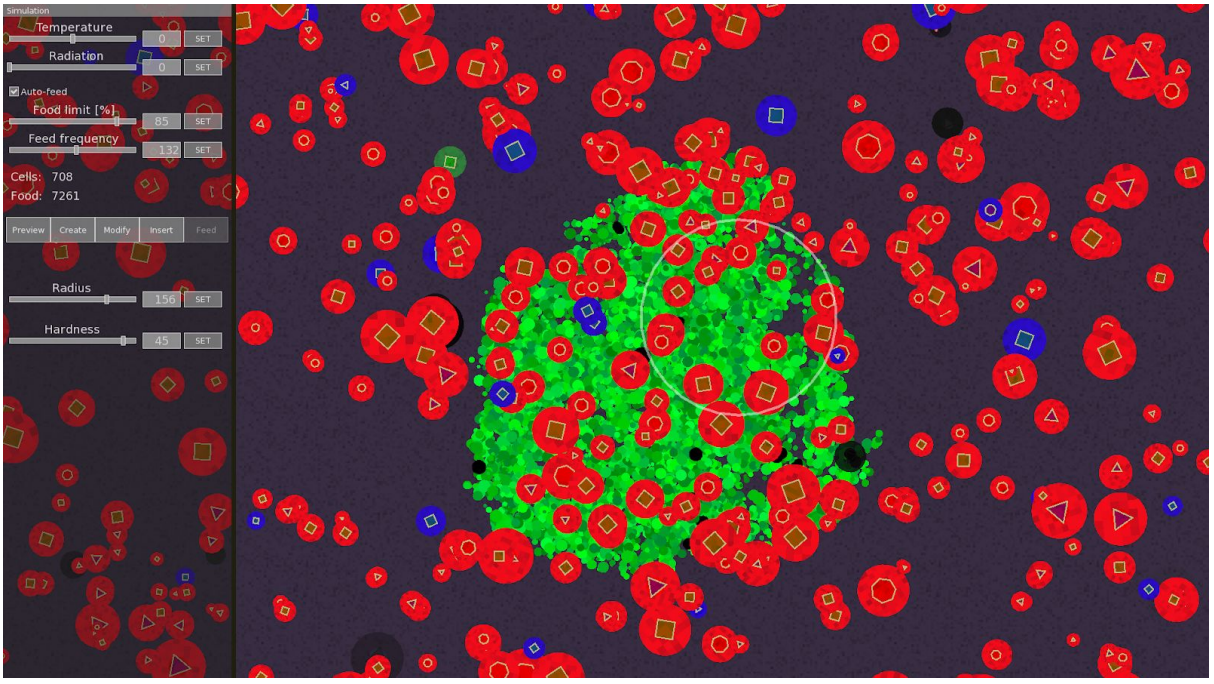
Obraz 3. Gracz obserwuje rozwój kolonii komórek, a także podgląda statystyki komórek i obrane przez nie cele. Szary okrąg oznacza wybraną komórkę, zielony - wybrane przez komórkę pożywienie, czerwony - wybraną inną komórkę do zaatakowania.



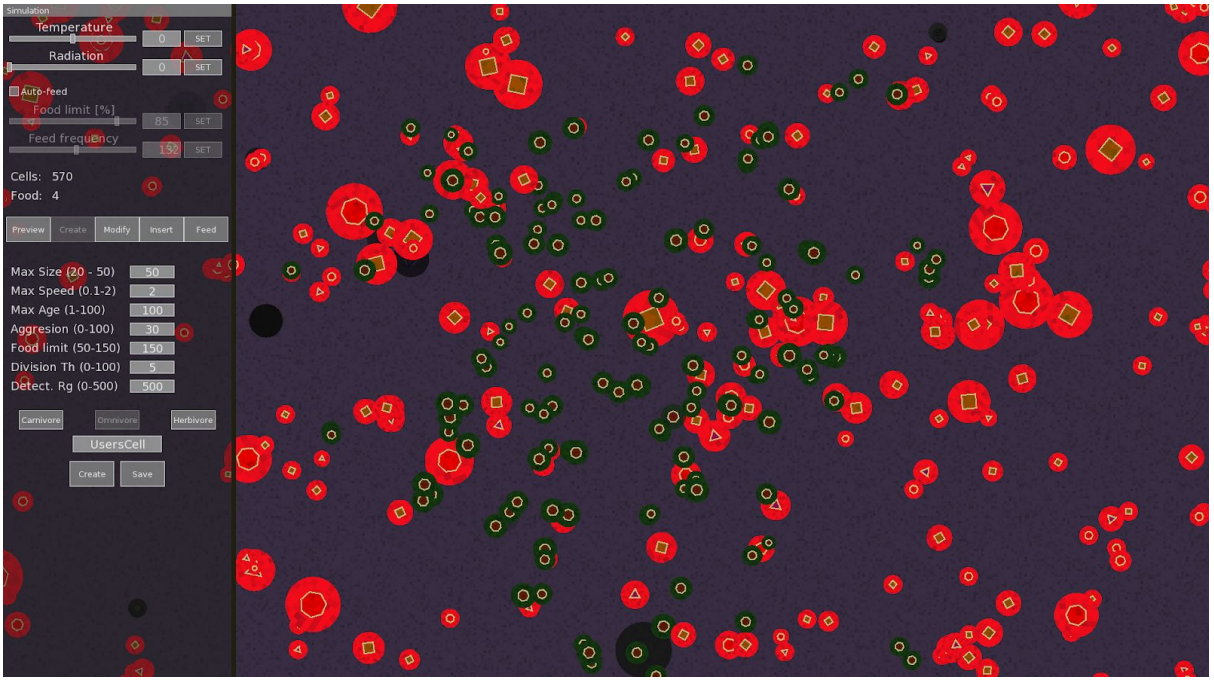
Obraz 4. Gracz obserwuje rozwój kolonii komórek. Celem urozmaicenia symulacji gracz zmienia geny kilku komórek. Następnie śledzi wybrane komórki.



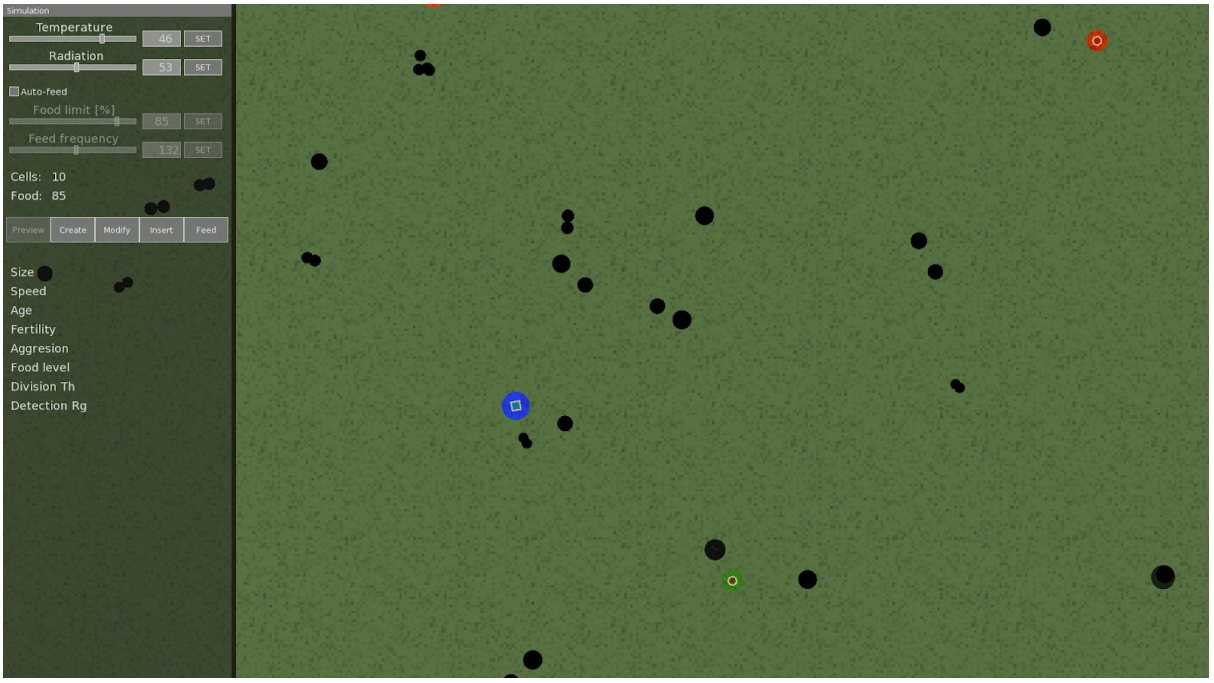
Obraz 5. Stworzone przez gracza agresywne komórki rozmnażają się oraz atakują komórki pasywne. Po pewnym czasie komórki agresywne stają się większością i zaczynają zabijać komórki pasywne.



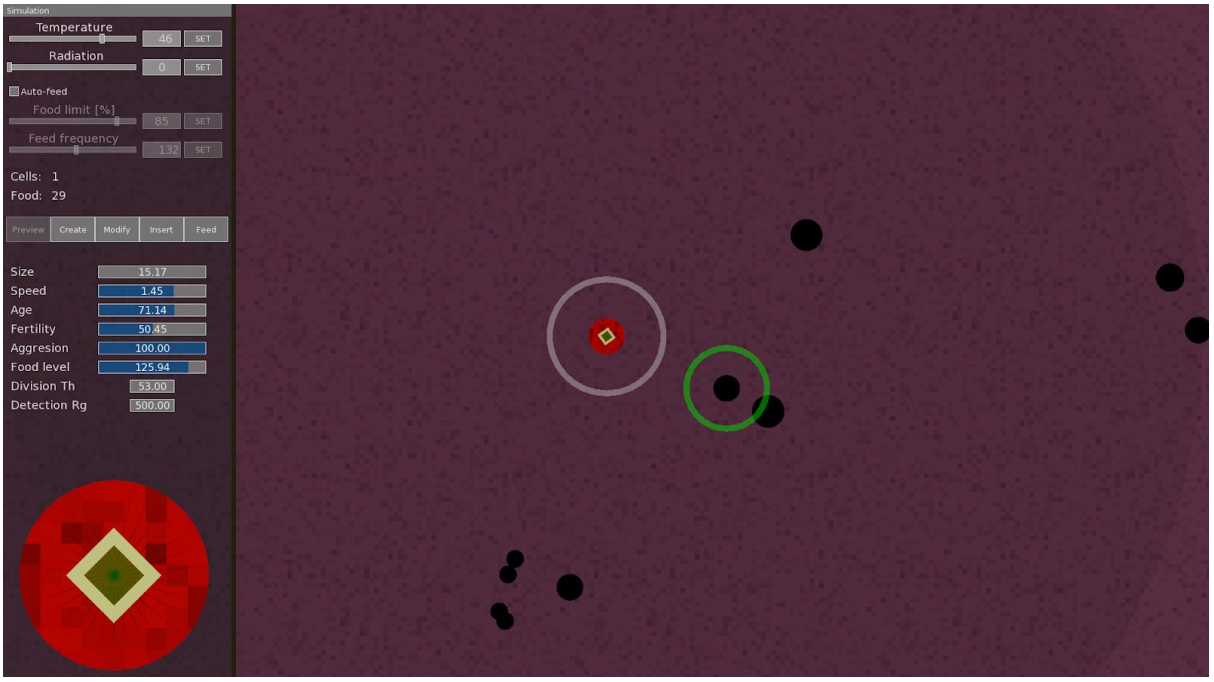
Obraz 6. Komórki pasywne zaczynają umierać ze względu na przeważającą ilość komórek agresywnych. Gracz używa narzędzia wstawiania pożywienia, by podnieść szanse komórek pasywnych na przeżycie. Gracz tworzy skupisko pożywienia.



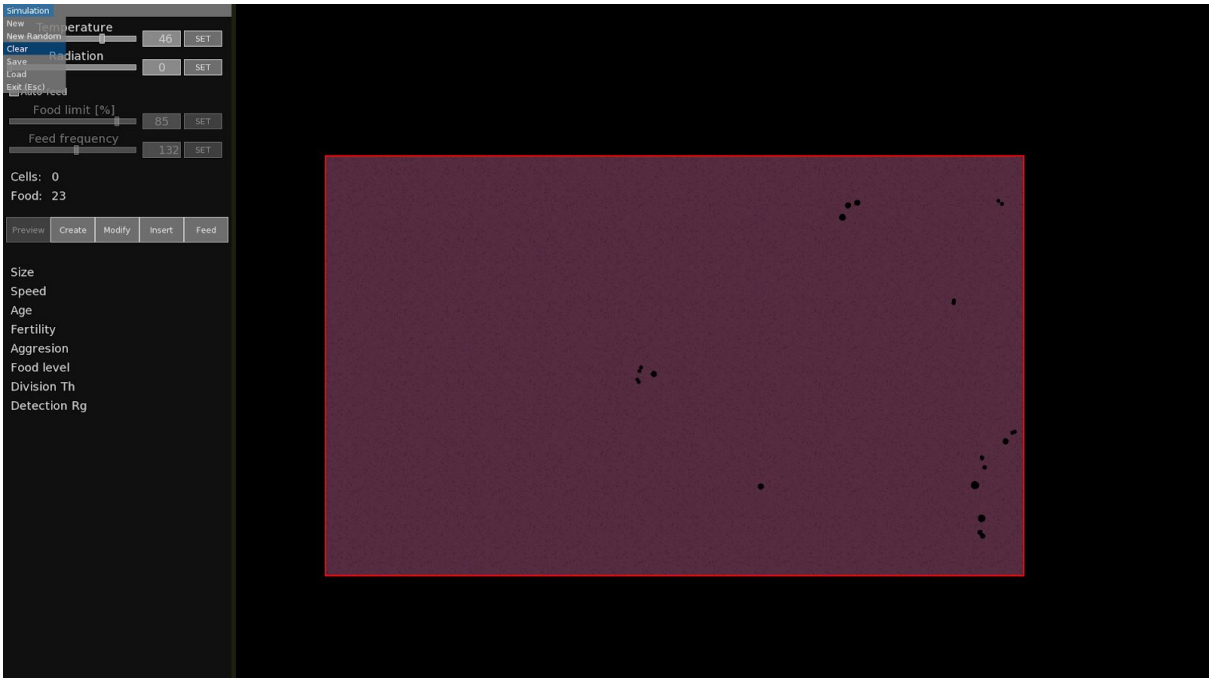
Obraz 7. Wszystkie komórki pasywne zostają uśmiercone. Komórki agresywne nie posiadają już ofiar, zatem zaczynają głodować. Po pewnym czasie głodne komórki agresywne zaczynają atakować się wzajemnie. Gracz, chcąc unieszkodliwić populację komórek agresywnych, umieszcza w środowisku komórki z własnoręcznie zdefiniowanym genotypem.



Obraz 8. Komórki stworzone przez gracza są słabsze, zatem od razu giną. Gracz podnosi poziom promieniowania celem wymuszenia mutacji genów. Jednakże za wysoki poziom promieniowania zabija większość kolonii komórek.



Obraz 9. Gracz obserwuje ostatnią żywą komórkę odżywiającą się pozostałościami po martwych komórkach.



Obraz 10. Ostatnia komórka umiera. Gracz wybiera z menu Simulation opcję Clear celem wyczyszczenia środowiska i przygotowania go do następnej symulacji.

6. Podsumowanie

Cele zrealizowane

Zrealizowane zostały wszystkie planowane funkcje oprócz algorytmu uczenia się komórek. Poniższa lista zawiera spis większych funkcjonalności:

- Plansza - Środowisko
- Narzędzie zaznaczania
- Narzędzie przesuwania
- Manipulacja planszą
- Wpływ komórek na środowisko
- Odczyt planszy z pliku
- Zapis planszy do pliku
- Automatyczne rozmieszczanie pożywienia
- Ręczne rozmieszczanie pożywienia
- Odżywianie komórek
- Rozwój komórek
- Mutacje genów komórek
- Walka komórek
- Poruszanie się komórek
- Wykrywanie kolizji
- Rozmnażanie komórek
- Wymiana genów
- Wykrywanie innych komórek
- Wykrywanie pożywienia przez komórki
- Przemieszczanie się w stronę wykrytych celów
- Panel modyfikacji środowiska
- Panel podglądu komórki
- Panel modyfikacji komórki
- Panel tworzenia nowej komórki
- Panel wstawiania zapisanych komórek
- Panel opcji manualnego karmienia
- Opcje zapisywania i tworzenia środowiska

Cele niezrealizowane

- Sztuczna inteligencja

Możliwe kierunki rozwoju systemu

System, po rozbudowaniu, można zastosować jako różnego rodzaju symulacje możliwe do wykorzystania w medycynie czy tym podobnych dziedzinach nauki.

Ponadto aplikacja została tak zaprojektowana, by można z łatwością zaimplementować przewidziane wcześniej rozszerzenia, tj.:

- Dodanie nowych cech komórek.
- Dodanie nowych czynników środowiska.
- Dodanie nowych form komórek.
- Dodanie opcji łączenia się komórek w większe organizmy.

Wnioski, obserwacje dot. pracy zespołowej

- Koniecznym elementem pracy zespołowej jest planowanie zadań oraz budowy projektu.
- Spotkania zespołu i dyskusja na temat pracy każdego członka zespołu umożliwiają dostrzeżenie faktycznego postępu prac oraz zrozumienie powiązań między projektowanymi elementami.
- Praca zespołowa przebiega lepiej, gdy wszyscy członkowie zespołu przebywają w jednym miejscu i wzajemnie się motywują.

- Praca zespołowa przyspiesza prace nad projektem, ponadto sprawia, że pojedynczy programista nie musi analizować i implementować każdego elementu samemu.
- Przeprowadzanie rewizji treści programu przez osoby z zespołu pozwala na znalezienie i poprawienie błędów, bądź rozwiązań nieoptymalnych zastosowanych w programie.

7. Literatura

- [1] Wpis w serwisie Wikipedia dotyczący Game Of Life, https://pl.wikipedia.org/wiki/Gra_w_%C5%BCycie (dostęp: 4.06.2019)
- [2] Strona domowa gry Cell Lab, https://www.cell-lab.net/ (dostęp: 4.06.2019)
- [3] Strona domowa biblioteki SFML, https://www.sfml-dev.org/ (dostęp: 4.06.2019)
- [4] Strona domowa biblioteki TGUI, https://tgui.eu/ (dostęp: 5.06.2019)
- [5] Dokumentacja biblioteki TGUI w wersji 0.8, https://tgui.eu/documentation/0.8/ (dostęp: 5.06.2019)